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. Постановка задачі 3](#_Toc136106800)

[2. Теоретична частина 4](#_Toc136106801)

[3. Хід роботи 5](#_Toc136106802)

[3.1. Безумовна оптимізація 5](#_Toc136106803)

[3.1.1. Визначення оптимального кроку обчислення похідних 6](#_Toc136106804)

[3.1.2. Визначення оптимальної схеми обчислення похідних 8](#_Toc136106805)

[3.1.3. Визначення оптимального значення параметру в алгоритмі Свенна 10](#_Toc136106806)

[3.1.4. Визначення оптимальної точності методу одновимірного пошуку 12](#_Toc136106807)

[3.1.5. Визначення оптимального методу одновимірного пошуку 14](#_Toc136106808)

[3.1.6. Визначення оптимального критерію закінчення 16](#_Toc136106809)

[3.1.7. Визначення необхідності наявності рестартів 18](#_Toc136106810)

[3.2. Умовна оптимізація 19](#_Toc136106811)

# 1. 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).

# 2. 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:

# 3. Хід роботи

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

## 3.1. 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

### 3.1.1. 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:

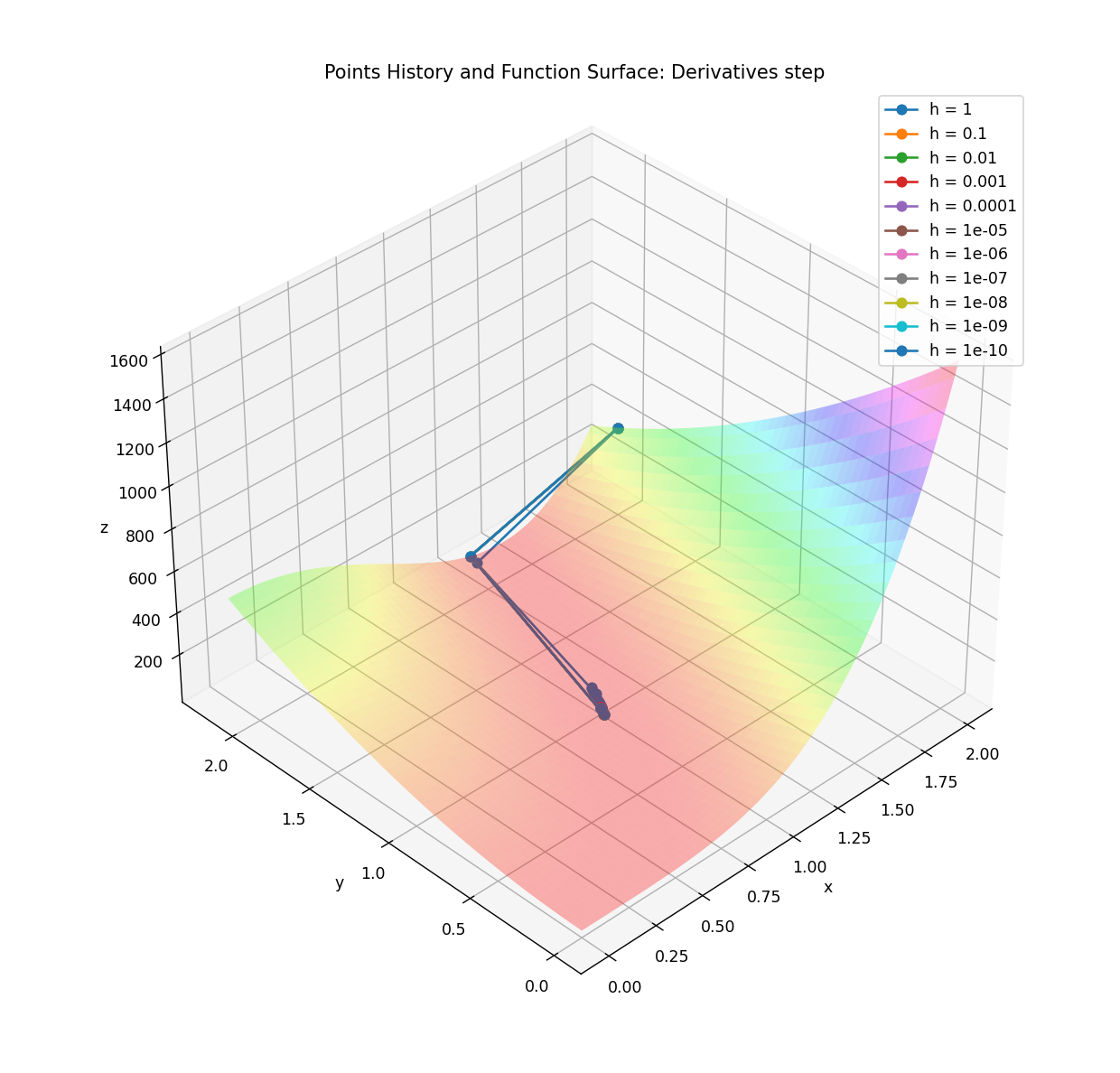


Рисунок 1 – шлях пошуку мінімуму в залежності від кроку похідних

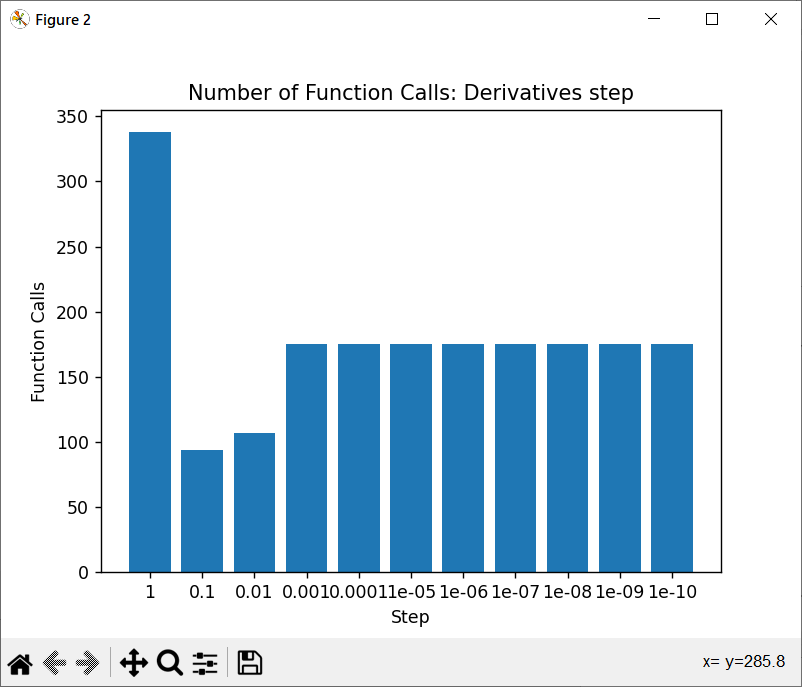
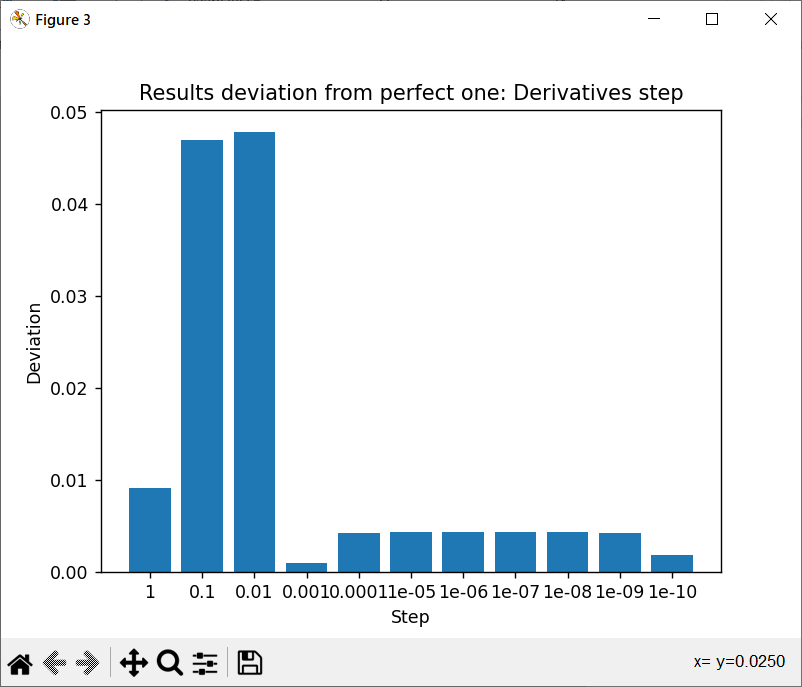
 

Рисунок 2 – кількість викликів функції в залежності від кроку похідних, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 1 – Результат тестування кроку обчислення похідних

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

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.

### 3.1.2. Визначення оптимальної схеми обчислення похідних

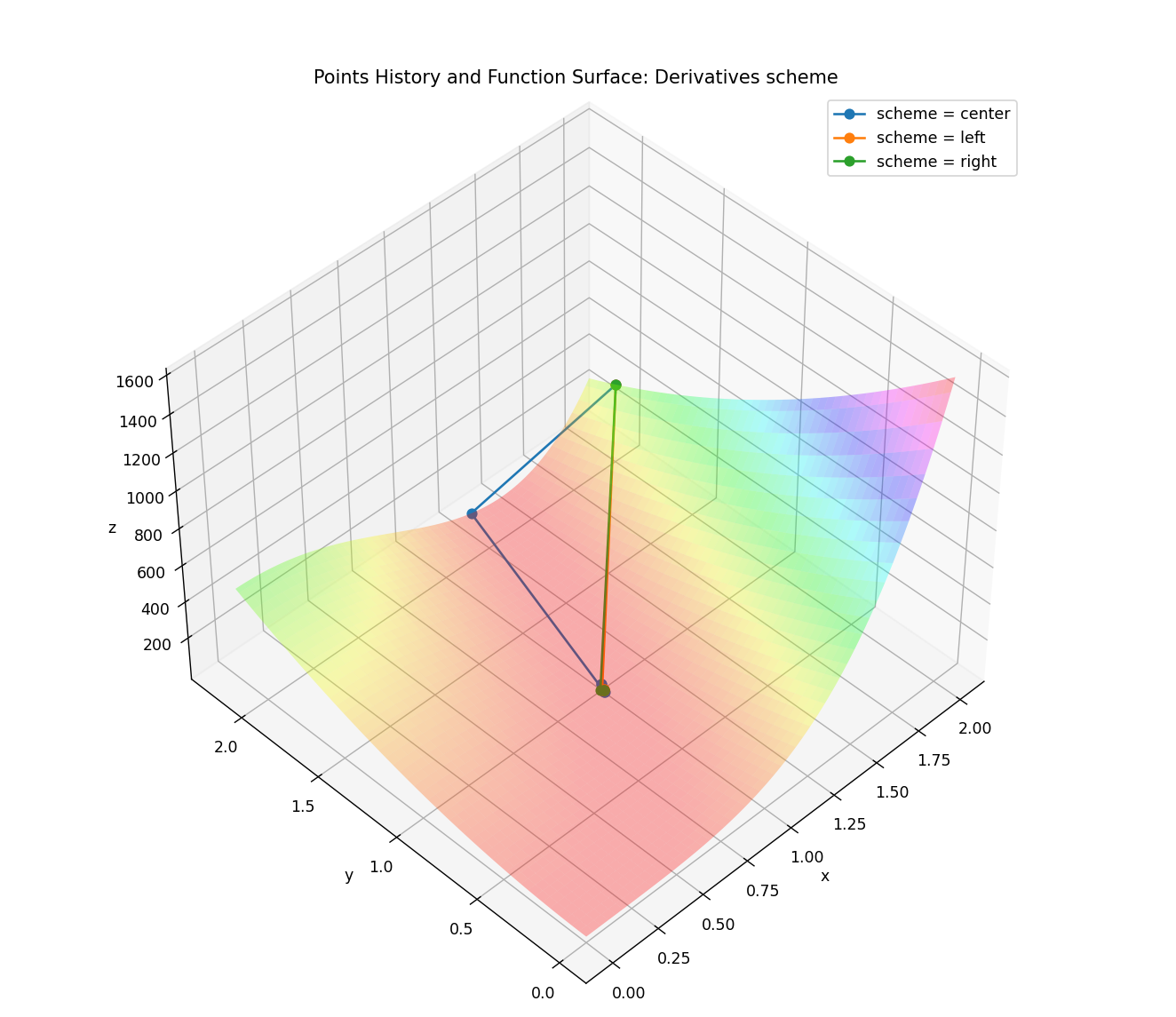


Рисунок 3 - шлях пошуку мінімуму в залежності від схеми обчислення похідних

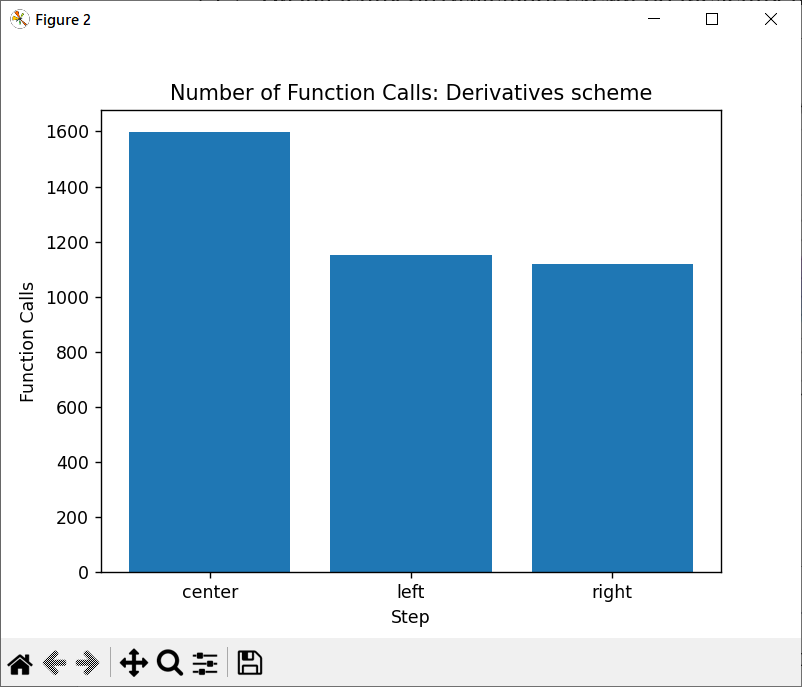
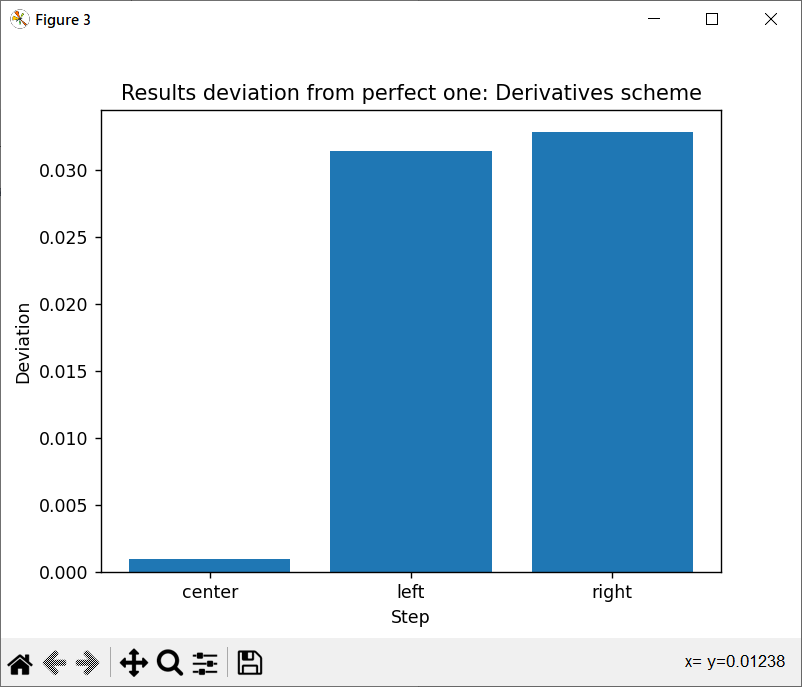
 

Рисунок 4 - кількість викликів функції в залежності від схеми обчислення похідних, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 2 - Результат тестування різних схем обчислення похідних

|  |  |  |  |
| --- | --- | --- | --- |
| Схема | Результат | Кількість викликів функції | Відхилення |
| Центральна | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 |
| Ліва | [0.98585819, 0.97200442] | 108 | 0.031364676232352076 |
| Права | [0.98518953, 0.9707299 ] | 75 | 0.03280379415667225 |

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.

### 3.1.3. 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:

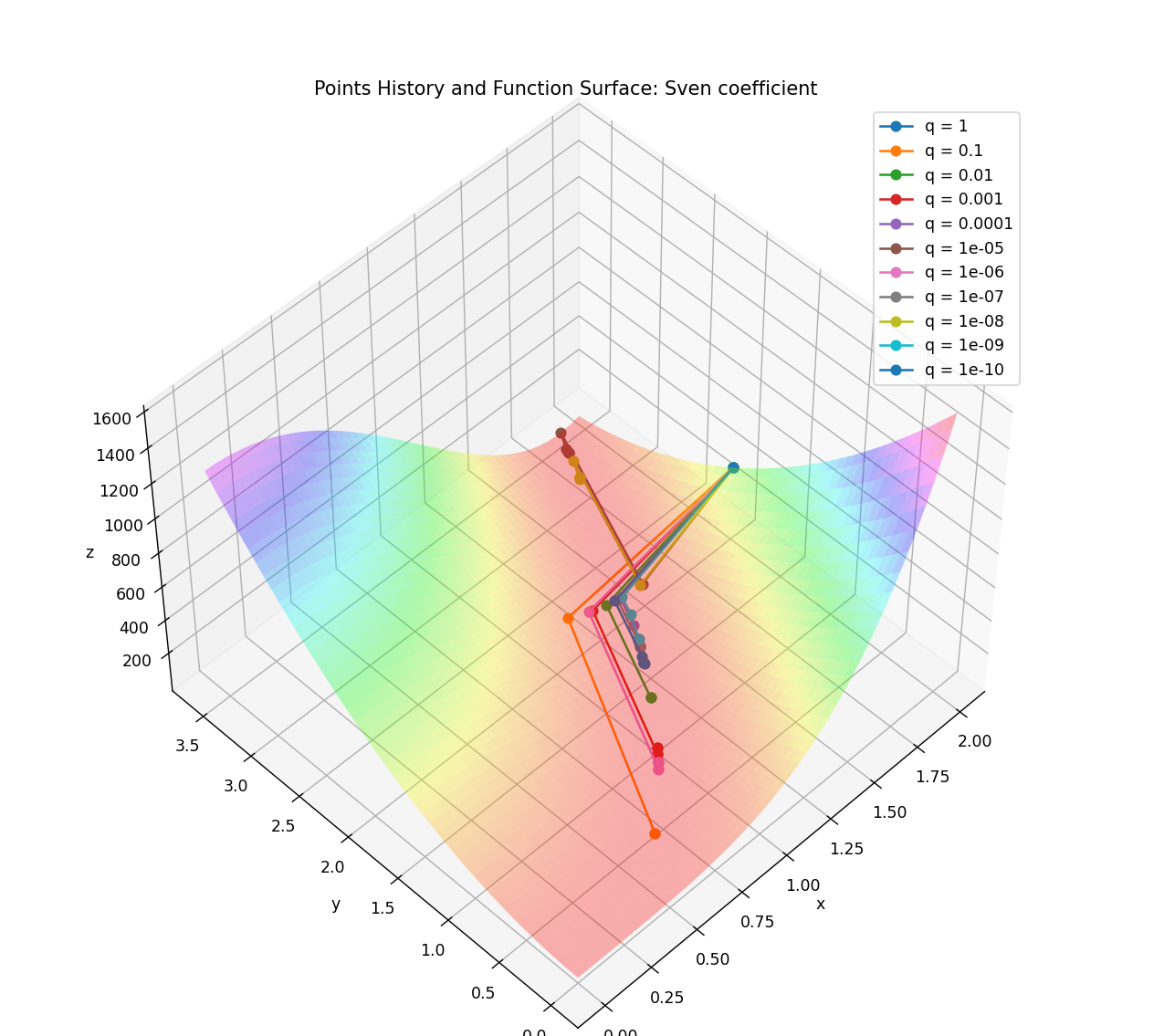


Рисунок 5 - шлях пошуку мінімуму в залежності від значення в алгоритмі Свенна

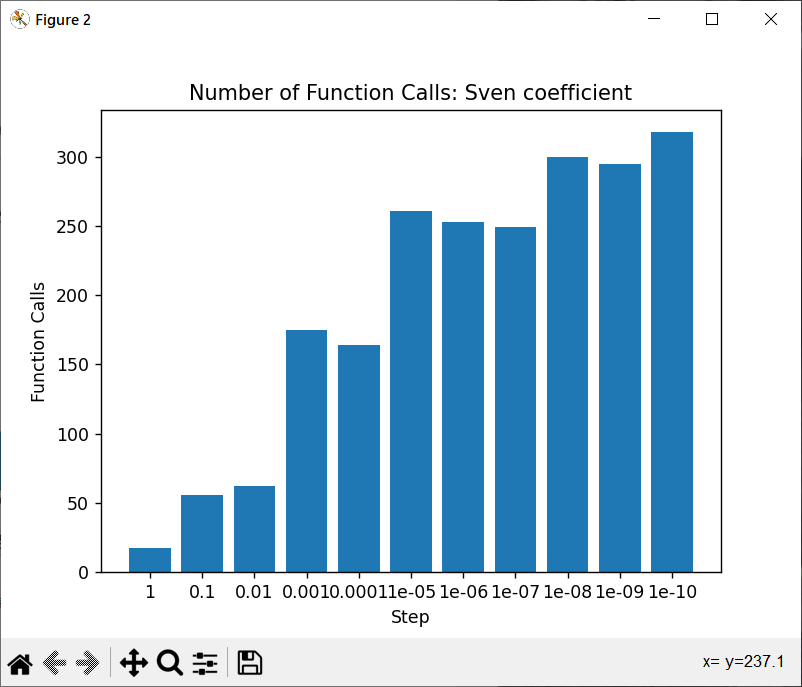
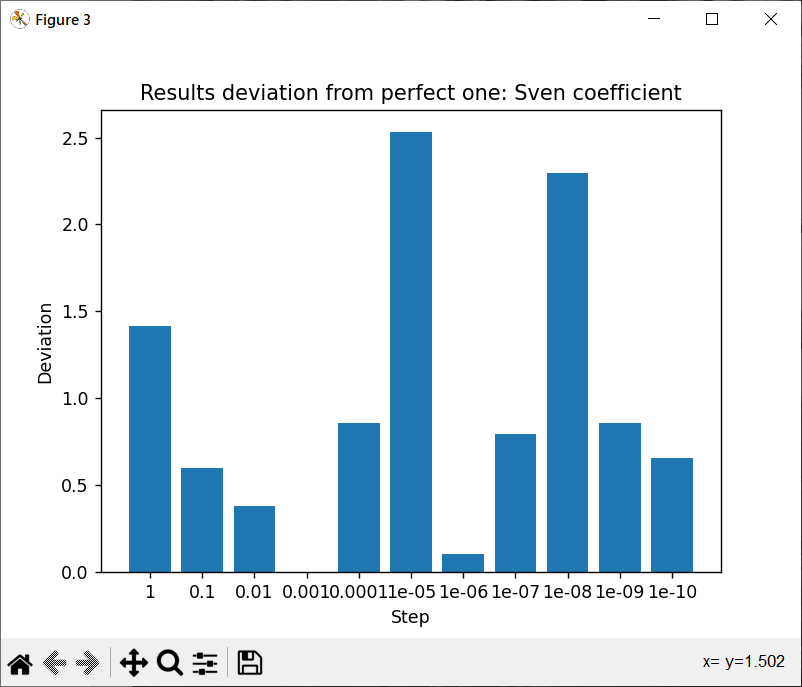
 

Рисунок 6 - кількість викликів функції в залежності від значення в алгоритмі Свенна, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 3 - Результат тестування різних коефіцієнтів в алгоритмі Свенна

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

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

### 3.1.4. 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]

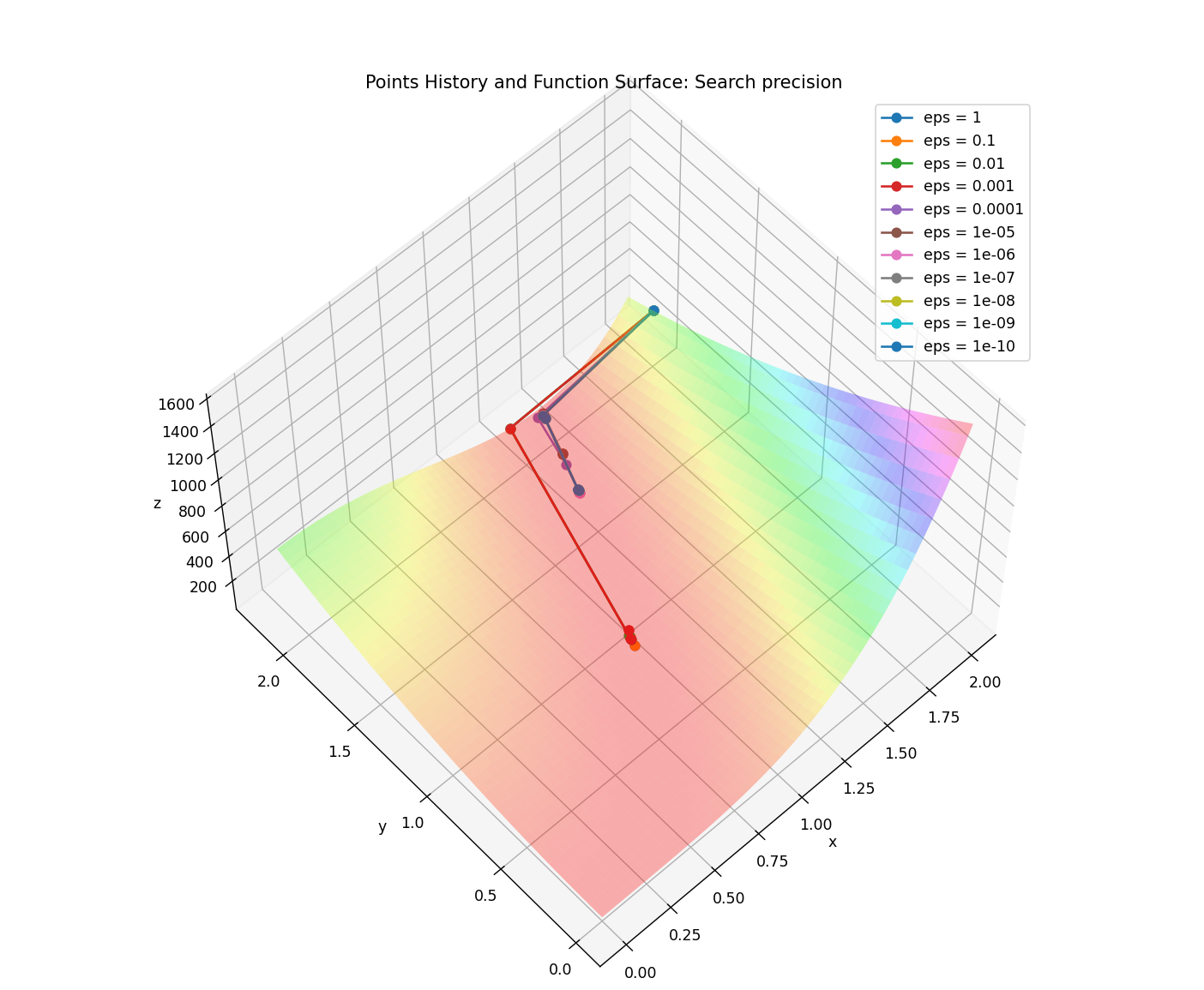


Рисунок 7 - шлях пошуку мінімуму в залежності від точності одновимірного пошуку

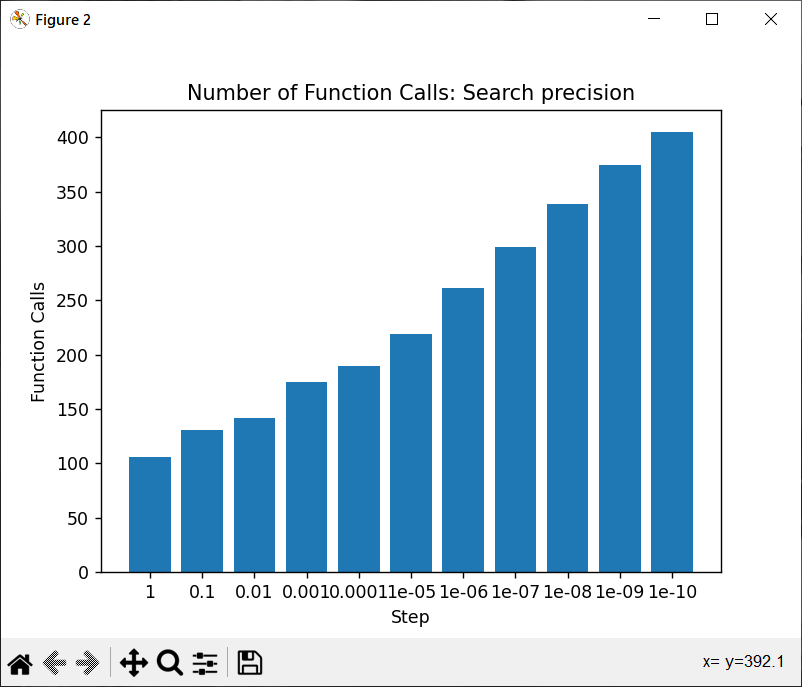
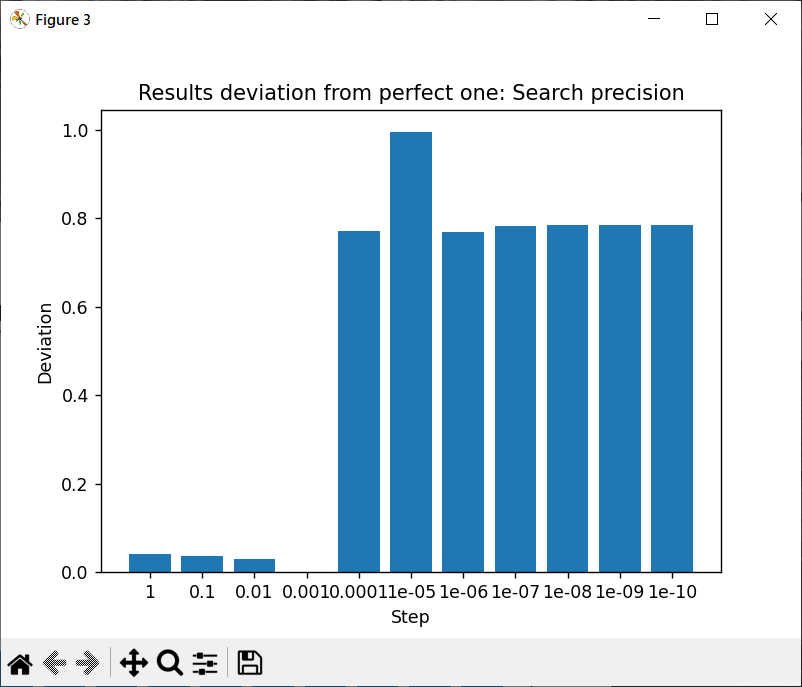
 

Рисунок 8 - кількість викликів функції в залежності від точності одновимірного пошуку, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 4 - Результат тестування різних точностей одновимірного пошуку

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

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

### 3.1.5. Визначення оптимального методу одновимірного пошуку

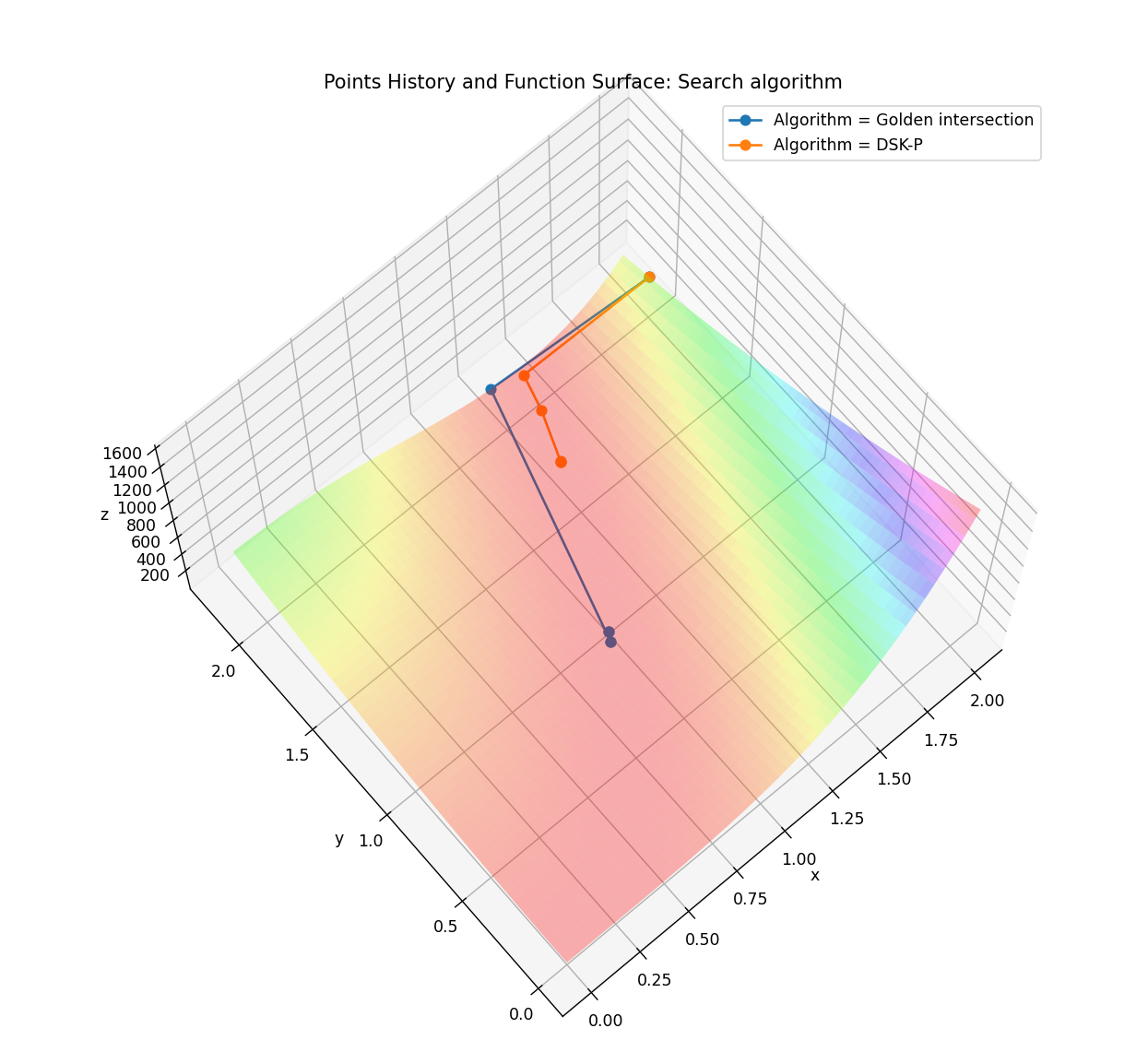


Рисунок 9 - шлях пошуку мінімуму в залежності від обраного методу одновимірного пошуку

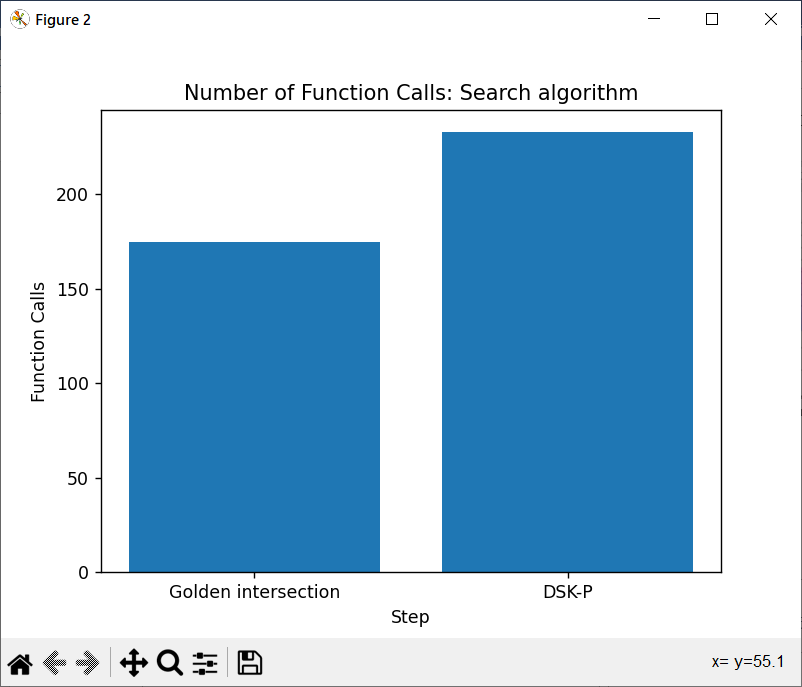
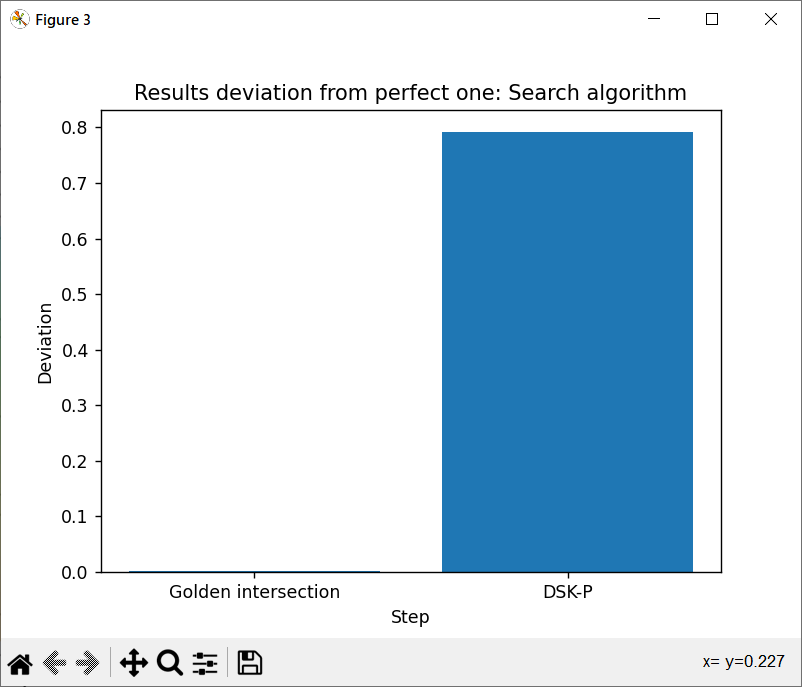
 

Рисунок 10 - кількість викликів функції в залежності від обраного методу одновимірного пошуку, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 5 - Результат тестування різних варіантів одновимірного пошуку

|  |  |  |  |
| --- | --- | --- | --- |
| Метод пошуку | Результат | Кількість викликів функції | Відхилення |
| Золотий перетин | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 |
| ДСК-П | [1.31871172, 1.72480984] | 233 | 0.7917868768208998 |

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.

### 3.1.6. Визначення оптимального критерію закінчення

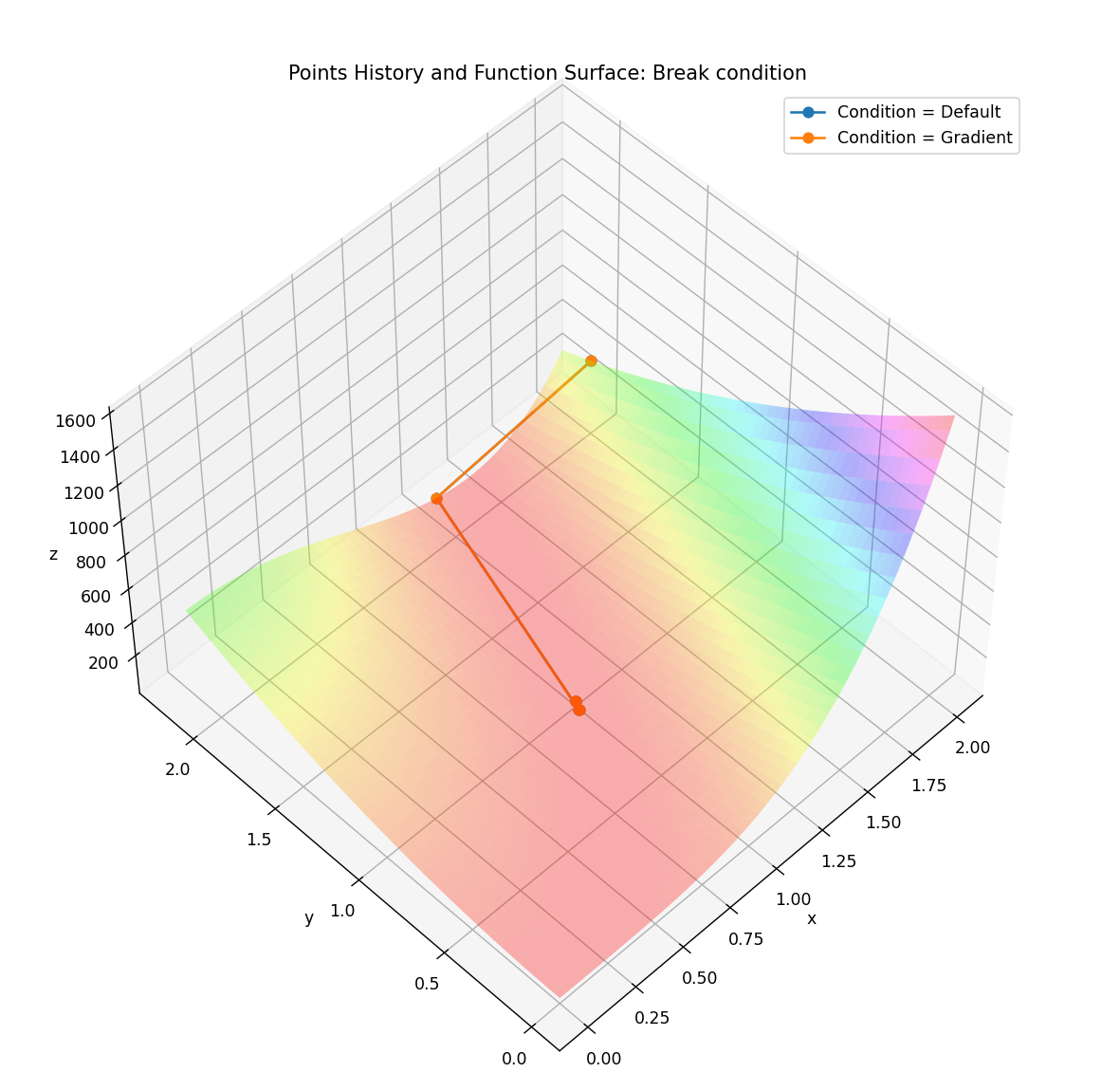


Рисунок 11 - шлях пошуку мінімуму в залежності від обраного критерію закінчення

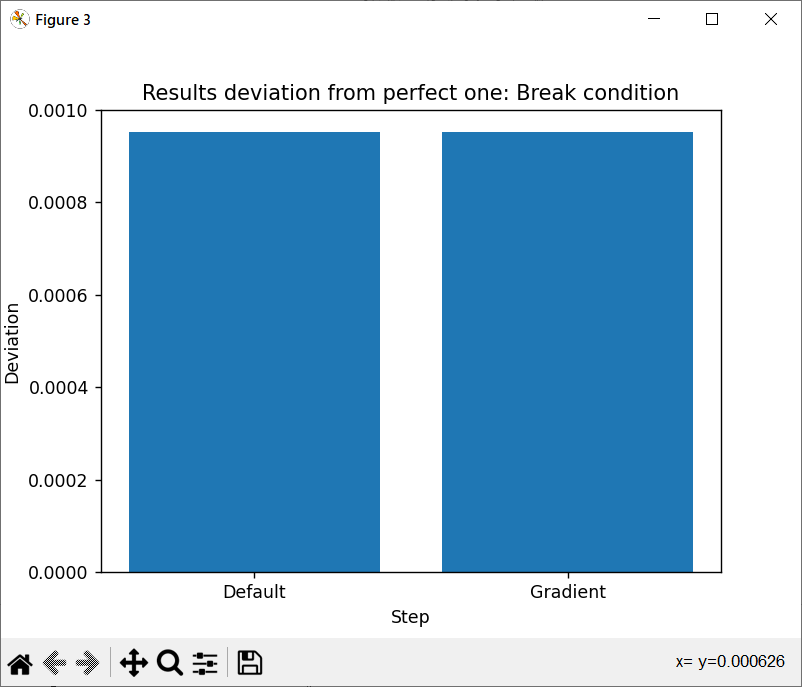
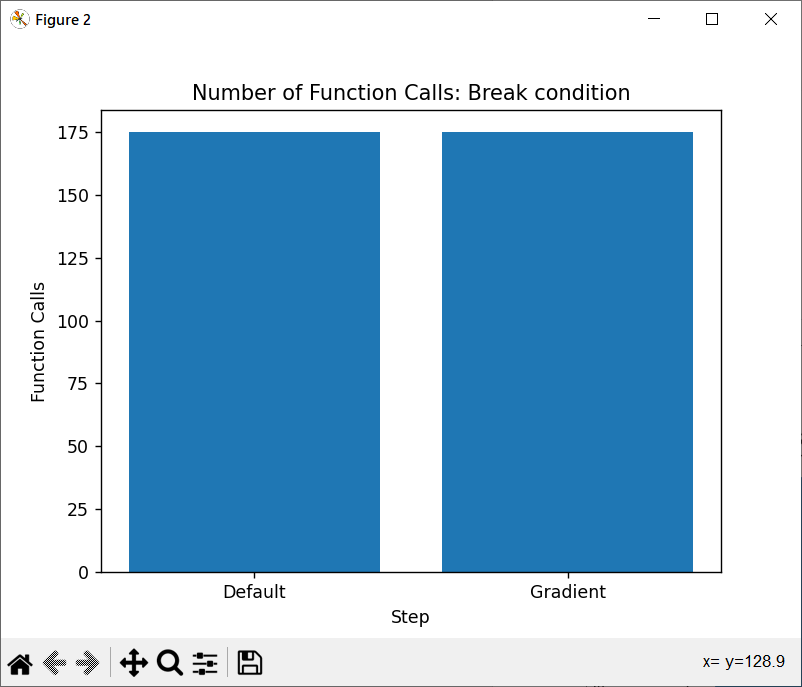


Рисунок 12 - кількість викликів функції в залежності від обраного критерію закінчення, та відхилення від ідеального розв’язку

Таблиця результатів:

Таблиця 5 - Результат тестування різних варіантів одновимірного пошуку

|  |  |  |  |
| --- | --- | --- | --- |
| Критерій | Результат | Кількість викликів функції | Відхилення |
| Звичайний | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 |
| Градієнт | [1.00019211, 1.00093298] | 175 | 0.0009525556292209975 |

Результати є абсолютно однаковими

### 3.1.7. 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.

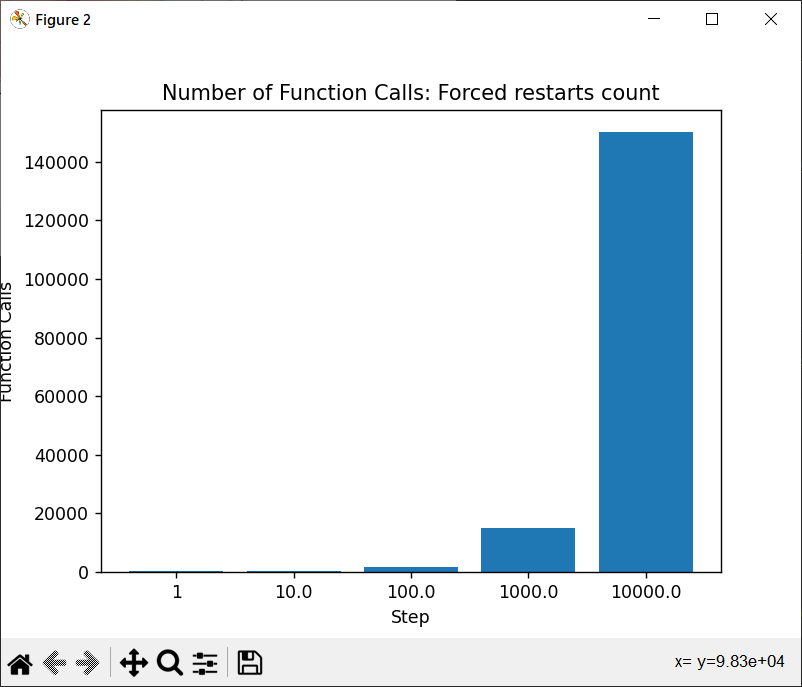
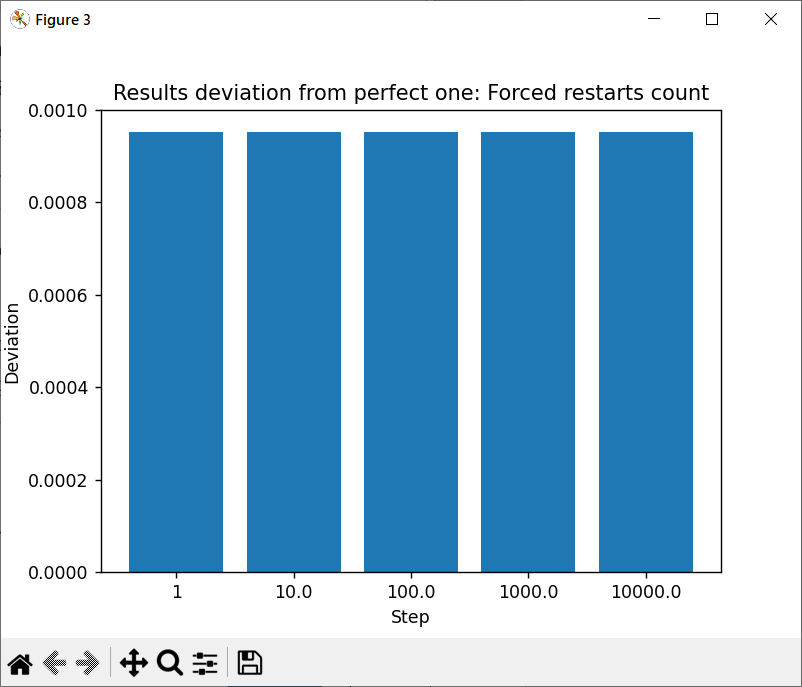


Рисунок 13 - кількість викликів функції в залежності від кількості рестартів, та відхилення від ідеального розв’язку

Результати є абсолютно ідентичними

## 3.2. Умовна оптимізація