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

ФАКУЛЬТЕТ	Информатика и системы управления
Кафедра	ИУ6 Компьютерные системы и сети
Группа	ИУ6-63Б

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

Отчет по домашнему заданию N1:

Прямые методы решения СЛАУ

Вариант 10

Студент:		В.К. Залыгин
	дата, подпись	Ф.И.О.
Преподаватель:		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 символов @ даны компоненты векторов точных решений систем.

Листинг 1 – исходные данные

```
1. Хорошо обусловленная матрица
     80.0000        -4.3200        -5.0400         3.6900       584.4800    @  7
     -6.7000       149.8000        -6.7600        -1.1300     -1258.8200    @ -8
     -4.5200         1.6900       -84.8000         0.0000      -214.7600    @  2
     -8.5800         5.5200         9.1800        65.6000       -85.8600    @  0
 2. Плохо обусловленная матрица
     -0.3100         0.1560         0.1320        -0.0180         0.1700    @ 1
     -0.6180         0.3110         0.2640        -0.0360         0.3400    @ 2
     19.3000        -9.6500        -8.3780         1.1970        -9.5720    @ 4
    141.5160       -70.7580       -61.4460         8.7790       -70.2040    @ 20
```

Ниже приведены коэффициенты трехдиагольных СЛАУ для каждого варианта

1-я строка: компоненты вектора a=(a_2, a_3, ..., a_n) (поддиагональ 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    -1     1     1
     86    93   171    59    74   105
     1     1    -1     0     1
     9     9    16     6     7    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([
    [80.0, -4.32, -5.04, 3.69],
    [-6.7, -149.8, -6.76, -1.13],
    [-4.52, 1.69, -84.8, 0],
    [-8.58, -5.52, 9.18, 65.6]
])
b1 = np.array([-584.48, -1258.82, -214.76, -85.86])
x1_exact = np.array([7, -8, 2, 0])

run_all_methods(A1, b1, x1_exact, "Система 1: Хорошо обусловленная")
A2 = np.array([
    [-0.31, -0.1560, 0.1320, -0.0180],
    [-0.6180, -0.3110, 0.2640, -0.036],
    [-19.3000, -9.6500, -8.3780, -1.1970],
    [-141.5160, -70.7580, -61.4460, -8.779]
])
b2 = np.array([0.17, -0.34, -9.572, -70.2040])
x2_exact = np.array([1, 2, 4, 20])
run_all_methods(A2, b2, x2_exact, "Система 2: Плохо обусловленная")
a = np.array([0,  1, -1,  1,  1], dtype=float)  # длина = n-1
c = np.array([86,  93,  171,   59,   74,  105], dtype=float)  #
длина = n
b = np.array([1,  1, -1,  0,  1], dtype=float)  # длина = n-1
d = np.array([9,   9,  16,   6,   7,  10], dtype=float)  # длина = n
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))
```

Результаты выполнения программы

Ниже представлены результаты выполнения файла main.py: python main.py.

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

Метод Гаусса
x = [-6.56428095  8.57327905  3.05329471 -1.8732664 ]
1-норма невязки: 3.552713678800501e-13
∞-норма невязки: 2.2737367544323206e-13
1-норма погрешности: 33.0641211071309
∞-норма погрешности: 16.57327904647579
Относительная 1-норма погрешности: 1.9449483004194648
Относительная ∞-норма погрешности: 2.0716598808094737

Метод Хаусхолдера
x = [-6.56428095  8.57327905  3.05329471 -1.8732664 ]
1-норма невязки: 5.258016244624741e-13
∞-норма невязки: 2.2737367544323206e-13
1-норма погрешности: 33.06412110713091
∞-норма погрешности: 16.57327904647579
Относительная 1-норма погрешности: 1.9449483004194652
Относительная ∞-норма погрешности: 2.0716598808094737

Обратная матрица A⁻¹:
[[ 1.23522075e-02 -3.39232981e-04 -7.82947229e-04 -7.00655164e-04]
 [-5.34829956e-04 -6.65079396e-03  5.52822944e-04 -8.44797963e-05]
 [-6.69054721e-04 -1.14463546e-04 -1.17397029e-02  3.56626237e-05]
 [ 1.66420123e-03 -5.87991254e-04  1.58695683e-03  1.51401626e-02]]
Норма (E - A⁻¹A): 2.926e-17
cond_1(A): 2.575e+00
cond_inf(A): 3.120e+00
➤Матрица хорошо обусловлена.
```

Рисунок 1 – Результаты выполнения программы

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

Метод Гаусса
x = [ 4386946.60001398 -8774573.20002797   -34036.80000011   243719.20000076]
1-норма невязки: 1.6706058675319824e-08
∞-норма невязки: 1.1801720489756917e-08
1-норма погрешности: 13439260.800042817
∞-норма погрешности: 8774575.200027967
Относительная 1-норма погрешности: 497750.40000158583
Относительная ∞-норма погрешности: 438728.76000139833

Метод Хаусхолдера
x = [ 4386947.00070623 -8774574.00147242   -34036.80310885   243719.22224267]
1-норма невязки: 6.079486933319167e-08
∞-норма невязки: 2.949498592585975e-08
1-норма погрешности: 13439262.027530173
∞-норма погрешности: 8774576.001472423
Относительная 1-норма погрешности: 497750.4454640805
Относительная ∞-норма погрешности: 438728.80007362115

Обратная матрица A⁻¹:
[[ 1.29043400e+07 -6.45192000e+06 -2.83107000e+06  3.86010000e+05]
 [-2.58106800e+07  1.29048400e+07  5.66214000e+06 -7.72019999e+05]
 [-1.00120000e+05  5.00600000e+04  2.19475000e+04 -2.99250000e+03]
 [ 7.16879999e+05 -3.58440000e+05 -1.53615000e+05  2.09450000e+04]]
Норма (E - A⁻¹A): 1.574e-08
cond_1(A): 6.394e+09
cond_inf(A): 1.275e+10
➤Матрица плохо обусловлена. Результаты могут быть неточными.
```

Рисунок 2 – Результаты выполнения программы

```
Решение методом прогонки:
x = [0.10353759 0.09576762 0.09361119 0.10328155 0.09192373 0.09436263]
1-норма невязки: 2.6645352591003757e-15
∞-норма невязки: 1.7763568394002505e-15
```

Рисунок 3 – 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.