Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №6:

Метод наискорейшего спуска поиска минимума функции многих переменных

Студент ИУ9-61Б (Группа) — Е.А. Матвеев (И.О. Фамилия)

Проверила — А. Б. Домрачева (И.О. Фамилия)

2023 г.

Содержание

1 Цель **3**

2 Постановка задачи **4**

3 Теоретические сведения о методе наискорейшего спуска . . . **5**

4 Реализация **6**

5 Тестирование **7**

6 Вывод **10**

1 Цель

Целью данной работы является изучение метода наискорейшего спуска: анализ сходимости и оценка погрешности результатов.

2 Постановка задачи

Дано: Непрерывная дважды дифференцируемая функция многих переменных $f : \mathbb{R}^2 \to \mathbb{R}$, имеющая локальный минимум в области G.

Требуется: Найти точку $x_0 \in \mathbb{R}^2$ такую, что x_0 - это локальный минимум в G функции f. Для достижения цели работы были поставлены следующие задачи:

1. Изучить теоретическое обоснование метода наискорейшего спуска.
2. Реализовать алгоритм метода наискорейшего спуска на языке программирования python3.
3. Проанализировать сходимость метода и сравнить полученный результат с аналитическим.

3 Теоретические сведения о методе наискорейшего спуска

Метод наискорейшего спуска, как и другие градиентные методы, основывается на вычислении последовательности точек $\{x_i\}$ таких, что $f(x_i) > f(x_{i+1})$. Нахождение каждой последующей точки x_{i+1} осуществляется с помощью градиента функции в точке x_i. Из-за этого в окрестности локального минимума функции (т.е. в искомой точке), точность вычисления градиента сильно влияет на выбор следующей точки, поэтому, в некоторых случаях, точки последовательности начинают распределяться вокруг локального минимума, не достигая его.

4 Реализация

На листинге 1 представлена реализация метода наискорейшего спуска.

Листинг 1: Реализация метода наискорейшего спуска

```
import numpy as np
import pandas as pd
from sympy import diff, symbols, lambdify, log, exp
import sympy
import warnings
warnings.filterwarnings('ignore')

x_arg, y_arg = sympy.symbols('x y')
fs = x_arg**4 + sympy.log(1 + x_arg**2 + y_arg**2) - y_arg**2 + sympy.exp((0.1*y_arg))

f = sympy.lambdify(
    [x_arg, y_arg],
    fs
)

def create_derivative(target_f, target_arg):
    d = sympy.diff(target_f, target_arg)
    l = sympy.lambdify([x_arg, y_arg], d)
    return l, d

dx, dxs = create_derivative(fs, x_arg)
dy, dys = create_derivative(fs, y_arg)
dxx, dxxs = create_derivative(dxs, x_arg)
dxy, dxys = create_derivative(dxs, y_arg)
dyy, dyys = create_derivative(dys, y_arg)

def gradf(x):
    return np.array([dx(*x), dy(*x)])

def fi1(x):
    return - dx(*x)**2 - dy(*x)**2

def fi2(x):
    return (dxx(*x)*dx(*x)**2) + (2*dxy(*x)*dx(*x)*dy(*x)) + (dyy(*x)*dy(*x)**2)

def find_t(x):
    return - fi1(x)/fi2(x)

def find_minimum(x, target):
    target = np.array(target)
    df = pd.DataFrame({'norm(shift)':[], 'x':[], 'y':[], 'distance':[]})
    x_cur = x
    i = 0
    while True:
        i += 1
        t = find_t(x_cur)
        g = gradf(x_cur)
        df = df.append(pd.DataFrame({
            'norm(shift)':[np.linalg.norm(t*g)],
            'x':[x_cur[0]], 'y':[x_cur[1]],
            'distance': [np.linalg.norm(x_cur - target)],
        }))
        x_cur = x_cur - t*g
        if abs(np.max(gradf(x_cur))) < eps:
            break

    df = df.reset_index()
    df = df.drop('index', axis=1)
    return x_cur, df

eps = 0.001
target = [0, 0.391403]
x_start = np.array([1, 0])
x_min, df = find_minimum(x_start, target)
```

5 Тестирование

В качестве функции f взята функция двух переменных:

$$f(x,y) = x^4 - y^2 + \log(1 + x^2 + y^2) + e^{0.1y}$$

Если точка (x_0, y_0) - точка локального минимума, то $x_0 = 0$. Найдем y_0, решив уравнение $f'(0, y) = 0$. Оно имеет 2 решения: 0.0391403 и 72.8378.

За точку (x_0, y_0) возьмем $(1,0)$.

На рисунке 1 представлена последовательность точек x_i, вычисленных алгоритмом. Столбец *distance* отвечает за растояние от точки x_i до аналитического решения задачи.

На рисунке 2 представлена последовательность точек x_i. Точки соединены отрезками. Полученная ломанная образует прямые углы - т.к. в каждой точке градиент по направлению одного отрезка равен нулю, а по направлению второго - максимален.

В результате работы программы получена точка $(0.00120717706, 0.391738706)$, что довольно близко к ожидаемому результату.

Абсолютная погрешность: 0.00035675094343429415

Относительная погрешность: 0.0009114670644688318

	norm(shift)	x	y	distance
1	0.34751	0.58315	-0.00834	0.70701
4	0.87529	-0.17489	-0.66612	1.07189
7	1.58385	-0.01048	0.06850	0.32308
10	5.26395	0.14976	-0.54428	0.94760
13	0.96594	1.33437	4.03660	3.88175
16	0.32826	0.56438	0.31609	0.56938
19	0.17442	-0.02411	0.29276	0.10155
22	0.13532	0.09176	-0.10988	0.50961
25	0.37864	0.86523	0.48455	0.87023
28	0.42393	-0.13720	0.83758	0.46680
31	0.64235	0.06178	0.54594	0.16643
34	0.43148	-0.02013	-0.08233	0.47416
37	2.69122	0.45944	-3.06214	3.48397
40	0.33658	0.04050	-0.57978	0.97203
43	0.77019	-2.21350	3.76542	4.03529
46	0.72688	-2.13450	-1.09096	2.59875
49	1.47277	-0.46361	-1.42552	1.87514
52	0.19371	-0.18931	-0.06791	0.49680
55	0.48761	-0.08646	0.60808	0.23329
58	0.15363	0.04547	0.07826	0.31643
61	0.22729	-0.06859	-0.25867	0.65368
64	5.41741	-0.74784	-2.73640	3.21597
67	0.75398	-1.13618	2.50937	2.40347
70	0.29925	-0.47027	0.43239	0.47205
73	0.02836	-0.01377	0.41174	0.02456
76	1.33534	-0.00808	0.15396	0.23758
79	0.24988	-0.04419	0.69783	0.30960
82	0.03357	0.03369	0.39137	0.03369

Рисунок 1 — Последовательность точек x_i

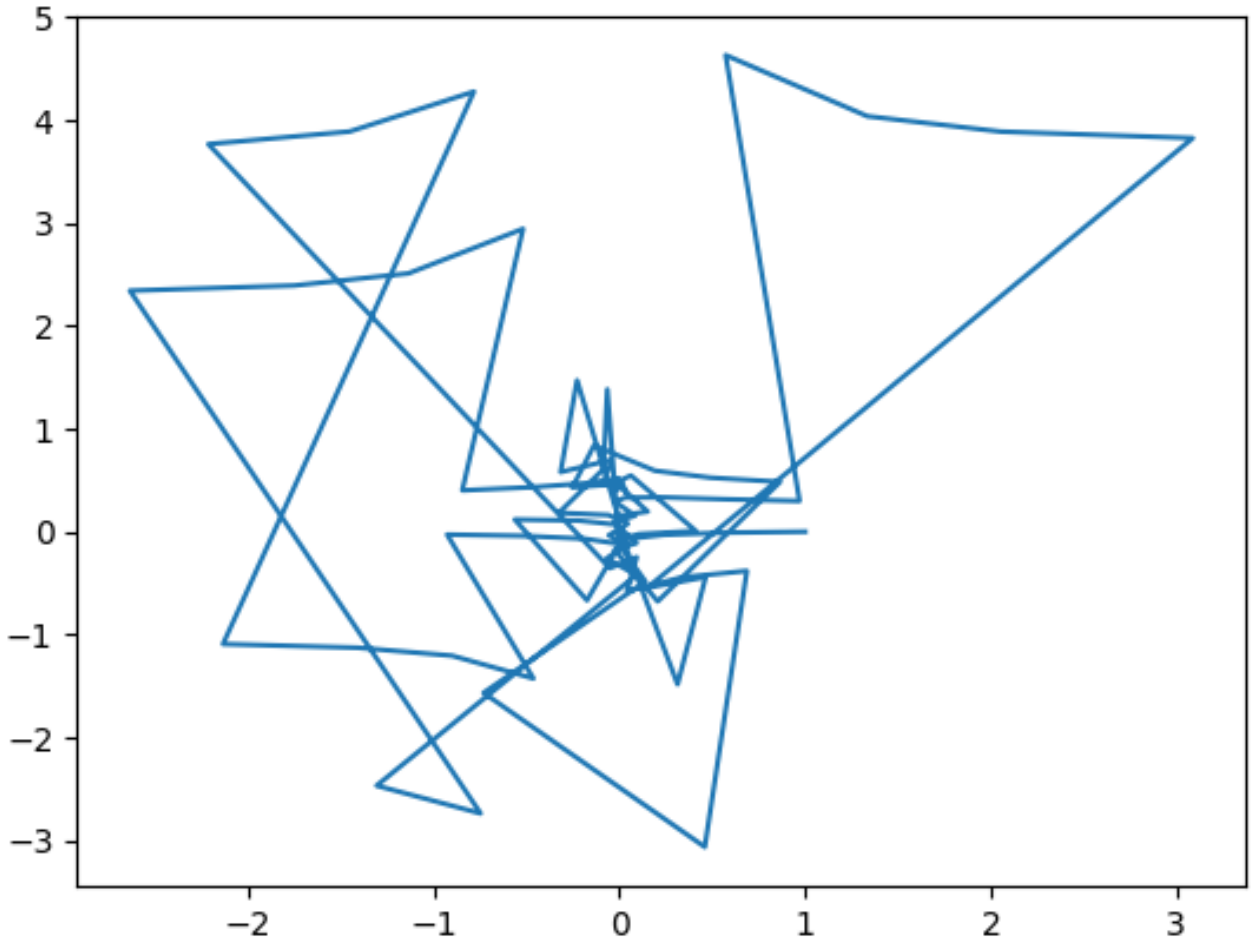

Рисунок 2 — Траектория точек x_i

6 Вывод

При выполнении лабораторной работы был изучен и реализован в программном коде метод наискорейшего спуска.

Метод испытан на тестовой функции, получены вычислительные погрешности. Имеет смысл применять данный метод при поиске глобального минимума функции (при условии, что он существует), если эта функция не имеет большого количества локальных минимумов в области поиска (достаточно выбрать несколько начальных точек), так как метод чувствителен к локальным минимумам. При несоблюдении этих условий стоит воспользоваться другими градиентными методами поиска глобального минимума.