НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із курсової роботи

із дисципліни «Методи оптимізацій»

на тему

*«Безумовна та умовна оптимізації»*

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-03  Орленко Антон Сергійович | *Ладогубець Т. С.* |

Київ – 2022

Зміст

[1. ВСТУП 3](#_Toc136703786)

[2. ПОСТАНОВКА ЗАДАЧІ 4](#_Toc136703787)

[3. ТЕОРЕТИЧНА ЧАСТИНА 5](#_Toc136703788)

[4. ХІД РОБОТИ 6](#_Toc136703789)

[4.1. Безумовна оптимізація 6](#_Toc136703790)

[4.1.1. Визначення оптимального кроку обчислення похідних 7](#_Toc136703791)

[4.1.2. Визначення оптимальної схеми обчислення похідних 9](#_Toc136703792)

[4.1.3. Визначення оптимального значення параметру в алгоритмі Свенна 11](#_Toc136703793)

[4.1.4. Визначення оптимальної точності методу одновимірного пошуку 13](#_Toc136703794)

[4.1.5. Визначення оптимального методу одновимірного пошуку 15](#_Toc136703795)

[4.1.6. Визначення оптимального критерію закінчення 17](#_Toc136703796)

[4.1.7. Визначення необхідності наявності рестартів 19](#_Toc136703797)

[4.2. Умовна оптимізація 20](#_Toc136703798)

[4.2.1. Метод штрафних функцій в залежності від розташування локального мінімуму 20](#_Toc136703799)

[4.2.2. Метод штрафних функцій в залежності від виду допустимої області (невипукла чи випукла) 27](#_Toc136703800)

[5. ВИСНОВОК 30](#_Toc136703801)

[6. СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 31](#_Toc136703802)

[7. ДОДАТОК 32](#_Toc136703803)

[7.1. Лістинг коду: 32](#_Toc136703804)

# 1. ВСТУП

Методи оптимізації відіграють важливу роль у багатьох галузях науки та технологій, де необхідно знайти найкращий розв'язок для задачі мінімізації або максимізації певної функції. Одним з найпоширеніших методів оптимізації є метод Пірсона, який базується на ітеративному покращенні поточного розв'язку шляхом використання градієнтних напрямків.

У даній курсовій роботі ми зосередимося на дослідженні збіжності метода Пірсона (алгоритм 2) при мінімізації функції Розенброка. Функція Розенброка є широко використовуваною для тестування ефективності різних методів оптимізації, оскільки вона представляє собою складну нелінійну функцію з багатьма локальними мінімумами та одним глобальним мінімумом.

Метою даної роботи є дослідження впливу різних факторів на збіжність метода Пірсона при мінімізації функції Розенброка. Ми будемо розглядати такі фактори, як початкова точка, величина кроку, критерій зупинки та похибка збіжності. Варіюючи ці фактори, ми будемо аналізувати, як вони впливають на швидкість збіжності метода Пірсона та точність отриманого розв'язку.

# 2. ПОСТАНОВКА ЗАДАЧІ

Дослідити збіжність метода Пірсона (алгоритм 2) при мінімізації функції Розенброка в залежності від:

1. Величини кроку h при обчисленні похідних.
2. Схеми обчислення похідних.
3. Виду методу одновимірного пошуку (ДСК-Пауелла або Золотого перетину).
4. Точності методу одновимірного пошуку.
5. Значення параметру в алгоритмі Свена.
6. Вигляду критерію закінчення.

().

1. Наявності рестартів.

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).

# 3. ТЕОРЕТИЧНА ЧАСТИНА

Алгоритм Пірсона (2) – квазіньютонівський алгоритм, що базується на апроксимації матриці, оберненої до матриці Гессе в точці. Вперше опублікований у журналі «The Computer Journal» у 1969р. Формула, що використовується для обчислення

Обчислення величини кроку (λ) здійснюється з використанням алгоритму Свенна для знаходження інтервалу невизначеності, та методу золотого перерізу або ДСК-Пауела для знаходження конкретного значення на цьому відрізку.

Формула для знаходження інтервалу невизначеності в алгоритмі Свенна:

# 4. ХІД РОБОТИ

## 4.1. Безумовна оптимізація

Перелік початкових налаштувань:

* Початковою точкою було обрано взяти [2 , 2]
* Початкове значення в алгоритмі Свенна:
* Початковий крок похідних:
* Початкова схема обрахунку похідних: центр
* Алгоритм одновимірного пошуку: золотий перетин
* Точність одновимірного пошуку:
* Очікувана точність:
* Наявність рестартів: так, 100 максимум
* Максимальна дозволена кількість ітерацій алгоритму Пірсона: 100000

### 4.1.1. Визначення оптимального кроку обчислення похідних

Було обрано наступний набір можливих значень h:

[1, 0.1, 0.01, 0.001, 0.0001, 1e-05, 1e-06, 1e-07, 1e-08, 1e-09, 1e-10]

В результаті було отримано наступні результати:

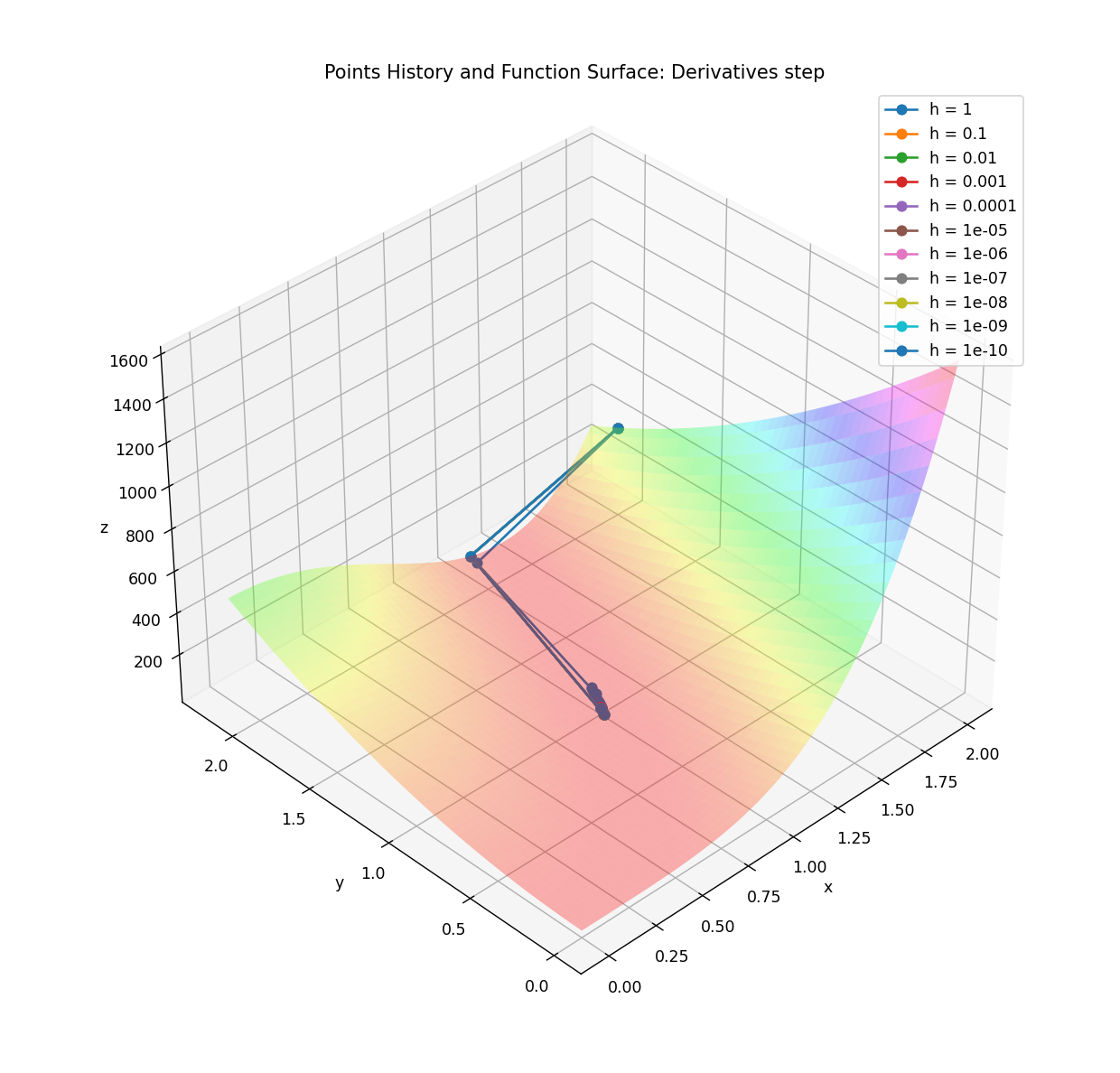


Рисунок 1 – шлях пошуку мінімуму в залежності від кроку похідних

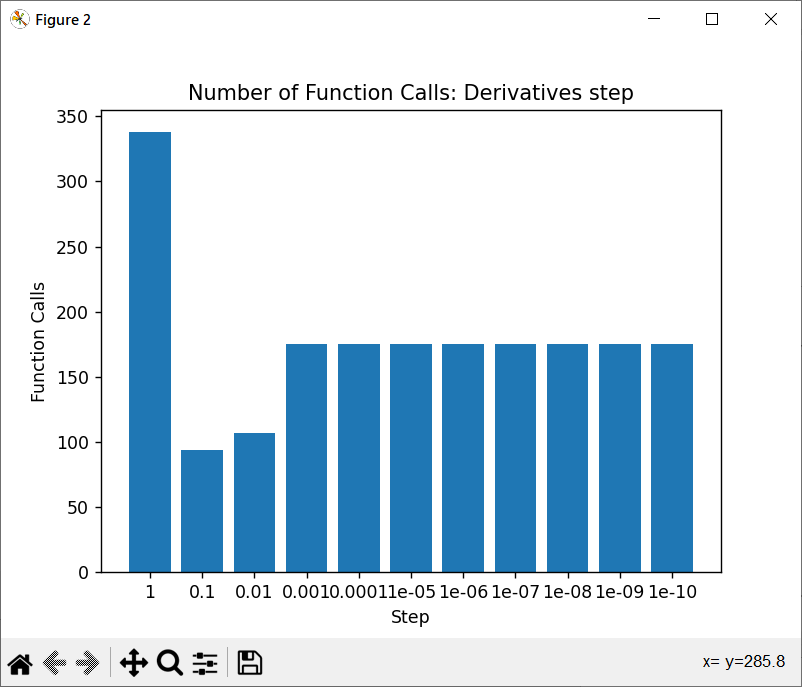
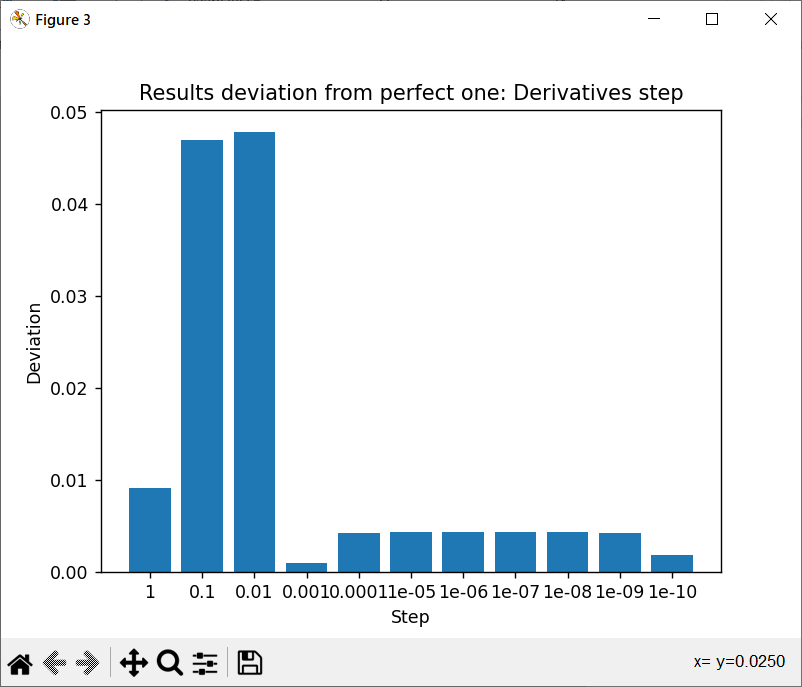
 

Рисунок 2 – кількість викликів функції в залежності від кроку похідних, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 1 – Результат тестування кроку обчислення похідних

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| h | Результат | Кількість викликів функції | Відхилення | F(x) |
| 1 | [0.99815752, 0.99106306] | 338 | 0.009124888558908533 | 0.0027652915362381923 |
| 0.1 | [0.9787037 , 0.95805489] | 94 | 0.04704173422541513 | 0.0004572946548854491 |
| 0.01 | [0.97834603, 0.9573132 ] | 107 | 0.04786498671996907 | 0.00047121212974840374 |
| 0.001 | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 | 3.01476057307639e-05 |
| 0.0001 | [0.99771406, 0.99640939] | 175 | 0.00425652278246184 | 0.00010049419151229233 |
| 1e-05 | [0.99767833, 0.9963414 ] | 175 | 0.004333077104449727 | 0.00010130387801978295 |
| 1e-06 | [0.99767817, 0.99634111] | 175 | 0.004333398600978763 | 0.00010130758064496741 |
| 1e-07 | [0.99767932, 0.9963433 ] | 175 | 0.004330935111146891 | 0.00010128141394159069 |
| 1e-08 | [0.99768256, 0.99634946] | 175 | 0.0043240018744388 | 0.00010120834667222108 |
| 1e-09 | [0.99770451, 0.99639125] | 175 | 0.004276957151178096 | 0.00010071179548206697 |
| 1e-10 | [0.99886943, 0.99858428] | 175 | 0.0018117473458245884 | 7.253519294337025e-05 |

Як можна побачити, кількість викликів функцій лишається приблизно на однаковому рівні (крім кроку в 1), у той час як найкраще наближення дає значення у 0.001.

В результаті обираємо значення 0.001.

### 4.1.2. Визначення оптимальної схеми обчислення похідних

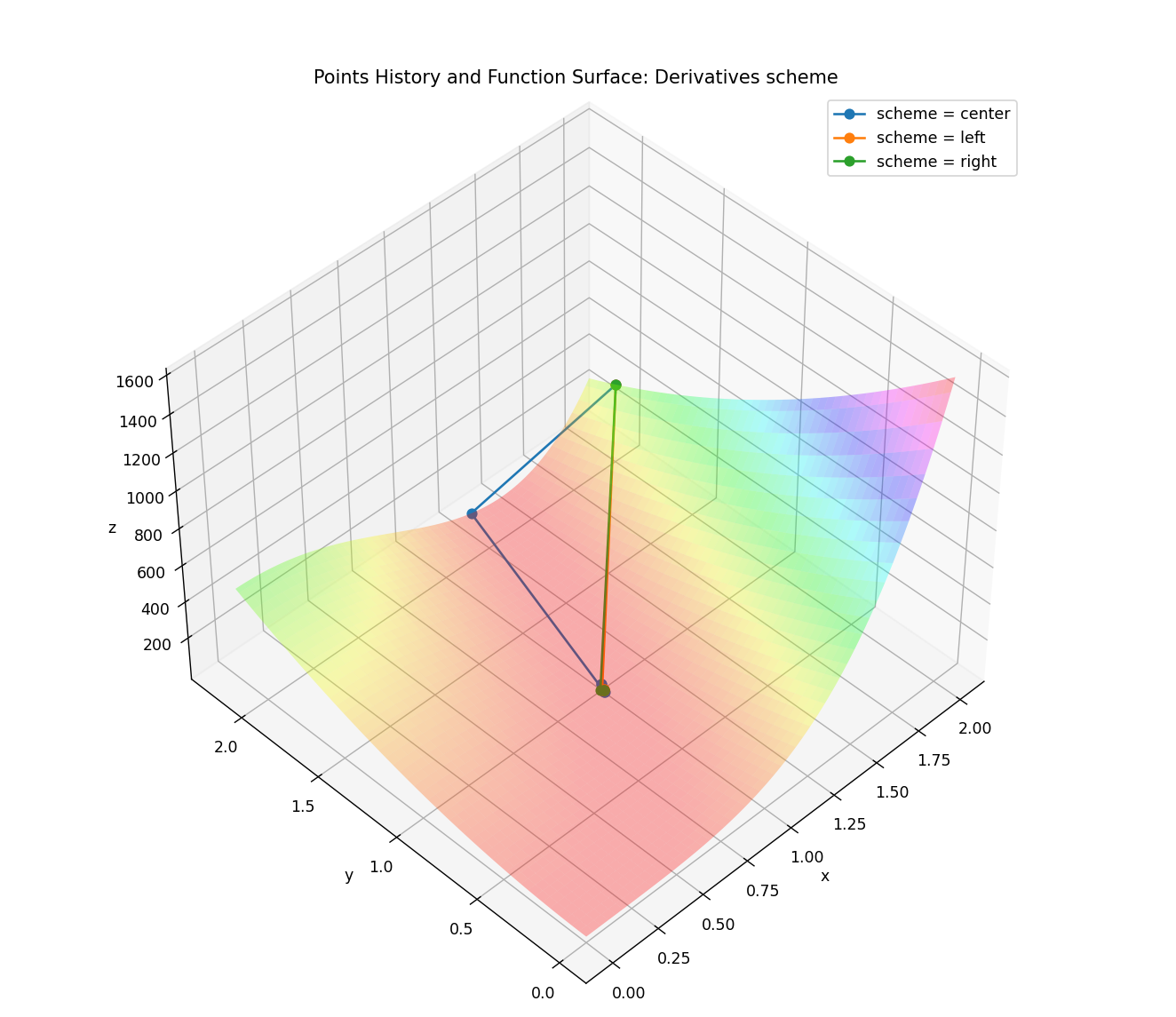


Рисунок 3 - шлях пошуку мінімуму в залежності від схеми обчислення похідних

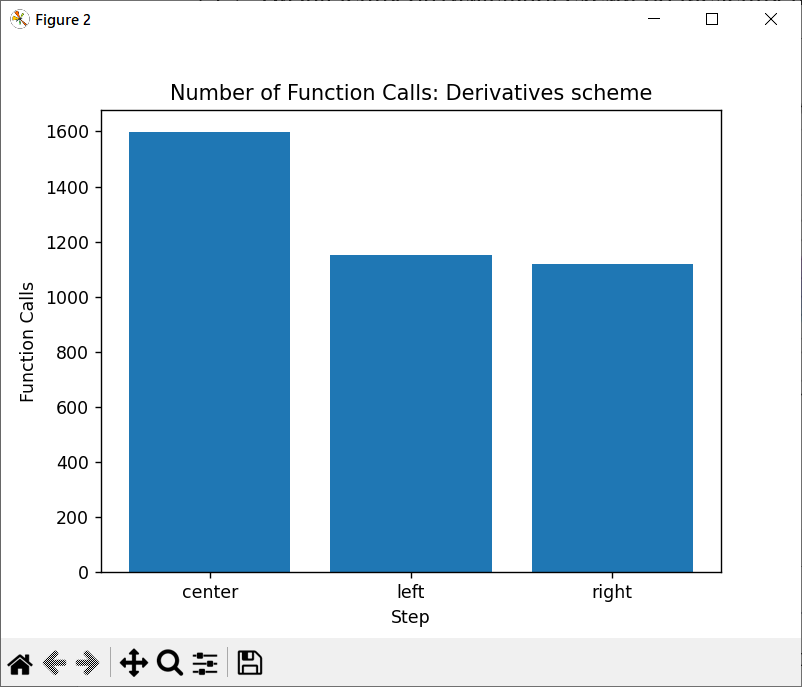
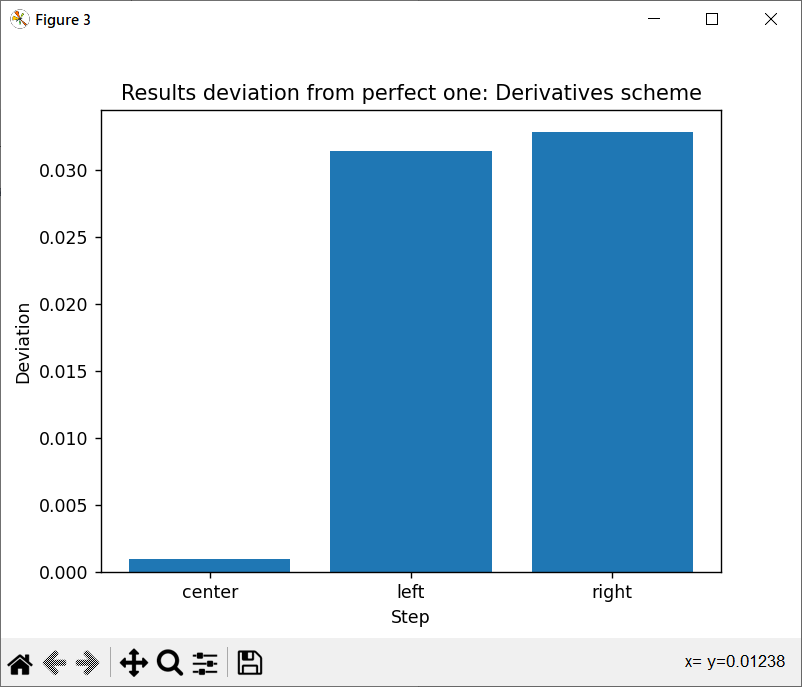
 

Рисунок 4 - кількість викликів функції в залежності від схеми обчислення похідних, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 2 - Результат тестування різних схем обчислення похідних

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Схема | Результат | Кількість викликів функції | Відхилення | F(x) |
| Центральна | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 | 3.01476057307639e-05 |
| Ліва | [0.98585819, 0.97200442] | 108 | 0.031364676232352076 | 0.00020076590711436186 |
| Права | [0.98518953, 0.9707299 ] | 75 | 0.03280379415667225 | 0.0002210790088010937 |

Не зважаючи на те, що ліва та праві схеми обчислення похідних дають меншу кількість обчислення функцій, вони й близько не наближаються до точності, що дає центральна схема. Отже, обираємо центральну схему обчислення.

### 4.1.3. Визначення оптимального значення параметру в алгоритмі Свенна

Наступним тестом я вирішив обрати тест коефіцієнту Свенна, адже, на мою думку, він має чи не найбільший вплив на точність обчислень. Для тестування було обрано наступні значення:

[1, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7, 1e-8, 1e-9, 1e-10]

Після тестування отримано наступні результати:

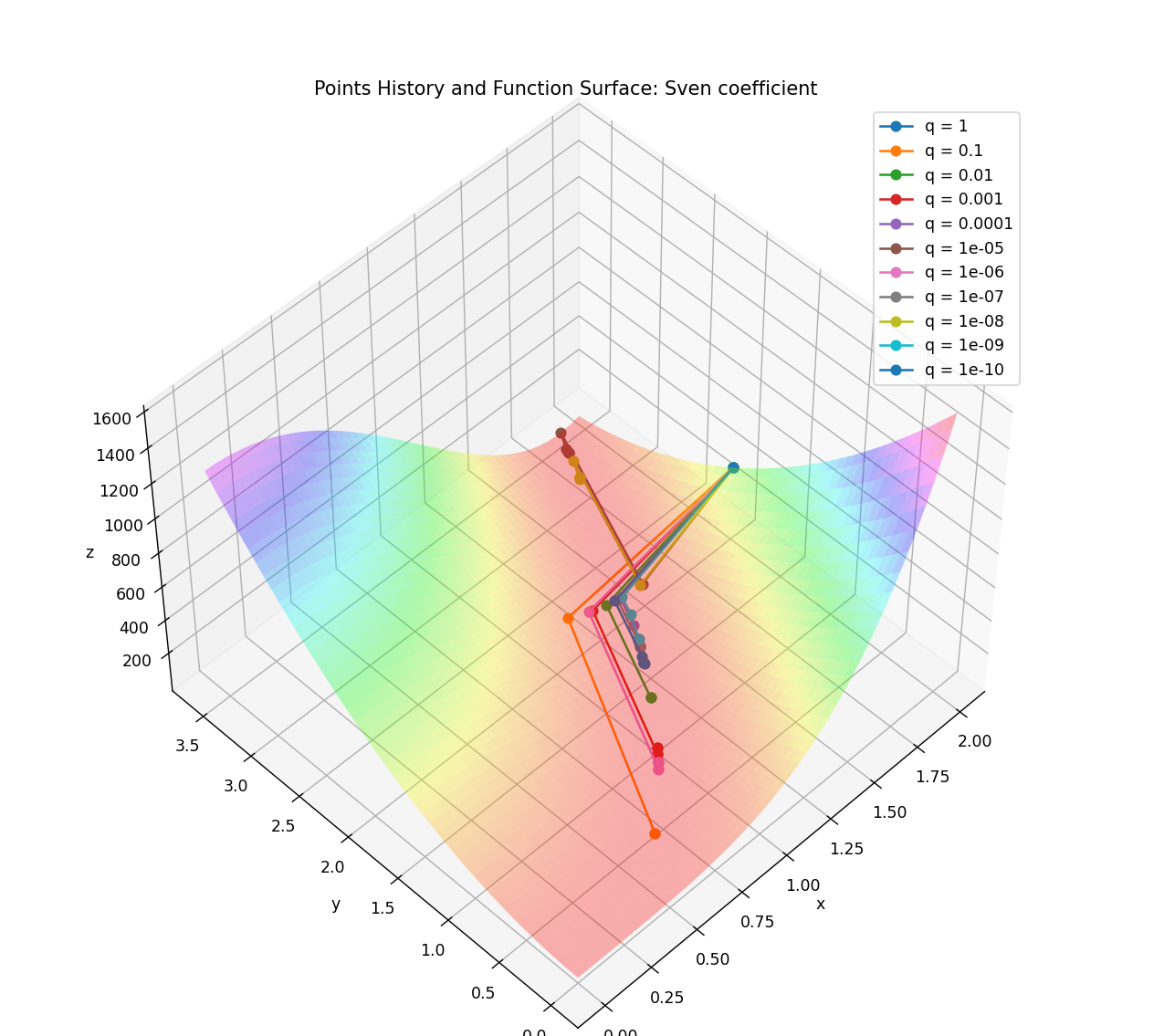


Рисунок 5 - шлях пошуку мінімуму в залежності від значення в алгоритмі Свенна

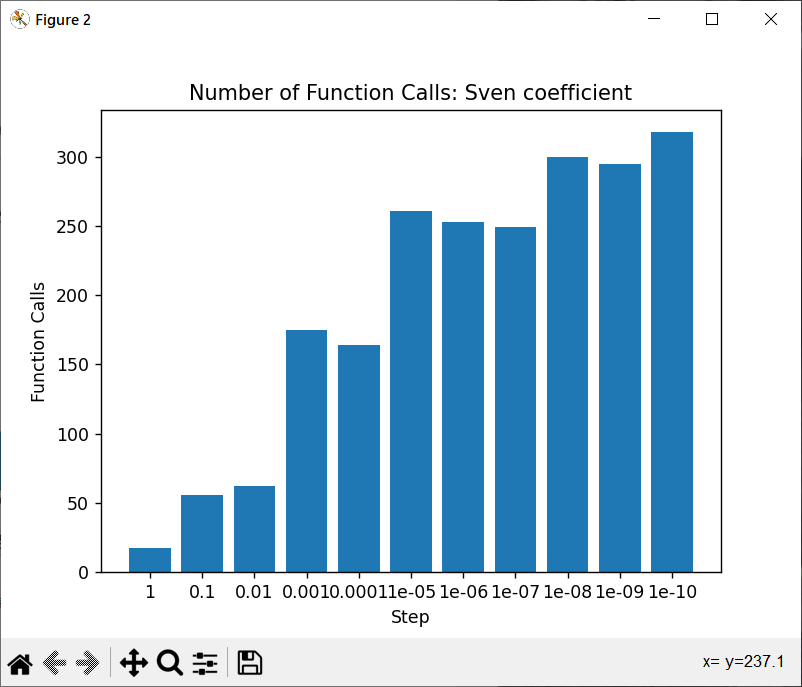
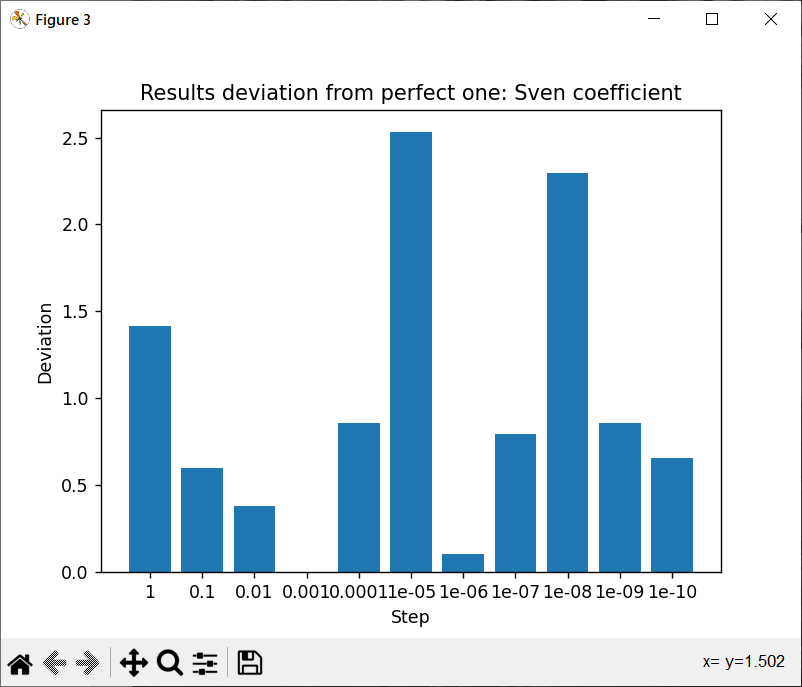
 

Рисунок 6 - кількість викликів функції в залежності від значення в алгоритмі Свенна, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 3 - Результат тестування різних коефіцієнтів в алгоритмі Свенна

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| q | Результат | Кількість викликів функції | Відхилення | F(x) |
| 1 | [2., 2.] | 17 | 1.4142135623730951 | 401.0 |
| 0.1 | [0.69519118, 0.48740334] | 56 | 0.5963755152661203 | 0.09459973376924201 |
| 0.01 | [1.16101169, 1.34610419] | 62 | 0.3817235567920611 | 0.026264777297228616 |
| 0.001 | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 | 3.01476057307639e-05 |
| 0.0001 | [1.34006805, 1.78443671] | 164 | 0.8549779114509599 | 0.12851873103689151 |
| 1e-05 | [1.83948724, 3.38884541] | 261 | 2.532058695752744 | 0.7073726812082758 |
| 1e-06 | [0.95337366, 0.90614631] | 253 | 0.1047975685522757 | 0.0029440864821395907 |
| 1e-07 | [1.31773984, 1.7296616 ] | 249 | 0.7958421090193304 | 0.10555093913283825 |
| 1e-08 | [1.77649002, 3.16005623] | 300 | 2.2953822513893827 | 0.6046502519070082 |
| 1e-09 | [1.34126369, 1.78492843] | 295 | 0.8559050991645947 | 0.13622887676496082 |
| 1e-10 | [1.26794807, 1.60016154] | 318 | 0.6572594946786471 | 0.07746740404411803 |

Найбільш підходящим значенням було обрано 0.001, адже за цього значення досягається найкращий результат, до того ж за адекватну кількість викликів функції.

### 4.1.4. Визначення оптимальної точності методу одновимірного пошуку

Було обрано наступний набір можливих значень eps:

[1, 0.1, 0.01, 0.001, 0.0001, 1e-05, 1e-06, 1e-07, 1e-08, 1e-09, 1e-10]

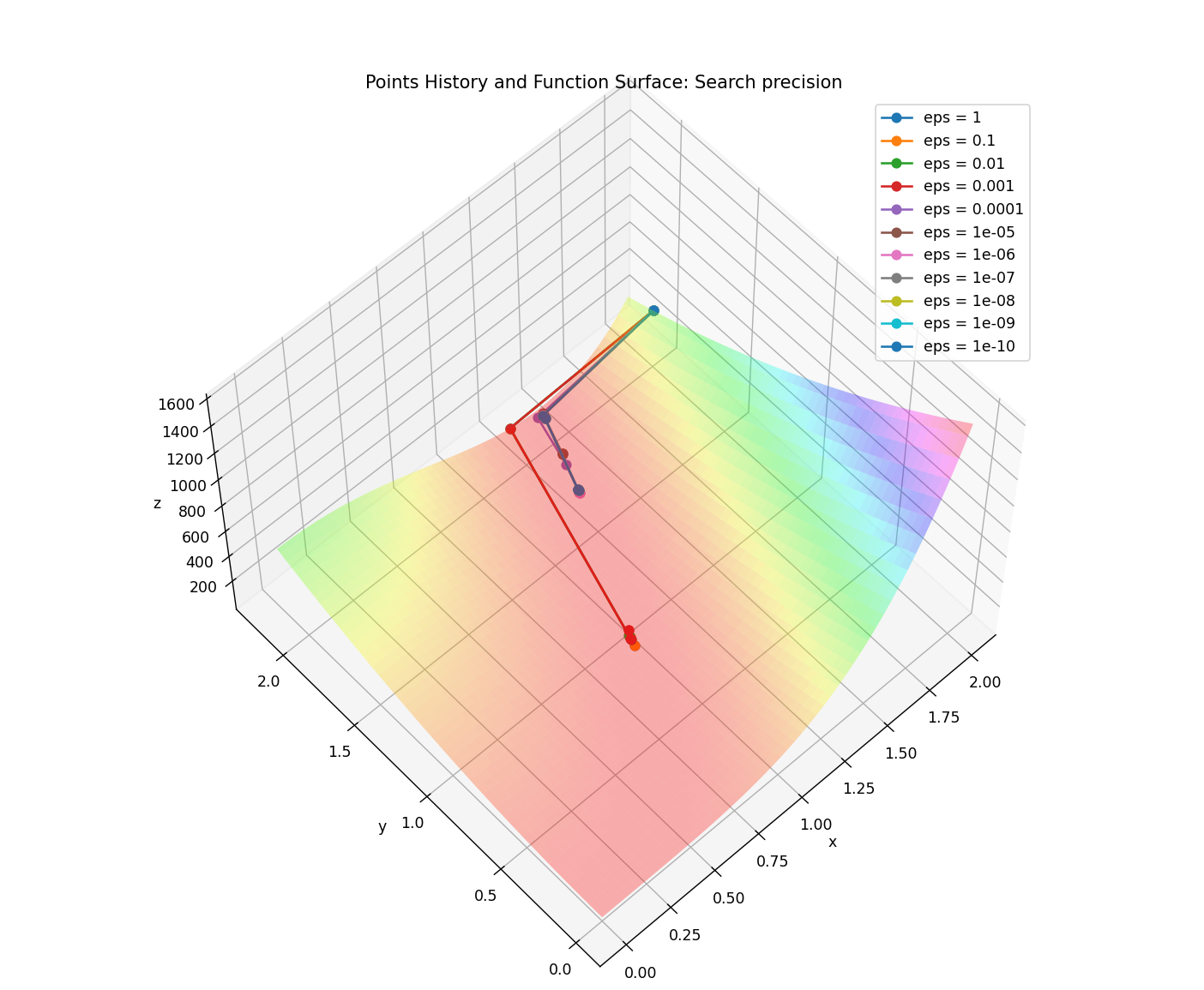


Рисунок 7 - шлях пошуку мінімуму в залежності від точності одновимірного пошуку

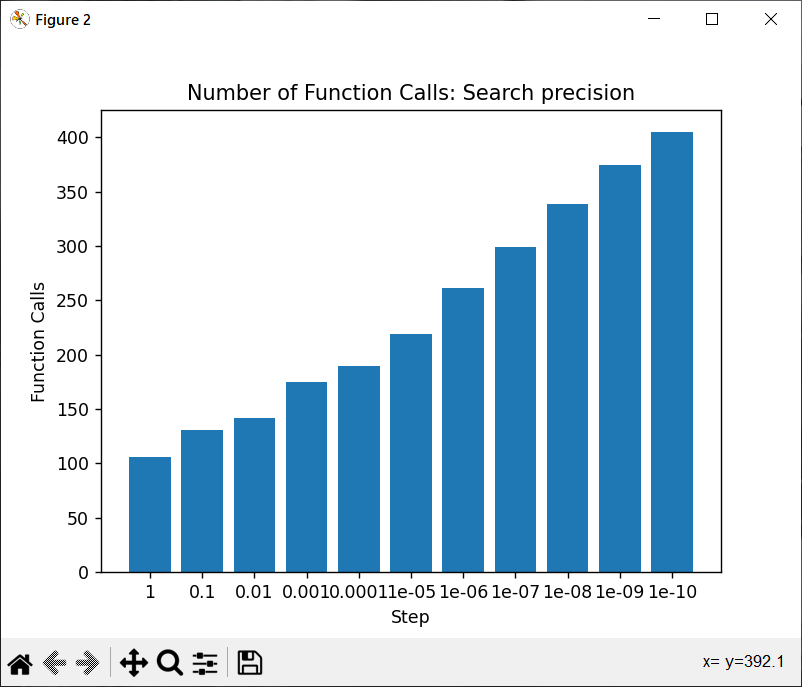
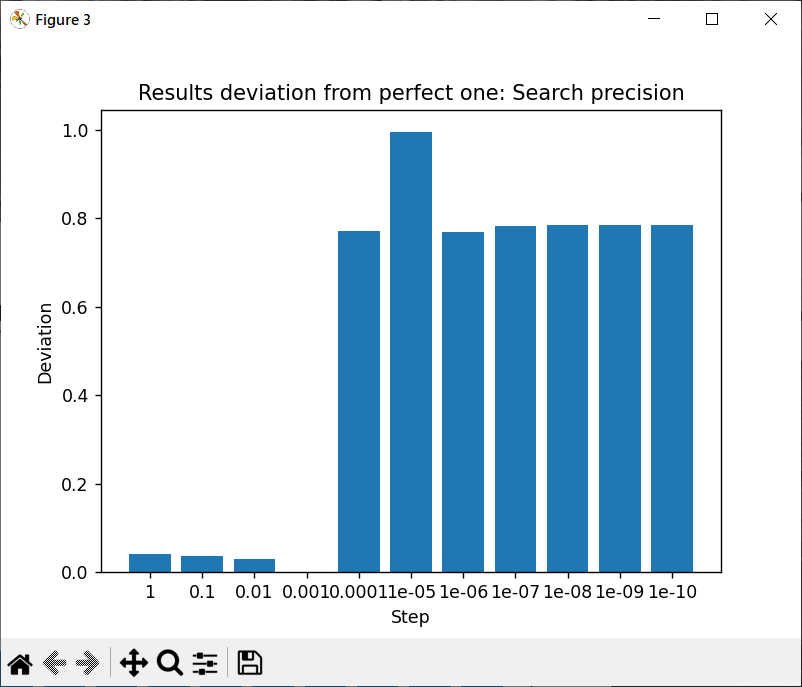
 

Рисунок 8 - кількість викликів функції в залежності від точності одновимірного пошуку, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 4 - Результат тестування різних точностей одновимірного пошуку

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| eps | Результат | Кількість викликів функції | Відхилення | F(x) |
| 1 | [0.98172253, 0.96416459] | 103 | 0.040227389056102625 | 0.00034892494111017197 |
| 0.1 | [0.98315254, 0.96720288] | 131 | 0.03687123220578365 | 0.00032153238648829744 |
| 0.01 | [0.98595322, 0.97335062] | 142 | 0.030124766253736997 | 0.00035277647087267506 |
| 0.001 | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 | 3.01476057307639e-05 |
| 0.0001 | [1.31048281, 1.70491682] | 190 | 0.7702644314293814 | 0.11189579887132856 |
| 1e-05 | [1.38978528, 1.91574491] | 219 | 0.9952493661033969 | 0.17676469288296112 |
| 1e-06 | [1.31054723, 1.70364085] | 261 | 0.7691228958327875 | 0.11574162030159302 |
| 1e-07 | [1.31583275, 1.71723455] | 299 | 0.783693647915111 | 0.11986119375598935 |
| 1e-08 | [1.31614728, 1.71804571] | 339 | 0.7845627733598215 | 0.12010729404255104 |
| 1e-09 | [1.31610589, 1.71793894] | 375 | 0.784448379777395 | 0.12007490326431854 |
| 1e-10 | [1.31609759, 1.71791754] | 405 | 0.7844254438232025 | 0.12006839683316929 |

Найкращою точністю є 0.001, адже за 175 викликів функції вдалося обчислити значення мінімуму з найкращою точністю.

### 4.1.5. Визначення оптимального методу одновимірного пошуку

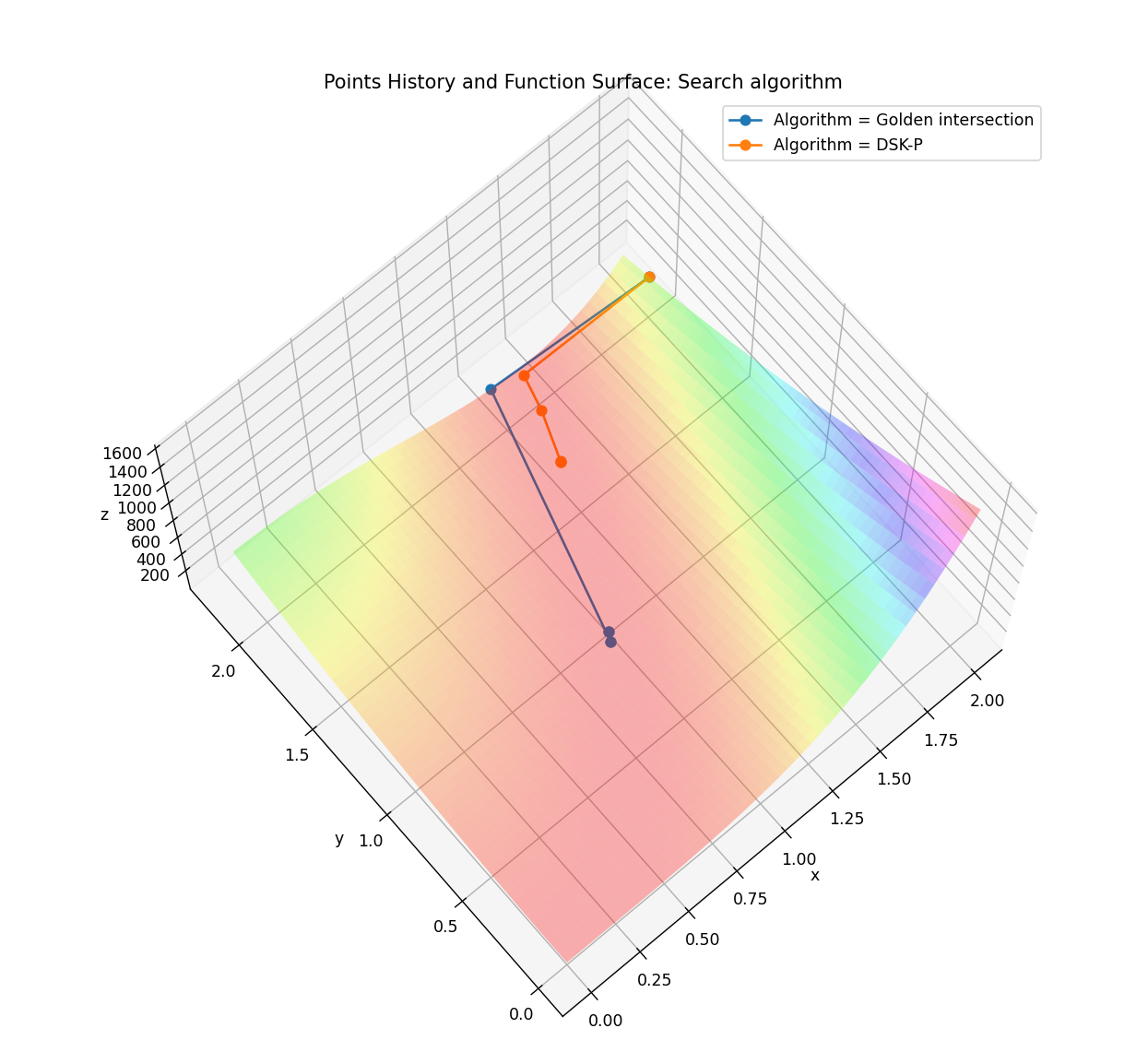


Рисунок 9 - шлях пошуку мінімуму в залежності від обраного методу одновимірного пошуку

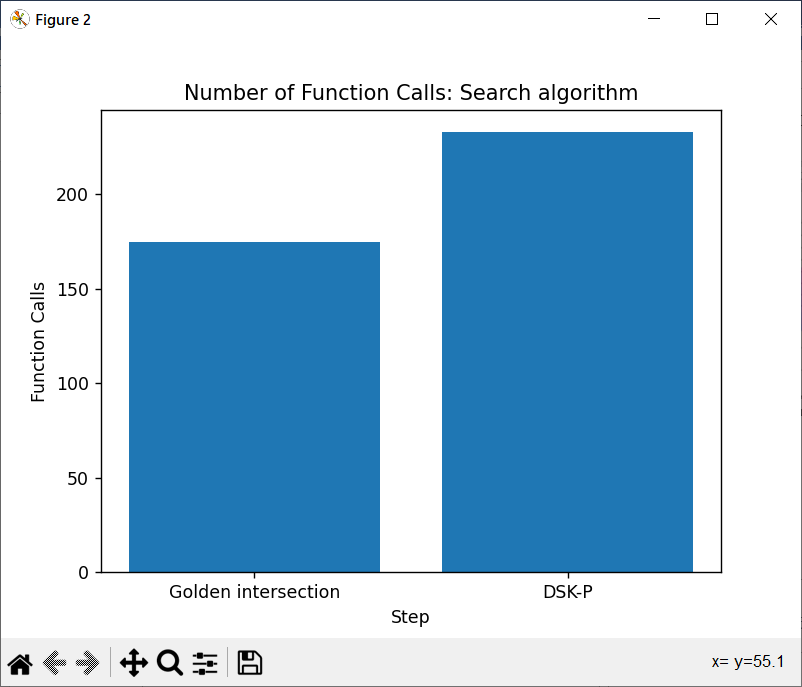
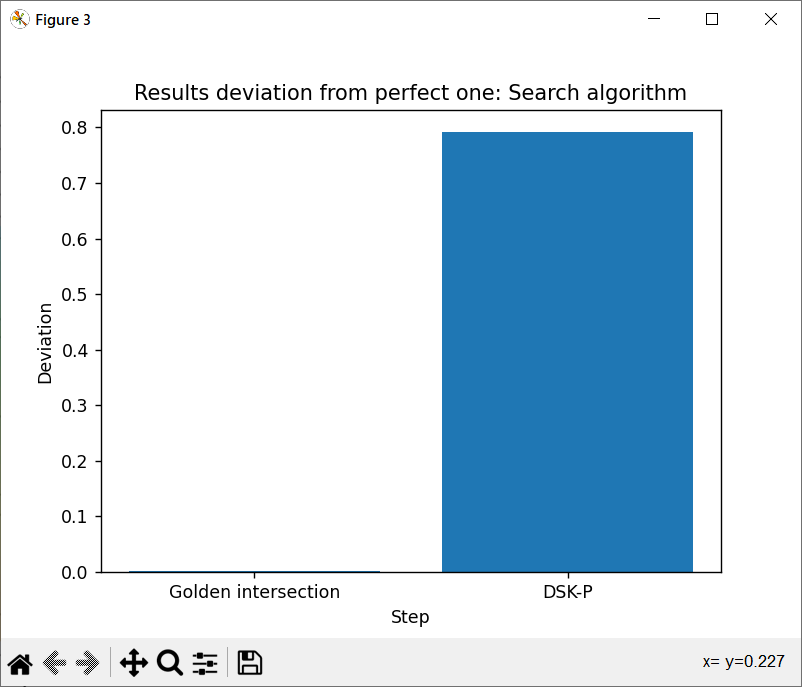
 

Рисунок 10 - кількість викликів функції в залежності від обраного методу одновимірного пошуку, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 5 - Результат тестування різних варіантів одновимірного пошуку

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Метод пошуку | Результат | Кількість викликів функції | Відхилення | F(x) |
| Золотий перетин | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 | 3.01476057307639e-05 |
| ДСК-П | [1.31871172, 1.72480984] | 233 | 0.7917868768208998 | 0.12171493809505807 |

Видно, що за використання методу Золотого перетину, отримуємо в 10 разів гіршу точність ніж за використання методу ДСК-Пауелла, виконавши у півтора рази менше обчислень цільової функції. Однак, оскільки ми говоримо про різницю у точності між 6 та 7 знаками після коми, має сенс і надалі використовувати метод золотого перетину.

### 4.1.6. Визначення оптимального критерію закінчення

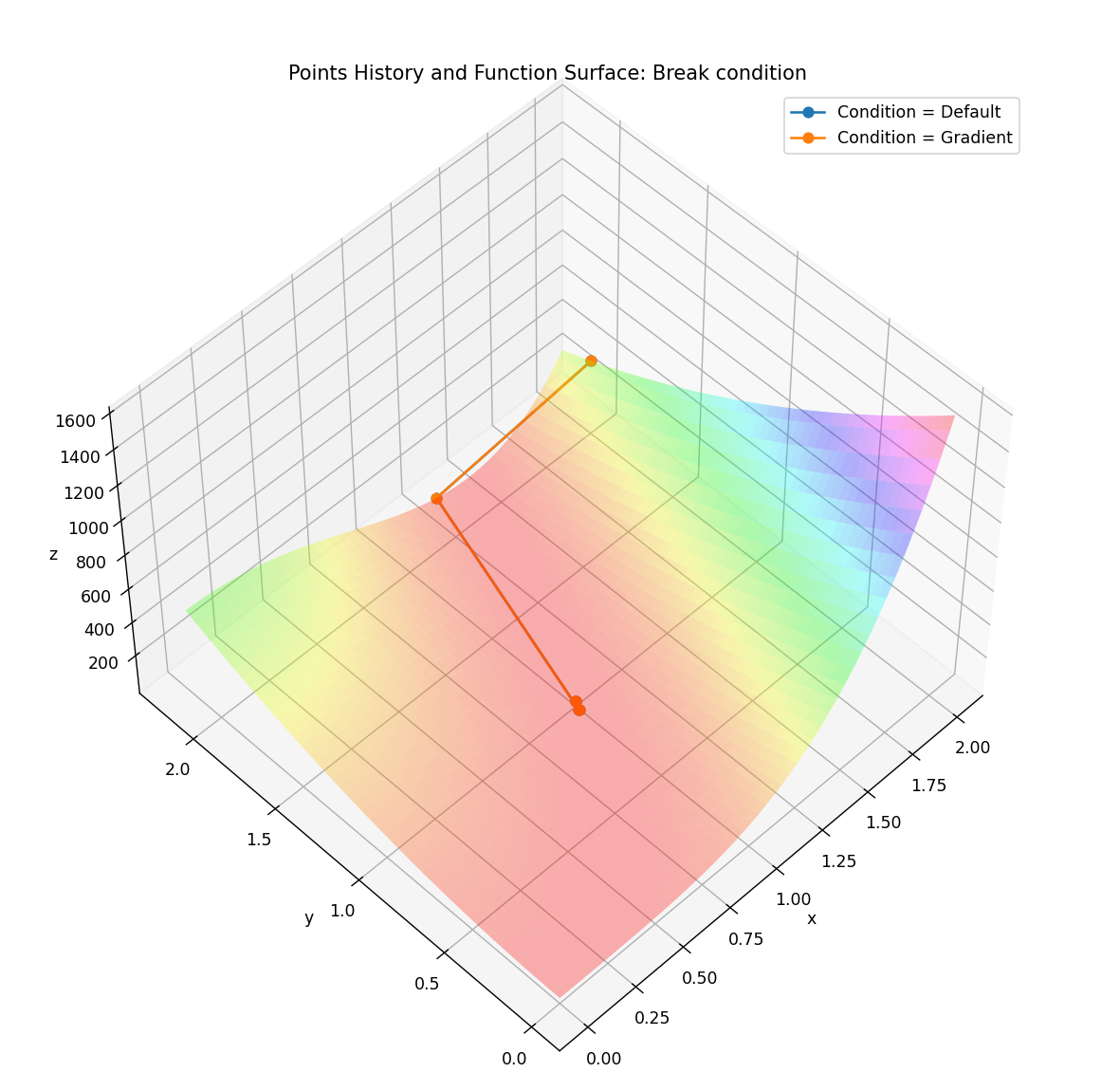


Рисунок 11 - шлях пошуку мінімуму в залежності від обраного критерію закінчення

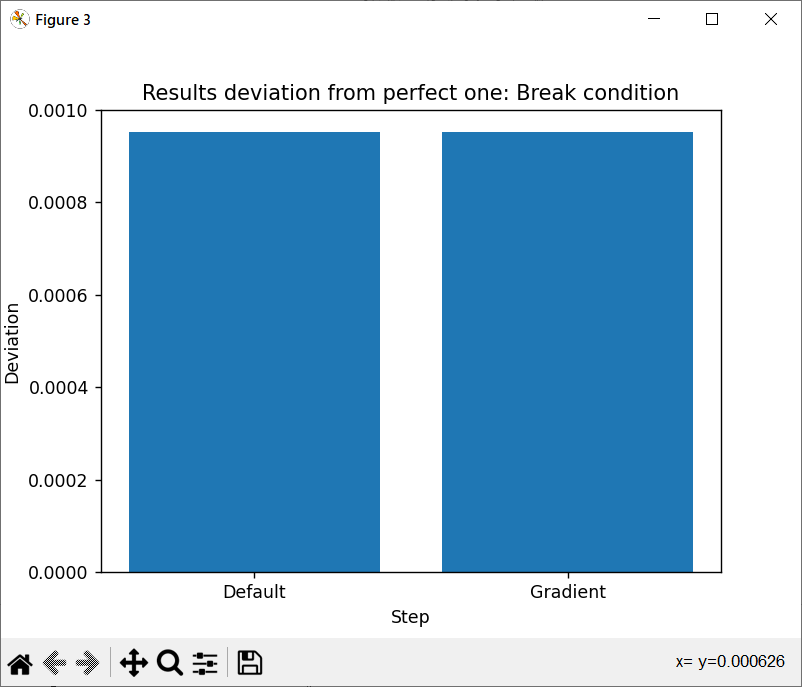
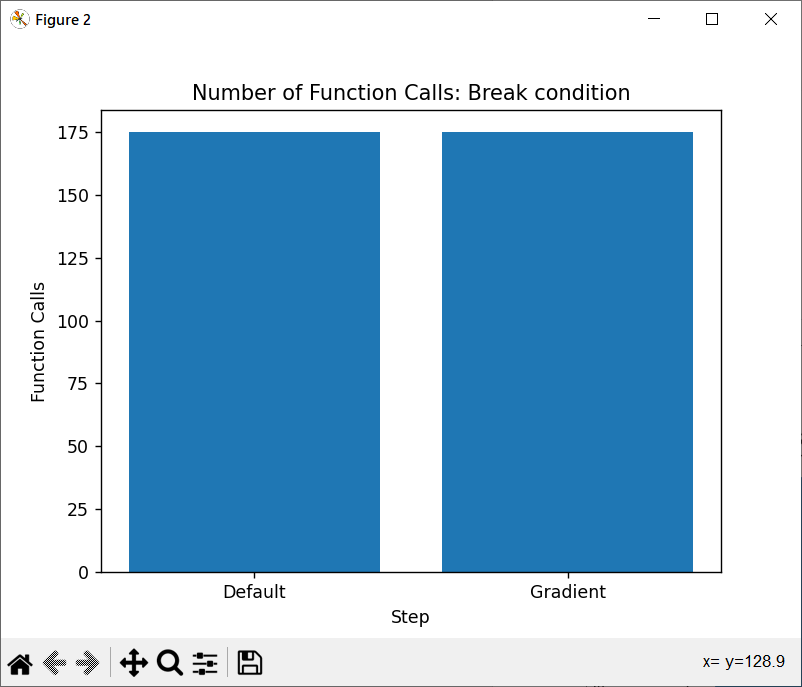


Рисунок 12 - кількість викликів функції в залежності від обраного критерію закінчення, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 6 - Результат тестування різних варіантів одновимірного пошуку

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерій | Результат | Кількість викликів функції | Відхилення | F(x) |
| Звичайний | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 | 3.01476057307639e-05 |
| Градієнт | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 | 3.01476057307639e-05 |

Результати є абсолютно однаковими

### 4.1.7. Визначення необхідності наявності рестартів

Під час попередніх досліджень, рестарт не відбувся жодного разу (умова для початку рестарту – не виконується обраний критерій закінчення та напрямок пошуку = 0 або знаменник з алгоритму Пірсона = 0 або кількість ітерацій більша за максимальну. Рестартом називаємо скидання матриці А до початкового значення та, відповідно, перерахунок напрямку руху. Однак новим початковим значенням х буде останнє обчислене значення перед початком рестарту.

Єдиним варіантом дослідження є прибрати критерії виходу та замінити їх на перевірку кількості здійснених рестартів. Протестуємо це на кількості рестартів у 1, 10, 100, 1000 та 10000.

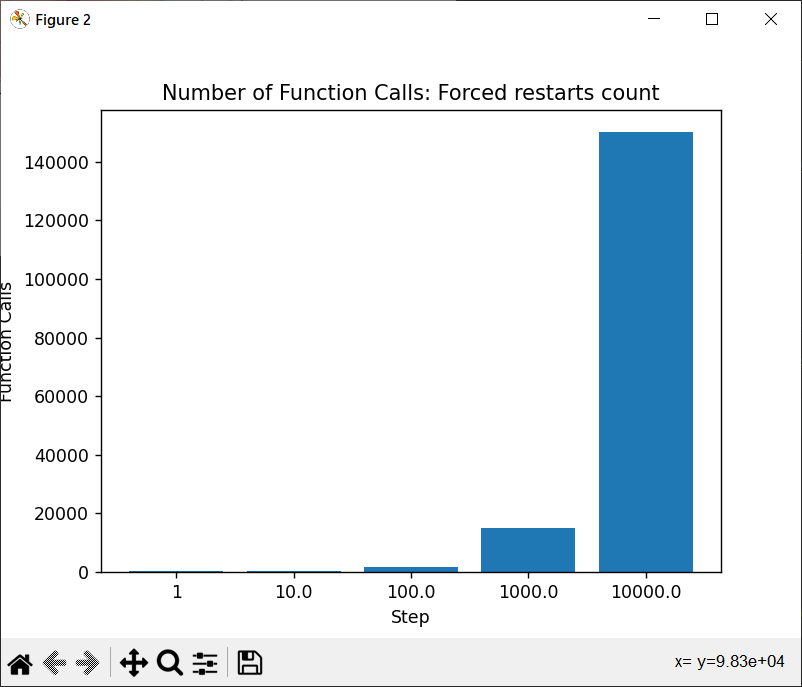
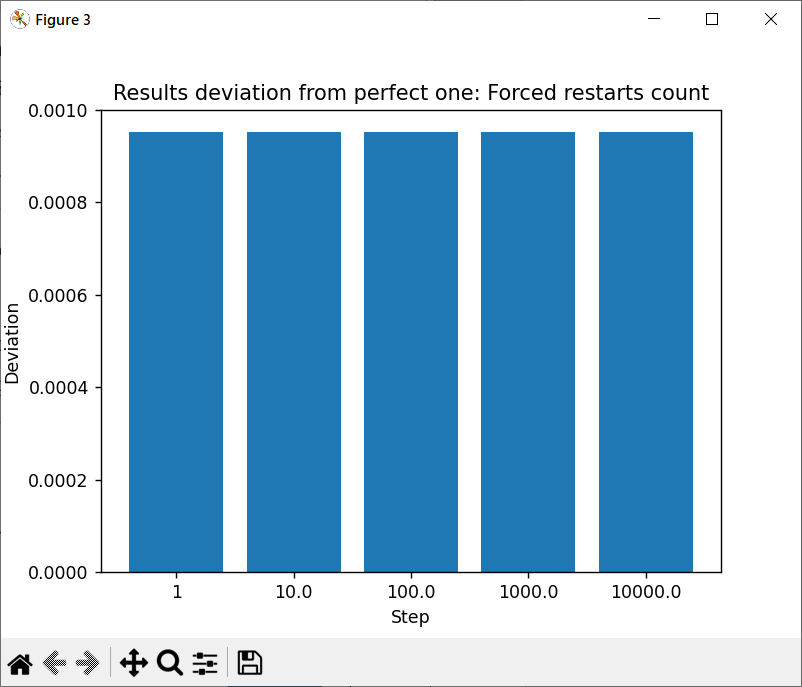


Рисунок 13 - кількість викликів функції в залежності від кількості рестартів, та відхилення від ідеального розв’язку

Результати з точки зору точності є абсолютно ідентичними, кількість же викликів функції поступово зростає в залежності від кількості рестартів.

## 4.2. Умовна оптимізація

### 4.2.1. Метод штрафних функцій в залежності від розташування локального мінімуму

Нехай допустима область задається наступним рівнянням:

Тоді цільова функція з урахуванням штрафу типу квадрату зрізки виглядатиме наступним чином:

В такому випадку локальний мінімум розташований всередині допустимої області.

Оскільки досліджується метод зовнішньої точки, початкова точка має також бути розташована поза допустимою областю.

Для початкового значення . зі зменшенням значення R знайдений мінімум, в теорії, мав би наближатись до точки глобального мінімуму через зменшення впливу штрафу. Однак, з першого погляду, практично отриманий результат дає протилежну картину:

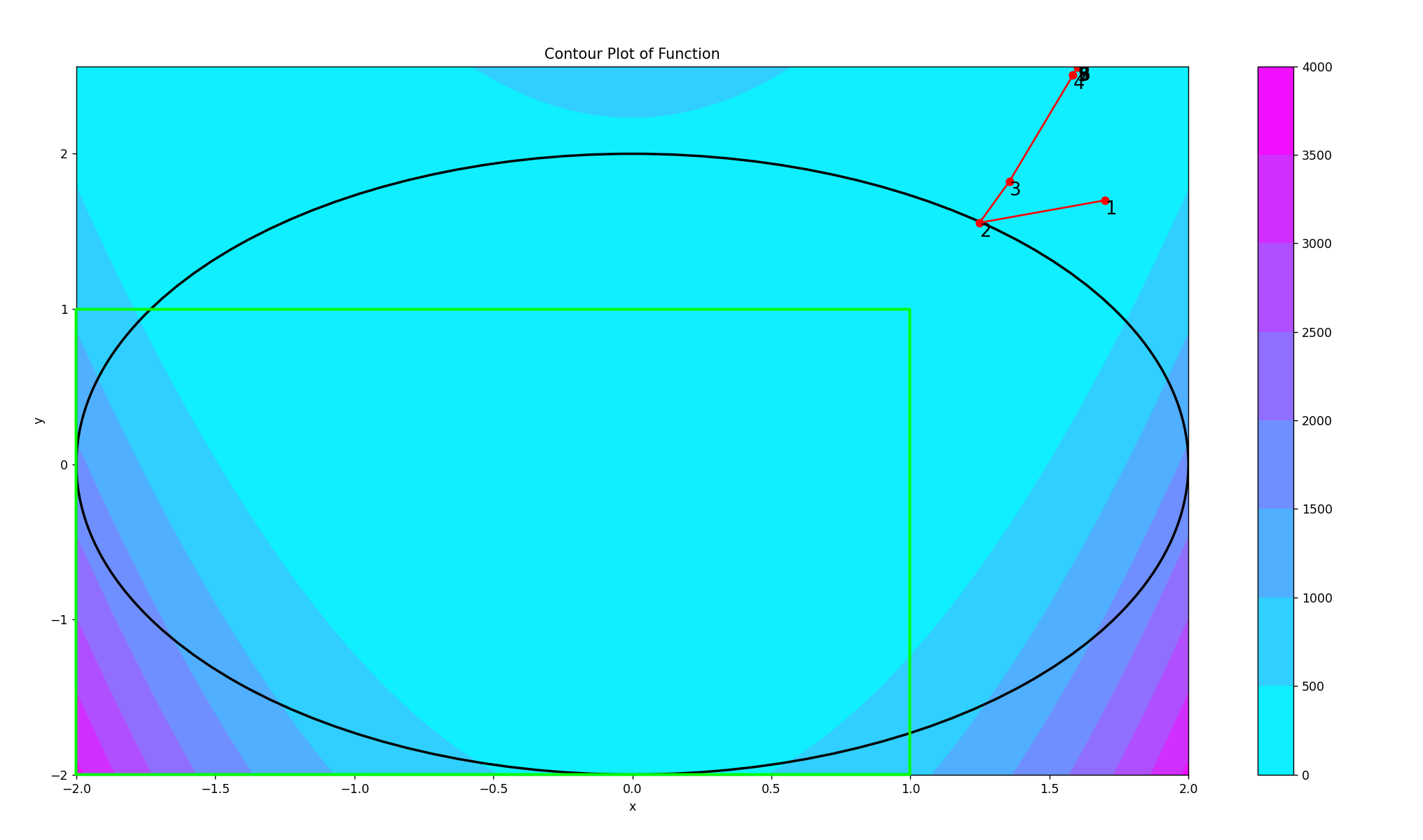


Рисунок 14 - шлях пошуку мінімуму в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 0.1 до 1e-08

Тим не менш, таку аномалію доволі легко пояснити залежністю точності другого алгоритму Пірсона від обраної початкової точки.

Якщо взяти початковою точкою, наприклад можна побачити, що дійсно значення наближатимуться до справжнього мінімуму у т. (1, 1):

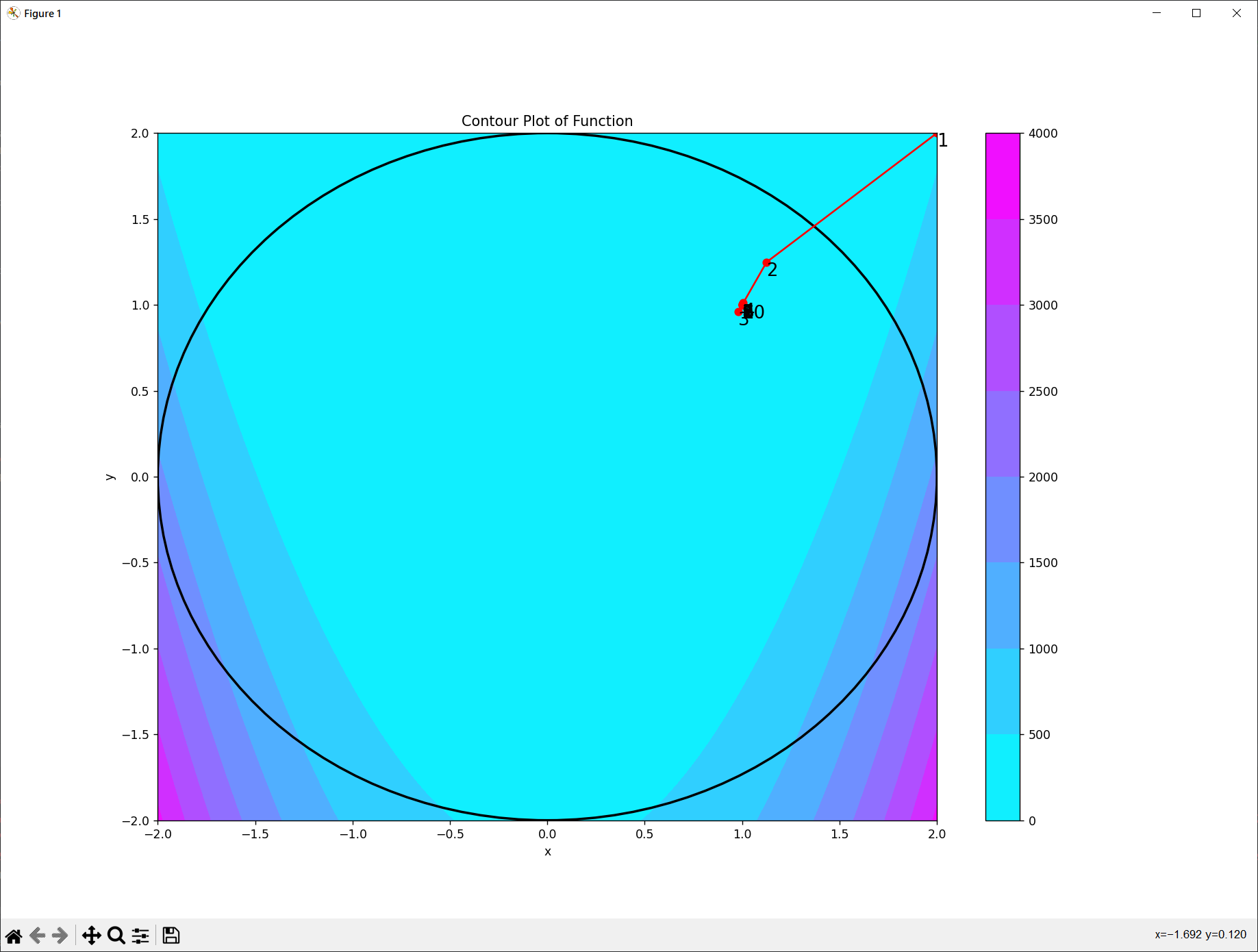


Рисунок 15 - шлях пошуку мінімуму в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 0.1 до 1e-08

Таблиця 7 - Результат тестування в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 0.1 до 1e-08

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| R | Результат | Кількість викликів функції | Відхилення | F(x) |
| 0.1 | [1.12560854 1.24879487] | 154 | 0.278704850407265 | 0.18661463991791244 |
| 0.01 | [0.98079959 0.96147746] | 117 | 0.043042322431450994 | 0.04506546622446953 |
| 0.001 | [1.00572022 1.01338692] | 174 | 0.014557829497065047 | 0.004246738902546976 |
| 0.0001 | [1.00131772 1.00328166] | 176 | 0.0035363341044153587 | 0.00043959635543656565 |
| 1e-05 | [1.00037729 1.00128166] | 176 | 0.0013360353132255533 | 6.777660046993592e-05 |
| 1e-06 | [1.00006735 1.00073435] | 176 | 0.000737429019838345 | 3.995548164718363e-05 |
| 1e-07 | [1.00018093 1.00091574] | 176 | 0.0009334447667215294 | 3.1107594849872036e-05 |
| 1e-08 | [1.00019211 1.00093298] | 176 | 0.0009506350843106926 | 3.024339270023064e-05 |

У випадку зі зростаючими значеннями R значення розташовуються вздовж границі допустимої області

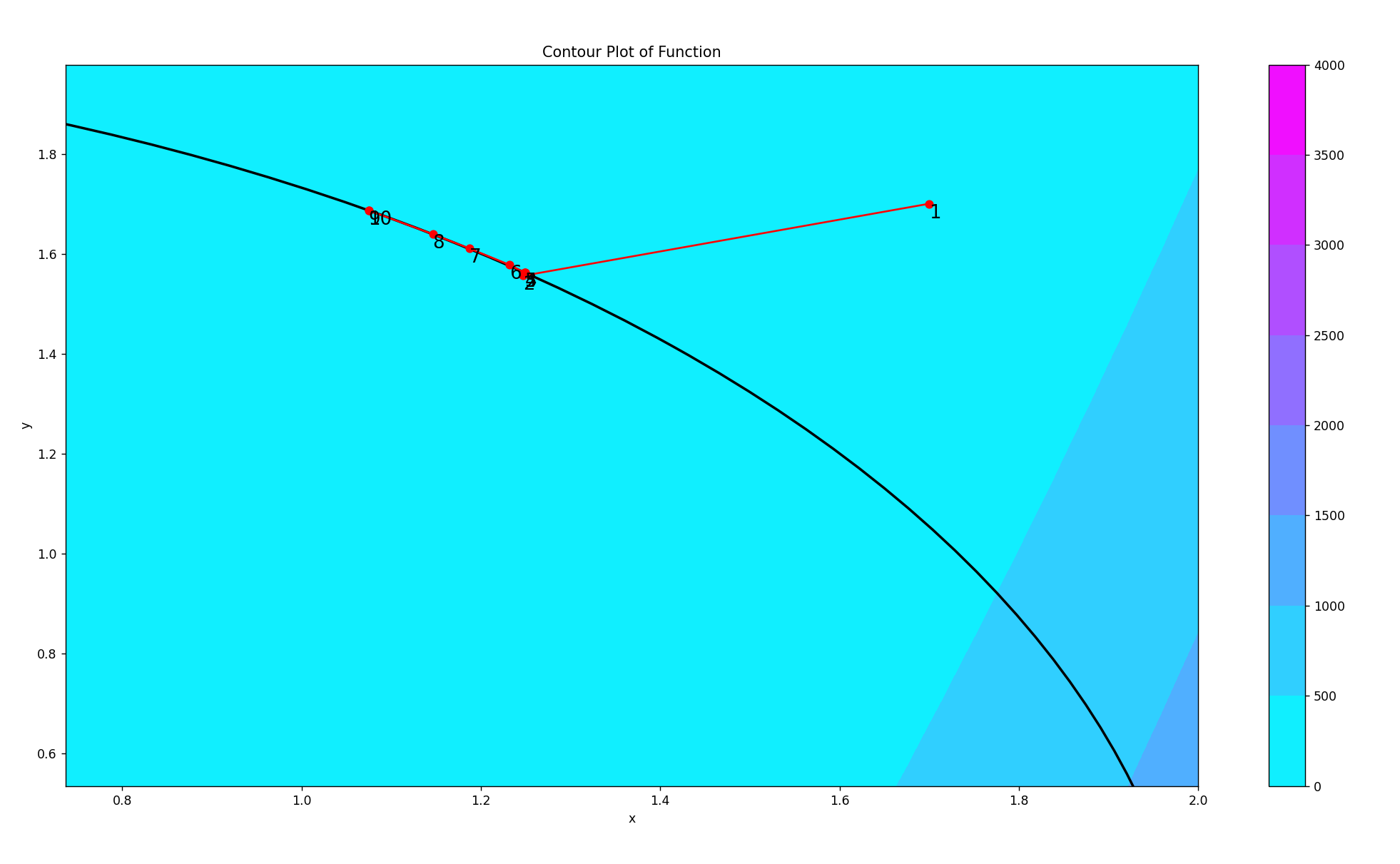


Рисунок 16 - шлях пошуку мінімуму в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 1 до 1e+08

Таблиця 8 - Результат тестування в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 1 до 1e+08

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| R | Результат | Кількість викликів функції | Відхилення | F(x) |
| 1 | [1.24727087 1.55577303] | 331 | 0.6082980669086513 | 0.061714188121624394 |
| 10 | [1.2492394 1.56153944] | 104 | 0.6143670030712335 | 0.06221861618210755 |
| 100 | [1.24893933 1.56155576] | 136 | 0.6142602547365157 | 0.06254895321115336 |
| 1000 | [1.24910295 1.561519 ] | 221 | 0.6142929836285831 | 0.06417187823885717 |
| 10000 | [1.23163431 1.57781262] | 162 | 0.6225125526435049 | 0.8360248897232399 |
| 1e+05 | [1.18752827 1.61022037] | 153 | 0.6383852689248706 | 4.954975214952758 |
| 1e+06 | [1.14641433 1.63907321] | 158 | 0.6556307854516481 | 11.255045670355134 |
| 1e+07 | [1.07477961 1.68664656] | 88 | 0.6907065114622399 | 28.306418335876668 |
| 1e+08 | [1.074792 1.68665186] | 88 | 0.6907131250249903 | 28.328508708671706 |

Підбиваючи підсумок, у випадку розташування точки локального мінімуму та початкової точки всередині допустимої області, найкращі значення було отримано за значень R близьких до 0, тобто коли вплив штрафу на цільову функцію був мінімальним. У випадку з розташуванням точки локального мінімуму всередині допустимої області, а початкової точки - поза допустимою областю, найкращий результат було отримано за значення R >= 1, однак не зважаючи від значення R усі знайдені точки розташовувались на границі допустимої області.

Нехай допустима область задається наступним рівнянням:

Тоді цільова функція з урахуванням штрафу типу квадрату зрізки виглядатиме наступним чином:

Тоді точка локального мінімуму розташована поза допустимою областю.

Візьмемо початковою точкою .

У випадку, коли значення R прямують до нуля, вплив штрафу постійно зменшується, а отже результати будуть приблизно рівні результатам вирішення задачі БО:

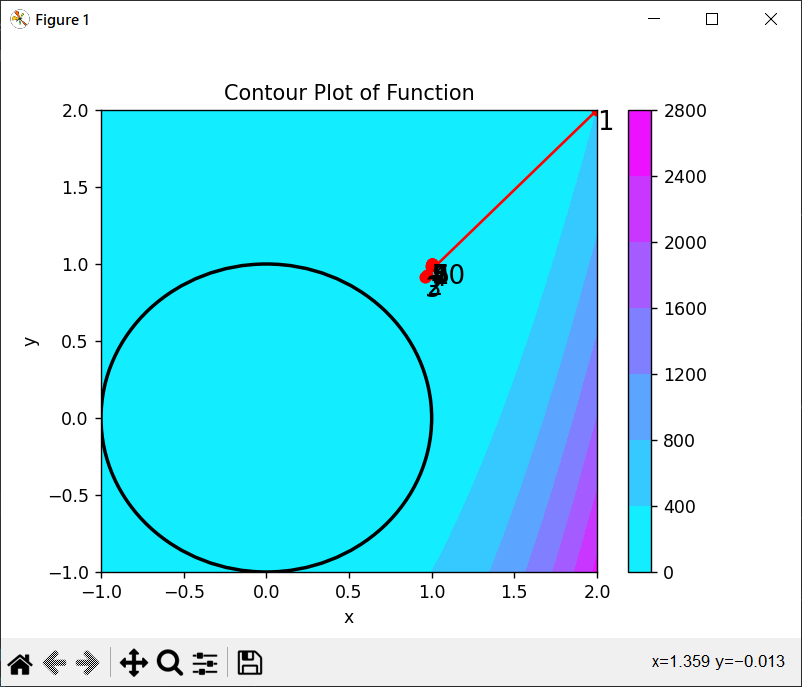


Рисунок 17 - шлях пошуку мінімуму в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 0.1 до 1e-08

За зростаючих значень R значення розташовані вздовж границі допустимої області:

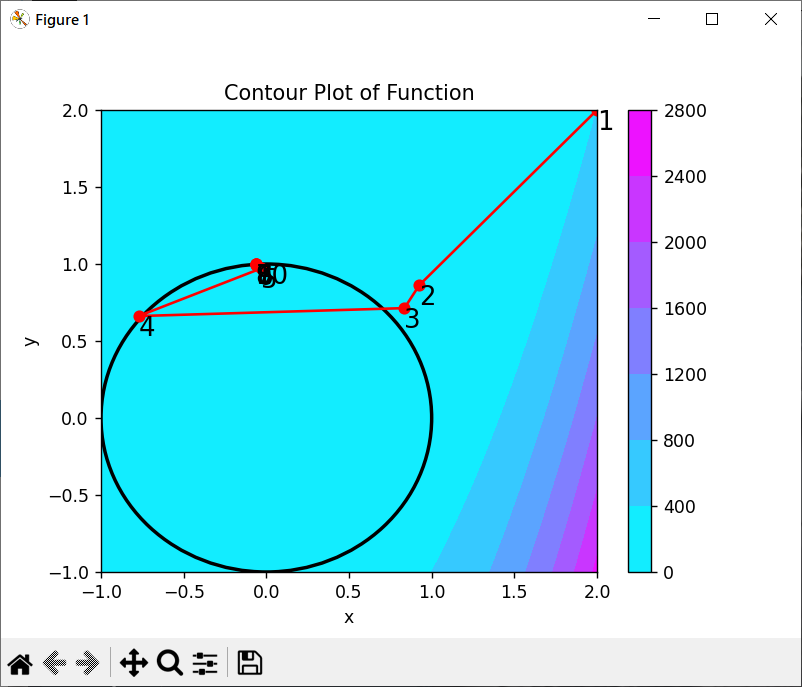


Рисунок 18 - шлях пошуку мінімуму в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 1 до 1e+08

### 4.2.2. Метод штрафних функцій в залежності від виду допустимої області (невипукла чи випукла)

Оскільки у п. 4.2.1. Метод штрафних функцій в залежності від розташування локального мінімуму фактично й досліджувались випуклі області, у цьому розділі розглядатимуться лише невипуклі допустимі області.

Візьмемо для прикладу наступне обмеження:

За такого обмеження значення мінімуму знаходиться всередині досліджуваної області. Після додавання штрафних функцій цільова функція виглядатиме наступним чином:

За значень R прямуючих до нуля вирішення задачі майже не відрізняється від вирішення задачі БО через зменшення впливу штрафу:

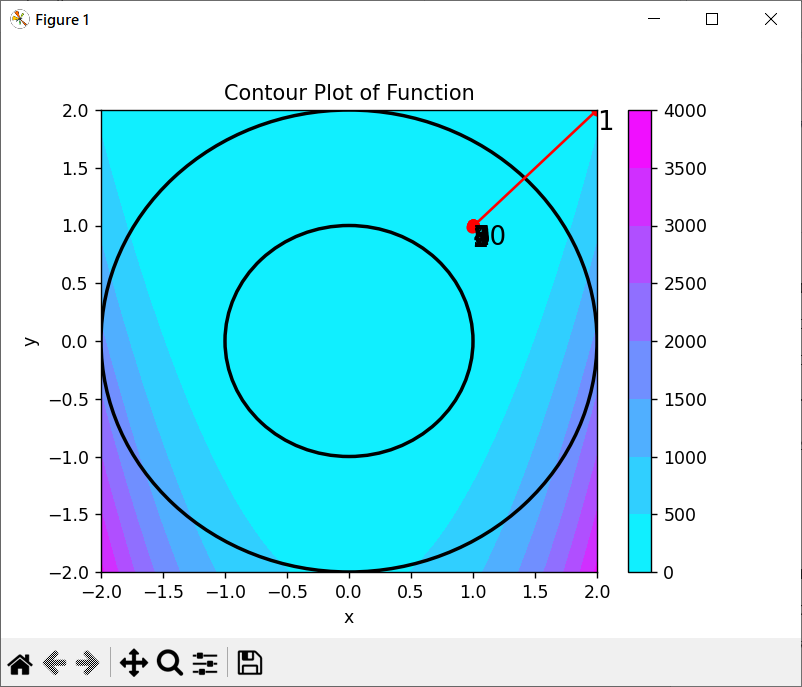


Рисунок 19 - шлях пошуку мінімуму в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 0.1 до 1e-08

За зростаючих значень R картина буде доволі схожою, адже доволі швидко шукана точка потрапляє допустиму область, а отже штраф втрачає свою вагу.

Взявши більш віддалену від мінімуму початкову точку, наприклад,

Можна побачити, що за великих значень R значення шукаються вздовж границі допустимої області, під кінець звалюючись в бік справжнього мінімуму:

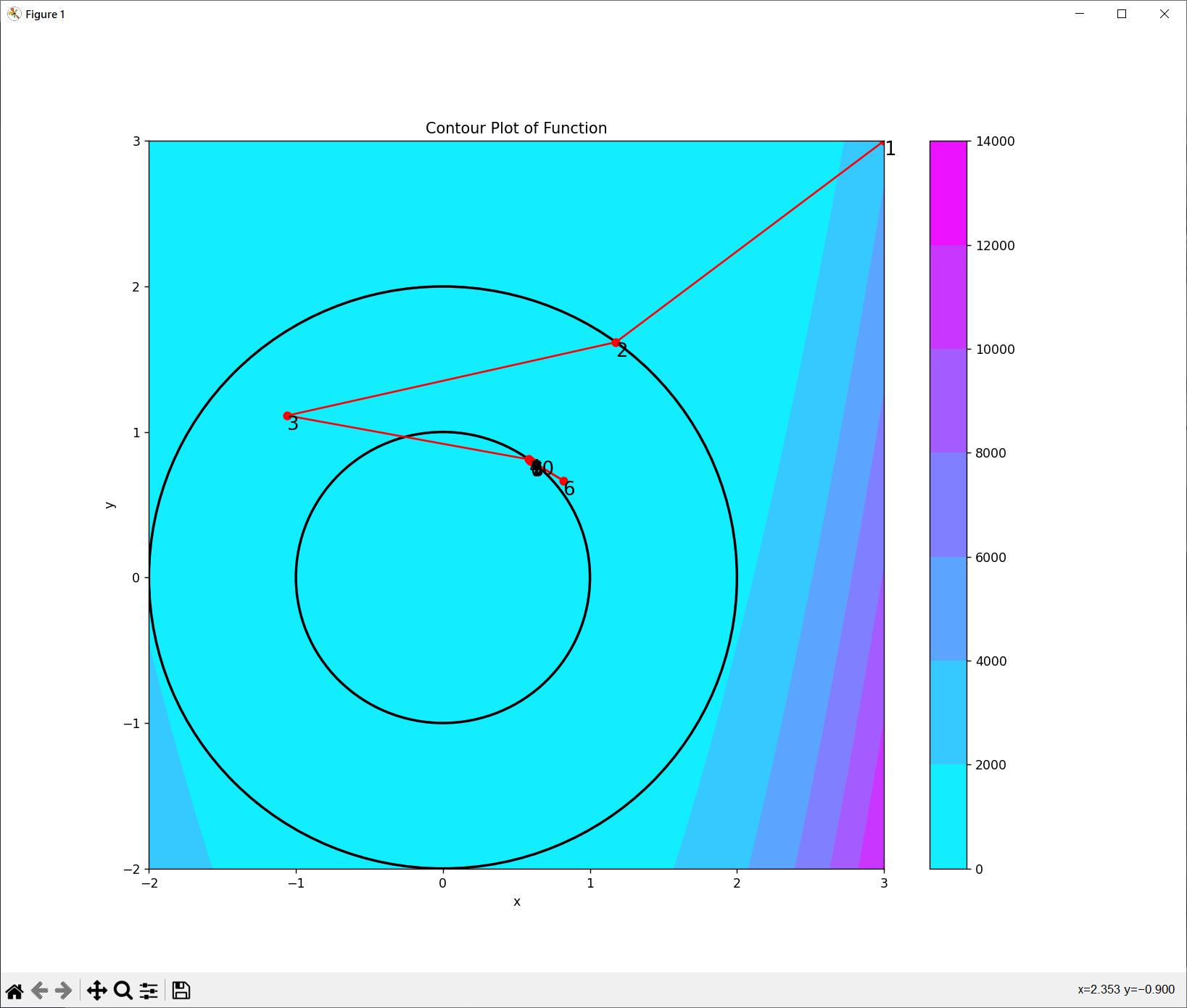


Рисунок 20 - шлях пошуку мінімуму в залежності від значення R. Початкова точка поза допустимою областю. Значення R від 1 до 1e+08

Таблиця 9

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| R | Результат | Кількість викликів функції | Відхилення | F(x) |
| 1 | [1.17669486 1.61721176] | 109 | 0.6420057826374193 | 5.441542369441039 |
| 10 | [-1.05833993 1.11283463] | 151 | 2.061430314024951 | 4.24201777087583 |
| 100 | [0.58440315 0.8116889 ] | 190 | 0.4562694511013784 | 22.277937989590388 |
| 1000 | [0.60689718 0.79478711] | 81 | 0.4434435234537494 | 18.341592557795654 |
| 10000 | [0.81710816 0.66685309] | 151 | 0.3800477432074451 | 0.033515465552758764 |
| 1e+05 | [0.59273039 0.80542429] | 98 | 0.4513626472409924 | 20.786092776864024 |
| 1e+06 | [0.59244842 0.8056137 ] | 97 | 0.4515355237064239 | 20.83390276271967 |
| 1e+07 | [0.59240617 0.80564669] | 97 | 0.4515594554810616 | 20.8414893554086 |
| 1е+08 | [0.59240855 0.80564454] | 97 | 0.45155823131547024 | 20.841035348279757 |

# 5. ВИСНОВОК

У цій курсовій роботі було проведено дослідження збіжності метода Пірсона (алгоритм 2) при мінімізації функції Розенброка. Було вивчено вплив різних факторів на швидкість збіжності та точність отриманого розв'язку.

Було досліджено параметри, які використовувались в основній формулі та  
обрано найкращі значення, звертавши увагу на кількість ітерацій та точність:

Крок обчислення похідних: 0.001

Метод обчислення похідних: центральний

Коефіцієнт алгоритму Свенна: 0.001

Точність одновимірного пошуку: 0.001

Алгоритм одновимірного пошуку: золотий перетин

Критерій закінчення: перший

Загалом, метод є дуже чутливим до початкової точки, а також вразливим до похибки обчислення матриці *А*. Однак, метод може бути використано для знаходження приблизного значення мінімуму функції, у випадку, коли будь-які аналітичні розрахунки, крім обрахунку значення функції, провести неможливо. Варто також зазначити, що не зважаючи на часом низьку точність, метод завжди забезпечував відносно невелику кількість обчислень цільової функції (приблизно 200-300 обчислень в залежності від заданих параметрів та початкової точки).

# 6. СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методи оптимізації без використання похідних: практикум з

дисципліни «Дослідження операцій»[Електронний ресурс]: навч. посіб. для

студ. спеціальності 113 «Прикладна математика», спеціалізації «Наука про дані

та математичне моделювання» / Т. С. Ладогубець, О. Д. Фіногенов; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 493 Кбайт). – Київ: КПІ ім. Ігоря Сікорського, 2020. – 45 с.

2. Химмельблау Д. Прикладное нелинейное программирование /

Химмельблау Д. — М. : Мир, 1975. — 535 с.

# 7. ДОДАТОК

## 7.1. Лістинг коду:

Код для програми, використаної для виконання курсової можна знайти за посиланням: https://github.com/Retro52/MOCourse