Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
сМосковский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ	Фундаментальные Науки
Кафедра	ИУ-6 Математическое моделирование
Группа	ИУ6-63Б

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

Отчет по домашнему заданию N1:

Прямые методы решения СЛАУ

Вариант 7

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

Москва, 2025

Оглавление

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 c=(c_1, c_2, ..., c_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

```
PS C:\Users\User\Desktop\вя1\VichMat\DZ1> python main.py

===== Система 1: Хорошо обусловленная =====

Метод Гаусса
x = [-8.  4. -3.  6.]
1-норма невязки: 2.8421709430404007e-13
∞-норма невязки: 1.1368683772161603e-13
1-норма погрешности: 4.440892098500626e-16
∞-норма погрешности: 4.440892098500626e-16
Относительная 1-норма погрешности: 2.114710523095536e-17
Относительная ∞-норма погрешности: 5.551115123125783e-17

Метод Хаусхолдера
x = [-8.  4. -3.  6.]
1-норма невязки: 4.547473508864641e-13
∞-норма невязки: 1.7053025658242404e-13
1-норма погрешности: 2.6645352591003757e-15
∞-норма погрешности: 1.7763568394002505e-15
Относительная 1-норма погрешности: 1.2688263138573217e-16
Относительная ∞-норма погрешности: 2.220446049250313e-16

Обратная матрица A⁻¹:
[[ 7.84649134e-03 -4.83061355e-04  1.08539010e-04  3.42170965e-04]
 [ 2.08004782e-05  7.29515364e-03  8.01316303e-06  7.04015628e-04]
 [-2.55006568e-04  3.53799392e-04 -9.00445956e-03  3.18960353e-04]
 [ 1.09814043e-03 -1.12705011e-03  5.63299890e-04  2.11140844e-02]]
Норма (E - A⁻¹A): 3.101e-17
cond_1(A): 3.518e+00
cond_inf(A): 3.379e+00
➤Матрица хорошо обусловлена.
```

Рисунок 1 – результаты выполнения программы.

```
===== Система 2: Плохо обусловленная =====

Метод Гаусса
x = [ 13.  14. -15.  10.]
1-норма невязки: 9.14823772291129e-14
∞-норма невязки: 5.684341886080802e-14
1-норма погрешности: 5.845391726211346e-09
∞-норма погрешности: 4.625983152095614e-09
Относительная 1-норма погрешности: 1.124113793502182e-10
Относительная ∞-норма погрешности: 3.0839887680637427e-10

Метод Хаусхолдера
x = [ 13.  14. -15.  10.]
1-норма невязки: 3.490541189421492e-13
∞-норма невязки: 2.8421709430404007e-13
1-норма погрешности: 4.059522495936108e-09
∞-норма погрешности: 3.202497822485384e-09
Относительная 1-норма погрешности: 7.80677403064636e-11
Относительная ∞-норма погрешности: 2.1349985483235894e-10

Обратная матрица A⁻¹:
[[  4474.74747449  -4141.41414116  19333.33333206  28989.8989881 ]
 [ 32727.2727252  -32393.93939189 160360.64151559 226748.47301237]
 [  1090.90909084  -1090.90909084   5444.4444441    7636.36363588]
 [  3272.72727252  -3272.72727252  16337.23450265  22907.79051752]]
Норма (E - A⁻¹A): 1.092e-07
cond_1(A): 7.226e+07
cond_inf(A): 1.881e+08
➤Матрица плохо обусловлена. Результаты могут быть неточными.


Решение методом прогонки:
x = [0.09858135 0.10002895 0.0918449  0.10344828 0.10177214 0.09604368]
1-норма невязки: 1.7763568394002505e-15
∞-норма невязки: 1.7763568394002505e-15
```

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.