Лабораторная 2

1 Задача 1: Минимизация квадратичной функции методами оптимизации

Функция:

$$f(x) = 3x^2 - 2x - 2, \quad x \in [-1, 1]$$

Цель задачи — найти минимум данной функции на отрезке с помощью методов: золотого сечения, дихотомии и поразрядного поиска.

1.1 Метод золотого сечения

Метод золотого сечения основан на делении интервала в пропорции золотого числа, которое примерно равно 0.618. На каждой итерации рассматриваются две точки внутри интервала, и в зависимости от значений функции в этих точках, интервал сужается до тех пор, пока его длина не станет меньше заданной точности.

Алгоритм:

1. Инициализация: $a = -1$, $b = 1$, точность $\mathrm{tol} = 10^{-5}$.
2. Вычисление коэффициента золотого сечения $gr = \frac{\sqrt{5}-1}{2}$.
3. На каждой итерации вычисляются точки c и d, которые делят интервал в пропорции золотого сечения.
4. Сравнение значений функции в точках $f(c)$ и $f(d)$. В зависимости от этого сужаем интервал.
5. Цикл продолжается до тех пор, пока длина интервала не станет меньше заданной точности.

Результат:

$$x_{\min} \approx -0.3333, \quad f(x_{\min}) \approx -2.3333$$

1.2 Метод дихотомии

Метод дихотомии заключается в последовательном делении интервала на две части с помощью точек, расположенных симметрично относительно середины интервала. Интервал сужается на каждой итерации, пока его длина не станет меньше заданной точности.

Алгоритм:

1. Инициализация: $a = -1$, $b = 1$, точность $\mathrm{tol} = 10^{-5}$, малое смещение $\delta = 10^{-6}$.

2. На каждой итерации вычисляются две точки $x_1 = \frac{a+b}{2} - \delta$ и $x_2 = \frac{a+b}{2} + \delta$.

3. Сравниваем значения функции $f(x_1)$ и $f(x_2)$, сужаем интервал в зависимости от этих значений.

4. Процесс повторяется, пока длина интервала не станет меньше заданной точности.

Результат:

$$x_{\min} \approx -0.3333, \quad f(x_{\min}) \approx -2.3333$$

1.3 Метод поразрядного поиска

Метод поразрядного поиска основан на последовательном изменении значения переменной влево или вправо с фиксированным шагом, пока не будет найдено приближение минимума. Если при смещении в обе стороны не достигается улучшения, шаг уменьшается.

Алгоритм:

1. Инициализация: начальная точка $x_0 = 0$, шаг $\alpha = 0.1$, точность $\mathrm{tol} = 10^{-5}$.

2. На каждой итерации сравниваем значения функции в точках $x - \mathrm{step}$, x и $x + \mathrm{step}$.

3. Если $f(x - \mathrm{step}) < f(x)$, смещаемся влево, если $f(x + \mathrm{step}) < f(x)$, смещаемся вправо.

4. Если ни одно смещение не уменьшает значение функции, уменьшаем шаг вдвое.

Результат:

$$x_{\min} \approx -0.3333, \quad f(x_{\min}) \approx -2.3333$$

2 Задача 2: Минимизация кубической функции методом парабол

Функция:

$$f(x) = 3x^3 + 6x^2 + x + 1, \quad x \in [-3, 3]$$

Цель задачи — найти минимум данной кубической функции на отрезке методом парабол.

2.1 Метод парабол

Метод парабол заключается в аппроксимации функции параболой через три точки на интервале и нахождении вершины этой параболы, которая является приближением минимума функции.

Алгоритм:

1. Инициализация: $a = -3$, $b = 3$, точность $\mathrm{tol} = 10^{-5}$.

2. Выбираем три точки: $x_1 = a$, $x_2 = \frac{a+b}{2}$, $x_3 = b$.

3. На каждой итерации вычисляются значения функции в точках $f(x_1), f(x_2), f(x_3)$.

4. Аппроксимируем параболу, вычисляем вершину параболы $x_{\min}$ с помощью формул для коэффициентов аппроксимации.

5. Если значение функции в точке $x_{\min}$ меньше, чем в x_2, то сужаем интервал, выбирая новые точки.

6. Процесс повторяется, пока длина интервала не станет меньше заданной точности.

Результат:

$$x_{\min} \approx -1.0, \quad f(x_{\min}) \approx -1.0$$

3 Код программы на Python

Python-код для задачи 1 и задачи 2:

```
import numpy as np

# 1. First task: Methods of optimization (golden section, dichotomy, c
def f(x):
    return 3*x**2 - 2*x - 2

# Method of golden section search
def golden_section_search(a, b, tol=1e-5):
    gr = (np.sqrt(5) - 1) / 2
    c = b - gr * (b - a)
    d = a + gr * (b - a)

    while abs(b - a) > tol:
        if f(c) < f(d):
            b = d
        else:
            a = c

        c = b - gr * (b - a)
        d = a + gr * (b - a)

    return (b + a) / 2

# Method of dichotomy search
def dichotomy_search(a, b, tol=1e-5, delta=1e-6):
    while abs(b - a) > tol:
        x1 = (a + b) / 2 - delta
        x2 = (a + b) / 2 + delta

        if f(x1) < f(x2):
            b = x2
        else:
            a = x1
```

```
    return (a + b) / 2

# Method of coordinate search
def coordinate_search(x0, tol=1e-5, alpha=0.1):
    x = x0
    step = alpha
    while abs(step) > tol:
        f_left = f(x - step)
        f_right = f(x + step)

        if f_left < f(x):
            x = x - step
        elif f_right < f(x):
            x = x + step
        else:
            step *= 0.5

    return x

# 2. Second task: Parabolic method

def f_cubic(x):
    return 3*x**3 + 6*x**2 + x + 1

# Method of parabolic search
def parabola_search(a, b, tol=1e-5):
    x1, x2, x3 = a, (a + b) / 2, b

    while abs(b - a) > tol:
        f1, f2, f3 = f_cubic(x1), f_cubic(x2), f_cubic(x3)

        num = (x2**2 - x3**2)*f1 + (x3**2 - x1**2)*f2 + (x1**2 - x2**2)
        denom = (x2 - x3)*f1 + (x3 - x1)*f2 + (x1 - x2)*f3

        x_min = 0.5 * num / denom

        if f_cubic(x_min) < f_cubic(x2):
            if x_min < x2:
                x3 = x2
            else:
                x1 = x2
            x2 = x_min
        else:
            if x_min < x2:
                x1 = x_min
            else:
                x3 = x_min
```

```
    return x_min

# Running the methods
if __name__ == '__main__':
    # Task 1: Optimization methods

    a, b = -1, 1

    # Golden section search
    x_min_golden = golden_section_search(a, b)
    print(f"Golden section method: x_min = {x_min_golden}, f(x_min) = {

    # Dichotomy search
    x_min_dichotomy = dichotomy_search(a, b)
    print(f"Dichotomy method: x_min = {x_min_dichotomy}, f(x_min) = {f(

    # Coordinate search
    x0 = 0
    x_min_coordinate = coordinate_search(x0)
    print(f"Coordinate search method: x_min = {x_min_coordinate}, f(x_

    # Task 2: Parabolic method for cubic function

    a, b = -3, 3

    # Parabolic search
    x_min_parabola = parabola_search(a, b)
    print(f"Parabolic method: x_min = {x_min_parabola}, f(x_min) = {f_
```