МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

Департамент математического и компьютерного моделирования

ЛАБОРАТОРНАЯ РАБОТА №1
По основной образовательной программе подготовки бакалавров
направлению 01.03.02 Прикладная математика и информатика
профиль «Системное программирование»

Студент группы

Исаенко Евгений Александрович______
(подпись)
«_____» _________________ 2023 г.

Преподаватель ______________________
(должность, ученое звание)

____
____
(подпись) (ФИО)

«______» ________________2025 г.

г. Владивосток
2025

Постановка задачи:

Минимизировать функцию $f(x) = \frac{1}{2} \bullet x^T A x + b \cdot x$, где $x \in R^6$,

$A_{6\times6} -$ произвольная положительно определенная матрица, $A \in R^{n\times n}$

$b -$ произвольный ненулевой вектор размерности 6, $b \in R^{n\times 1}$

$x_0 -$ произвольный начальный ненулевой вектор размера 6, отдаленный от точного решения, $x \in R^{n\times 1}$

$$A = \begin{bmatrix} \\ 1.981 & 1.197 & 1.257 & 1.597 & 1.295 & 1.954 \\ 1.197 & 0.962 & 0.647 & 0.961 & 0.875 & 1.165 \\ 1.257 & 0.647 & 1.188 & 1.308 & 0.855 & 1.539 \\ 1.597 & 0.961 & 1.308 & 2.047 & 1.566 & 2.187 \\ 1.295 & 0.875 & 0.855 & 1.566 & 1.523 & 1.882 \\ 1.954 & 1.165 & 1.539 & 2.187 & 1.882 & 3.013 \end{bmatrix}$$

Проверка матрицы A, что она является положительно определённой.

$$b = \begin{bmatrix} 0.535 \\ 0.535 \\ 0.603 \\ 0.102 \\ 0.635 \\ 0.249 \\ 0.1 \end{bmatrix}$$

$$x_0 = \begin{bmatrix} 0.012 \\ 0.105 \\ 0.003 \\ 0.571 \\ 0.268 \\ 0.006 \end{bmatrix}$$

$x_* = - A^{-1} b,$

Метод градиента:

$x_{k+1} = x_k - \lambda f'(x_k)$, где $\lambda = 10^{-4}$

Первая производная функции: $f'(x) = \frac{1}{2}\left(A^T + A\right)x + b$.

Приравнивая производную к нулю, получаем вектор $x_{точ} \in R^{n\times 1}$.

$$x_{exact} = \begin{bmatrix} -0.25819402 \\ -1.09001505 \\ 1.92115774 \\ -2.53189952 \\ 2.02882 \\ 0.14505099 \end{bmatrix}$$

Алгоритм отработал за 131320 шагов. Условие выхода из цикла: $\|x_{k+1} - x_k\| < \xi = 10^{-5}$

Промежуточные результаты:

$$x_{\frac{m}{4}} = x_{\text{КШ}/4} = \begin{bmatrix} -0.20078213 \\ -0.62693422 \\ 0.30505736 \\ -0.586342 \\ 0.18702476 \\ 0.45997848 \end{bmatrix}$$

$$x_{\frac{m}{2}} = x_{\text{КШ}/2} = \begin{bmatrix} -0.12522822 \\ -0.83434712 \\ 0.55635759 \\ -1.10882805 \\ 0.44020368 \\ 0.61232217 \end{bmatrix}$$

$$x_{\frac{3m}{4}} = x_{3\text{КШ}/4} = \begin{bmatrix} -0.08537192 \\ -0.94532492 \\ 0.75589358 \\ -1.42354565 \\ 0.69747781 \\ 0.60346446 \end{bmatrix}$$

$$x_{m} = x_{\text{КШ}} = \begin{bmatrix} -0.07203739 \\ -1.01401618 \\ 0.9283046 \\ -1.63734926 \\ 0.92206637 \\ 0.55014213 \end{bmatrix}$$

Промежуточные значения функционала:

$$f\left(x_{\frac{m}{4}}\right) = f\left(x_{\text{КШ}/4}\right) = \text{-0.4976260398593633}$$

$$f\left(x_{\frac{m}{2}}\right) = f\left(x_{\text{КШ}/2}\right) = \text{-0.6464878383143928}$$

$$f\left(x_{\frac{3m}{4}}\right) = f\left(x_{3\text{КШ}/4}\right) = \text{-0.7140151051959476}$$

$$f\left(x_{m}\right) = f\left(x_{\text{КШ}}\right) = \text{-0.7549378452167426}$$

Значение функционала в точке x_*: $f\left(x_*\right) =$ -0.842900263326454 (точное решение)

Погрешности метода градиента:

$$\|x_{m1} - x_{ex1}\| = 0.186$$
$$\|x_{m2} - x_{ex2}\| = 0.076$$
$$\|x_{m3} - x_{ex3}\| = -0.993$$
$$\|x_{m4} - x_{ex4}\| = 0.895$$
$$\|x_{m5} - x_{ex5}\| = -1.107$$
$$\|x_{m6} - x_{ex6}\| = 0.405$$

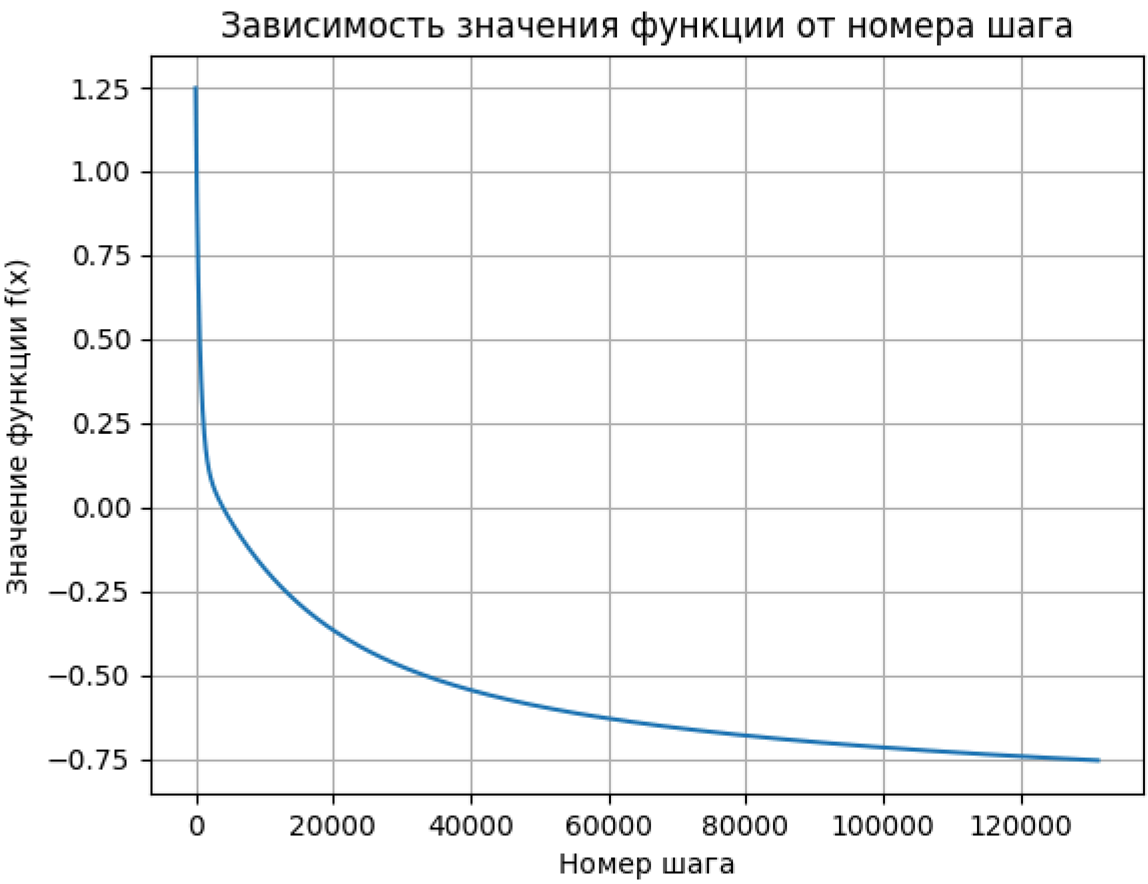

Рисунок 1 – График зависимости значения функции от номера шага методом градиентного спуска

Алгоритм содержится в приложении 1.

Приложения

```
import numpy as np
import matplotlib.pyplot as plt


def generate_positive_definite_matrix(size=6):
    A = np.random.rand(size, size)
    # симметричная, положительна определенная (A * A^T)
    A = np.dot(A, A.T)
    return A


def gradient(x, A, b):
    # градиент функции f(x)
    return 0.5 * (A.T + A) @ x + b


def function_value(x, A, b):
    # значения функции f(x)
    return 0.5 * np.dot(x.T, np.dot(A, x)) + np.dot(b, x)


def gradient_descent(A, b, x0, learning_rate=1e-4, tolerance=1e-5,
max_iterations=1000000):
    # метод градиентного спуска для минимизации функции f(x)
    x = x0
    f_values = [function_value(x, A, b)]  # Список для хранения значений
функции
    x_values = [x]
    for _ in range(max_iterations):
        grad = gradient(x, A, b)
        x_new = x - learning_rate * grad
        f_values.append(function_value(x_new, A, b))
        x_values.append(x_new)
```

```python
        # проверка на сходимость
        if np.linalg.norm(x_new - x) < tolerance:
            break

        x = x_new
    return x, f_values, x_values, len(f_values)

A = generate_positive_definite_matrix()
b = np.array([0.53453240, 0.60307323, 0.10238401, 0.63493003, 0.24934701,
0.10047803])
x0 = np.array([0.01247024, 0.10472905, 0.00346109, 0.57095621, 0.26825007,
0.00579221])

print('исходная матрица', np.around(A, 3))
print('b: ', np.around(b, 3))
print('x0: ', np.around(x0, 3))

x_exact = -2 * np.linalg.inv((A.T + A)) @ b
solution, function_values, arg_values, N = gradient_descent(A, b, x0)

print("x N:", solution)
print("(x*):", x_exact)
print("значение функционала в x*", function_value(x_exact, A, b))
print("Погрешность:", solution - x_exact)
f_dif = function_value(solution, A, b) - function_value(x_exact, A, b)
print("f_dif: ", f_dif)

# Вывод значений на определенных шагах
indices = [N // 4, N // 2, 3 * N // 4, N - 1]
for idx in indices:
    if idx < len(function_values):
```

```
        print(f"x({idx+1}): {arg_values[idx]}")

        print(f"f(x({idx+1})): {function_values[idx]}")


# Построение графика

plt.plot(range(len(function_values)), function_values)

plt.xlabel('Номер шага')

plt.ylabel('Значение функции f(x)')

plt.title('Зависимость значения функции от номера шага')

plt.grid()

plt.show()
```