

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский технологический университет «МИСиС»

КАФЕДРА

Автоматизированных систем управления

НАПРАВЛЕНИЕ

09.03.01 «Информатика и вычислительная техника»

ПРОФИЛЬ

Интеллектуальные системы обработки информации и управления

ЛАБОРАТОРНАЯ РАБОТА

по дисциплине

Методы оптимизации

на тему: «Численные методы одномерной минимизации с использованием
производной»

Студент(ка) ак. группы БИВТ-18-1

аббревиатура

подпись

Гузев В. Н.
И.О. Фамилия

Оценка с учетом защиты

оценка

дата

Преподаватель

подпись

И.О. Фамилия

Москва 2021

Цель работы

Приобретение практических навыков для решения задач одномерной минимизации численными методами.

Формулировка задачи

Требуется найти безусловный минимум функции одной переменной $y = f(x)$ на отрезке $[a, b]$, где функция является выпуклой. То есть найти такую точку $x^* \in [a, b]$, что $x^* = \min_{x \in [a, b]} f(x)$.

Методы, рассмотренные в лабораторной работе 1, используются при минимальных требованиях к целевой функции $y = f(x)$ – она должна быть унимодальной.

В данной работе предполагается, что целевая функция $y = f(x)$ является выпуклой и дифференцируемой (один раз или дважды). Причём, производные могут быть вычислены в произвольно выбранных точках.

№	Функция	Тип экстремума	Интервал	Погрешность	Методы
7	$f(x) = x + \frac{1}{\ln(x)}$	min	[1.5; 3]	0.001	1, 2

Графическое представление функции на заданном интервале

Функция на интервале $[1.5;3]$ представлена следующим образом:

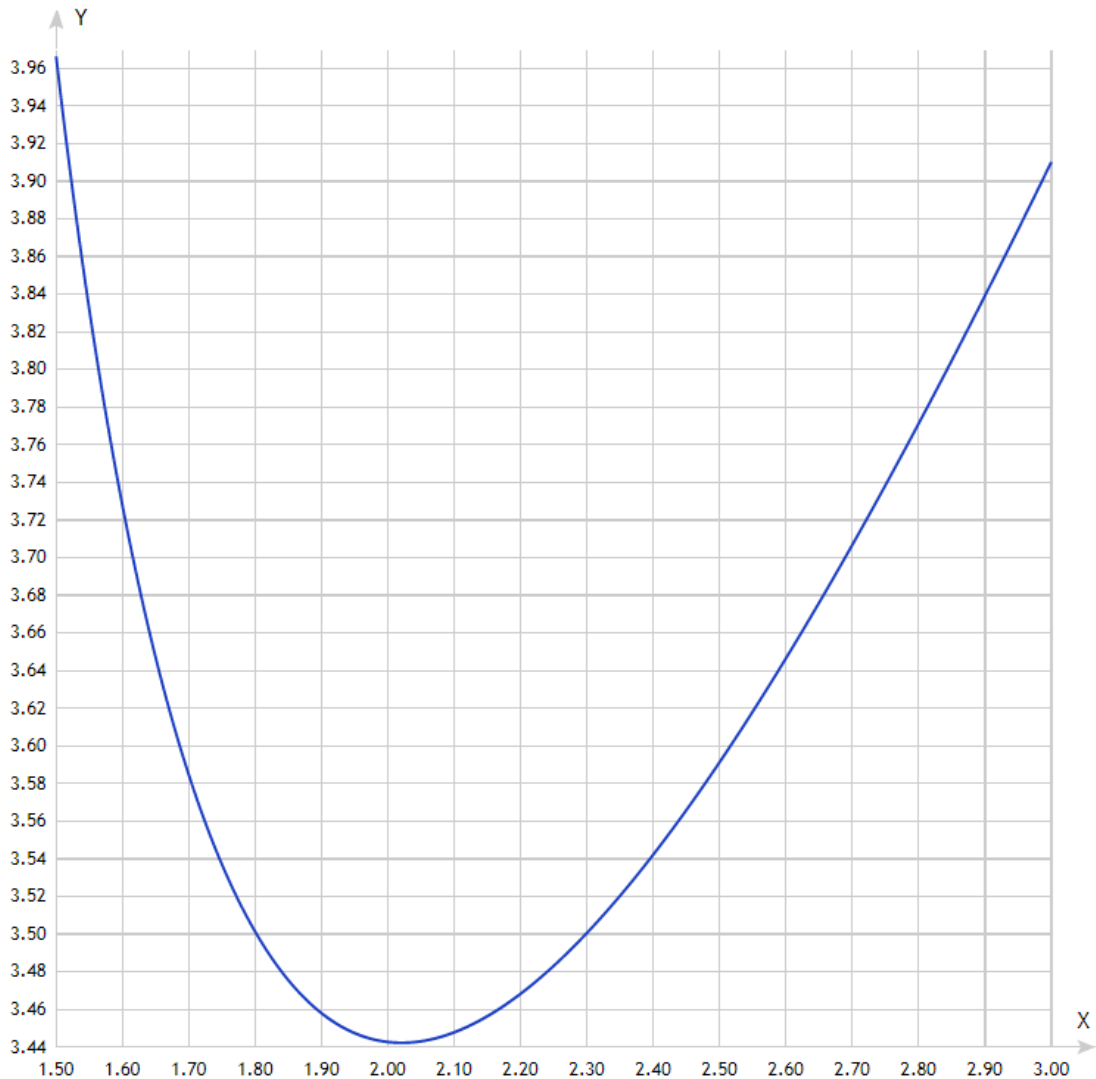


Рисунок 1 – Графическое представление функции

Согласно сервису WolframAlpha, минимум функции достигается в точке:

$$\min\left\{x + \frac{1}{\log(x)} \mid 1.5 \leq x \leq 3\right\} \approx 3.44228 \text{ at } x \approx 2.02075$$

Листинги программ

В листинге 1 представлена программа на Python для метода средней точки.

```
import math
k = 0
def func(x):
    global k
    k += 1
    return x + 1 / math.log(x)
def first_derivative(x):
    global k
    k += 1
    return 1 - 1 / (x * (math.log(x) ** 2))
left = 1.5
right = 3
epsilon = 0.001
x = (left + right) / 2
dy = first_derivative(x)
while abs(dy) > epsilon:
    if dy < 0:
        left = x
    else:
        right = x
    x = (left + right) / 2
    dy = first_derivative(x)
print("x =", x, "за", k, "вызовов функций")
```

Листинг 1 - Алгоритм средней точки

В листинге 2 представлена программа на Python для метода хорд.

```
import math
k = 0
def func(x):
    global k
    k += 1
    return x + 1 / math.log(x)
def first_derivative(x):
    global k
    k += 1
    return 1 - 1 / (x * (math.log(x) ** 2))
left = 1.5
right = 3
epsilon = 0.001
d_left = first_derivative(left)
d_right = first_derivative(right)
```

```

continue_search = True
if d_left * d_right >= 0:
    continue_search = False
while continue_search:
    x = left - d_left / (d_left - d_right) * (left - right)
    d_x = first_derivative(x)
    if abs(d_x) <= epsilon:
        break
    if abs(d_x) > epsilon:
        if d_x > 0:
            right = x
            d_right = first_derivative(right)
        if d_x <= 0:
            left = x
            d_left = first_derivative(left)
result = 0
if d_left > 0 and d_right > 0:
    result = left
elif d_left < 0 and d_right < 0:
    result = right
else:
    result = (left if abs(d_left) < epsilon else right)
print("x =", result, "за", k, "вызовов функций")

```

Листинг 2 – Алгоритм хорд

Результаты вычислений

Программа, реализующая алгоритм средней точки, выдала результат за 11 вызовов функций:

```
x = 2.020751953125 за 11 вызовов функций
```

Программа, реализующая алгоритм хорд, выдала результат за 41 вызовов функций:

```
x = 2.0214331394997 за 41 вызовов функций
```

Сравнительная характеристика методов

Исходя из представленных результатов, легко сделать вывод, что с задачей лучше справился алгоритм средней точки.

Результат вычисления у алгоритма средней точки более точный, чем у алгоритма хорд.

В то же время вычислительная сложность метода средней точки намного меньше, чем у алгоритма хорд. Судя по вызовам функций, алгоритм хорд обрабатывает заданный отрезок в 4 раза медленнее, чем алгоритм средней точки.

Выводы

В ходе работы были представлены метод средней точки и метод хорд. Для каждого из этих методов сделана программа на языке Python и выполнена с применением входных параметров из условия задачи. Были проанализированы результаты их работы, а так же сделаны выводы, что лучшим алгоритмом в данном случае является метод средней точки.