Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Кафедра «Информатика»
кафедра

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 5

Метод Розенброка
Тема

Преподаватель ______ В. В. Тынченко
Подпись, дата Инициалы, Фамилия

Студент КИ19-17/1Б, №031939174 ______ А. К. Никитин
Номер группы, зачетной книжки Подпись, дата Инициалы, Фамилия

Красноярск 2021

1 Постановка задачи

Разработать программу, реализующую метод Розенброка.

Найти безусловный экстремум функции, выбранной в соответствии с заданием, с использованием разработанной программы.

Функция: $(x_2 + x_1 - 1)^2 + 2(x_1 - 2)^2 \to min$

2 Описание метода

Суть метода состоит во вращении системы координат в соответствии с изменением скорости убывания целевой функции. Новые направления координатных осей определяются таким образом, чтобы одна из них соответствовала направлению наиболее быстрого убывания целевой функции, а остальные находятся из условия ортогональности. Идея метода состоит в следующем.

Из начальной точки x[0] осуществляют спуск в точку x[1] по направлениям, параллельным координатным осям. На следующей итерации одна из осей должна проходить в направлении y_1 = x[1] - x[0], а другая - в направлении, перпендикулярном к y_1. Спуск вдоль этих осей приводит в точку x[2], что дает возможность построить новый вектор x[2] - x[1] и на его базе новую систему направлений поиска. В общем случае данный метод эффективен при минимизации овражных функций, так как результирующее направление поиска стремится расположиться вдоль оси оврага.

3 Исходные тексты программ

На листинге 1 представлен код программы, реализующий задание.

Листинг 1 – Метод Розенброка

```
import numpy as np
from const import f
import math


def init_directions(length):
```

Продолжение листинга 1

```
    directions = np.zeros((length, length))
    for i, row in enumerate(directions):
        row[i] = 1.
    return np.array(directions)


def calc_a_param(lambdas, directions):
    dunno = []
    for i in range(len(directions)):
        dunno.append(sum(lambdas[i:] * directions[i:]))
    a_list = np.where(lambdas == 0, directions, dunno)
    return a_list


def rosenbrock_algorithm(x0, alpha=2, beta=-0.5, epsilon=0.01, N=4, 
steps_list=None):
    x0 = np.array(x0).astype(float)
    if steps_list is None:
        steps_list = [1.] * len(x0)
    steps = np.array(steps_list).astype(float)

    directions = init_directions(len(x0))
    y = x0.copy()

    iterations = 0
    while True:
        iterations += 1
        l = 0
        unchanged_y = y.copy()

        for i in range(len(x0)):
            if f(y + steps[i] * directions[i]) < f(y):
                y += steps[i] * directions[i]
                steps[i] *= alpha
            else:
                steps[i] *= beta

        if f(y) < f(unchanged_y):
            continue

        elif f(y) == f(x0) and l <= N:
```

Окончание листинга 1

```
            l += 1
            if all(abs(steps) < epsilon):
                return y, iterations
            continue

        else:
            if math.sqrt(sum(np.square(y - x0))) <= epsilon:
                return y, iterations

            lambdas = sum((y - x0) * np.transpose(directions))

            a_list = calc_a_param(lambdas, directions)

            b = a_list[0]
            for i, a in enumerate(a_list):
                directions[i] = b / np.linalg.norm(b)
                b = a - (np.transpose(a) * directions[i]) * directions[i]

            x0 = y.copy()
            steps = np.array(steps_list).astype(float)


if __name__ == '__main__':
    print(rosenbrock_algorithm((8, 9), alpha=2, beta=-0.5, steps_list=[1, 2],
epsilon=0.6))
```

4 Исследование влияния параметров метода на точность и скорость нахождения решения

Из рисунка 1 можно заключить, что, в целом, чем меньше параметр эпсилон, тем точнее решение, однако тем больше шагов необходимо алгоритму для нахождения ответа.

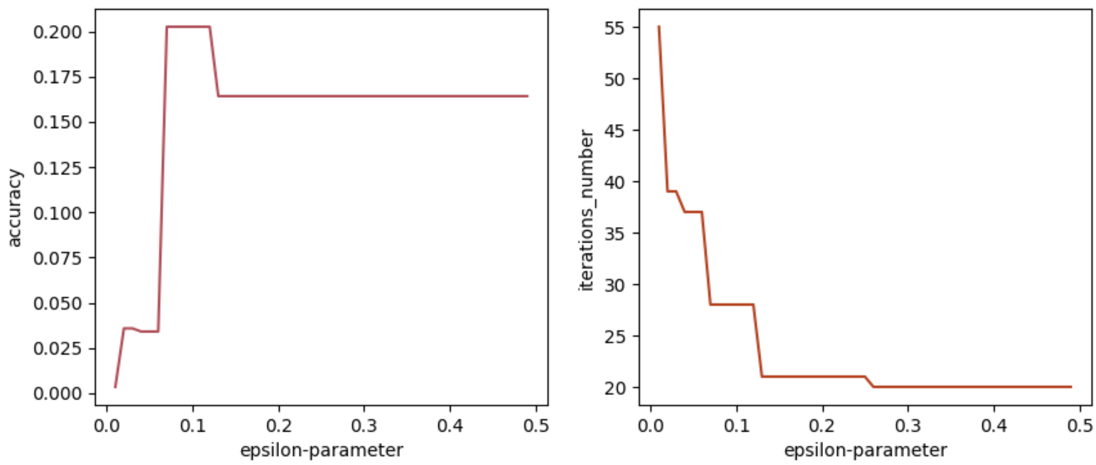

Рисунок 1 – Влияние параметра ε на точность и производительность

Из рисунка 2 можно заключить, что параметр α не имеет никаких ярко выраженных зависимостей с точностью и производительностью алгоритма.

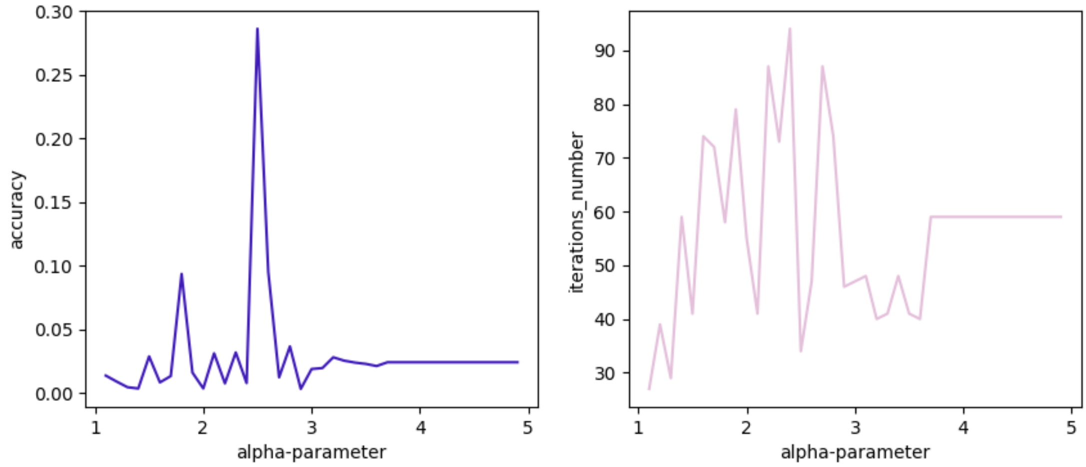

Рисунок 2 - Влияние параметра α на точность и производительность

Из рисунка 3 можно заключить, что величина β маловероятно имеет явную зависимость с изменением с точностью, однако чем ближе параметр к значению -1, тем дольше работает алгоритм.

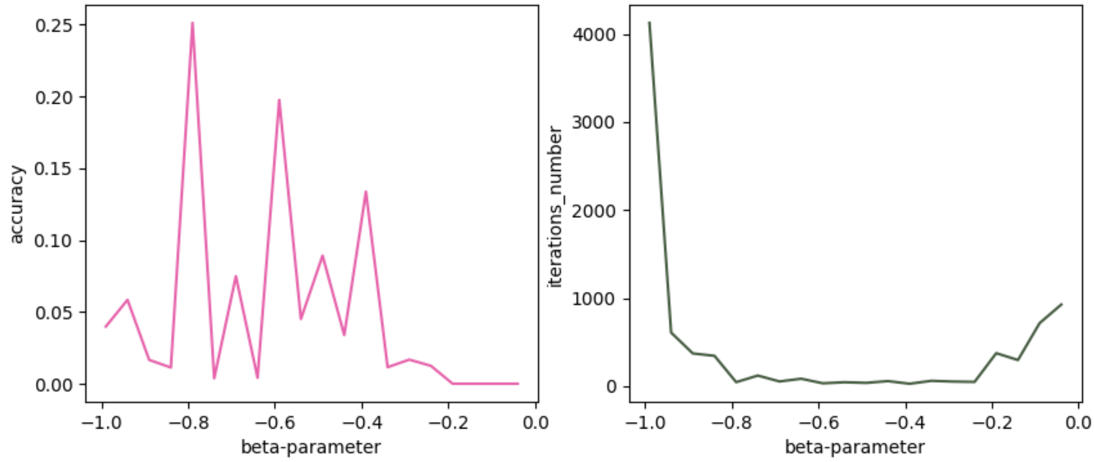

Рисунок 3 - Влияние параметра β на точность и производительность

Из рисунка 4 можно заключить, что параметр N в данном случае никак не влияет на работу алгоритма.

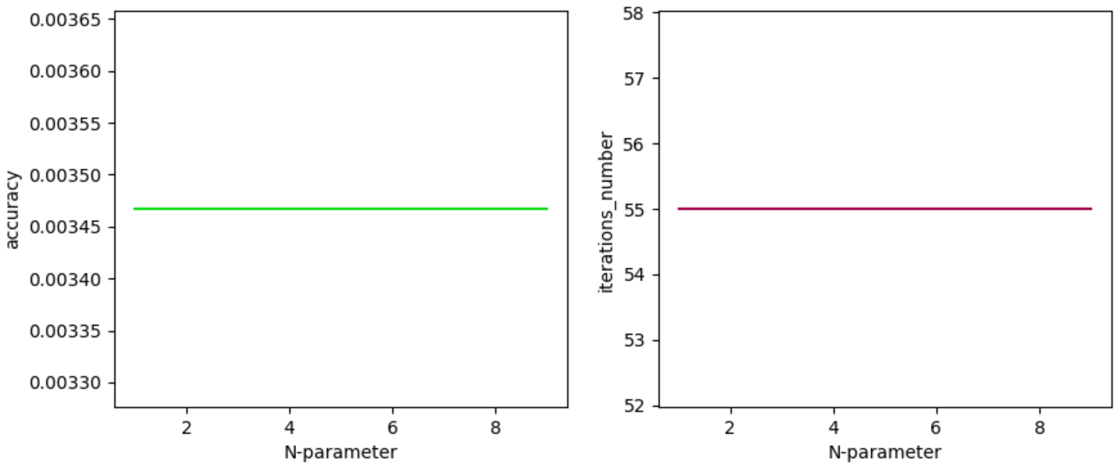

Рисунок 4 – Влияние параметра N на точность и производительность

5 Вывод

В результате данной работы был реализован и проанализирован метод Розенброка для поиска локального минимума многомерной функции. Также были проанализированы гиперпараметры метода и их влияние на работу функции.