



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

ФАКУЛЬТЕТ «Робототехники и комплексной автоматизации»  
КАФЕДРА «Системы автоматизированного проектирования (РК-6)»

## ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №4 по дисциплине «Вычислительная математика»

Студент:	Меньшикова Елизавета Андреевна
Группа:	РК6-51Б
Тип задания:	лабораторная работа
Тема:	Устойчивость прямых методов решения СЛАУ

Студент

\_\_\_\_\_  
подпись, дата

Меньшикова Е.А.  
Фамилия, И.О.

Преподаватель

\_\_\_\_\_  
подпись, дата

\_\_\_\_\_  
Фамилия, И.О.

Москва, 2023

# Содержание

<b>Устойчивость прямых методов решения СЛАУ</b>	<b>3</b>
Задание . . . . .	3
Цель выполнения лабораторной работы . . . . .	4
1 Решение СЛАУ с помощью метода Гаусса . . . . .	4
2 Решение СЛАУ с помощью метода прогонки . . . . .	6
3 Выбор "универсального" метода . . . . .	9
4 Генерация случайных невырожденных матриц . . . . .	9
5 Проведение эксперимента с большим количеством случайных матриц . .	10
6 Расширение генератора невырожденных матриц . . . . .	13
7 Решение СЛАУ с помощью разложения Холецкого . . . . .	13
8 Анализ методов, учитывая разложение Холецкого . . . . .	16
9 Распределение спектральных радиусов и чисел обусловленности . . . . .	17
10 Влияние спектрального радиуса на вычислительную устойчивость алго- ритма . . . . .	19
11 Влияние отношения максимального по модулю собственного числа к ми- нимальному по модулю собственному числу на вычислительную устой- чивость алгоритма . . . . .	19
12 Влияние числа обусловленности на вычислительную устойчивость алго- ритма . . . . .	20
Заключение . . . . .	21

# Устойчивость прямых методов решения СЛАУ

## Задание

Решение систем линейных алгебраических уравнений (СЛАУ) – ключевой этап огромного множества задач вычислительной математики и анализа данных. В случае плотных матриц сравнительно небольшой размерности, для нахождения решения СЛАУ часто применяются прямые методы, такие как метод Гаусса, метод прогонки или разложение Холецкого. В то же время известно, что многие прямые методы обладают вычислительной неустойчивостью и могут приводить к некорректному решению для некоторых матриц коэффициентов. В этой лабораторной работе рассматриваются матрицы нескольких видов и с помощью генерации большого количества случайных матриц демонстрируется наличие или отсутствие вычислительной неустойчивости у метода Гаусса, метода прогонки и разложения Холецкого.

### Требуется (базовая часть):

1. Написать функцию `gauss(A, b, pivoting)`, которая возвращает решение СЛАУ  $Ax = b$ , полученное с помощью метода Гаусса. Если параметр `pivoting=True`, то решение должно находиться с частичным выбором главного элемента. Если `pivoting=False`, то выбора главного элемента происходить не должно.

2. Написать функцию `thomas(A, b)`, которая возвращает решение СЛАУ  $Ax = b$ , полученное с помощью метода прогонки.

3. Среди реализованных методов, включая два варианта метода Гаусса, выбрать тот, который минимизирует вычислительные погрешности для случая квадратных матриц общего вида. В рамках задания такой метод будем называть "универсальным".

4. Разработать и описать алгоритм генерации случайных невырожденных матриц размерности  $6 \times 6$  с элементами  $a_{ij} \in \mathbb{R}$ ,  $|a_{ij}| < 1$  общего и 3-х диагонального вида.

5. Сгенерировав 1000 случайных матриц  $A^{(j)}$  каждого типа с 32-битным float представлением элементов необходимо провести следующий эксперимент:

- (a) Выбрать "специальный" вычислительно-эффективный метод по типу матрицы.
- (b) Для каждой СЛАУ  $A^{(j)} = [1, 1, 1, 1]^T$  найти решение с помощью "универсального" и "специального" методов, а затем найти относительную погрешность вычислений с помощью среднеквадратичной и супремум-нормы. Вывести на экран распределения погрешностей в виде гистограмм.

(c) Является ли выбранный "специальный" метод вычислительно устойчивым? Почему?

### Требуется (продвинутая часть):

1. Расширить генератор из задания 4 для получения положительно-определенных матриц.

2. Написать функцию `cholesky(A, b)`, которая возвращает решение СЛАУ  $Ax = b$ , полученное с помощью разложения Холецкого.

3. Провести требуемый в задании 5 анализ учитывая метод Холецкого (сравнить с

"универсальным" методом).

4. Для всех рассмотренных ранее матриц вывести на экран распределение спектральных радиусов и распределение чисел обусловленности в виде гистограмм. Сделать вывод.

5. Влияет ли значение спектрального радиуса матрицы на вычислительную устойчивость рассмотренных алгоритмов? Если да, то как?

6. Влияет ли значение отношения максимального по модулю собственного числа к минимальному по модулю собственному числу на вычислительную устойчивость алгоритма? Если да, то как?

7. Влияет ли число обусловленности на вычислительную устойчивость алгоритма? Если да, то как?

## Цель выполнения лабораторной работы

Цель выполнения лабораторной работы: исследование прямых методов решения системы линейных алгебраических уравнений таких как метод Гаусса, метод прогонки и разложение Холецкого, анализ их вычислительной неустойчивости при работе с матрицами коэффициентов определенных видов.

## 1 Решение СЛАУ с помощью метода Гаусса

Система линейных алгебраических уравнений в матричном виде представлена в (1).

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}, \quad (1)$$

где  $\mathbf{A} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{x} \in \mathbb{R}^n$ ,  $\mathbf{b} \in \mathbb{R}^n$  и  $n$  – количество уравнений в системе. В кратком виде СЛАУ можно записать так:

$$\mathbf{Ax} = \mathbf{b}. \quad (2)$$

В лабораторной работе рассматриваются прямые методы решения СЛАУ, то есть методы, дающие точное решение. Одним из таких методов является метод Гаусса, также имеющий название метода последовательного исключения. Метод подразделяется на две стадии: прямой и обратный ходы. Во время прямого хода матрица  $\mathbf{A}$  приводится к треугольному виду с помощью элементарных преобразований. В обратный ход находится решение  $\mathbf{x}$  с помощью рекурсивной подстановки.

В начале метода необходимо рассмотреть расширенную матрицу вида:

$$\tilde{\mathbf{A}} = [\mathbf{A}, \mathbf{b}] = \left[ \begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} & b_n \end{array} \right].$$

Приведение матрицы  $\mathbf{A}$  к треугольному виду включает в себя последовательное обнуление элементов, находящихся под главной диагональю. Осуществить это можно путем элементарного преобразования (3).

$$(i) \longrightarrow (i) - \frac{a_{ik}}{a_{kk}}(k), \quad i = k + 1, \dots, n \quad (3)$$

где  $(i)$  обозначает  $i$ -тую строку,  $k$  – номер итерации.

Обратный ход метода Гаусса осуществляется с помощью формул:

$$\begin{aligned} x_n &= \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}}, \\ x_{n-1} &= \frac{b_{n-1}^{(n-2)} - a_{n-1,n}^{(n-2)} x_n}{a_{n-1,n-1}^{(n-2)}}, \\ &\dots \\ x_1 &= \frac{b_1 - \sum_{i=2}^n a_{1i} x_i}{a_{11}}. \end{aligned}$$

При приведении матрицы  $\mathbf{A}$  к треугольному виду необходимо вычислять значение множителя  $m_{ik} = \frac{a_{ik}}{a_{kk}}$  из выражения (3), то есть делить все элементы столбца, находящиеся под диагональным элементом, на соответствующий диагональный элемент. Если диагональный элемент будет равен нулю, найти множитель будет невозможно ввиду необходимости деления на ноль. Эта проблема может быть решена перестановкой строк матрицы  $\mathbf{A}$ . Если  $a_{kk} \ll a_{ik}$ , то значение  $m_{ik}$  будет очень велико, вследствие чего значительно вырастет погрешность округления ввиду многократно используемого в умножении  $m_{ik}$  при приведении матрицы к треугольному виду. Также деление на малый элемент  $a_{kk}$  в обратном ходе метода Гаусса может привести к росту накопленных погрешностей. Избежать описанные проблемы можно путем перестановки строк матрицы  $\mathbf{A}$  таким образом, чтобы диагональным элементом  $a_{kk}$  являлся наибольший по модулю элемент  $k$ -того столбца.

$$|a_{kk}^{(k)}| = \max_{k \leq i \leq n} |a_{ik}^{(k)}|. \quad (4)$$

Описанный алгоритм называется частичным выбором главного элемента.

В листинге 1 представлена функция `gauss(A, b, pivoting)`, которая возвращает решение СЛАУ (2), полученное с помощью метода Гаусса. Функция принимает в качестве параметров матрицу  $\mathbf{A}$ , столбец свободных коэффициентов  $\mathbf{b}$ , а также параметр `pivoting`. Если `pivoting = True`, то решение должно находиться с частичным выбором главного элемента по формуле (4). Если `pivoting = False`, то выбора главного элемента происходить не должно.

Листинг 1 – Функция `gauss(A, b, pivoting)`, возвращающая решение СЛАУ (1), полученное с помощью метода Гаусса.

---

```

def gauss(A, b, pivoting=False):
    A = np.copy(A)
    b = np.copy(b)
    n = len(b)

    for i in range(0, n - 1):
        if pivoting:
            max_i = np.argmax(np.abs(A[:, i][i:])) + i

            b[[i, max_i]] = b[[max_i, i]]
            A[[i, max_i]] = A[[max_i, i]]

        for j in range(i + 1, n):
            b[j] = b[j] - (A[j, i] / A[i, i]) * b[i]
            A[j] = A[j] - (A[j, i] / A[i, i]) * A[i]

    x = np.zeros([n])

    for i in range(n - 1, -1, -1):
        x[i] = (b[i] - np.sum(A[i][i + 1:] * x[i + 1:])) / A[i, i]

    return x[:n]

```

---

## 2 Решение СЛАУ с помощью метода прогонки

Перед рассмотрением метода прогонки необходимо обратиться к понятию ленточных и трехдиагональных матриц. Квадратная матрица  $A \in \mathbb{R}^{n \times n}$  называется ленточной, если существуют такие  $p, q \in \mathbb{Z}$ , где  $1 < p, q < n$ , что  $j - i \geq p \implies a_{ij} = 0$  и  $i - j \geq q \implies a_{ij} = 0$ . Ширина ленты определяется как  $w = p + q - 1$ . Переменная  $p$  означает количество ненулевых диагоналей под главной диагональю включительно, а  $q$  – количество ненулевых диагоналей над главной диагональю включительно. Трехдиагональная матрица – частный случай ленточной, при котором  $p = 2$ ,  $q = 2$ ,  $w = 3$ . Примером трехдиагональной матрицы является матрица (3).

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & \dots & \dots & \dots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & & & \vdots \\ 0 & a_{32} & a_{33} & a_{34} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & \ddots & a_{n-1,n-2} & a_{n-1,n-1} & a_{n-1,n} \\ 0 & \dots & \dots & \dots & 0 & a_{n,n-1} & a_{nn} \end{bmatrix}. \quad (5)$$

Система линейных алгебраических уравнений (1), содержащая трехдиагональную

матрицу, будет иметь вид:

$$\begin{bmatrix} a_{11} & a_{12} & 0 & \dots & \dots & \dots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & & & \vdots \\ 0 & a_{32} & a_{23} & a_{34} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & \ddots & a_{n-1,n-2} & a_{n-1,n-1} & a_{n-1,n} \\ 0 & \dots & \dots & \dots & 0 & a_{n,n-1} & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}, \quad (6)$$

где каждая строка эквивалентна рекуррентному соотношению вида:

$$a_{i,i-1}x_{i-1} + a_{ii}x_i + a_{i,i+1}x_{i+1} = b_i. \quad (7)$$

Методом последовательного исключения можно привести СЛАУ (6) к верхней треугольной форме:

$$\begin{bmatrix} \tilde{a}_{11} & \tilde{a}_{12} & 0 & \dots & \dots & 0 \\ 0 & \tilde{a}_{22} & \tilde{a}_{23} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & \ddots & \tilde{a}_{n-1,n-1} & \tilde{a}_{n-1,n} \\ 0 & \dots & \dots & \dots & 0 & \tilde{a}_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ \vdots \\ \vdots \\ \tilde{b}_{n-1} \\ \tilde{b}_n \end{bmatrix}. \quad (8)$$

Из СЛАУ (8) можно вывести рекуррентное соотношение  $x_{i-1}$  через  $x_i$ :

$$x_{i-1} = \frac{\tilde{b}_{i-1} - \tilde{a}_{i-1,i}x_i}{\tilde{a}_{i-1,i-1}}, \quad (9)$$

Для удобства дальнейших преобразований необходимо ввести коэффициенты  $\gamma_i = -\frac{\tilde{a}_{i-1,i}}{\tilde{a}_{i-1,i-1}}$ ,  $\beta_i = \tilde{b}_{i-1}$ . Тогда выражение (7) примет вид:

$$x_{i-1} = \gamma_i x_i + \beta_i. \quad (10)$$

Подстановка (10) в (7) дает:

$$\begin{aligned} & a_{i,i-1}(\gamma_i x_i + \beta_i) + a_{ii}x_i + a_{i,i+1}x_{i+1} = b_i \\ \implies & x_i = \frac{-a_{i,i+1}}{a_{i,i-1}\gamma_i + a_{ii}}x_{i+1} + \frac{b_i - a_{i,i-1}\beta_i}{a_{i,i-1}\gamma_i + a_{ii}}. \end{aligned} \quad (11)$$

Сравнение полученного выражения с (10) позволяет получить рекуррентные соотношения для определения  $\gamma_i$  и  $\beta_i$ :

$$\gamma_{i+1} = \frac{-a_{i,i+1}}{a_{i,i-1}\gamma_i + a_{ii}}, \quad (12)$$

$$\beta_{i+1} = \frac{b_i - a_{i,i-1}\beta_i}{a_{i,i-1}\gamma_i + a_{ii}}. \quad (13)$$

Дальнейшие шаги предполагают последовательное вычисление всех коэффициентов  $\gamma_{i+1}$  и  $\beta_{i+1}$  по формулам (12) и (13) решение СЛАУ с помощью обратного хода метода Гаусса по формуле (10).

Необходимо также найти  $\beta_1, \gamma_1, x_n$ . Выражение (5) для первой строки записывается как:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 &= b_1 \\ \implies x_1 &= -\frac{a_{12}}{a_{11}}x_2 + \frac{b_1}{a_{11}}. \end{aligned} \quad (14)$$

Если сравнить (14) с (9), то получается:

$$\begin{aligned} \frac{-a_{12}}{a_{10}\gamma_1 + a_{11}} &= -\frac{a_{12}}{a_{11}}, \\ \frac{b_1 - a_{10}\beta_1}{a_{10}\gamma_1 + a_{11}} &= \frac{b_1}{a_{11}}, \end{aligned}$$

из чего следует  $\gamma_1 = \beta_1 = 0$ .

Для нахождения  $x_n$  требуется рассмотреть выражение (5) для последней строки матрицы. Его подстановка в (8) дает:

$$x_n = \frac{b_n - a_{n,n-1}\beta_n}{a_{nn} + a_{n,n-1}\gamma_n}. \quad (15)$$

В листинге 2 представлена функция `thomas(A, b)`, в которой находится решение СЛАУ (1) по формулам (10), (12) и (13), а также учитывает (15). Входными параметрами являются матрица **A** и столбец свободных коэффициентов **b**.

Листинг 2 – Функция `thomas(A, b)`, возвращающая решение СЛАУ (2), полученное с помощью метода прогонки.

---

```
def thomas(A, b):
    n = len(b)

    y = np.zeros(n + 1)
    bt = np.zeros(n + 1)

    for i in range(n):
        a = ((A[i, i - 1] if i > 0 else 0),
             A[i, i],
             (A[i, i + 1] if i < n - 1 else 0))

        y[i + 1] = -a[2] / (a[1] + y[i] * a[0])
        bt[i + 1] = (b[i] - a[0] * bt[i]) / (a[0] * y[i] + a[1])

    x = np.zeros(n + 1)

    for i in range(n - 1, -1, -1):
        x[i] = y[i + 1] * x[i + 1] + bt[i + 1]

    return x[:n]
```

---



### 3 Выбор "универсального" метода

Метод Гаусса является базовым методом решения СЛАУ, содержащих матрицу общего вида, однако в некоторых случаях, описанных в пункте 1, может приводить к высоким вычислительным погрешностям. Частичным решением этих проблем является метод Гаусса с частичным выбором главного элемента. Путем перестановки строк для выбора главного элемента метод позволяет уменьшить влияние ошибок округления и представления чисел в памяти компьютера, а также округления при вычитания одной строки матрицы из другой. Метод прогонки же предназначен для решения СЛАУ с трехдиагональными матрицами, что обуславливает высокую вычислительную погрешность при использовании метода с матрицами общего вида. Таким образом, среди реализованных методов, "универсальным" то есть минимизирующим вычислительные погрешности для случая квадратных матриц общего вида, будет метод Гаусса с частичным выбором главного элемента.

### 4 Генерация случайных невырожденных матриц

Для генерации случайной невырожденной матрицы размерности  $6 \times 6$  с элементами  $|a_{ij}| < 1$  общего вида используется следующий алгоритм:

1. Генерируется случайная невырожденная нижняя треугольная матрица  $L$  размерности  $6 \times 6$  с диагональными элементами, равными 1.
2. Генерируется случайная невырожденная верхняя треугольная матрица  $U$  размерности  $6 \times 6$ .
3. Умножить матрицы  $L$  и  $U$ , чтобы получить невырожденную искомую матрицу  $A$ .

Для генерации случайной невырожденной нижней треугольной матрицы  $L$  можно использовать следующий алгоритм:

1. Создать единичную матрицу  $L$  размерности  $6 \times 6$ .
2. Для каждого элемента матрицы  $L_{ij}$ , где  $i > j$ , сгенерировать случайное число от  $-1$  до  $1$  и присвоить его этому элементу.

Для генерации случайной верхней треугольной матрицы  $U$  используется генерация верхней треугольной матрицы.

В результате умножения матриц  $L$  и  $U$  получится матрица, которая будет невырожденной и общего вида. Для того чтобы элементы получившейся матрицы были по модулю меньше 1, в написанной функции `generate_simple(size, limits)`, представленной в листинге 3, элементы делятся на модуль максимального по модулю элемента матрицы, умноженного на коэффициент 1,1. Действие применяется и для дальнейших описываемых функций генерации невырожденных матриц. Функция принимает на вход параметр `size` – размер матрицы и `limits` – требуемые граничные значения элементов матрицы.

Листинг 3 – Функция `generate_simple(size, limits)`, генерирующая невырожденную матрицу общего вида.

---

```
def generate_simple(size, limits):
    L = np.eye(size[0], size[1]) + np.tril(np.random.rand(size[0], size[1]), -1)
    U = np.triu(np.random.rand(size[0], size[1]), 0)

    A = np.dot(L, U)
    A = A / (np.max(np.abs(A)) * 1.1) * (limits[1] - limits[0]) + limits[0]

    return np.matrix(A, dtype=np.float32)
```

---

Для генерации случайной невырожденной трехдиагональной матрицы используется аналогичный алгоритм:

1. Сгенерировать случайную невырожденную ленточную матрицу  $L$  размерности  $6 \times 6$  с параметрами  $p = 2, q = 1$ .
2. Сгенерировать случайную невырожденную ленточную матрицу  $U$  размерности  $6 \times 6$  с параметрами  $p = 1, q = 2$ .
3. Умножить матрицы  $L$  и  $U$ , чтобы получить невырожденную трехдиагональную матрицу  $A$ .

Функция `generate_diagonal(size, limits)`, генерирующая невырожденную матрицу трехдиагонального вида, представлена в [листинге 4](#). Функция принимает на вход параметр *size* – размер матрицы и *limits* – требуемые граничные значения элементов матрицы.

Листинг 4 – Функция `generate_diagonal(size, limits)`, генерирующая невырожденную матрицу трехдиагонального вида.

---

```
def generate_diagonal(size, limits):
    L = np.eye(size[0], size[1]) + np.diag(np.random.rand(size[0] - 1), -1)
    U = np.diag(np.random.rand(size[0]), 0) + np.diag(np.random.rand(size[1] - 1), 1)

    A = np.dot(L, U)
    A = (A / (np.max(np.abs(A)) * 1.1))
    A[A > 0] = A[A > 0] * (limits[1] - limits[0]) + limits[0]

    return np.matrix(A, dtype=np.float32)
```

---

## 5 Проведение эксперимента с большим количеством случайных матриц

Для оценки вычислительной эффективности методов следует рассчитать количество операций, необходимых для решения СЛАУ различными методами. Учитываться будут только время вычислений операций умножения и деления, так как оно значительно больше вычислительного времени операций сложения и вычитания.

Для метода Гаусса на  $k$ -й итерации из всего  $(n-1)$  итераций прямого хода требуется  $(n-k)$  делений для вычисления множителя  $\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$  из (3). Далее необходимо для  $(n-k)$  строк произвести  $(n-k+1)$  перемножений, т.е. суммарно  $(n-k)(n-k+1)$ . Таким образом, на  $k$ -й итерации  $n-k + (n-k)(n-k+1) = (n-k)(n-k+2)$  умножений и делений.

Произведя суммирование по  $k = 1, \dots, n-1$ , получается общее число умножений и делений в прямом ходе метода Гаусса:

$$\begin{aligned} \sum_{k=1}^{n-1} (n-k)(n-k+2) &= \sum_{k=1}^{n-1} (n-k)^2 + 2 \sum_{k=1}^{n-1} (n-k) = \sum_{k=1}^{n-1} k^2 + 2 \sum_{k=1}^{n-1} k \\ &= \frac{(n-1)n(2n-1)}{6} + 2 \frac{(n-1)n}{2} = \frac{2n^3 + 3n^2 - 5n}{6}. \end{aligned} \quad (16)$$

В обратном ходе метода Гаусса для нахождения  $x_k$  требуется  $(n-k)$  перемножений и одно деление, что в сумме дает:

$$\sum_{k=1}^n (n-k+1) = \sum_{k=1}^n k = \frac{n(n+1)}{2} = \frac{n^2+n}{2}. \quad (17)$$

Суммируя (16) и (17), для нахождения решения методом Гаусса требуется  $\frac{2n^3+3n^2-5n}{6} + \frac{n^2+n}{2}$  операций, то есть  $O(n^3)$ .

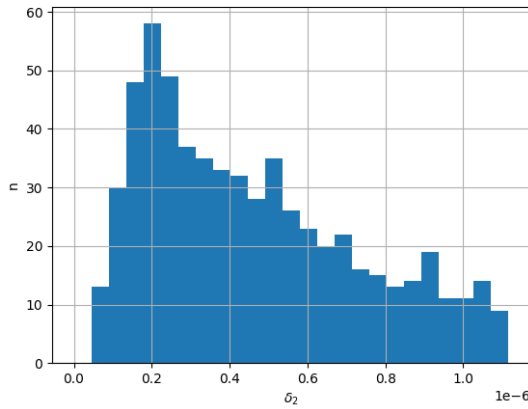
Метод прогонки является модификацией метода Гаусса для случая СЛАУ с трехдиагональными матрицами и требует  $O(n)$  операций для их решения. Следовательно, является более вычислительно-эффективным методом для подобного типа матриц.

Таким образом, "специальным" методом для решения СЛАУ с трехдиагональной матрицей является метод прогонки, а с матрицей общего вида – метод Гаусса.

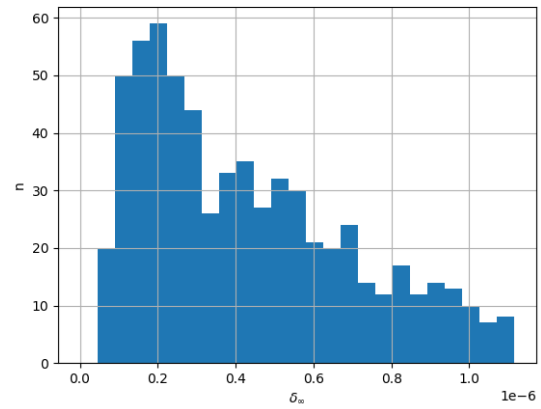
Среднеквадратичная и супремум-нормы  $\mathbf{x} = [x_1, \dots, x_n] \in \mathbb{R}^n$  определяются соответственно следующими формулами:

$$\begin{aligned} \|\mathbf{x}\|_2 &= \left( \sum_{i=1}^n x_i^2 \right)^{\frac{1}{2}}, \\ \|\mathbf{x}\|_\infty &= \max_{i \in [1;n]} |x_i|. \end{aligned}$$

Таким образом вычисляется относительная погрешность среднеквадратичной нормы  $\delta_2 = \frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2}$ , где  $\mathbf{x}$  – решение СЛАУ "универсальным" методом, а  $\tilde{\mathbf{x}}$  – решение СЛАУ "специальным" методом, и супремум-нормы  $\delta_\infty = \frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|_\infty}{\|\mathbf{x}\|_\infty}$ . Распределения перечисленных погрешностей в виде гистограмм для каждого "специального" метода представлены на рисунках 1 – 2, где  $n$  – количество решений СЛАУ.

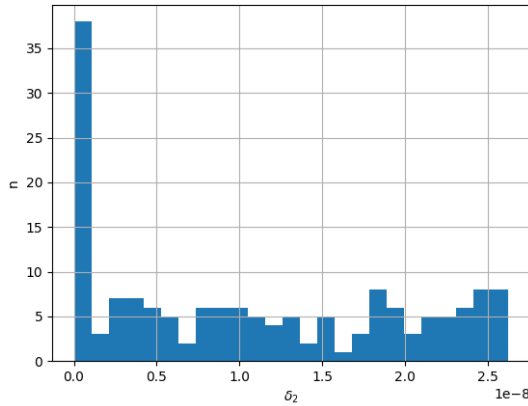


а)

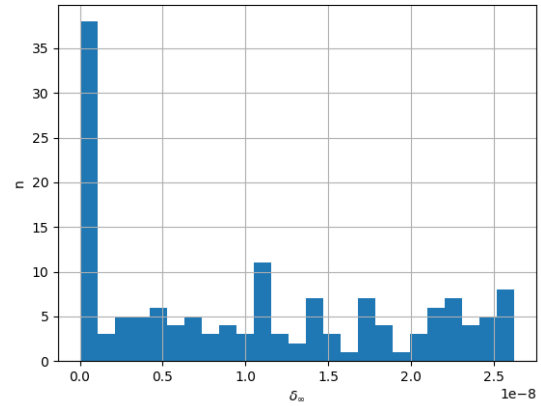


б)

Рисунок 1 — Гистограмма распределения относительных погрешностей среднеквадратичной (а) и супремум-норм (б) метода Гаусса



а)



б)

Рисунок 2 — Гистограмма распределения относительных погрешностей среднеквадратичной (а) и супремум-норм (б) метода прогонки

Анализ графиков показывает, что относительная погрешность для метода Гаусса ограничена значениями меньшего порядка, чем у метода прогонки. Оба наблюдаемых графика указывают, что относительные погрешности стремятся к нулю. Это свидетельствует о **высокой численной устойчивости** обоих методов при решении систем своих характерных матриц.

Метод Гаусса не является **вычислительной устойчивым** для СЛАУ, содержащих матрицы общего вида, так как требует выполнения операций с элементами матрицы, что включает деление на значения, которые близки к нулю. Это может вызвать ошибки из-за потери точности.

Метод прогонки же является вычислительно устойчивым методом для решения СЛАУ с трехдиагональными матрицами.<sup>[1]</sup>

## 6 Расширение генератора невырожденных матриц

Матрица  $\mathbf{A}$  называется положительно-определенной, если она симметричная, и верным является неравенство  $\mathbf{x}^T \mathbf{A} \mathbf{x} > 0$  для любого вектора  $\mathbf{x} \neq 0$ .

1. Создать невырожденную нижнюю треугольную матрицу  $\mathbf{L}$  размером  $6 \times 6$ , заполнить её элементы случайными значениями из отрезка  $(-1, 1)$ .

2. Получить матрицу  $\mathbf{A}$  путем умножения  $\mathbf{L}$  на транспонированную  $\mathbf{L}^T$ :  $\mathbf{A} = \mathbf{L} \cdot \mathbf{L}^T$ .

Этот алгоритм использует свойство разложения Холецкого, который будет рассмотрен в пункте 7. Разложение Холецкого позволяет представить положительно определенную симметричную матрицу в виде произведения нижнетреугольной матрицы на транспонированную этой матрицы. Поэтому если можно получить разложение Холецкого для случайно сгенерированной матрицы  $\mathbf{A}$ , то  $\mathbf{A}$  является положительно определенной.

Для генерации положительно-определенных матриц была написана функция `generate_positive_definite(size, limits)`, представленная в листинге 5. Функция принимает на вход параметры *size* – размер матрицы и *limits* – требуемые граничные значения элементов матрицы.

Листинг 5 – Функция `generate_positive_definite(size, limits)`, генерирующая положительно-определенную матрицу.

---

```
def generate_positive_definite(size, limits):
    L = np.eye(size[0], size[1]) + np.tril(np.random.rand(size[0], size[1]), 0)

    A = np.dot(L, np.transpose(L))
    A = A / (np.max(np.abs(A)) * 1.1)

    return np.matrix(A, dtype=np.float32)
```

---

## 7 Решение СЛАУ с помощью разложения Холецкого

Разложение Холецкого - это способ представления симметричной положительно определенной матрицы  $\mathbf{A}$  в виде произведения нижнетреугольной матрицы  $\mathbf{L}$  на её транспонированную:

$$\mathbf{A} = \mathbf{L} \mathbf{L}^T$$

Шаги для решения системы уравнений с помощью разложения Холецкого:

1. Вычисление разложения Холецкого: нахождение нижнетреугольной матрицы  $\mathbf{L}$  такой, что  $\mathbf{A} = \mathbf{L} \mathbf{L}^T$ .

В качестве примера рассмотрена положительно определенная матрица размерности  $3 \times 3$ , демонстрирующая разложение Холецкого:

$$\begin{aligned} \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} &= \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix} \\ &= \begin{bmatrix} l_{11}^2 & l_{11}l_{21} & l_{11}l_{31} \\ l_{11}l_{21} & l_{21}^2 + l_{22}^2 & l_{21}l_{31} + l_{22}l_{32} \\ l_{11}l_{31} & l_{21}l_{31} + l_{22}l_{32} & l_{31}^2 + l_{32}^2 + l_{33}^2 \end{bmatrix}. \end{aligned} \quad (18)$$

По приведенному равенству можно вычислить элементы матрицы  $\mathbf{L}$  следующим образом:

$$\begin{aligned} l_{11} &= \sqrt{a_{11}}, \\ l_{21} &= \frac{a_{21}}{l_{11}}, \\ l_{31} &= \frac{a_{31}}{l_{11}}, \\ l_{22} &= \sqrt{a_{22} - l_{21}^2}, \\ l_{32} &= \frac{1}{l_{22}} (a_{32} - l_{21}l_{31}), \\ l_{33} &= \sqrt{a_{33} - l_{31}^2 - l_{32}^2}. \end{aligned}$$

Отсюда справедливы следующие формулы для матрицы произвольной размерности:

$$\begin{aligned} l_{ii} &= \sqrt{a_{ii} - \sum_{j=1}^{i-1} l_{ij}^2}, \quad i = 1, \dots, n, \\ l_{ij} &= \frac{1}{l_{jj}} \left( a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk} \right), \quad j < i, \end{aligned} \quad (19)$$

где  $n$  - размерность матрицы.

Далее решение разделяется на два последовательных шага:

$$\mathbf{Ax} = \mathbf{b} \implies \mathbf{LL}^T \mathbf{x} = \mathbf{b} \implies \begin{cases} \mathbf{Ly} = \mathbf{b}, \\ \mathbf{L}^T \mathbf{x} = \mathbf{y}, \end{cases}$$

2. Сначала нужно найти решение матричного уравнения  $\mathbf{Ly} = \mathbf{b}$ .

Для демонстрации решения данной СЛАУ возьмем  $\mathbf{L}$  с размерностью  $3 \times 3$ , тогда:

$$\mathbf{Ly} = \mathbf{b} \implies \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Данное матричное уравнение приводит к системе вида:

$$\begin{cases} l_{11}y_1 = b_1, \\ l_{21}y_1 + l_{22}y_2 = b_2, \\ l_{31}y_1 + l_{32}y_2 + l_{33}y_3 = b_3 \end{cases}$$

И далее можно записать решение данной системы:

$$\begin{cases} y_1 = \frac{b_1}{l_{11}}, \\ y_2 = \frac{b_2 - l_{21}y_1}{l_{22}}, \\ y_3 = \frac{b_3 - l_{31}y_1 - l_{32}y_2}{l_{33}} \end{cases}$$

Отсюда справедлива следующая формула для матрицы произвольной размерности  $n$ :

$$y_i = \frac{1}{l_{ii}} \left( b_i - \sum_{k=1}^{i-1} l_{ik}y_k \right), \quad i = \overline{1 \dots n} \quad (20)$$

3. Имея значения  $\mathbf{y}$ , можно приступить к решению системы  $\mathbf{L}^T \mathbf{x} = \mathbf{y}$ .

Для демонстрации решения данной СЛАУ также возьмем  $\mathbf{L}$  с размерностью  $3 \times 3$ , тогда:

$$\mathbf{L}^T \mathbf{x} = \mathbf{y} \implies \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Данное матричное уравнение приводит к системе вида:

$$\begin{cases} l_{11}x_1 + l_{12}x_2 + l_{13}x_3 = y_1, \\ l_{22}x_2 + l_{23}x_3 = y_2, \\ l_{33}x_3 = y_3 \end{cases}$$

И далее можно записать решение данной системы:

$$\begin{cases} x_1 = \frac{y_1 - l_{12}x_2 - l_{13}x_3}{l_{11}}, \\ x_2 = \frac{y_2 - l_{23}x_3}{l_{22}}, \\ x_3 = \frac{y_3}{l_{33}} \end{cases}$$

Отсюда справедлива следующая формула для матрицы произвольной размерности  $n$ :

$$x_i = \frac{1}{l_{ii}} \left( y_i - \sum_{k=i+1}^n l_{ik}x_k \right), \quad i = \overline{n \dots 1} \quad (21)$$

В **листинге 6** представлена функция `cholesky(A, b)`, вычисляющая решение СЛАУ (2) с помощью разложения Холецкого.

Листинг 6 – Функция `cholesky(A, b)`, вычисляющая решение СЛАУ (2) с помощью разложения Холецкого (формулы (??), (??), (??)). Функция принимает в качестве параметров матрицу  $\mathbf{A}$ , столбец свободных коэффициентов  $\mathbf{b}$ .

---

```

def cholesky(A, b):
    L = np.zeros(A.shape)
    n = len(b)

    for i in range(n):
        for j in range(0, i):
            L[i, j] = (A[i, j] - np.sum(L[i, :j] * L[j, :j])) / L[j, j]

        L[i, i] = np.sqrt(A[i, i] - np.sum(L[i, :i] ** 2))

    y = np.zeros(n)

    for i in range(n):
        y[i] = (b[i] - np.sum(L[i, :i] * y[:i])) / L[i, i]

    L = np.transpose(L)
    x = np.zeros(n)

    for i in range(n - 1, -1, -1):
        x[i] = (y[i] - np.sum(L[i, i + 1:] * x[i + 1:])) / L[i, i]

    return x

```

---

## 8 Анализ методов, учитывая разложение Холецкого

"Специальным" методом решения СЛАУ для положительно-определенных матриц является разложение Холецкого, так как данный метод наиболее вычислительно-эффективен к матрицам подобного типа. Распределения относительных погрешностей среднеквадратичной и супремум-норм в виде гистограмм для разложения Холецкого представлены на рисунке 3, где  $n$  – количество решений СЛАУ.

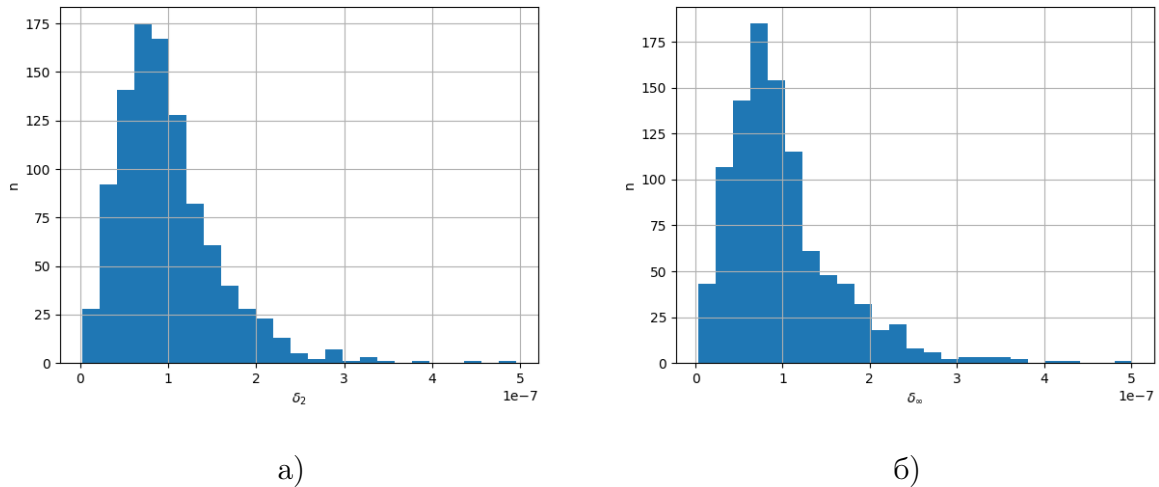


Рисунок 3 — Гистограмма распределения относительных погрешностей среднеквадратичной (а) и супремум-норм (б) разложения Холецкого

Анализируя рисунок 3, можно заметить, что большинство относительных погрешностей находятся в пределах значений порядка  $10^{-7}$ , это может свидетельствовать о хорошей численной устойчивости метода на данном наборе данных.



Разложение Холецкого является численно устойчивым для положительно-определенных матриц, так как для таких матриц разложение всегда существует и единственно. Однако для матриц общего вида, разложение Холецкого может быть численно неустойчивым из-за возможных численных ошибок, которые могут привести к отрицательным значениям под корнем при вычислении разложения.

## 9 Распределение спектральных радиусов и чисел обусловленности

Пусть  $\mathbf{A} \in \mathbb{R}^{n \times n}$ . Тогда собственным вектором матрицы  $\mathbf{A}$  называется такой вектор  $\mathbf{x} \neq \mathbf{0}$ , что:

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x},$$

где  $\lambda$  – называется собственным числом матрицы  $\mathbf{A}$ , ассоциированным с собственным вектором.

Спектральным радиусом матрицы  $\mathbf{A}$  называется число  $\rho(\mathbf{A}) \in \mathbb{R}$  такое, что:

$$\rho(\mathbf{A}) = \max_{i \in [1, m]} |\lambda_i|,$$

где  $\lambda_i$  – одно из собственных чисел матрицы.

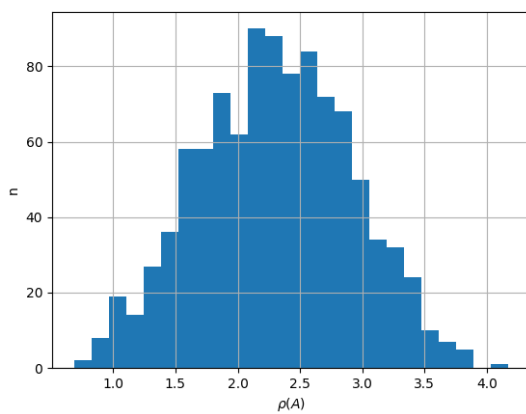
Пусть  $\|\cdot\|$  – норма в пространстве  $\mathbb{R}^n$ . Тогда следующий функционал является нормой матрицы:  $\|\cdot\|$

$$\|\mathbf{A}\| = \max_{\|\mathbf{x}\|=1} \|\mathbf{A}\mathbf{x}\|.$$

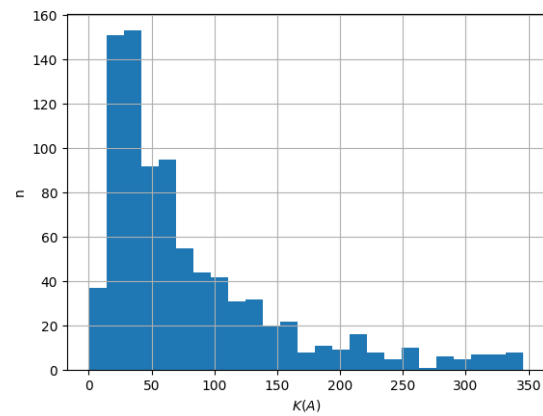
Число обусловленности матрицы  $\mathbf{A}$  описывается выражением:

$$K(\mathbf{A}) = \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\|$$

Распределение спектральных радиусов и чисел обусловленности в виде гистограмм для каждого типа рассмотренных ранее матриц представлено на рисунках 4 – 6.



а)



б)

Рисунок 4 — Гистограмма распределения спектральных радиусов (а) и чисел обусловленности (б) матрицы общего вида  $\mathbf{A}$

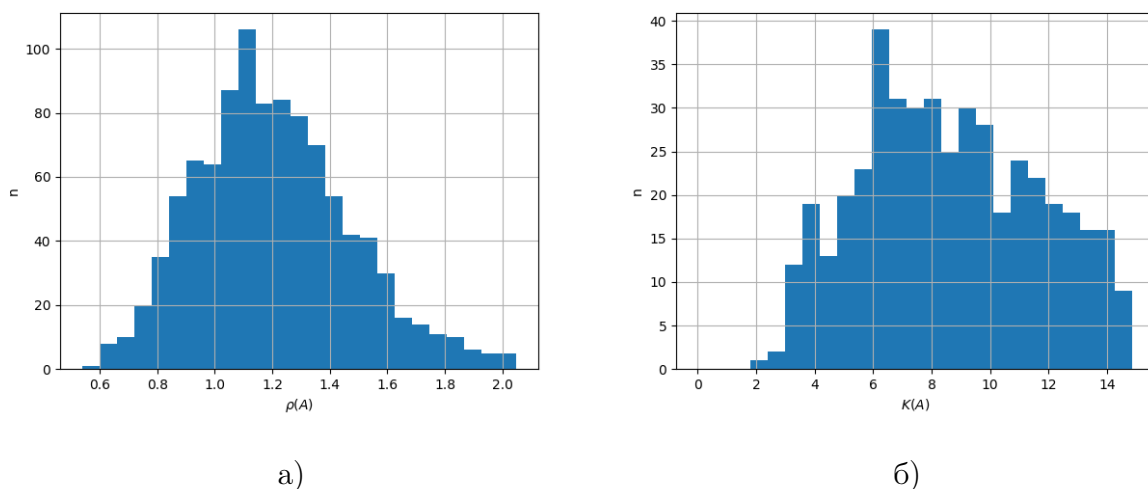


Рисунок 5 — Гистограмма распределения спектральных радиусов (а) и чисел обусловленности (б) трехдиагональной матрицы  $\mathbf{A}$

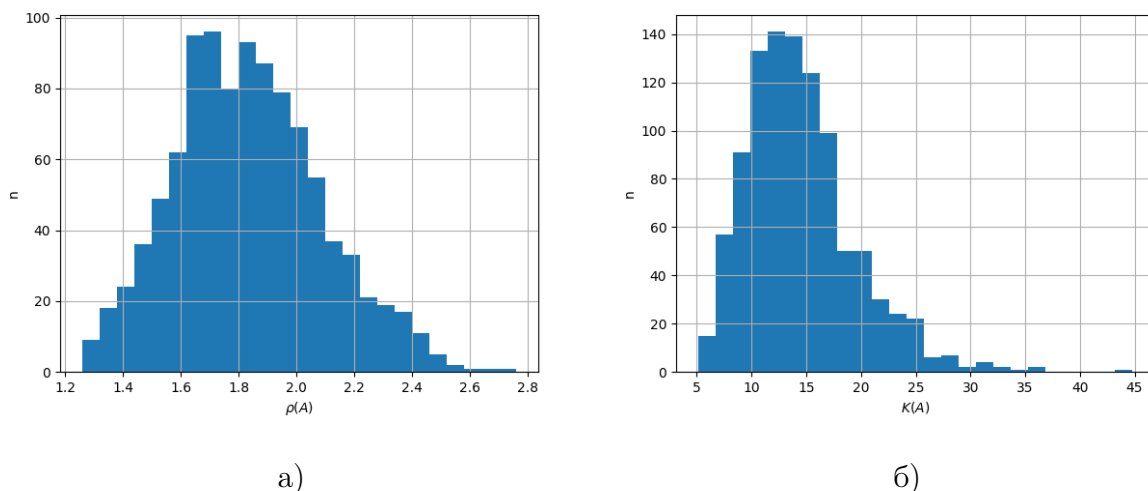


Рисунок 6 — Гистограмма распределения спектральных радиусов (а) и чисел обусловленности (б) положительно-определенной матрицы  $\mathbf{A}$

Гистограмма чисел обусловленности может показать, как распределены числа обусловленности матрицы и, если большинство значений находятся в определенном диапазоне, это может указывать на хорошую численную устойчивость матрицы. Численная устойчивость матрицы относится к её способности сохранять точность вычислений при использовании численных методов. Исходя из рисунков 4 – 6(б), хорошую численную устойчивость демонстрируют трехдиагональные и положительно-определенные

матрицы. Матрицы общего вида характеризуются разбросом значений чисел обусловленности и их сравнительно большим значением, что говорит об их более низкой численной устойчивости. Распределение спектральных радиусов говорит о спектре собственных значений матрицы. Так, например, для матриц общего вида распределение спектральных радиусов рассредоточено и имеет среднее значение, равное приблизительно 2,5. Большой спектральный радиус может указывать на более плохую обусловленность матриц, что означает, что матрица более чувствительна к ошибкам входных данных. Для трехдиагональной и положительно-определенной матриц распределение строится вокруг значения спектрального радиуса, равного 1 и имеет меньшее рассредоточение значений, что является следствием их структуры и свойств.

## 10 Влияние спектрального радиуса на вычислительную устойчивость алгоритма

Спектральный радиус является показателем скорости сходимости решений системы итерационными методами решения СЛАУ, однако и в рассматриваемых методах значение спектрального радиуса оказывает влияние на вычислительную устойчивость. Большие значения спектрального радиуса могут сделать алгоритмы нахождения СЛАУ более чувствительными к погрешностям в исходных данных или округлению при вычислениях. Это может привести к большим ошибкам и неточности в результатах. Малые значения спектрального радиуса могут сделать алгоритмы нахождения СЛАУ более устойчивыми к вычислительным ошибкам. Они могут лучше справляться с округлением и погрешностями, что приводит к более точным результатам.

Значение спектрального радиуса прямо пропорционально зависит от числа обусловленности, которое в свою очередь напрямую влияет на вычислительную устойчивость:

$$K(A) \geq \frac{\rho(A)}{\min_{i \in [1, m]} |\lambda_i|}$$

В целом, меньшие значения спектрального радиуса обычно предпочтительны для вычислительной устойчивости алгоритмов нахождения СЛАУ. Они обеспечивают меньшую чувствительность к погрешностям и более устойчивое решение.

## 11 Влияние отношения максимального по модулю собственного числа к минимальному по модулю собственному числу на вычислительную устойчивость алгоритма

Отношение максимального по модулю собственного числа к минимальному по модулю собственному числу (далее - отношение матрицы) является важным показателем вычислительной устойчивости алгоритма нахождения решения СЛАУ.

Чем больше отношение матрицы, тем менее устойчивым будет алгоритм нахождения решения СЛАУ ввиду того, что матрица близка к вырожденной или плохо



3. Ошибка округления: числа обусловленности также влияют на ошибку округления в вычислениях. При больших числах обусловленности матрицы, малейшие погрешности в вычислениях могут привести к значительным ошибкам в решении СЛАУ.

В целом, числа обусловленности матрицы имеют существенное влияние на вычислительную устойчивость алгоритмов нахождения решения СЛАУ. При больших числах обусловленности матрицы необходимо использовать более устойчивые алгоритмы и методы вычислений, чтобы получить точное решение СЛАУ.

## Заключение



В лабораторной работе были рассмотрены такие прямые методы решения СЛАУ, как метод Гаусса, метод Гаусса с частичным выбором главного элемента, метод прогонки и разложение Холецкого, проведен анализ их вычислительной эффективности и вычислительной неустойчивости, рассмотрены параметры матрицы и их влияние на вычислительную неустойчивость алгоритмов. Результаты демонстрируют, что прямые методы успешно применяются для решения СЛАУ, однако могут проявлять вычислительную неустойчивость при решении СЛАУ с определенными матрицами коэффициентов. Проведенное исследование помогает понять, как применение прямых методов решения СЛАУ зависит от типа и структуры матрицы коэффициентов, а также позволяет оценить их применимость в контексте конкретных задач вычислительной математики.

## Список использованных источников

1. Першин А.Ю. Лекции по курсу «Вычислительная математика». Москва, 2018-2021. С. 140. URL: <https://archrk6.bmstu.ru/index.php/f/810046>.
2. Соколов, А.П. Инструкция по выполнению лабораторных работ (общая). Москва: Соколов, А.П., 2018-2021. С. 9. URL: <https://archrk6.bmstu.ru>. (облачный сервис кафедры РК6).
3. Соколов, А.П. Инструкция по выполнению заданий к семинарским занятиям (общая). Москва: Соколов, А.П., 2018-2022. С. 7. URL: <https://archrk6.bmstu.ru>. (облачный сервис кафедры РК6).
4. Першин А.Ю. Сборник задач семинарских занятий по курсу «Вычислительная математика»: Учебное пособие. / Под редакцией Соколова А.П. [Электронный ресурс]. Москва, 2018-2021. С. 20. URL: <https://archrk6.bmstu.ru>. (облачный сервис кафедры РК6).
5. Першин А.Ю., Соколов А.П., Гудым А.В. Сборник постановок задач на лабораторные работы по курсу «Вычислительная математика»: Учебное пособие. [Электронный ресурс]. Москва, 2023. С. 47. URL: <https://archrk6.bmstu.ru>. (облачный сервис кафедры РК6).

## Выходные данные

Меньшикова Е.А. Отчет о выполнении лабораторной работы №4 по дисциплине «Вычислительная математика». [Электронный ресурс] — Москва: 2023. — 22 с. URL: <https://sa2systems.ru:88> (система контроля версий кафедры РК6)

Постановка:  ассисент кафедры РК-6, PhD Першин А.Ю.  
Решение и вёрстка:  студент группы РК6-51Б, Меньшикова Е.А.

2023, осенний семестр