Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №4

по дисциплине «Методы оптимизации»

Вариант: 17

Преподаватель:
Селина Елена Георгиевна

Выполнил:
Тимошкин Роман Вячеславович
Группа: P3231

Санкт-Петербург, 2025

Задание

Найти экстремум функции методами покоординатного спуска, градиентного спуска и наискорейшего спуска. Три итерации каждого метода выполнить вручную + написать программу на одном из языков программирования.

Уравнение:

$$f(x_1, x_2) = 6x_1^2 + x_2^2 - x_1x_2 + 4x_1 - 8x_2 + 1, \quad M0 = (2, 2)$$

Решение методом покоординатного спуска

Итерация 1

Шаг 1: Минимизация по x_1 при фиксированном $x_2 = 2$

Функция превращается в одномерную:

$$f(x_1, 2) = 6x_1^2 + 2^2 - x_1(2) + 4x_1 - 8(2) + 1 = 6x_1^2 - 2x_1 + 4x_1 - 16 + 4 + 1 = 6x_1^2 + 2x_1 - 11$$

Находим минимум, приравняв производную к нулю:

$$\frac{d}{dx_1}(6x_1^2 + 2x_1 - 11) = 12x_1 + 2 = 0$$

Решение:

$$x_1 = -\frac{2}{12} = -\frac{1}{6}$$

Точка: $(-\frac{1}{6}, 2)$.

Шаг 2: Минимизация по x_2 при фиксированном $x_1 = -\frac{1}{6}$

Теперь функция:

$$f\left(-\frac{1}{6}, x_2\right) = 6\left(-\frac{1}{6}\right)^2 + x_2^2 - \left(-\frac{1}{6}\right)x_2 + 4\left(-\frac{1}{6}\right) - 8x_2 + 1$$

$$= 6\left(\frac{1}{36}\right) + x_2^2 + \frac{1}{6}x_2 - \frac{2}{3} - 8x_2 + 1 = \frac{1}{6} + x_2^2 + \frac{1}{6}x_2 - \frac{2}{3} - 8x_2 + 1 = x_2^2 - \frac{47}{6}x_2 + \frac{29}{6}$$

Находим минимум:

$$\frac{d}{dx_2}\left(x_2^2 - \frac{47}{6}x_2 + \frac{29}{6}\right) = 2x_2 - \frac{47}{6} = 0$$

Решение:

$$x_2 = \frac{47}{12}$$

Точка: $\left(-\frac{1}{6}, \frac{47}{12}\right)$.

Вычислим значение функции:

$$f\left(-\frac{1}{6}, \frac{47}{12}\right) = 6\left(-\frac{1}{6}\right)^2 + \left(\frac{47}{12}\right)^2 - \left(-\frac{1}{6} \cdot \frac{47}{12}\right) + 4\left(-\frac{1}{6}\right) - 8\left(\frac{47}{12}\right) + 1$$

$$= 6 \cdot \frac{1}{36} + \frac{2209}{144} + \frac{47}{72} - \frac{4}{6} - \frac{376}{12} + 1 = \frac{1}{6} + \frac{2209}{144} + \frac{47}{72} - \frac{2}{3} - \frac{94}{3} + 1$$

$$\approx -24.2083$$

Проверим критерий остановки:

$$|f(M_1) - f(M_0)| = |-24.2083 - (-19)| = |-5.2083| > 0.0001$$

$$\left(-\frac{1}{6} - 2\right)^2 + \left(\frac{47}{12} - 2\right)^2 \approx 2.97 > 0.0001$$

Алгоритм продолжается.

Итерация 2

Шаг 1: Минимизация по x_1 при $x_2 = \frac{47}{12}$

Функция:

$$f\left(x_1, \frac{47}{12}\right) = 6x_1^2 + \left(\frac{47}{12}\right)^2 - x_1\left(\frac{47}{12}\right) + 4x_1 - 8\left(\frac{47}{12}\right) + 1$$

Берем производную:

$$\frac{d}{dx_1}\left(6x_1^2 - \frac{47}{12}x_1 + 4x_1\right) = 12x_1 - \frac{47}{12} + 4 = 0$$

Решаем:

$$12x_1 + \frac{1}{12} = 0 \quad \Rightarrow \quad x_1 = -\frac{1}{144}$$

Новая точка:

$$M_2 = \left(-\frac{1}{144}, \frac{47}{12}\right)$$

Шаг 2: Минимизация по x_2 при $x_1 = -\frac{1}{144}$

Функция:

$$f\left(-\frac{1}{144}, x_2\right) = 6\left(-\frac{1}{144}\right)^2 + x_2^2 - \left(-\frac{1}{144}\right)x_2 + 4\left(-\frac{1}{144}\right) - 8x_2 + 1$$

Берем производную:

$$\frac{d}{dx_2}\left(x_2^2 - 8x_2 + C\right) = 2x_2 - 8 = 0$$

Решаем:

$$x_2 = 4$$

Новая точка:

$$M_3 = \left(-\frac{1}{144}, 4\right)$$

Проверка критерия остановки

$$|f(M_3) - f(M_2)| = \left|f\left(-\frac{1}{144}, 4\right) - f\left(-\frac{1}{6}, \frac{47}{12}\right)\right|$$

$$\left(-\frac{1}{144} + \frac{1}{6}\right)^2 + \left(4 - \frac{47}{12}\right)^2 \approx 0.0833 > \epsilon$$

Вывод: Метод продолжается.

Итерация 3

После второй итерации была получена следующая точка:

$$M_2 = \left(-\frac{1}{144}, 4\right)$$

Теперь выполняем очередной цикл минимизации.

Шаг 1: Минимизация по x_1 при $x_2 = 4$

Функция:

$$f\left(x_1, 4\right) = 6x_1^2 + 16 - 4x_1 + 4x_1 - 32 + 1 = 6x_1^2 - 15$$

Берем производную и приравниваем к нулю:

$$\frac{d}{dx_1}\left(6x_1^2 - 15\right) = 12x_1 = 0$$

Решаем:

$$x_1 = 0$$

Следовательно, получаем точку:

$$M_3 = (0, 4)$$

Шаг 2: Минимизация по x_2 при $x_1 = 0$

Функция:

$$f\left(0, x_2\right) = 6(0)^2 + x_2^2 - 0 \cdot x_2 + 4(0) - 8x_2 + 1 = x_2^2 - 8x_2 + 1$$

Берем производную:

$$\frac{d}{dx_2}\left(x_2^2 - 8x_2 + 1\right) = 2x_2 - 8 = 0$$

Решаем:

$$x_2 = 4$$

Точка остаётся прежней:

$$M_3 = (0, 4) \quad \text{(не изменилась)}$$

Проверка критерия остановки

Разница значений функции:

$$|f(M_3) - f(M_2)| = |f(0, 4) - f\left(-\frac{1}{144}, 4\right)|$$

Подставляем:

$$f(0, 4) = 0 + 16 - 0 + 0 - 32 + 1 = -15$$

$$f\left(-\frac{1}{144}, 4\right) \approx -14.9998$$

$$|-15 - (-14.9998)| = 0.0002 > \epsilon$$

Разница координат:

$$\left(0 - \left(-\frac{1}{144}\right)\right)^2 + (4 - 4)^2 = \left(\frac{1}{144}\right)^2 = \frac{1}{20736} \approx 4.8 \times 10^{-5}$$

Также больше ϵ, следовательно, критерий остановки не выполнен, и процесс минимизации продолжается.

Код программы

```
import numpy as np


def f(x1, x2):
    return 6 * x1 ** 2 + x2 ** 2 - x1 * x2 + 4 * x1 - 8 * x2 + 1


def coordinate_descent(x1_init, x2_init, epsilon=0.0001, max_iter=100):
    x1, x2 = x1_init, x2_init
    prev_f = f(x1, x2)
    print(f"Начальная точка: ({x1}, {x2}), f = {prev_f}")

    for iteration in range(1, max_iter + 1):
        print(f"\nИтерация {iteration}")

        # Шаг 1: Минимизация по x1 при фиксированном x2
        print(f"Шаг 1: Минимизация по x1 при x2 = {x2}")
        # Производная по x1: 12*x1 - x2 + 4 = 0 => x1 = (x2 - 4)/12
        new_x1 = (x2 - 4) / 12
        print(f"Новый x1 = {new_x1}")

        # Шаг 2: Минимизация по x2 при фиксированном x1
        print(f"Шаг 2: Минимизация по x2 при x1 = {new_x1}")
        # Производная по x2: 2*x2 - x1 - 8 = 0 => x2 = (x1 + 8)/2
        new_x2 = (new_x1 + 8) / 2
        print(f"Новый x2 = {new_x2}")

        # Вычисляем новое значение функции
        current_f = f(new_x1, new_x2)
        print(f"Точка: ({new_x1}, {new_x2}), f = {current_f}")

        # Проверка критерия остановки
        delta_f = abs(current_f - prev_f)
        delta_x = (new_x1 - x1) ** 2 + (new_x2 - x2) ** 2
        print(f"Разница значений функции: {delta_f}")
        print(f"Разница координат: {delta_x}")

        if delta_f < epsilon and delta_x < epsilon:
            print("\nКритерий остановки достигнут.")
            break

        # Обновляем значения для следующей итерации
        x1, x2 = new_x1, new_x2
        prev_f = current_f

    return x1, x2


# Начальная точка из примера
```

```
x1_init = 2
x2_init = 2

x1_opt, x2_opt = coordinate_descent(x1_init, x2_init)

print("\nРезультат:")
print(f"Оптимальная точка: ({x1_opt}, {x2_opt})")
print(f"Значение функции: {f(x1_opt, x2_opt)}")
```

Решение методом градиентного спуска

Выбор шага и начальная точка

Метод градиентного спуска использует обновление координат по формуле:

$$x^{(k+1)} = x^{(k)} - \alpha \nabla f(x^{(k)})$$

где α — шаг градиентного спуска, $\nabla f(x)$ — градиент функции. Возьмём начальную точку $M_0 = (2, 2)$ и фиксированный шаг $\alpha = 0.1$.

Итерация 1

Градиент функции:

$$\nabla f(x_1, x_2) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \right)$$

Частные производные:

$$\frac{\partial f}{\partial x_1} = 12x_1 - x_2 + 4, \quad \frac{\partial f}{\partial x_2} = 2x_2 - x_1 - 8$$

Подставляем $M_0 = (2, 2)$:

$$\nabla f(2, 2) = (12(2) - 2 + 4, 2(2) - 2 - 8) = (26, -6)$$

Обновляем координаты:

$$x_1^{(1)} = 2 - 0.1 \times 26 = -0.6, \quad x_2^{(1)} = 2 - 0.1 \times (-6) = 2.6$$

Вычисляем значение функции:

$$f(M_0) = 6(2)^2 + (2)^2 - 2 \cdot 2 + 4 \cdot 2 - 8 \cdot 2 + 1 = 17$$

$$f(M_1) \approx -11.72$$

Проверяем критерий остановки:

$$|f(M_1) - f(M_0)| = |-11.72 - 17| = 28.72 > 0.0001$$

Алгоритм продолжается.

Итерация 2

Градиент в новой точке $(-0.6, 2.6)$:

$$\nabla f(-0.6, 2.6) = (-6.8, -2.2)$$

Обновляем координаты:

$$x_1^{(2)} = -0.6 - 0.1 \times (-6.8) = 0.08, \quad x_2^{(2)} = 2.6 - 0.1 \times (-2.2) = 2.82$$

Вычисляем значение функции:

$$f(M_2) \approx -13.4748$$

Проверяем критерий остановки:

$$|f(M_2) - f(M_1)| = |-13.4748 - (-11.72)| = 1.7548 > 0.0001$$

Алгоритм продолжается.

Итерация 3

Градиент в новой точке $(0.08, 2.82)$:

$$\nabla f(0.08, 2.82) = (-1.86, -2.44)$$

Обновляем координаты:

$$x_1^{(3)} = 0.08 - 0.1 \times (-1.86) = 0.266, \quad x_2^{(3)} = 2.82 - 0.1 \times (-2.44) = 3.064$$

Вычисляем значение функции:

$$f(M_3) \approx -13.4504$$

Получили $f(M_3) > f(M_2)$, значит нужно уменьшить шаг.

$$\alpha = 0.05$$

Обновляем координаты:

$$x_1^{(3)} = 0.08 - 0.05 \times (-1.86) = 0.173, \quad x_2^{(3)} = 2.82 - 0.05 \times (-2.44) = 2.942$$

Вычисляем значение функции:

$$f(M_3') \approx -13.518$$

Проверяем критерий остановки:

$$|f(M_3') - f(M_2)| = |-13.518 - (-13.4748)| = 0.0432 > 0.0001$$

Алгоритм продолжается.

Код программы

```
import numpy as np


def f(x1, x2):
    return 6 * x1 ** 2 + x2 ** 2 - x1 * x2 + 4 * x1 - 8 * x2 + 1


def gradient(x1, x2):
    df_dx1 = 12 * x1 - x2 + 4
    df_dx2 = 2 * x2 - x1 - 8
    return np.array([df_dx1, df_dx2])


def gradient_descent(x1_init, x2_init, alpha=0.1, epsilon=0.0001, max_iter=100):
    x1, x2 = x1_init, x2_init
    prev_f = f(x1, x2)
    print(f"Начальная точка: ({x1}, {x2}), f = {prev_f}")

    for iteration in range(1, max_iter + 1):
        print(f"\nИтерация {iteration}")

        # Вычисляем градиент
        grad = gradient(x1, x2)
        print(f"Градиент: ({grad[0]}, {grad[1]})")

        # Обновляем координаты
        new_x1 = x1 - alpha * grad[0]
        new_x2 = x2 - alpha * grad[1]
        print(f"Новая точка: ({new_x1}, {new_x2})")

        # Вычисляем новое значение функции
        current_f = f(new_x1, new_x2)
        print(f"Значение функции: {current_f}")

        if current_f > prev_f:
            alpha /= 2
            continue

        # Проверка критерия остановки
        delta_f = abs(current_f - prev_f)
        delta_x = (new_x1 - x1) ** 2 + (new_x2 - x2) ** 2
        print(f"Разница значений функции: {delta_f}")
        print(f"Разница координат: {delta_x}")

        if delta_f < epsilon and delta_x < epsilon:
            print("\nКритерий остановки достигнут.")
            break

        # Обновляем значения для следующей итерации
```

```
        x1, x2 = new_x1, new_x2
        prev_f = current_f

    return x1, x2


# Начальная точка из примера
x1_init = 2
x2_init = 2

x1_opt, x2_opt = gradient_descent(x1_init, x2_init)

print("\nРезультат:")
print(f"Оптимальная точка: ({x1_opt}, {x2_opt})")
print(f"Значение функции: {f(x1_opt, x2_opt)}")
```

Метод наискорейшего спуска

Целевая функция:

$$f(x_1, x_2) = 6x_1^2 + x_2^2 - x_1x_2 + 4x_1 - 8x_2 + 1$$

Начальная точка:

$$M_0 = (2, 2)$$

Задаем точность $\varepsilon = 0.0001$

Итерация 1

Градиент функции:

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}\right)$$

Частные производные:

$$\frac{\partial f}{\partial x_1} = 12x_1 - x_2 + 4, \frac{\partial f}{\partial x_2} = 2x_2 - x_1 - 8.$$

Подставляем начальную точку $M_0 = (2, 2)$:

$$\frac{\partial f}{\partial x_1}(M_0) = 12(2) - 2 + 4 = 26, \frac{\partial f}{\partial x_2}(M_0) = 2(2) - 2 - 8 = -6.$$

Проверяем условие остановки:

$$\|\nabla f(M_0)\| = \sqrt{26^2 + (-6)^2} \approx 26.68 > \varepsilon.$$

Антиградиент:

$$S_0 = -\nabla f(M_0) = (-26, 6)$$

Обновляем координаты:

$$x_{11} = 2 - 26\lambda_1, x_{21} = 2 + 6\lambda_1.$$

Подставляем:

$$f(x_{11}, x_{21}) = 6(2 - 26\lambda_1)^2 + (2 + 6\lambda_1)^2 - (2 - 26\lambda_1)(2 + 6\lambda_1) + 4(2 - 26\lambda_1) - 8(2 + 6\lambda_1) + 1$$

$$= 17 - 712\lambda_1 + 4248\lambda_1^2.$$

Минимизируем по λ_1:

$$\lambda_1 = \frac{712}{8496} \approx 0.0838.$$

Обновляем точку:

$$M_1 = (2 - 26 \cdot 0.0838, 2 + 6 \cdot 0.0838) \approx (-0.179, 2.503).$$

Итерация 2

Вычисляем градиент в новой точке:

$$\frac{\partial f}{\partial x_1}(M_1) = 12(-0.179) - 2.503 + 4 = -0.651, \frac{\partial f}{\partial x_2}(M_1) = 2(2.503) - (-0.179) - 8 = -2.815.$$

Проверяем условие остановки:

$$\|\nabla f(M_1)\| = \sqrt{(-0.651)^2 + (-2.815)^2} \approx 2.89 > \varepsilon.$$

Антиградиент:

$$S_1 = (0.651, 2.815)$$

Обновляем координаты:

$$x_{12} = -0.179 + \lambda_2 \cdot 0.651, x_{22} = 2.503 + \lambda_2 \cdot 2.815.$$

Подставляем:

$$f(x_{12}, x_{22}) = 8.634\lambda_2^2 - 8.35\lambda_2 - 13.3347$$

Находим λ_2:

$$\lambda_2 \approx 0.0992.$$

Обновляем точку:

$$M_2 = (-0.179 + 0.0992 \cdot 0.651, 2.503 + 0.0992 \cdot 2.815) \approx (-0.114, 2.782).$$

Итерация 3

Вычисляем градиент:

$$\frac{\partial f}{\partial x_1}(M_2) = 12(-0.114) - 2.782 + 4 = -0.14, \frac{\partial f}{\partial x_2}(M_2) = 2(2.782) - (-0.114) - 8 = -2.322.$$

Проверяем условие остановки:

$$\|\nabla f(M_2)\| = \sqrt{(-0.14)^2 + (-2.322)^2} \approx 2.33 > \varepsilon.$$

Антиградиент:

$$S_2 = (0.14, 2.322)$$

Обновляем координаты:

$$x_{13} = -0.114 + \lambda_3 \cdot 0.14, x_{23} = 2.782 + \lambda_3 \cdot 2.322.$$

Подставляем:

$$f(x_{13}, x_{23}) = 5.184\lambda_3^2 - 5.435\lambda_3 - 13.677$$

Находим λ_3:

$$\lambda_3 \approx 0.0955.$$

Обновляем точку:

$$M_3 = (-0.114 + 0.0955 \cdot 0.14, 2.782 + 0.0955 \cdot 2.322) \approx (-0.101, 3.003).$$

Продолжаем процесс до достижения $\|\nabla f(M_k)\| < \varepsilon$.

Код программы

```
import numpy as np


def f(x1, x2):
    return 6 * x1 ** 2 + x2 ** 2 - x1 * x2 + 4 * x1 - 8 * x2 + 1


def grad_f(x1, x2):
    df_dx1 = 12 * x1 - x2 + 4
    df_dx2 = 2 * x2 - x1 - 8
    return np.array([df_dx1, df_dx2])


def golden_section_search(x, d, tol=1e-5):
    a, b = 0, 1  # Начальные границы поиска
    phi = (1 + np.sqrt(5)) / 2  # Золотое сечение
    resphi = 2 - phi
    c = a + resphi * (b - a)
    d = a + phi * (b - a)
    while abs(b - a) > tol:
        if f(*(x - c * d)) < f(*(x - d * d)):
            b = d
        else:
            a = c
        c = a + resphi * (b - a)
        d = a + phi * (b - a)
    return (a + b) / 2


def steepest_descent(x0, tol=1e-4, max_iter=100):
    x = np.array(x0, dtype=float)
    for i in range(max_iter):
        grad = grad_f(*x)
        if np.linalg.norm(grad) < tol:
            break
        d = -grad  # Направление антиградиента
        alpha = golden_section_search(x, d)  # Оптимальный шаг
        x += alpha * d  # Обновление точки
        print(f"Iteration {i + 1}: x = {x}, f(x) = {f(*x)}")
    return x


# Начальная точка
x0 = [2, 2]
opt_x = steepest_descent(x0)
```