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

Студент(ка) ак. группы БИВТ-18-1 ____________ Гузев В. Н.

аббревиатура подпись И.О. Фамилия

Оценка с учетом защиты ______________________ ____________

оценка дата

Преподаватель ____________ ____________

подпись И.О. Фамилия

Москва 2020

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 унимодальной. То есть найти такую точку $x^* \in [a, b]$, что $x^* = \min_{x \in [a,b]} f(x)$.

Целевая функция по № варианта:

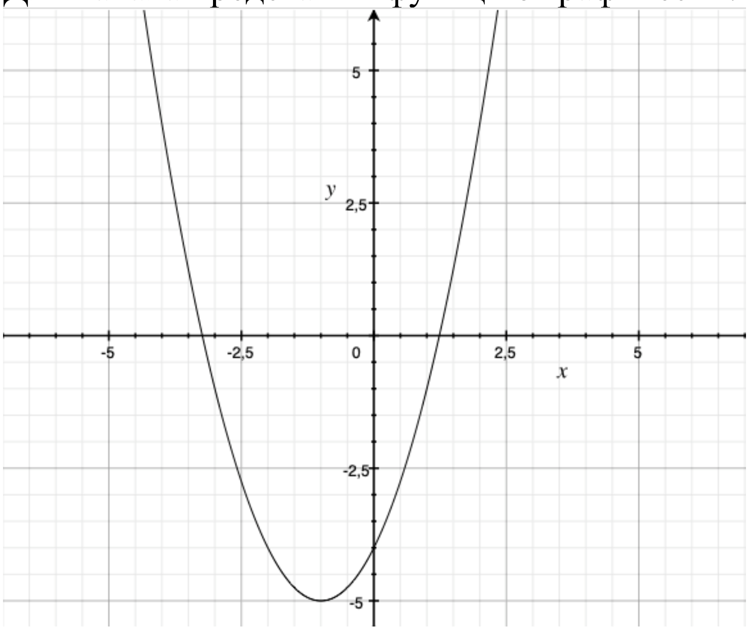
№	Функция	a	b	ε	Метод 1	Метод 2	Метод 3
7	$f(x) = x^2 + 2x - 4$	-2	1	0.0001	1	4	6

Исследование функции

Требуется найти:

a) Графическое представление функции на заданном интервале
b) Найденное точное значение минимума функции и координата точки где он достигается

Для начала представим функцию графически:



Затем аналитически найдём точное значение её минимума.
Чтобы найти глобальный минимум, необходимо для начала найти критические точки. Для их поиска воспользуемся производной первого порядка:

$$f'(x) = \frac{d}{dx}(x^2 + 2x - 4) = 2x + 2 = 2(x + 1)$$

Найдём критические точки:

$$2(x + 1) = 0$$
$$x = -1$$

Единственная критическая точка функции $x = -1$.
При члене x^2 стоит положительный коэффициент, поэтому парабола направлена вверх, а точка $x = -1$ является глобальным минимумом со значением функции в ней:

$$f(x) = (-1)^2 + 2 * (-1) - 4 = -5$$

Листинги программ

Для решения поставленной задачи целесообразно использовать язык высокого уровня Swift.

```
import Foundation

enum Errors: Error {
   case invalidParameters
}

func f(_ x: Double) -> Double {
   return pow(x, 2) + 2 * x - 4
}

let a = -2.0
let b = 1.0
let epsilon = 0.0001
let delta = Double.random(in: 0..<(epsilon / 2)) // Константа различимости

guard a <= b && epsilon > 0 else {
   print("Неверные параметры")
   throw Errors.invalidParameters
}

// Метод равномерного поиска

var N = Int(ceil((b - a) / epsilon))

var min1_x = a
var min1_y = f(a)

for i in 1...N {
   let xi = a + Double(i) * (b - a) / Double(N)
   let yi = f(xi)
```

```
    if yi < min1_y {
        min1_y = yi
        min1_x = xi
    }
}

print("""
    Минимальное значение функции по методу равномерного поиска:
\(min1_y),
    достигается в точке x = \(min1_x) за \(N) шагов\n\n
    """)

// Метод дихотомии

var a_var = a
var b_var = b

N = 0

while (b_var - a_var) >= epsilon {
    let x1 = (a_var + b_var) / 2.0 - delta
    let x2 = (a_var + b_var) / 2.0 + delta
    if f(x1) < f(x2) {
        b_var = x2
    } else {
        a_var = x1
    }
    N += 1
}

let min2_x = f(a_var) < f(b_var) ? a_var : b_var
let min2_y = f(min2_x)

print("""
    Минимальное значение функции по методу дихотомии: \(min2_y),
    достигается в точке x = \(min2_x) за \(N) шагов\n\n
    """)

// Метод Фибоначчи

a_var = a
b_var = b
```

```
var F1 = 1
var F2 = 1
var F = [F1, F2]

while F.last! <= Int(ceil((b - a) / epsilon)) {
    F.append(F1 + F2)
    F1 = F2
    F2 = F.last!
}

N = F.count - 1

var x1 = a_var +
            Double(F[N - 2]) / Double(F[N]) *
                (b_var - a_var)
var x2 = a_var +
            Double(F[N - 1]) / Double(F[N]) *
                (b_var - a_var)
var f1 = f(x1)
var f2 = f(x2)

var k = 1

while k < N - 1 {
    if f1 > f2 {
        a_var = x1
        x1 = x2
        f1 = f2
        x2 = a_var + Double(F[N - k - 1]) / Double(F[N - k]) * (b_var - a_var)
        f2 = f(x2)
    } else {
        b_var = x2
        x2 = x1
        f2 = f1
        x1 = a_var + Double(F[N - k - 2]) / Double(F[N - k]) * (b_var - a_var)
        f1 = f(x1)
    }
    k += 1
}

x2 = x1 + delta
if f(x1) > f(x2) {
    a_var = x1
```

```
} else {
   b_var = x2
}

let min3_x = (a_var + b_var) / 2
let min3_y = f(min3_x)

print("""
   Минимальное значение функции по методу Фибоначчи: \(min3_y),
   достигается в точке x = \(min3_x) за \(k) шагов\n\n
   """)
```

Результаты вычислений

Результаты вычислений представлены в виде снимка экрана:

```
Минимальное значение функции по методу равномерного поиска: -5.0,
достигается в точке x = -1.0 за 30000 шагов


Минимальное значение функции по методу дихотомии: -4.99999999795299,
достигается в точке x = -1.0000452438950422 за 19 шагов


Минимальное значение функции по методу Фибоначчи: -4.999999999902766,
достигается в точке x = -1.0000098607238375 за 22 шагов


Program ended with exit code: 0
```

Сравнительная характеристика методов

В ходе вычислений все методы дали приемлимые результаты, с учётом допустимого отклонения ε.

Точное значение дал метод равномерного поиск, но это обусловлено равномерным шагом, который привёл алгоритм ровно в целочисленную точку минимума. В то время как другие методы использовали принцип «разделяй и властвуй», и результат их работы скорее можно охарактеризовать как доверительный интервал, в котором находится точка минимума.

Наиболее эффективным по количеству выполняемых операций является метод дихотомии, наименее эффективным – метод равномерного поиска.

Разбор программы с записью экрана

Разбор доступен по ссылке: Запись ЛР №1

Выводы

Мы сравнили алгоритмы поиска локального минимума функции, и сделали вывод, что наиболее точным является результат по методу равномерного поиска. Тем не менее, его эффективность по количеству выполняемых операций очень низкая, поэтому в серьёзных вычислениях целесообразнее применять метод дихотомии или метод Фибоначчи.