МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Дальневосточный федеральный университет»
(ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

Кафедра информатики, математического и компьютерного моделирования

Лабораторная работа №1

Направление подготовки
01.03.02 «Прикладная математика и информатика»
профиль « Математическое и информационное обеспечение математической деятельности »

Выполнил студент
гр. Б9120-01.03.02
Агличеев А.О. ________
(ФИО) *(подпись)*

Проверил

____________ ________
(ФИО) *(подпись)*

« 18 » марта 2023 г.

г. Владивосток
2023

1 Постановка задачи

Минимизировать функцию $f(x) = \frac{1}{2} \cdot x^T A x + b \cdot x$

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

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

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

$$A = \begin{pmatrix} 3.5250663 & 3.47620458 & 3.48870342 & 3.25241547 & 3.13102997 & 2.8216509 \\ 3.47620458 & 3.67981631 & 3.58848911 & 3.3682241 & 3.07763483 & 2.89993938 \\ 3.48870342 & 3.58848911 & 3.64737011 & 3.35467961 & 3.04393178 & 2.84412883 \\ 3.25241547 & 3.3682241 & 3.35467961 & 3.22609546 & 2.86574917 & 2.71725572 \\ 3.13102997 & 3.07763483 & 3.04393178 & 2.86574917 & 2.87270201 & 2.5177417 \\ 2.8216509 & 2.89993938 & 2.84412883 & 2.71725572 & 2.5177417 & 2.34092264 \end{pmatrix}$$

$\lambda_1 = 18.8098291$, $\lambda_2 = 0.272692101$, $\lambda_3 = 0.00307129084$, $\lambda_4 = 0.0250115652$, $\lambda_5 = 0.103202992$, $\lambda_6 = 0.0781657514 \Rightarrow A$ - положительно определенная

$$b = \begin{pmatrix} 1.36886891 \\ 1.20398408 \\ 1.61577703 \\ 1.64312545 \\ 1.65507064 \\ 1.22275572 \end{pmatrix}$$

$$x_0 = \begin{pmatrix} -24.91430618 \\ -18.34836885 \\ 17.2160659 \\ -2.16716165 \\ 14.89750953 \\ 19.38128351 \end{pmatrix}$$

2 Метод градиента

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

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

Присваивая производную к нулю, получаем вектор $x_{\text{точ}} \in \mathbb{R}^{6\times 1}$

$$x_{\text{точ}} = \begin{pmatrix} 12.45715309 \\ 9.17418443 \\ -8.60803295 \\ 1.08358083 \\ -7.44875477 \\ -9.69064175 \end{pmatrix}$$

Алгоритм отработал за 135 906 шагов. Условие выхода из цикла: $||x_{k+1} - x_k|| < \xi = 10^{-5}$

Промежуточные результаты:

$$x_{\frac{m}{4}} = \begin{pmatrix} -9.05315651 \\ -5.74846129 \\ 8.15985564 \\ -11.06010065 \\ 0.20722274 \\ 20.2155591 \end{pmatrix}$$

$$x_{\frac{m}{2}} = \begin{pmatrix} -0.8997728 \\ -2.97659935 \\ 3.51196569 \\ -11.32658265 \\ -5.55490725 \\ 19.12635553 \end{pmatrix}$$

$$x_{\frac{3m}{2}} = \begin{pmatrix} 3.02589772 \\ -1.41248026 \\ 1.02621484 \\ -10.5377917 \\ -7.89978232 \\ 17.08988487 \end{pmatrix}$$

$$x_{m} = \begin{pmatrix} 5.07484822 \\ -0.22291392 \\ -0.47848709 \\ -9.56146077 \\ -8.77861289 \\ 14.78756135 \end{pmatrix}$$

Промежуточные значения функционала:

$f\left(x_{\frac{m}{4}}\right) = 2.8697312$

$f\left(x_{\frac{m}{2}}\right) = -1.17168854$

$f\left(x_{\frac{3m}{4}}\right) = -2.2177977$

$f\left(x_{m}\right) = -2.66331479$

Погрешности метода градиента:

$$x_{\text{точ}} = \begin{pmatrix} 12.45715309 \\ 9.17418443 \\ -8.60803295 \\ 1.08358083 \\ -7.44875477 \\ -9.69064175 \end{pmatrix} \quad x_m = \begin{pmatrix} 5.07484822 \\ -0.22291392 \\ -0.47848709 \\ -9.56146077 \\ -8.77861289 \\ 14.78756135 \end{pmatrix}$$

$|x_{m1} - x_{\text{точ}}| = 7.38230487$

$|x_{m2} - x_{\text{точ}}| = 9.39709835$

$|x_{m3} - x_{\text{точ}}| = 8.12954586$

$|x_{m4} - x_{\text{точ}}| = 10.6450416$

$|x_{m5} - x_{\text{точ}}| = 1.32985812$

$|x_{m6} - x_{\text{точ}}| = 24.4782031$

$|f(x_m) - f(x_{\text{точ}})| = 1.44064724$

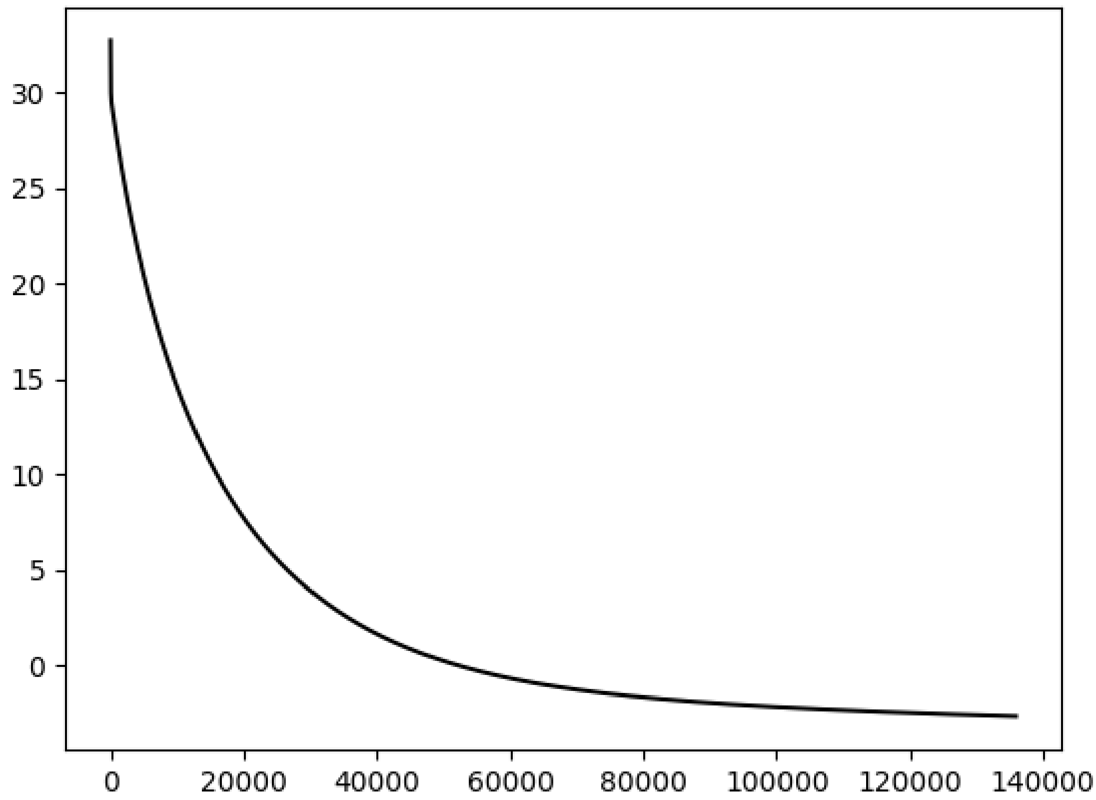

Рис. 1: График зависимости значения функции от номера шага методом градиентного спуска

Алгоритм содержится в приложении 1.

3 Приложения

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


def generate_pd_matrix(n: int, m: int) -> np.ndarray:
  matrix = np.random.uniform(0.5, 1, (n, m))
  return np.matmul(matrix, matrix.transpose())
```

```
def f(x: np.ndarray, a: np.ndarray, b: np.ndarray) -> np.ndarray:
  return .5 * x.transpose() @ a @ x + b.transpose() @ x


def f_derivative(a: np.ndarray, b: np.ndarray, x: np.ndarray) -> np.ndarray:
  return .5 * np.matmul(np.add(a.transpose(), a), x) + b


def gradient_descent(a: np.ndarray, b: np.ndarray, initial_x: np.ndarray, h
   =10e-4, precision=10e-5):
  counter = 0
  res_tuples = []
  error = float('inf')
  prev = None
  current = initial_x
  while error >= precision:
    prev = current
    current = np.subtract(prev, h * f_derivative(a, b, prev))
    error = np.linalg.norm(np.subtract(current, prev), 'fro')
    counter += 1
    res_tuples.append((counter, current))
  return res_tuples


if __name__ == '__main__':
  matrix_a = np.loadtxt("matrix_a.txt", usecols=range(6))
  vector_b = np.loadtxt("vector_b.txt", usecols=range(1), ndmin=2)
  x_sol = np.linalg.solve(.5 * np.add(matrix_a.transpose(), matrix_a), -
   vector_b)
  vector_x0 = x_sol * -2

  ans = gradient_descent(matrix_a, vector_b, vector_x0)
  f_values = np.ravel([f(i[1], matrix_a, vector_b).flatten() for i in ans]).
   tolist()
  plt.plot([i[0] for i in ans], f_values, 'black')
  plt.savefig('fig1.png')
```