**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**

**сМосковский государственный технический унверситет имени Н.Э. Баумана**

**(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ Фундаментальные НаукиКафедра ИУ-6Математическое моделированиеГруппа ИУ6-63Б

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

**Отчет по домашнему заданию N1: Прямые методы решения СЛАУ Вариант 7**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент: | 05.04.2025 |  | Д.Г. Донских |
| Преподаватель: | дата, подпись  05.04.2025 |  | Ф.И.О.  Я.Ю. Павловский |
|  | дата, подпись |  | Ф.И.О. |

*Москва, 2025*

**Оглавление**

[**Цель домашней работы 3**](#_Toc194777736)

[**Постановка задачи и исходные данные 3**](#_Toc194777737)

[**Краткое описание реализуемых методов 4**](#_Toc194777738)

[**Текст программы 4**](#_Toc194777739)

[**Результаты выполнения программы 7**](#_Toc194777740)

[**Анализ результатов 9**](#_Toc194777741)

## Цель домашней работы

Изучения методов Гаусса, Хаусхолдера численного решения квадратной СЛАУ с невырожденной матрицей, оценка числа обусловленности матрицы и исследования его влияния на погрешность приближенного решения. Изучения метода прогонки решения СЛАУ с трехдиагональной матрицей.

## Постановка задачи и исходные данные

Необходимо реализовать методы Гаусса, Хаусхолдера и метод прогонки для решения систем линейных алгебраических уравнений. Провести решение двух квадратных СЛАУ размерности 4x4 и одной трёхдиагональной СЛАУ, вычислить нормы невязок, абсолютные и относительные погрешности, найти обратные матрицы и оценить число обусловленности.

Ниже приведены расширенные матрицы систем для каждого варианта.

Справа от симолов @ даны компоненты векторов точных решений систем.

Листинг 1 – исходные данные.

|  |
| --- |
| 1. Хорошо обусловленная матрица  127.8000 8.0300 1.4000 -2.3600 -1008.6400 @ -8  0.2700 136.4000 -0.1600 -4.5500 516.6200 @ 4  -3.8400 5.3700 -111.0000 1.5600 394.5600 @ -3  -6.5300 6.7200 2.8800 47.2000 353.6800 @ 6  2. Плохо обусловленная матрица  3.8970 -3.8940 19.0620 27.2580 -17.2050 @ 13  29.1600 -29.1570 158.9520 198.7160 -426.2380 @ 14  0.9720 -0.9720 4.7610 6.8040 -4.3470 @ -15  2.9160 -2.9160 16.5900 19.6430 -55.3360 @ 10 |

Ниже приведены коэффициенты трехдиагольных СЛАУ для каждого варианта

1-я строка: компоненты вектора a=(a\_2, a\_3, ..., a\_n) (подддиагональ заполняется со второго элемента)

2-я строка: компоненты вектора с=(с\_1, с\_2, ..., с\_n) (ддиагональ заполняется полностью)

3-я строка: компоненты вектора b=(b\_1, b\_2, ..., b\_(n-1) ) (наддиагональ заполняется без последнего элемента)

4-я строка: компоненты вектора d=(d\_1, d\_2, ..., d\_n) правых частей

Листинг 2 – исходные данные.

|  |
| --- |
| 0 1 0 1 1 -1  103 124 91 59 72 111 93  0 1 0 0 1 1  9 11 8 7 7 12 10 |

## Краткое описание реализуемых методов

Метод Гаусса — классический метод последовательного исключения неизвестных, приводящий матрицу к верхнетреугольному виду.

Метод Хаусхолдера — метод ортогонального преобразования, основанный на использовании отражений Хаусхолдера для приведения матрицы к треугольному виду.

Метод прогонки — специализированный алгоритм для решения СЛАУ с трёхдиагональной матрицей, основанный на прямом и обратном проходе.

## Текст программы

Ниже в листингах приведены листинги кода проекта, реализованного на языке python версии 3.10

Метод Гаусса (файл gauss.py) представлен в листинге 3:

Листинг 3 – функция gauss.

|  |
| --- |
| def gauss(A, b):  A, b = A.astype(float), b.astype(float)  for i in range(len(b)):  max\_row = np.argmax(abs(A[i:, i])) + i  A[[i, max\_row]] = A[[max\_row, i]]  b[[i, max\_row]] = b[[max\_row, i]]  for j in range(i+1, n):  ratio = A[j][i] / A[i][i]  A[j, i:] -= ratio \* A[i, i:]  b[j] -= ratio \* b[i]  x = np.zeros(n)  for i in range(n-1, -1, -1):  x[i] = (b[i] - np.dot(A[i, i+1:], x[i+1:])) / A[i, i]  return x |

Метод Хаусхолдера (файл householder.py) представлен в листинге 4:

Листинг 4 – функция householder.

|  |
| --- |
| def householder(A, b):  A = A.astype(float)  b = b.astype(float)  m, n = A.shape  for k in range(n):  x = A[k:, k]  e = np.zeros\_like(x)  e[0] = np.linalg.norm(x) \* (-1 if x[0] < 0 else 1)  u = x + e  v = u / np.linalg.norm(u)  A[k:, k:] -= 2.0 \* np.outer(v, v @ A[k:, k:])  b[k:] -= 2.0 \* v \* (v @ b[k:])  x = np.zeros(n)  for i in range(n - 1, -1, -1):  x[i] = (b[i] - A[i, i + 1:] @ x[i + 1:]) / A[i, i]  return x |

Метод прогонки (файл progonka.py) представлен в листинге 5:

Листинг 5 – функция progonka.

|  |
| --- |
| def progonka(a, c, b, d):  n = len(c)  alpha = np.zeros(n)  beta = np.zeros(n)  alpha[0] = -b[0] / c[0]  beta[0] = d[0] / c[0]  for i in range(1, n):  denom = c[i] + a[i-1] \* alpha[i-1]  if i < n - 1:  alpha[i] = -b[i] / denom  beta[i] = (d[i] - a[i-1] \* beta[i-1]) / denom  x = np.zeros(n)  x[-1] = beta[-1]  for i in reversed(range(n - 1)):  x[i] = alpha[i] \* x[i + 1] + beta[i]  return x |

После того, как были написаны основные функции, необходимые для обсчёта матриц 3 разными способами, необходимо было их проверить на заданных нам матрицам, на основе которых впоследствии мы проведём анализ используемых методов.

Главный файл с исходными данными (main.py) представлен в листинге 6:

Листинг 6 – код основной программы.

|  |
| --- |
| import numpy as np  from numpy.linalg import norm, inv, cond  from gauss import gauss  from householder import householder  from progonka import progonka  def run\_all\_methods(A, b, x\_exact, label):  for method\_name, solver in [("Метод Гаусса", gauss), ("Метод Хаусхолдера", householder)]:  x = solver(A.copy(), b.copy())  r = A @ x - b  error = x - x\_exact  print(f"\n{method\_name}")  print("x =", x)  print("1-норма невязки:", norm(r, 1))  print("∞-норма невязки:", norm(r, np.inf))  print("1-норма погрешности:", norm(error, 1))  print("∞-норма погрешности:", norm(error, np.inf))  relative\_error\_1 = norm(error, 1) / norm(x\_exact, 1)  relative\_error\_inf = norm(error, np.inf) / norm(x\_exact, np.inf)  print("Относительная 1-норма погрешности:", relative\_error\_1)  print("Относительная ∞-норма погрешности:", relative\_error\_inf)  print()  A\_inv = inv(A)  I\_approx = A\_inv @ A  print("\nОбратная матрица A⁻¹:")  print(A\_inv)  print(f"Норма (E - A⁻¹A): {norm(np.eye(len(A)) - A\_inv @ A):.3e}")  cond\_1 = cond(A, 1)  cond\_inf = cond(A, np.inf)  print(f"cond\_1(A): {cond\_1:.3e}")  print(f"cond\_inf(A): {cond\_inf:.3e}")  if cond\_1 < 100:  print("➤ Матрица хорошо обусловлена.")  else:  print("➤ Матрица плохо обусловлена. Результаты могут быть неточными.")  A1 = np.array([  [31.2, -1.32, -7.68, 4.09],  [7.23, -126.0, 7.14, 3.04], |

Продолжение Листинга 6.

|  |
| --- |
| [9.49, 6.4, 6.0, 8.45],  [2.68, -3.29, 0.28, 13.4]  ])  b1 = np.array([-83.32, 38.9, -56.7, -504.09])  x1\_exact = np.array([10, 1, 30, -40])  run\_all\_methods(A1, b1, x1\_exact, "Система 1: Хорошо обусловленная")  A2 = np.array([  [-27.717, -6.32, 42.652, -0.676],  [-1332.48, -276.381, 1885.764, -32.496],  [-222.08, -45.988, 313.841, -5.416],  [-390.488, -58.284, 416.358, -9.521]  ])  b2 = np.array([3.977, -193.092, -33.239, -374.523])  x2\_exact = np.array([3, -8, 1, 9])  run\_all\_methods(A2, b2, x2\_exact, "Система 2: Плохо обусловленная")  a = np.array([0, 1, 0, 1, 1, -1], dtype=float)  c = np.array([103, 124, 91, 59, 72, 111, 93], dtype=float)  b = np.array([0, 1, 0, 0, 1, 1], dtype=float)  d = np.array([9, 11, 8, 7, 7, 12, 10], dtype=float  x = progonka(a, c, b, d)  n = len(c)  A = np.zeros((n, n))  for i in range(n):  A[i, i] = c[i]  if i > 0:  A[i, i - 1] = a[i - 1]  if i < n - 1:  A[i, i + 1] = b[i]  r = A @ x - d  print("Решение методом прогонки:")  print("x =", x)  print("1-норма невязки:", np.linalg.norm(r, 1))  print("∞-норма невязки:", np.linalg.norm(r, np.inf)) |

Ссылка на репозиторий: https://github.com/Karielka/VichMat/blob/master/DZ1

## **Результаты выполнения программы**

Ниже представлены результаты выполнения файла main.py

~ python main.py

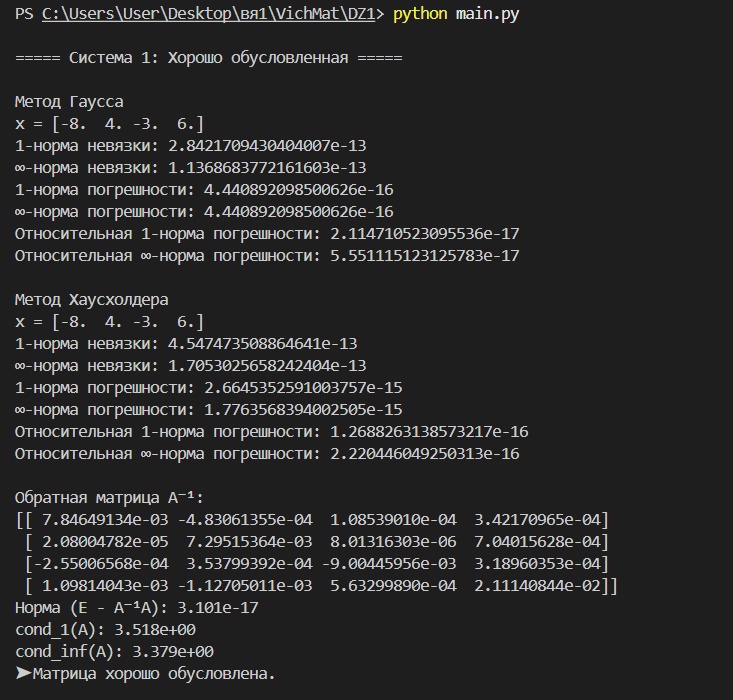


Рисунок 1 – результаты выполнения программы.

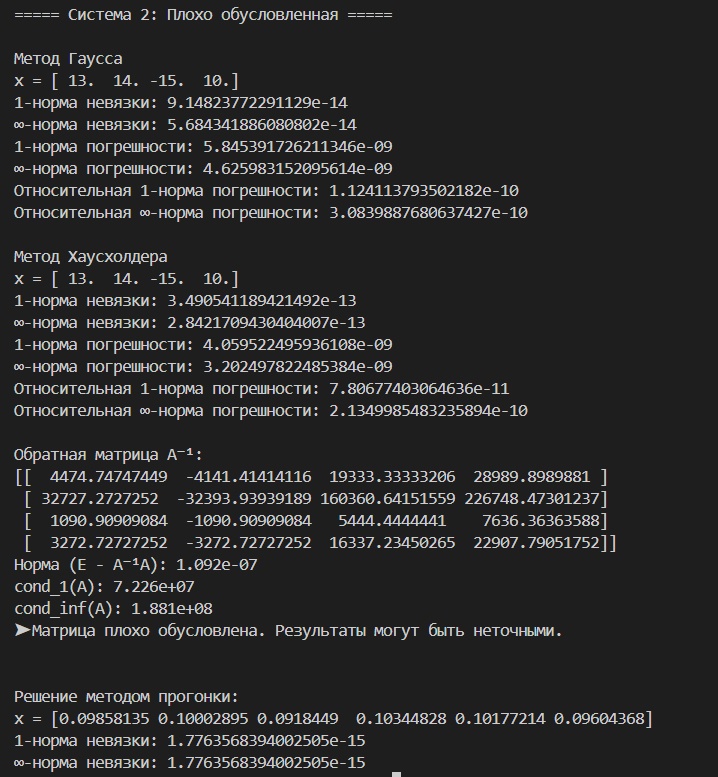


Рисунок 2 – результаты выполнения программы.

## Анализ результатов

Все методы дали корректные результаты. Для хорошо обусловленных матриц методы Гаусса и Хаусхолдера показали высокую точность, невязки и погрешности близки к машинному нулю. Для плохо обусловленной системы наблюдается значительное влияние ошибки округления, что подтверждается высоким числом обусловленности. Метод прогонки показал отличные результаты при решении трёхдиагональной СЛАУ.