**Национальный исследовательский технологический университет «МИСиС»**

**Институт Информационных технологий и компьютерных наук (ИТКН)**

**Курс «Методы оптимизации»**

Лабораторная работа № 1

по теме

«Численные методы одномерной минимизации»

Вариант №23

Выполнил:

Студент группы БИВТ-20-1

Смирнов А.А.

Проверил:

к.ф.-м.н., доцент, Лычев А.В.

Москва, 2023

Цель: приобретение практических навыков для решения задач одномерной минимизации численными методами.

# Ход работы:

## Формулировка

Вариант задания – №23.

Рассматриваемая функция –

Требуется найти безусловный минимум функции одной переменной y = f(x) на отрезке [a, b], где функция является унимодальной. То есть найти такую точку .

1. Графическое представление функции

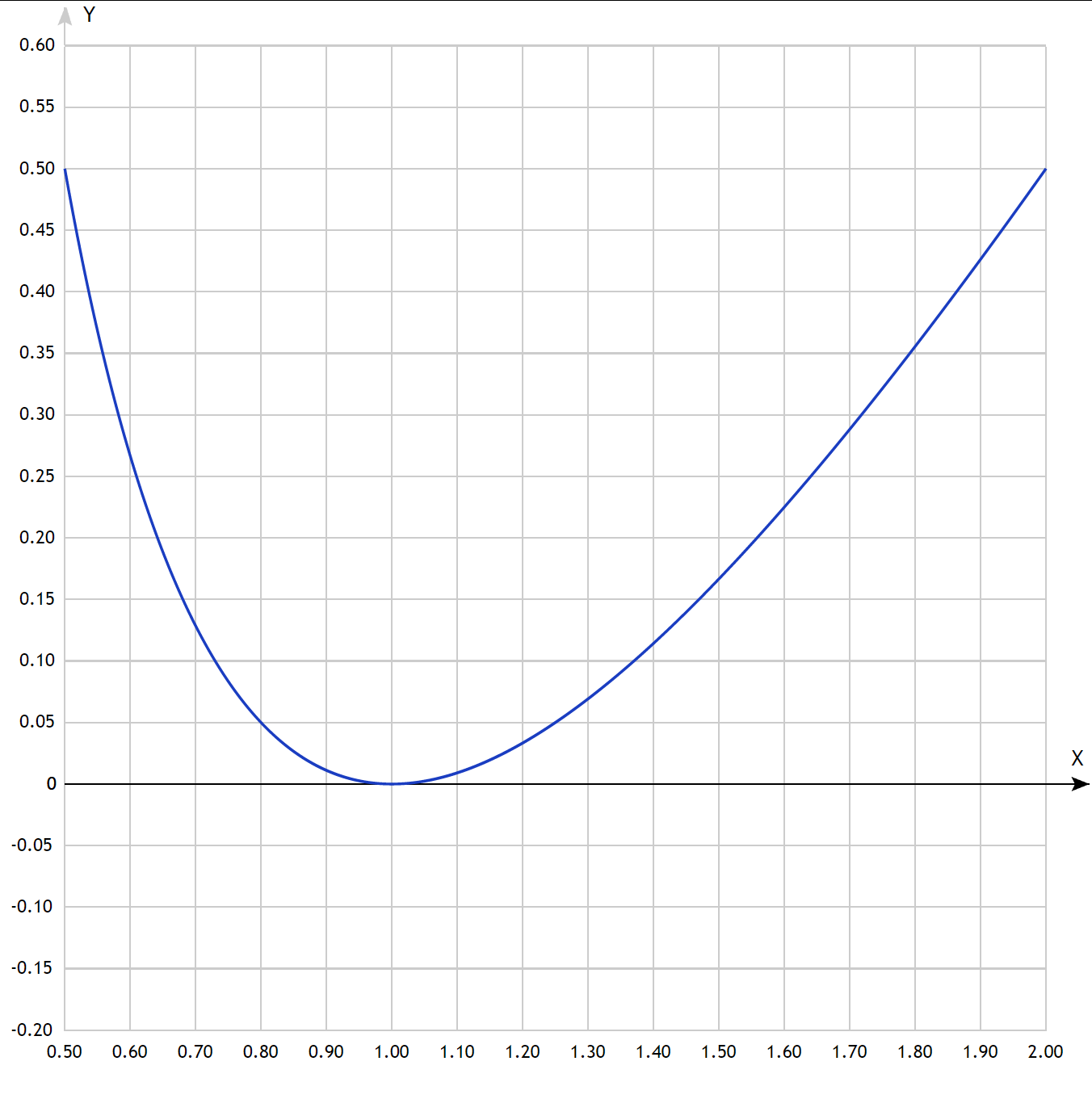


Рисунок 2.1 – График рассматриваемой функции на заданном отрезке.

## Поиск координат минимума аналитическим путем

– не входит в рассматриваемый отрезок

В итоге получаем:

– точка экстремума (по рисунку 2.1 видно, что это минимум).

– координата минимума.

## Листинг программ

Для выполнения данной лабораторной работы я реализовал несколько функций.

### 4.1 Функция для поиска значения исследуемой функции

def f(x):

return (x-1)\*\*2/x

### 4.2 Функция реализующая метод равномерного поиска

def uniform\_search\_method(a, b, epsilon):

iter = 0

x = a

while (x < b - epsilon):

if (f(x) < f(x + epsilon)):

return [(round(x, 7), round(f(x), 7)), iter]

x += epsilon

iter += 1

### 4.3 Функция реализующая метод дихотомии

def dichotomy\_mehod(a, b, epsilon, iter = 0):

x = (a + b) / 2

if (f(x - epsilon) < f(x + epsilon)):

b = x

else:

a = x

if(abs(b - a) >= 2 \* epsilon):

return dichotomy\_mehod(a, b, epsilon, iter + 1)

return [(round(x, 7), round(f(x), 7)), iter]

### 4.4 Функция реализующая метод Фибоначчи

def fibonacci\_sum(n):

return fib\_iter(1, 0, n)

def fib\_iter(a, b, count):

if (count == 0):

return b

return fib\_iter(a + b, a, count - 1)

def find\_fibonacci\_number(N):

iter = 1

while(fibonacci\_sum(iter) < N):

iter += 1

return iter

def fibonacci\_method(a, b, epsilon):

iter = 0

S = find\_fibonacci\_number((b - a) / epsilon)

K = 1

Fs = fibonacci\_sum(S)

l = 1/Fs \* (b-a)

x1 = a + l \* fibonacci\_sum(S - 1 - K)

x2 = b - l \* fibonacci\_sum(S - 1 - K)

A = f(x1)

B = f(x2)

while (True):

if (A < B):

b = x2

else:

a = x1

K += 1

if (K == S - 1):

break

iter += 1

if (A < B):

x2 = x1

B = A

x1 = a + l \* fibonacci\_sum(S - 1 - K)

A = f(x1)

else:

x1 = x2

A = B

x2 = b - l \* fibonacci\_sum(S - 1 - K)

B = f(x2)

x2 = x1 + epsilon

B = f(x2)

if (A < B):

b = x1

else:

a = x1

x = (a + b)/2

return [(round(x, 7), round(f(x), 7)), iter]

## Результаты вычислений

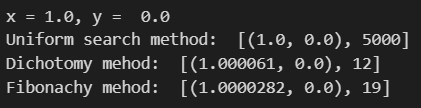


Рисунок 5.1 – Результаты вычислений минимума разными способами.

## Сравнение методов

Таблица 6.1 – Сравнительная характеристика методов.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название метода | Число итераций | Количество вычислений функций | Найденное решение | Значение функции |
| Метода равномерного поиска | 5000 | 5001 | 1.0000 | 0. 0000 |
| Метод дихотомии | 12 | 24 | 1.0000 | 0. 0000 |
| Метод квадратичной интерполяции | 19 | 21 | 1.0000 | 0. 0000 |

По данной таблице видно, что при вычислении минимума методом дихотомии число итераций минимально. Однако функция вычисляется дважды за итерацию. Метод Фибоначчи имеет немного больше итераций, но вычисление функции происходит 1 раз за итерацию. Оба этих метода показали близкую точность.

Метод равномерного поиска оказался худшим по числу итераций и количеству вычислений функции.

Вывод: в результате выполнение лабораторной работы я приобрел практические навыки для решения задач одномерной минимизации численными методами. Также для заданной функции нашел минимум сначала аналитическим способом, а затем 3-мя численными методами: метод равномерного поиска, метод дихотомии, метод Фибоначчи – и сравнил данные методы.