Министерство образования Российской Федерации

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Н.Э. БАУМАНА

Факультет: Информатика и системы управления
Кафедра: Информационная безопасность (ИУ8)

ТЕОРИЯ СИСТЕМ И СИСТЕМНЫЙ АНАЛИЗ

Лабораторная работа №1 на тему:

«Исследование методов прямого поиска экстремума унимодальной функции одного переменного»

Вариант 7

Преподаватель:
Строганов И.С.

Студент:
Заботин Д.В.,

Группа:
ИУ8-31

Москва 2021

Цель работы

Исследование функционирование и провести сравнительный анализ различных алгоритмов прямого поиска экстремума (пассивный поиск, метод дихотомии) на примере унимодальной функции одного переменного.

Постановка задачи

На интервале *[a, b]* задана унимодальная функция одного переменного *f(x)*. Используя методы *последовательного поиска* (дихотомии, золотого сечения и Фибоначчи), отыскать интервалы неопределенности $\varepsilon = 0{,}1$. Провести сравнение с методом *оптимального пассивного поиска*. Результат в зависимости от числа точек разбиения *N* представить в виде таблицы.

Ход работы

На интервале [1, 4] задана унимодальная функция одного переменного $f(x) = -\sqrt{(x)} * \sin x - 1{,}5$. Используя метод дихотомии, найти интервал нахождения минимума *f(x)* при заданной наибольшей допустимой длине интервала неопределенности $\varepsilon = 0{,}1$. Провести сравнение с методом оптимального пассивного поиска. Интервал делится на $N + 1$ частей точками с координатами:

$x_k = a + k\frac{b-a}{N+1}$, $k = 1, \ldots, N$.

Точность поиска равна $\Delta N = \frac{b-a}{N+1} = \frac{\varepsilon}{2}$, наибольшая длина интервала неопределенности: $\varepsilon = \frac{2*(b-a)}{N+1}$, откуда число точек определяется:

$$N = \frac{2*(b-a)}{\varepsilon} - 1.$$

Среди вычисленных значений функции выбирается минимальное и дается ответ в виде: $x_{min} \pm \Delta N$,

$$y_{min} = f(x_{min}).$$

Вычислительная сложность: $O(N)$.

Скорость сходимости (скорость убывания интервала неопределенности в зависимости от числа N вычисленных значений функции): $l_N = \frac{2}{N+1}$.

Для заданной функции имеем:

$$\Delta N = \frac{\varepsilon}{2} = \frac{0.1}{2} = 0.05 \qquad N = \frac{2*(4-1)}{0.1} - 1 = 59$$

```
Метод оптимального пассивного поиска:
+-----+------+-----------+
|   N |    x |      f(x) |
|-----+------+-----------|
|   1 | 1.05 | -2.38884  |
|   2 | 1.1  | -2.43471  |
|   3 | 1.15 | -2.47883  |
|   4 | 1.2  | -2.521    |
|   5 | 1.25 | -2.561    |
|   6 | 1.3  | -2.59863  |
|   7 | 1.35 | -2.63369  |
|   8 | 1.4  | -2.666    |
|   9 | 1.45 | -2.69538  |
|  10 | 1.5  | -2.72168  |
|  11 | 1.55 | -2.74472  |
|  12 | 1.6  | -2.76437  |
|  13 | 1.65 | -2.7805   |
|  14 | 1.7  | -2.79297  |
|  15 | 1.75 | -2.80169  |
|  16 | 1.8  | -2.80655  |
|  17 | 1.85 | -2.80748  |
|  18 | 1.9  | -2.80438  |
|  19 | 1.95 | -2.79722  |
|  20 | 2    | -2.78594  |
|  21 | 2.05 | -2.77051  |
|  22 | 2.1  | -2.75091  |
|  23 | 2.15 | -2.72713  |
|  24 | 2.2  | -2.69919  |
|  25 | 2.25 | -2.66711  |
|  26 | 2.3  | -2.63092  |
|  27 | 2.35 | -2.59067  |
|  28 | 2.4  | -2.54642  |
|  29 | 2.45 | -2.49826  |
```

Рисунок 1 - Результат работы оптимального пассивного поиска (часть 1).

```
|  30 | 2.5  | -2.44627  |
|  31 | 2.55 | -2.39055  |
|  32 | 2.6  | -2.33122  |
|  33 | 2.65 | -2.26841  |
|  34 | 2.7  | -2.20226  |
|  35 | 2.75 | -2.13291  |
|  36 | 2.8  | -2.06054  |
|  37 | 2.85 | -1.98532  |
|  38 | 2.9  | -1.90743  |
|  39 | 2.95 | -1.82706  |
|  40 | 3    | -1.74443  |
|  41 | 3.05 | -1.65974  |
|  42 | 3.1  | -1.57321  |
|  43 | 3.15 | -1.48508  |
|  44 | 3.2  | -1.39558  |
|  45 | 3.25 | -1.30495  |
|  46 | 3.3  | -1.21344  |
|  47 | 3.35 | -1.12131  |
|  48 | 3.4  | -1.02881  |
|  49 | 3.45 | -0.936197 |
|  50 | 3.5  | -0.843745 |
|  51 | 3.55 | -0.751716 |
|  52 | 3.6  | -0.660376 |
|  53 | 3.65 | -0.569995 |
|  54 | 3.7  | -0.48084  |
|  55 | 3.75 | -0.393176 |
|  56 | 3.8  | -0.307269 |
|  57 | 3.85 | -0.223381 |
|  58 | 3.9  | -0.141771 |
|  59 | 3.95 | -0.062692 |
+-----+------+-----------+

Результат: x_min = 1.850 ± 0.05, y_min = -2.807
```

Рисунок 2 - Результат работы оптимального пассивного поиска (часть 2).

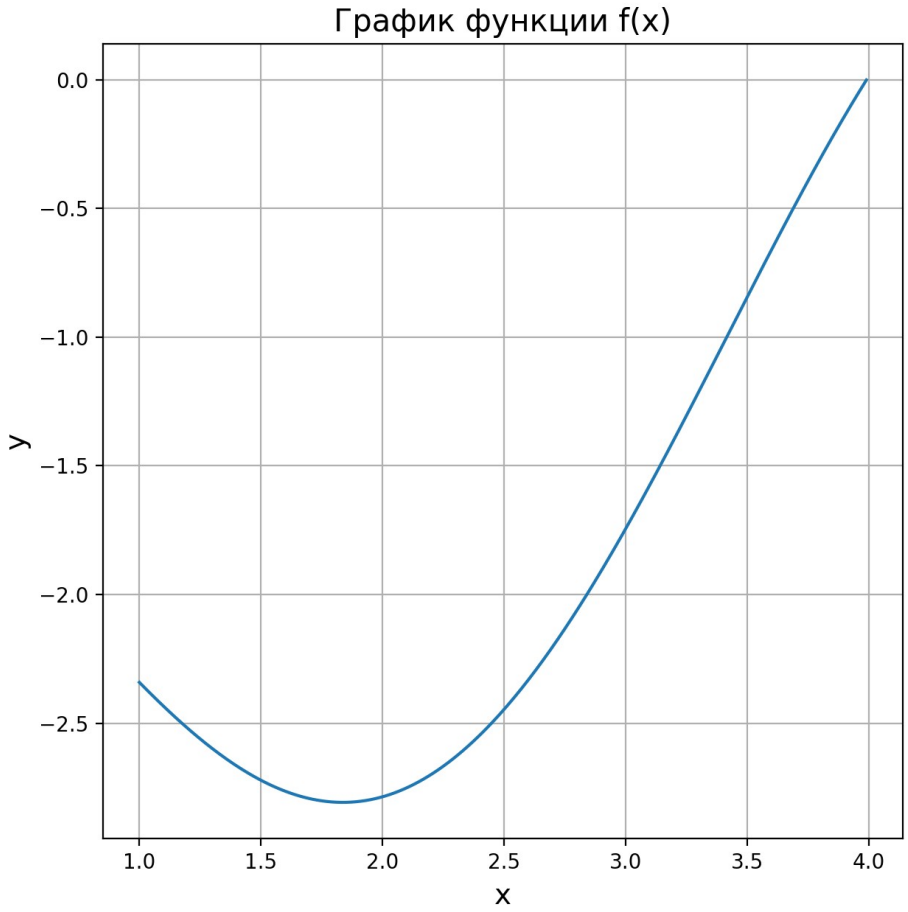

Рисунок 3 - График функции заданный вариантом.

Для метода дихотомии на отрезке $[a_k, b_k]$ длиной l_k выбираем две точки точки:

$$x_{k1}=\frac{b_k+a_k}{2}-\delta \qquad x_{k1}=\frac{b_k+a_k}{2}+\delta,$$

где $\delta > 0$ — некоторое достаточно малое число, причем $2\delta < \varepsilon$.

Вычисляем значения функции $f(x_{k1})$ и $f(x_{k2})$ в этих точках и выполняем процедуру исключения отрезка. В результате получаем новый отрезок $[a_{k+1}, b_{k+1}] \subset [a_k, b_k]$. Если длина данного отрезка $l_k > \varepsilon$, то алгоритм метода дихотомии переходит к следующему шагу. Но если $l_{k+1} < \varepsilon$, то вычисления прекращаются, результат: $x=\frac{b_{k+1}+a_{k+1}}{2}$.

Метод дихотомии:

a	b	x1	x2	f(x1)	f(x2)	f(x1) > f(x2) - ?	a_k+1	b_k+1	length
1	4	2.49	2.51	-2.45697	-2.43542	no	1	2.51	3
1	2.51	1.745	1.765	-2.80099	-2.80356	yes	1.745	2.51	1.51
1.745	2.51	2.1175	2.1375	-2.74306	-2.73347	no	1.745	2.1375	0.765
1.745	2.1375	1.93125	1.95125	-2.80039	-2.79699	no	1.745	1.95125	0.3925
1.745	1.95125	1.83812	1.85812	-2.80762	-2.80725	no	1.745	1.85812	0.20625
1.745	1.85812	1.79156	1.81156	-2.80601	-2.80712	yes	1.79156	1.85812	0.113125
1.79156	1.85812	1.81484	1.83484	-2.80724	-2.80762	yes	1.81484	1.85812	0.0665625

Результат: x_min = 1.8365 ± 0.0333, y_min = -2.8076

Рисунок 4 - Результат работы поиска минимума методом дихотомии.

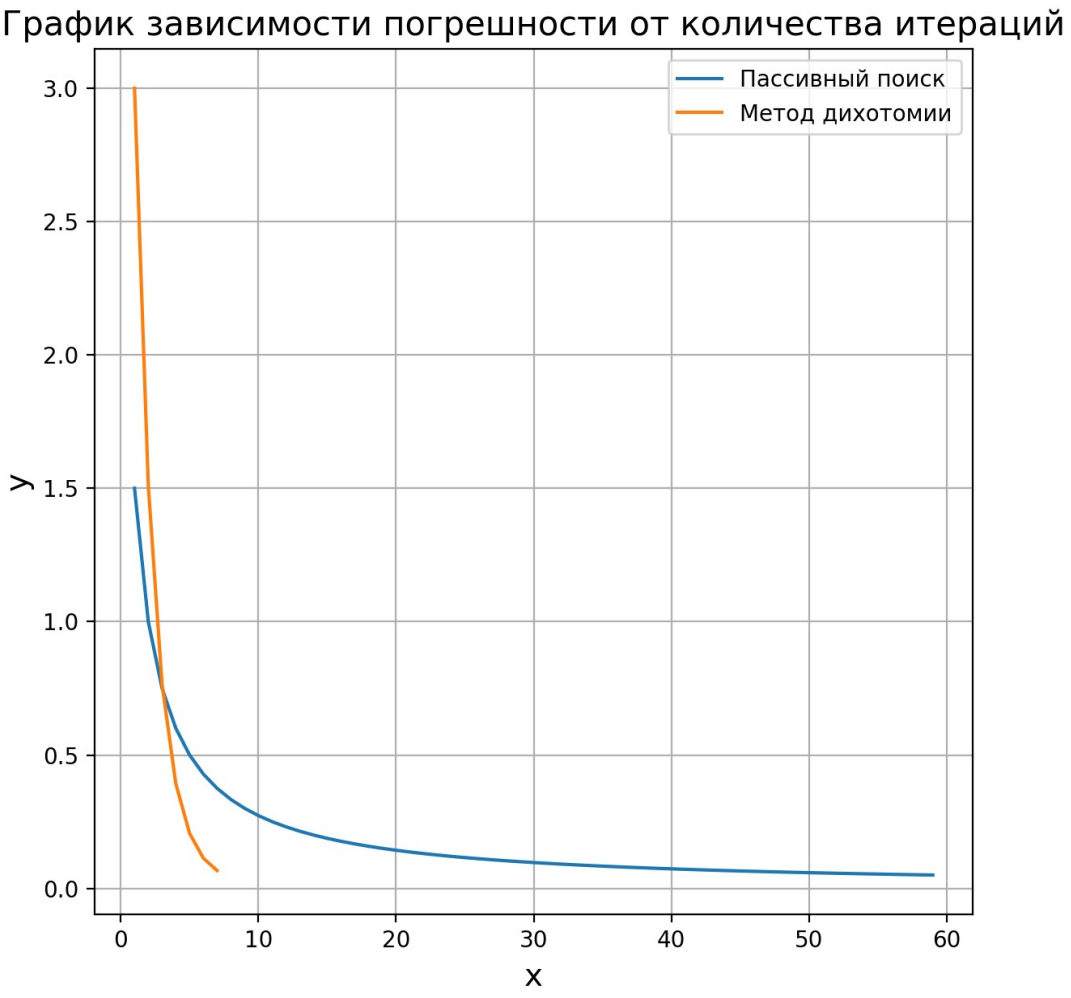

Рисунок 5 - График зависимости погрешности от количества итераций.

Вывод

В данной лабораторной работе были изучены методы поиска экстремума унимодальной функции одного переменного: оптимальный пассивный поиск и поиск методом дихотомии. Для выполнения лабораторной работы был использован язык программирования Python. На нем была написана программа, в которой были реализованы соответсвующие функции поиска экстремума, а также были реализованы функции вывода результатов в виде таблиц и графиков.

Из результатов видно, что метод дихотомии значительно эффективнее метода пассивного поиска для нахождения экстремума унимодальной функции одного переменного.

Исходный код программы

```
# Copyright 2021 DimaZzZz101 zabotin.d@list.ru

"""
                    Лабораторная работа №1
        Исследование методов прямого поиска экстремума унимодальной функции одного переменного
    Цель работы: исследовать функционирование и провести сравнительный анализ различных алгоритмов прямого поиска экстремума (пассивный поиск, метод дихотомии, золотого сечения, Фибоначчи) на примере унимодальной функции одного переменного.
                    Вариант 7
"""


# Подключение необходимых библиотек.
from tabulate import tabulate
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np


# Функция приведения числа к фиксированной точности.
def to_fixed(num_obj, digits=0):
    return f"{num_obj:.{digits}f}"


# Заданные константы
_epsilon = 0.1      # Интервал непределенности.
```

```
_begin = 1          # Левая граница.
_end = 4            # Правая гранциа.


# Список со созначениями длин при пассивном поиске.
list_of_length_passive = [(_end - _begin) / (n + 1) for n in range(1, 60)]

# Список со значениями длин при методе дихотомии.
list_of_length_dichotomy = []


# Функция из варинта.
def func_from_task(x):
    return -np.sqrt(x) * np.sin(x) - 1.5


# Сравнение погрешностей методов пассивного поиска и дихотомии.
def compare_inaccuracy(name="", length_passive=None,
length_dichotomy=None):
    if length_passive is None:
        length_passive = []

    if length_dichotomy is None:
        length_dichotomy = []

    x1 = [n for n in range(1, len(length_passive) + 1)]
    x2 = [n for n in range(1, len(length_dichotomy) + 1)]

    y1 = length_passive
    y2 = length_dichotomy
```

```
    fig = plt.figure(figsize=(7, 7))
    plt.plot(x1, y1, label="Пассивный поиск")
    plt.plot(x2, y2, label="Метод дихотомии")
    plt.title(name, fontsize=15)        # Заголовок.
    plt.xlabel("x", fontsize=14)        # Ось абсцисс.
    plt.ylabel("y", fontsize=14)        # Ось ординат.
    plt.grid(True)                      # Включение отображение сетки.
    plt.legend()
    plt.show()
    save = "Graphics/" + name + ".png"  # Указание пути до файла.
    fig.savefig(save)                   # Сохранение графика по заданному пути.


# Функция подсчета N - количества итераций.
def count_of_iterations(begin, end, epsilon):
    return 2 * (end - begin) / epsilon - 1


# Функция подсчета точности.
def count_delta(epsilon):
    return epsilon / 2


# Функция вывода таблицы для метода пассивного поиска на экран.
def create_table_passive_search(x_array, y_array, n):
    pd.set_option('display.max_rows', n)

    table = pd.DataFrame({
        'N': list(range(1, n + 1)),
```

```
        'x': x_array,
        'f(x)': y_array,
    })

    table.set_index('N', inplace=True)

    print(tabulate(table, headers='keys', tablefmt='psql'), end='\n\n')


# Метод оптимального пассивного поиска.
def optimal_passive_search(begin, end, epsilon):
    delta = count_delta(epsilon)

    list_of_x = np.arange(begin + delta, end, delta)
    list_of_y = [func_from_task(x) for x in list_of_x]

    n = int(count_of_iterations(begin, end, epsilon))

    # Задание начального состояния переменных для поиска минимума.
    y_min = 5
    x_min = 0

    # Поиск y_min и соответсвующего x_min.
    for x in list_of_x:
        y = func_from_task(x)
        if y < y_min:
            x_min = x
            y_min = y

    # Вывод таблицы со всеми x и y.
```

```
    print("Метод оптимального пассивного поиска:")
    create_table_passive_search(list_of_x, list_of_y, n)

    # Вывод результата - искомых x_min и y_min.
    print(f"Результат: x_min = {to_fixed(x_min, 3)} ± {delta}, y_min =
{to_fixed(y_min, 3)}", end='\n\n')



# Функция печати таблицы для метода дихотомии.
def create_table_dichotomy(list_a, list_b, list_x1, list_x2, list_f1, list_f2,
list_yes_no, list_a_, list_b_, length):
    pd.set_option('display.max_rows', None)

    table = pd.DataFrame({
        'a': list_a,
        'b': list_b,
        'x1': list_x1,
        'x2': list_x2,
        'f(x1)': list_f1,
        'f(x2)': list_f2,
        'f(x1) > f(x2) - ?': list_yes_no,
        'a_k+1': list_a_,
        'b_k+1': list_b_,
        'length': length
    })
    table.set_index('a', inplace=True)

    print(tabulate(table, headers='keys', tablefmt='psql'), end='\n\n')
```

```
# Поиск методом дихотомии.
def dichotomy_search(begin, end, epsilon):
    new_a = begin
    new_b = end

    delta = 0.01

    # Списки для сохранения данных будущей таблицы
    list_a = []
    list_b = []

    list_x1 = []
    list_x2 = []

    list_f1 = []
    list_f2 = []

    list_yes_no = []

    list_a_ = []
    list_b_ = []

    while True:
        # Сохраняем текущие начало и конец отрезка
        list_a.append(new_a)
        list_b.append(new_b)

        # Вычисляем расстояние между концами отрезка.
        length = new_b - new_a
```

```
        # Вычисление точности.
        precision = length / 2

        # Запоминаем текущую дляину отрезка
        list_of_length_dichotomy.append(length)

        # Вычисляем, затем сохраняем полученные значения точек в списки.
        x1 = (new_a + new_b) / 2 - delta
        x2 = (new_a + new_b) / 2 + delta

        list_x1.append(x1)
        list_x2.append(x2)

        # К текущим значениям точек находим соответсвующие значения
функций.
        f_x1 = func_from_task(x1)
        f_x2 = func_from_task(x2)

        # Сохраняем их.
        list_f1.append(f_x1)
        list_f2.append(f_x2)

        if f_x1 > f_x2:
            new_a = x1
            list_yes_no.append('yes')
        else:
            new_b = x2
            list_yes_no.append('no')

        # Сохраняем новые начало и конец отрезка.
```

```
        list_a_.append(new_a)
        list_b_.append(new_b)

        # Условие выхода из цикла.
        if length < epsilon:
            x_part1 = (new_a + new_b) / 2       # Запоминаем полученный минимальный x.

            y_min = func_from_task(x_part1)     # Ему в соответсвие вычисляем значение функции в нем.

            # Вывод таблицы для метода дихотомии.
            print("Метод дихотомии:")
            create_table_dichotomy(list_a, list_b, list_x1,
                                   list_x2, list_f1, list_f2,
                                   list_yes_no, list_a_, list_b_,
                                   list_of_length_dichotomy)

            # Вывод результата.
            print(f'Результат: x_min = {to_fixed(x_part1, 4)} ± {to_fixed(precision, 4)}, y_min = {to_fixed(y_min, 4)}')
            break


# Функция вывода графика.
def show_graphic(name="", x=None, y=None):
    if y is None:
        y = []

    if x is None:
```

```
        x = []

    fig = plt.figure(figsize=(7, 7))
    plt.plot(x, y)

    plt.title(name, fontsize=15)        # Заголовок.
    plt.xlabel("x", fontsize=14)        # Ось абсцисс.
    plt.ylabel("y", fontsize=14)        # Ось ординат.
    plt.grid(True)                      # Включение отображение сетки.
    plt.show()
    save = "Graphics/" + name + ".png"
    fig.savefig(save)



# Вывод пояснительной надписи.
print('Лабораторная работа №1', 'Вариант 7', 'Функция: -sqrt(x) * sin(x) - 1.5',
sep='\n', end='\n\n')

# Метод пассивного поиска.
optimal_passive_search(_begin, _end, _epsilon)

# Метод дихотомии.
dichotomy_search(_begin, _end, _epsilon)

# Построение графика функции.
x_array = np.arange(_begin, _end, 0.01)
y_array = func_from_task(x_array)
show_graphic("График функции f(x)", x_array, y_array)
```

```
# Построение графика сравнения работы медода пассивного поиска и метода дихотомии дихотомии, в зависимости от N.
compare_inaccuracy("График зависимости погрешности от количества итераций", list_of_length_passive,
                   list_of_length_dichotomy)
```