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

по дисциплине: «Вычислительная математика»

Студент	Косенков Александр Александрович
Группа	РК6-64Б
Тип задания	Лабораторная работа №4
Тема лабораторной работы	Устойчивость прямых методов решения СЛАУ

Студент _____ **Косенков А.А.**
подпись, дата *фамилия, и.о.*

Преподаватель _____ **Першин А.Ю.**
подпись, дата *фамилия, и.о.*

Преподаватель _____ **Соколов А.П.**
подпись, дата *фамилия, и.о.*

Оценка _____

Москва, 2020 г.

Оглавление

Задание на лабораторную работу 3
Цель выполнения лабораторной работы 5
Выполненные задачи 5
1. Реализация функции для решения СЛАУ методом Гаусса. 6
2. Реализация функции для решения СЛАУ методом прогонки. 9
3. Реализация функции для решения СЛАУ методом разложения Холецкого. 10
5. Реализация функции генерации случайных матриц. 13
6. Анализ решения СЛАУ для матриц общего вида. 16
7. Анализ решения СЛАУ для матриц со строгим диагональным преобладанием 25
8. Анализ решения СЛАУ для трехдиагональных матриц. 30
9. Анализ решения СЛАУ для положительно определенных матриц. 35
Заключение 40
Список использованных источников 42

Задание на лабораторную работу

Задача 17 (Устойчивость прямых методов решения СЛАУ)

Требуется:

1. Написать функцию *gauss(A, b, pivoting)*, которая возвращает решение СЛАУ $\boldsymbol{Ax} = \boldsymbol{b}$, полученное с помощью метода Гаусса. Если $pivoting = True$, то решение должно находиться с полным выбором главного элемента. Если $pivoting = False$, то выбора главного элемента происходить не должно.
2. Написать функцию $thomas(A, b)$, которая возвращает решение СЛАУ $\boldsymbol{Ax} = \boldsymbol{b}$, полученное с помощью метода прогонки.
3. Написать функцию $cholesky(A, b)$, которая возвращает решение СЛАУ $\boldsymbol{Ax} = \boldsymbol{b}$, полученное с помощью разложения Холецкого.
4. Из предложенных выше методов выбрать тот, который минимизирует вычислительные погрешности для случая матриц общего вида. В рамках задания мы будем называть этот метод «точным», предполагая, что он дает точное решение СЛАУ.
5. Для случаев матриц общего вида, матриц со строгим диагональным преобладанием, трехдиагональных матриц и положительно определенных матриц провести следующий анализ:
   (a) Выбрать метод, минимизирующий количество арифметических операций для данного класса матриц. Такой метод мы будем называть «методом по умолчанию».
   (b) Разработать и описать алгоритм генерации случайных невырожденных вещественно-значных матриц размерности 4×4 с элементами $|a_{ij}| < 1$.
   (c) Используя разработанный алгоритм, сгенерировать 1000 случайных матриц $\boldsymbol{A}^{(j)}$.

(d) Вывести на экран распределение спектральных радиусов и распределение чисел обусловленности матриц $\boldsymbol{A}^{(j)}$ в виде гистограмм.

(e) Для каждой СЛАУ $\boldsymbol{A}^{(j)}\boldsymbol{x} = [1, 1, 1, 1]^T$ найти решение с помощью «точного метода» и «метода по умолчанию», а затем найти относительную погрешность вычислений с помощью среднеквадратичной и супремум-нормы.

(f) Вывести на экран распределения погрешностей в виде гистограмм.

(g) Ответить на следующие вопросы:

i. Является ли выбранный метод, примененный к матрицам данного класса, вычислительно устойчивым? Почему?

ii. Влияет ли значение спектрального радиуса матрицы на вычислительную устойчивость алгоритма? Если да, то как?

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

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

Цель выполнения лабораторной работы

Цель выполнения лабораторной работы – исследовать вычислительную устойчивость метода Гаусса, метода прогонки и разложения Холецкого на примере нахождения решений СЛАУ с различными матрицами коэффициентов.

Выполненные задачи

1. Реализация функции для решения СЛАУ методом Гаусса.
2. Реализация функции для решения СЛАУ методом прогонки.
3. Реализация функции для решения СЛАУ методом разложения Холецкого.
4. Выбор «точного» метода.
5. Реализация функции генерации случайных матриц.
6. Анализ решения СЛАУ для матриц общего вида.
7. Анализ решения СЛАУ для матриц со строгим диагональным преобладанием.
8. Анализ решения СЛАУ для трехдиагональных матриц.
9. Анализ решения СЛАУ для положительно определенных матриц.

В ходе лабораторной работы для программной реализации задач был использован язык Python v3.6.9, а также прикладные библиотеки numpy v1.18.1, matplotlib v3.2.0, scipy v1.4.1.

1. Реализация функции для решения СЛАУ методом Гаусса.

Метод Гаусса состоит в приведении матрицы коэффициентов A к верхнетреугольному виду с помощью элементарных преобразований, и последовательного решения полученных уравнений.

Первый этап называется прямым ходом Гаусса и заключается в последовательном обнулении элементов, находящихся под главной диагональю, которое реализовано с помощью применения к расширенной матрице $\tilde{A}$ [1](5.5) элементарных преобразований вида:

$$a_{ij}^{(1)} = a_{ij} - \frac{a_{i1}}{a_{11}} a_{1j} \tag{1.1}$$

$$b_i^{(1)} = b_i - \frac{a_{i1}}{a_{11}} b_1. \tag{1.2}$$

Программная реализация прямого хода подразумевает вычисление множителя $m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$, последовательное вычисление коэффициентов в каждой строке на k-й итерации по формуле (1.1) и вычисление новых значений коэффициентов вектора $\bar{b}$ по формуле (1.2) (Листинг 1).

В случае, когда диагональный элемент равен нулю, вычисление множителя m_{ik} не представляется возможным. Помимо этого, диагональный элемент может быть крайне мал и может привести к вычислительной погрешности, которая в данном рассматриваемом методе на каждой итерации будет накапливаться. Во избежание подобных проблем была реализована модификация метода Гаусса в виде полного выбора главного элемента. Данный выбор подразумевает перестановку строк и столбцов матрицы так, что

$$\left|a_{kk}^{(k)}\right| = \max_{k \le i,j \le n} \left|a_{ij}^{(k)}\right|. \tag{3}$$

При этом, перестановка столбцов местами влияет на порядок значений в итоговом векторе решений $\bar{x}$, так как перестановка столбцов матрицы подразумевает изменение порядка переменных в уравнениях. Для учета этого, используется матрица перестановок P, которая представляет собой единичную матрицу той же размерности, что и исходная матрица A. При перестановке столбцов матрицы A, столбцы матрицы P также переставляются. После получения вектора решений $\bar{x}$, порядок следования переменных восстанавливается умножением на матрицу перестановок слева, что также реализовано в функции *gauss(A, b, pivoting)* (Листинг 1).

Листинг 1 – функция решения СЛАУ методом Гаусса

```
def gauss(A, b, pivoting=True):
    # Если система несовместна и имеет бесконечное множество решений
```

```
    if is_matrix_singular(A):
        raise ValueError("Matrix is singular.")

    n = len(A)

    A_gauss = np.copy(A)
    b_gauss = np.copy(b)

    # Матрица перестановок
    P_matrix = np.eye(n)

    # Прямой ход Гаусса
    for k in range(n - 1):
        if pivoting:
            # Выбор элемента
            max_ind_flat = abs(A_gauss[k:, k:]).argmax()
            # Так как были отброшены k строк и столбцов - индексы необходимо
нормализовать
            max_ind_row = max_ind_flat // (n - k) + k
            max_ind_col = max_ind_flat % (n - k) + k

            # Перестановка
            if max_ind_col != k:
                A_gauss[:, [k, max_ind_col]] = A_gauss[:, [max_ind_col, k]]
                P_matrix[:, [k, max_ind_col]] = P_matrix[:, [max_ind_col, k]]

            if max_ind_row != k:
                A_gauss[[k, max_ind_row]] = A_gauss[[max_ind_row, k]]
                b_gauss[[k, max_ind_row]] = b_gauss[[max_ind_row, k]]
        else:
            if A_gauss[k, k] == 0:
                raise ValueError("Element on the main diagonal is zero. Try
setting pivot flag to True.")

        # Элементарные преобразования
        for row in range(k + 1, n):
            multiplier = A_gauss[row, k] / A_gauss[k, k]
            A_gauss[row, k:] = A_gauss[row, k:] - multiplier * A_gauss[k, k:]
            b_gauss[row] = b_gauss[row] - multiplier * b_gauss[k]

    x = gauss_substitution(A_gauss, b_gauss)

    x_normalize = np.dot(P_matrix, x)
    return x_normalize
```

Дальнейшее получение непосредственных значений вектора решения $\bar{x}$ называется обратным ходом Гаусса и представляет собой обратную подстановку в виде следующих операций:

$$x_n = \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}},$$

$$x_{n-1} = \frac{b_{n-1}^{(n-2)} - a_{n-1,n}^{(n-2)} x_n}{a_{n-1,n-1}^{(n-2)}},$$

...

$$x_1 = \frac{b_1 - \sum_{i=2}^{n} a_{1i} x_i}{a_{11}}. \quad (2.1)$$

Данный обход является обратным ходом Гаусса с обратной подстановкой. Помимо этого, для нижних треугольных матриц существует обратный ход Гаусса с прямой подстановкой, где вычисление значений вектора решения $\bar{x}$ осуществляется, начиная с 1-го индекса [2]:

$$x_1 = \frac{b_1}{l_{11}},$$

$$x_2 = \frac{b_2 - l_{21} x_1}{l_{22}},$$

...

$$x_n = \frac{b_n - \sum_{i=1}^{n-1} l_{ni} x_i}{l_{nn}}. \quad (2.2)$$

Данная реализация обратного хода была применена в дальнейшем для реализации метода разложения Холецкого, поэтому также была описана в функции *gauss_substitution(A, b, type)*, реализующей обратный ход Гаусса, где аргумент *type* может принимать значения *"forward"* и *"backward"* и обозначает направление подстановки (Листинг 2).

Листинг 2 – функция, реализующая обратный ход Гаусса

```
def gauss_substitution(A, b, type="backward"):
    n = len(A)
    x = np.zeros((n, 1))

    if type == "backward":
        for k in range(n - 1, -1, -1):
            x[k] = (b[k] - np.dot(A[k, k + 1:], x[k + 1:])) / A[k, k]
    elif type == "forward":
        for k in range(n):
            x[k] = (b[k] - np.dot(A[k, :k], x[:k])) / A[k, k]
    else:
        raise ValueError("Unknown type of substitution")

    return x
```

Помимо этого, в рамках рассматриваемой задачи лабораторной работы будем считать, что СЛАУ должно иметь единственное решение, т.е. быть совместной, откуда на матрицу коэффициентов A накладывается условие невырожденности или отсутствия сингулярности. Данное условие проверяется в начале реализации метода Гаусса (Листинг 1).

2. Реализация функции для решения СЛАУ методом прогонки.

Метод прогонки является методом решения СЛАУ для трехдиагональных матриц, так как использует свойства данных матриц и уменьшает количество арифметических операций до $O(n)$ по сравнению с методом Гаусса, имеющим вычислительную сложность $O(n^3)$[1](Гл. 5.1.6).

Данный метод подразумевает использование рекуррентных соотношений для вычислений коэффициентов γ_i и β_i:

$$\gamma_{i+1} = -\frac{a_{i,i+1}}{a_{i,i-1}\gamma_i - a_{ii}}, \qquad \beta_{i+1} = \frac{b_i - a_{i,i-1}\beta_i}{a_{i,i-1}\gamma_i + a_{ii}} \tag{3}$$

с начальными условиями $\gamma_1 = \beta_1 = 0$.

После последовательного вычисления коэффициентов γ_i и β_i, с помощью рекурентного соотношения

$$x_{i-1} = \gamma_i x_i + \beta_i \tag{4}$$

с начальным условием $x_n = \frac{b_n - a_{n,n-1}\beta_n}{a_{nn} + a_{n,n-1}\gamma_n}$ вычисляются все компоненты вектора решения СЛАУ $\bar{x}$ (Листинг 3).

Листинг 3 - функция решения СЛАУ методом прогонки.

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

    gamma = np.zeros((n, 1))
    beta = np.zeros((n, 1))

    for i in range(n - 1):
        if i != 0:
            A_i_prev = A[i, i - 1]
        else:
            A_i_prev = 0

        gamma[i + 1] = - A[i, i + 1] / (A_i_prev * gamma[i] + A[i, i])
        beta[i + 1] = (b[i] - A_i_prev * beta[i]) / (A_i_prev * gamma[i] + A[i,
i])

    x = np.zeros((n, 1))
    # Начальное условие x_n
    x[n - 1] = (b[n - 1] - A[n - 1, n - 2] * beta[n - 1]) / (A[n - 1, n - 1] +
A[n - 1, n - 2] * gamma[n - 1])

    for i in range(n - 1, 0, -1):
        x[i - 1] = gamma[i] * x[i] + beta[i]

    return x
```

3. Реализация функции для решения СЛАУ методом разложения Холецкого.

Разложение Холецкого представляет собой разложение исходной матрицы коэффициентов A на нижнюю и верхнюю треугольные матрицы вида:

$$A = LL^T, \tag{5}$$

где L – матрица с ненулевыми элементами на диагонали.

Реализация данного метода подразумевает собой использование выражений для определения коэффициентов матрицы L:

$$l_{ii} = \sqrt{a_{ii} - \sum_{j=1}^{i-1} l_{ij}^2}, i = 1, \dots, n, \tag{6.1}$$

$$l_{ij} = \frac{1}{l_{jj}} \left(\sum_{k=1}^{j-1} l_{ik} l_{jk} \right), j < i, \tag{6.2}$$

где l_{ij} – элементы нижней треугольной матрицы L.

Вычисление коэффициентов матрицы L программно реализовано в функции *cholesky(A, b)* (Листинг 4), где также помимо непосредственного вычисления коэффициентов l_{ij} присутствуют проверки на то, что диагональные элементы матрицы L ненулевые, что является условием разложения Холецкого, и проверку на то, что подкоренное выражение в (6.1) положительно, так как в ином случае матрица переходит их действительного множества в комплексное, что также не удовлетворяет условиям рассматриваемой задачи.

После вычисления всех коэффициентов l_{ij} происходит вычисление компонентов вектора решения СЛАУ $\bar{x}$ в соответствии с выражениями следующей системы:

$$\begin{cases} Ly = b \\ L^T x = y \end{cases}. \tag{6.3}$$

Значения компонентов вектора $\bar{y}$ из первого уравнения системы (6.3) определяются с помощью обратного хода Гаусса с прямой подстановкой, в то время как компоненты вектора $\bar{x}$ - с помощью обратного хода Гаусса с обратной подстановкой.

Данное определение вектора решения СЛАУ $\bar{x}$ также программно реализовано в функции *cholesky(A, b)* (Листинг 4).

Листинг 4 – функция решения СЛАУ методом разложения Холецкого.

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

    # L - down triangle matrix with non-null diagonal elements
    L = np.zeros((n, n))

    for i in range(n):
        # i + 1 because there are i + 1 elements in each row in L matrix
        for j in range(i + 1):
            tmp_sum = sum([L[i, p] * L[j, p] for p in range(j)])  # max p is
p=i-1, because max j is j=i

            if i == j:  # Diagonal element
                under_sqrt_expression = A[i, i] - tmp_sum
                if under_sqrt_expression < 0:
                    print("Check negative: ", under_sqrt_expression)
                    raise ValueError("Negative value under sqrt. A matrix isn't
positive definitive")
                elif under_sqrt_expression == 0:
                    raise ValueError("L matrix has zero diag element. A matrix
isn't positive definitive")

                L[i, j] = np.sqrt(under_sqrt_expression)

            else:
                L[i, j] = ((A[i, j] - tmp_sum) / L[j, j])

    y = gauss_substitution(L, b, type="forward")
    x = gauss_substitution(L.T, y, type="backward")

    return x
```

Для проверки правильности написания функций, для вектора правой части $\bar{b} = \{3; 2; 3\}$ и матриц:

$$A_{gauss} = \begin{bmatrix} 1 & 5 & 3 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}, A_{thomas} = \begin{bmatrix} 6 & 3 & 4 \\ 3 & 6 & 5 \\ 4 & 5 & 10 \end{bmatrix}, A_{cholesky} = \begin{bmatrix} 1 & 5 & 0 \\ 3 & 4 & 5 \\ 0 & 7 & 8 \end{bmatrix}$$

были протестированы функции *gauss*, *thomas* и *cholesky* соответственно. Полученное решение сравнивалось с решением, полученным с помощью метода *np.linalg.solve*. Результаты тестирования приведены на рис. 1.

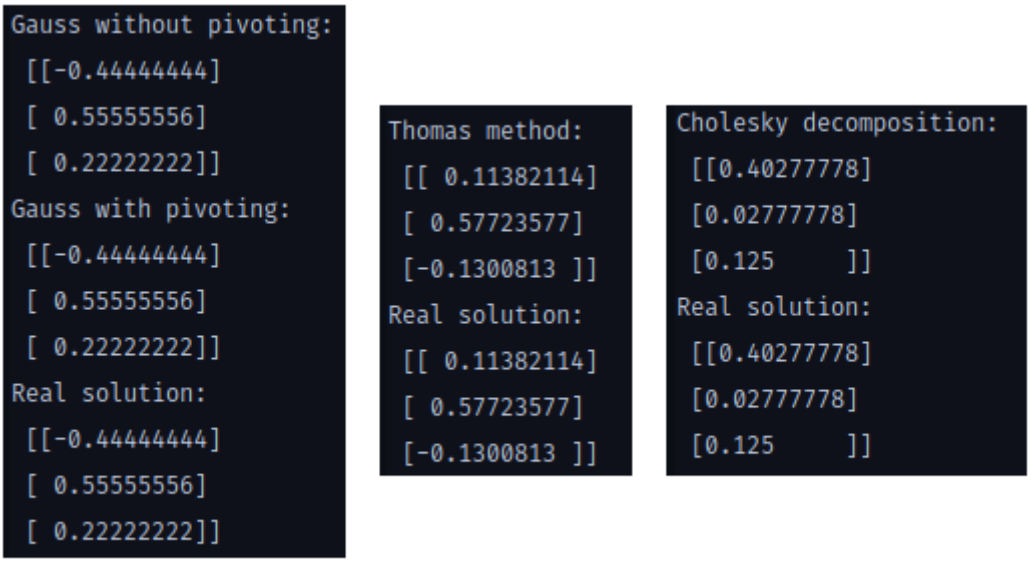

Рис. 1 – Результаты тестирования методов решения СЛАУ функцией *gauss* (а), *thomas* (б) и *cholesky* (в).

Как видно на рис. 1, функции были реализованы верно и дают правильное решение.

4. Выбор «точного» метода.

Для рассматриваемых в рамках лабораторной работы методов, выбор точного метода для матриц общего вида возможен из метода Гаусса и метода Гаусса с выбором главного элемента. Данные методы работают с матрицами общего вида, в то время как метод разложения Холецкого применим только для положительно определенных матриц, а метод прогонки для трехдиагональных матриц.

Из возможных методов наиболее точным является метод Гаусса с выбором главного элемента, поскольку по определению уменьшает накапливаемую погрешность вычисления, которая может присутствовать в методе Гаусса без выбора главного элемента. Таким образом, элементарные преобразования, присутствующие в методе Гаусса в реализации прямого хода являются более точными.

Таким образом, в качестве «точного» метода для матриц общего вида был выбран метод Гаусса с выбором главного элемента.

5. Реализация функции генерации случайных матриц.

Для генерации случайных матриц, принадлежащих тому или иному классу, была реализована функция *generate_rand_matrix(n, type)*, где n – размерность генерируемой матрицы, $type$ – тип генерируемой матрицы, который может принимать значения:

- “default” – матрицы общего вида;
- “DP” – (Diagonal Prevalence) матрицы со строгим диагональным преобладанием;
- “PD” – (Positive Definitive) положительно определенные матрицы;
- “TD” – (Tri-Diagonal) трехдиагональные матрицы.

Генерация матрицы общего вида подразумевает генерацию случайного числа для каждого коэффициента матрицы в интервале $(-1; 1)$.

Далее с помощью функции *is_matrix_singular*, представленной в листинге 5, происходит проверка на сингулярность в соответствии с условием отличия от нуля определителя матрицы. При сингулярности сгенерированной матрицы происходит повторная генерация до тех пор, пока матрица не станет невырожденной.

Листинг 5 – функция проверки матрицы на сингулярность.

```
def is_matrix_singular_old(A):
    return np.linalg.det(A) == 0
```

Затем, в зависимости от требуемого типа матрицы происходит модификация сгенерированной матрицы общего вида до требуемого вида с помощью функции вида *modify_to_<type>*.

Матрицей со строгим диагональным преобладанием является матрица, которая удовлетворяет условию:

$$|a_{ii}| > \sum_{\substack{j=1 \\ i \neq j}}^{n} |a_{ij}|, i = 1, \dots, n. \quad (7)$$

Данное условие достигается модификацией матрицы общего вида, которая заключается в увеличении диагональных элементов на сумму всех остальных элементов в строке, в случае если диагональный элемент изначально меньше этой суммы. Данная модификация реализована в функции *modify_to_DP* (Листинг 6).

Так как по условию поставленной задачи, сгенерированная матрица должна иметь значения коэффициентов, удовлетворяющих выражению

$$|a_{ij}| < 1, \tag{8}$$

то рассмотренное ранее увеличение диагональных элементов может привести к нарушению данного условия. Поэтому, после модификации матрицы, происходит деление всей матрицы на скаляр, равный максимальному элементу матрицы. Таким образом, удается соблюсти условие (8) и требование строгого диагонального преобладания, так как делению подвергаются все коэффициенты.

Листинг 6 – функция приведения матрицы к матрице со строгим диагональным преобладанием.

```
def modify_to_DP(matrix):
    n = len(matrix)

    for i in range(n):
        remaining_sum = np.sum([np.abs(matrix[i, j]) for j in range(n) if i != 
j])
        if matrix[i, i] == 0:
            matrix[i, i] = remaining_sum + 0.01
        elif np.abs(matrix[i, i]) <= remaining_sum:
            if matrix[i, i] > 0:
                matrix[i, i] += remaining_sum
            else:
                matrix[i, i] -= remaining_sum

    while is_matrix_singular(matrix):
        matrix = modify_to_not_singular(matrix)
        max_elem = np.max(np.abs(matrix))
        if max_elem >= 1:
            matrix /= (max_elem + 0.001)

    return matrix
```

Трехдиагональной матрицей является частный случай ленточной матрицы со значениями параметров $p = q = 2$, откуда ширина ленты $w = p + q - 1 = 3$ [1](Гл. 5.1.6). По определению ленточных матриц, элементы за пределами ленты являются нулевыми, иными словами:

$$a_{ij} = 0, \qquad j - i \geq p = 2 \tag{9.1}$$

$$a_{ij} = 0, \qquad i - j \geq q = 2 \tag{9.2}$$

Данное зануление реализовано в функции *modify_to_TD* (Листинг 7).

Листинг 7 – функция приведения матрицы к трехдиагональной.

```
def modify_to_TD(matrix):
    n = len(matrix)
    p = 2
    q = 2

    for i in range(n):
        for j in range(n):
            if (j - i) >= p or (i - j) >= q:
                matrix[i, j] = 0

    while is_matrix_singular(matrix):
        matrix = modify_to_not_singular(matrix)
        max_elem = np.max(np.abs(matrix))
        if max_elem >= 1:
            matrix /= (max_elem + 0.001)

    return matrix
```

Матрица является положительно определенной, если она симметричная и удовлетворяет условию:

$$\bar{x}^T A\bar{x} > 0, \qquad \forall \bar{x} \neq 0. \tag{10}$$

Свойством положительно определенной матрицы является удовлетворение критерию Сильвестра [3], который заключается в неотрицательности всех угловых миноров матрицы [4]. По этой причине данный критерий является условием проверки принадлежности сгенерированной матрицы классу положительно определенных матриц, реализованной в функции *silvester_criterion* (Листинг 8)

Листинг 8– функция проверки выполнения критерия Сильвестра.

```
def silvester_criterion(A):
    n = len(A)

    for k in range(2, n, 1):
        if np.linalg.det(A[:k, :k]) <= 0:
            return False
    return True
```

Получить матрицу, принадлежащую множеству положительно полуопределенных матриц можно путем умножения ее на саму себя транспонированную [5](C1 Corollary C.2). Множество положительно определенных матриц является подмножеством положительно полуопределенных матриц. Данное подмножество отличается от множества положительно полуопределенных матриц удовлетворению условию несингулярности (C1 Corollary C.l), поэтому, для гарантированной генерации положительно определенной матрицы требуется привести ее заведомо к несингулярной, что достигается с помощью прибавления к матрицы единичной, умноженной на случайный положительный коэффициент, что реализовано в функции *modify_to_not_singular*. После чего, при неудовлетворении условию (8) происходит деление всей матрицы на максимальный элемент. Данное

прибавление не нарушает симметричности матрицы и эмпирически не нарушает принадлежности классу положительно определенных матриц. Модификация матрицы общего вида для приведения к классу положительно определенных матриц программно реализована в функции *modify_to_TD* (Листинг 9).

Листинг 9– функция приведения матрицы к положительно определенной.

```
def modify_to_PD(matrix):
    symm_matrix = np.dot(matrix, matrix.T)

    max_elem = np.max(np.abs(symm_matrix))
    if max_elem >= 1:
        symm_matrix /= (max_elem + 0.001)

    while is_matrix_singular(symm_matrix):
        symm_matrix = modify_to_not_singular(symm_matrix)
        max_elem = np.max(np.abs(symm_matrix))
        if max_elem >= 1:
            symm_matrix /= (max_elem + 0.001)

    return symm_matrix
```

Также каждый метод в силу проверок гарантированно возвращает несингулярную матрицу, что является обязательным условием генерации.

При этом после генерации происходит проверка сгенерированной матрицы на принадлежность тому или иному классу, для точной гарантии того, что матрица имеет заданный тип.

6. Анализ решения СЛАУ для матриц общего вида.

В качестве метода, минимизирующего количество арифметических операций (метода «по умолчанию») для матриц общего вида был выбран метод Гаусса без выбора главного элемента. Поскольку, как было отмечено в п. 4, для решения СЛАУ с матрицами общего вида из рассматриваемых методов подходит метод Гаусса и метод Гаусса с выбором главного элемента. Поскольку полный выбор главного элемента увеличивает количество арифметических и логических операций, метод Гаусса без выбора главного элемента лучше удовлетворяет условию метода «по умолчанию»

Для анализа решений СЛАУ для матриц общего вида, а также для анализа характеристик таких матриц, с помощью функции *generate_rand_matrix* были сгенерированы 1000 случайных матриц $\boldsymbol{A}^{(j)}$ общего вида.

Затем, с помощью реализованной функции *spectral_radius* были вычислены значения спектральных радиусов для каждой матрицы. Данная функция реализует вычисление спектрального радиуса в соответствии с его определением:

$$\rho(\boldsymbol{A}) = \max_{i\in[1,n]}|\lambda_i|\,. \qquad (11)$$

Вычисление собственных чисел реализовано с помощью библиотечной функции $numpy.linalg.eigvals$.

Распределение спектральных радиусов для сгенерированных матриц представлено в виде гистограммы на рис. 2.

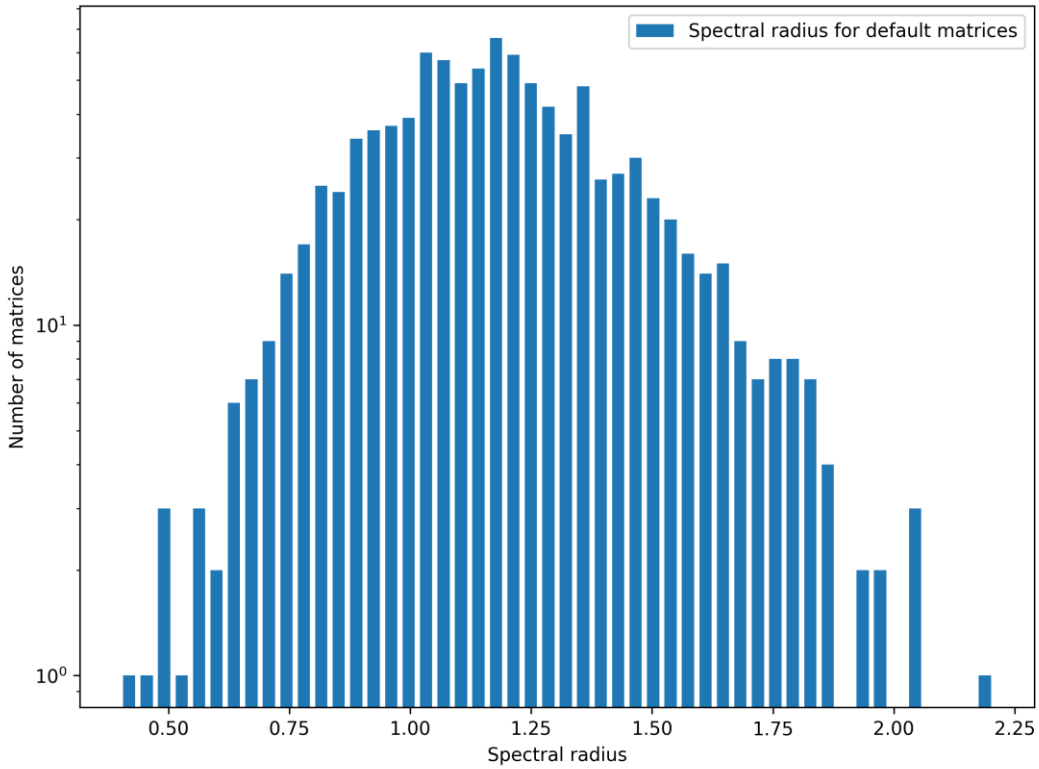

Рис. 2 – Распределение спектральных радиусов для случайно сгенерированных матриц общего вида.

Также для каждой матрицы было получено число обусловленности, вычисленной с помощью библиотечной функции $numpy.linalg.cond()$.

Число обусловленности является показателем устойчивости решения СЛАУ к ее малым изменениям и связывает погрешность с невязкой решения [1](Гл. 5.2.8).

Распределение чисел обусловленности для сгенерированных матриц представлено в виде гистограммы на рис. 3.

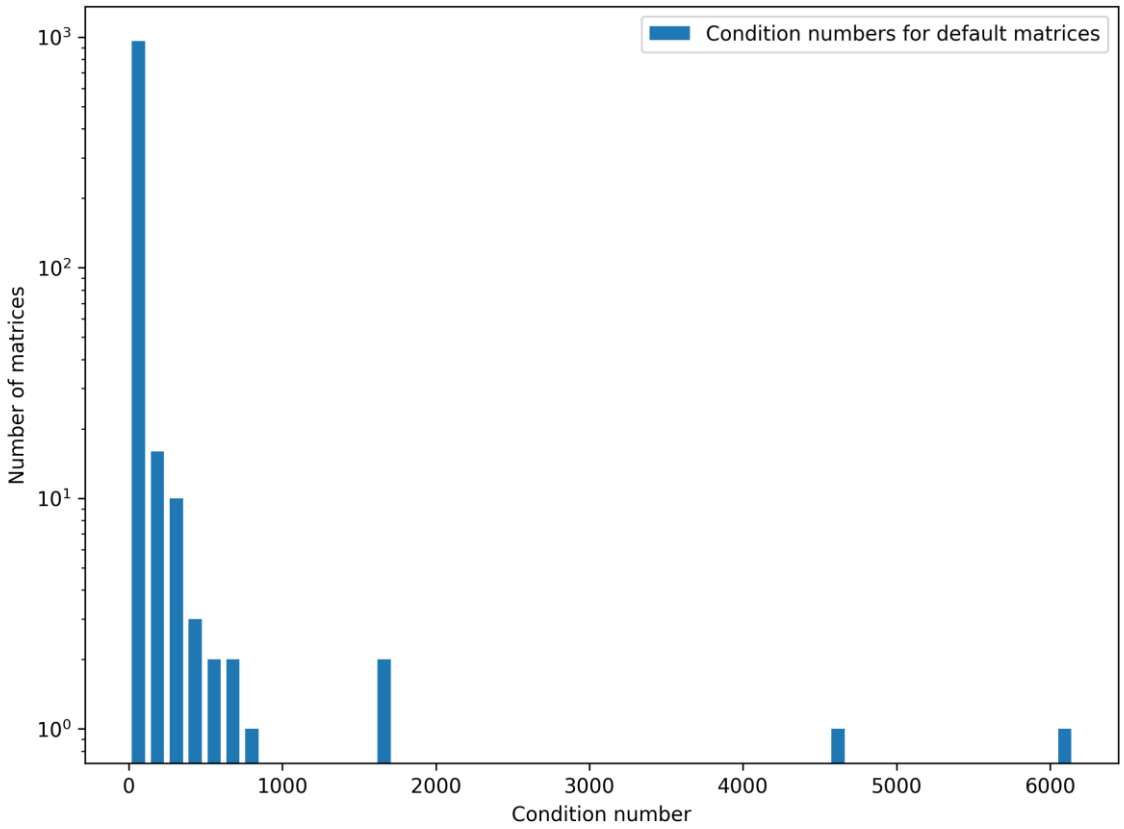

Рис. 3 - Распределение чисел обусловленности для случайно сгенерированных матриц общего вида.

Анализ гистограммы рис. 3 показывает значительные скачки значения числа обусловленности для очень малого количества матриц. В ходе исследования было обнаружено, что подобное отклонение сильнее всего сказывается на числах обусловленности для положительно определенных матриц (рис. 4).

Было сделано предположение, что данные значения являются флуктуациями, вызванными в ходе вычислительной погрешности. Действительно, вычисление числа обусловленности подразумевает вычисление обратной матрицы [1](Теорема 5.2.11):

$$K(A) = \|A\|\|A^{-1}\|. \quad (12)$$

Обратная матрица существует тогда и только тогда, когда матрица не является сингулярной.

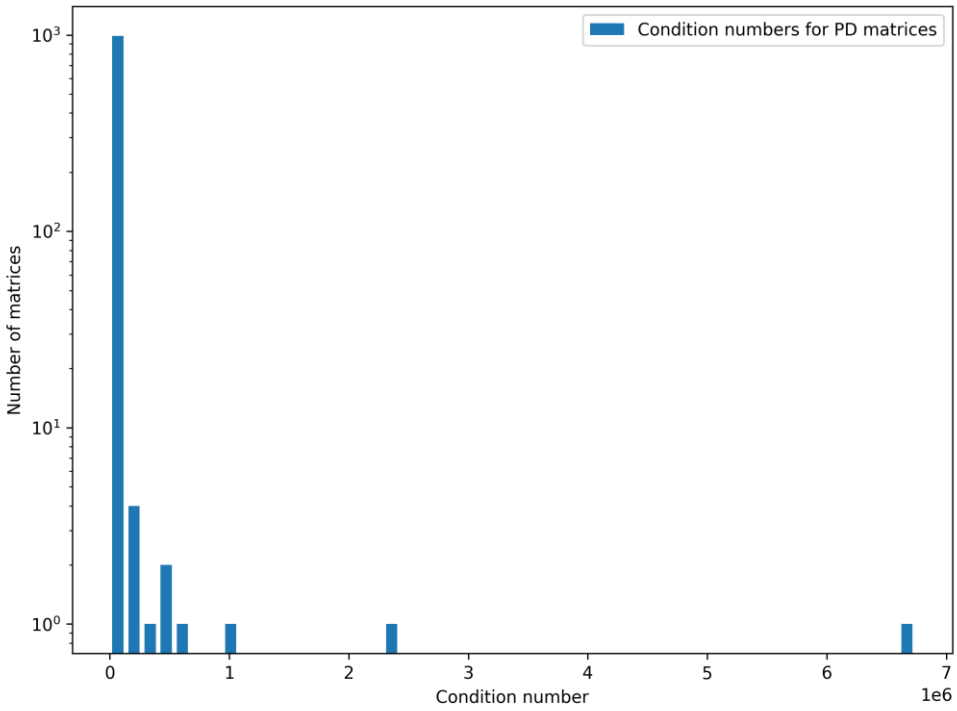

Рис. 4 - Распределение чисел обусловленности для случайно сгенерированных положительно определенных матриц (предварительный анализ).

Для каждого вида матриц была выведена обратная матрица и определитель исходной матрицы для максимального числа обусловленности:

```
Matrix for type default with max cond=6157.413167886158 is:
 [[-0.69803509 -0.87203465 -0.00442701 -0.07710671]
 [-0.49368933 -0.76208261  0.66035672  0.12724388]
 [ 0.67162325 -0.43693897  0.6181144   0.52310444]
 [ 0.81337314 -0.38477423  0.36854148  0.51484705]]
 Inverse matrix is:
 [[  269.32704812  -442.87457369   978.58047703  -844.48340881]
 [ -160.31797661   261.83560613  -578.32835138   498.88128687]
 [  140.83829013  -231.92826903   516.618017    -446.49017416]
 [ -646.12270395  1061.37360485 -2348.02181004  2028.53827184]]
 determinant is = -0.00026062133430467606
Matrix for type DP with max cond=29.53562629784669 is:
 [[-0.12304759 -0.02041533  0.06621525  0.03493015]
 [-0.9532408   2.20395423  0.04991701  0.54198512]
 [ 0.9365823  -0.93388225 -2.95406504  0.79080856]
 [-0.4995564   0.04817379 -0.72670208 -2.04516277]]
 Inverse matrix is:
 [[-7.3277975  -0.11302808 -0.11688708 -0.2003048 ]
 [-3.6310119   0.38704293 -0.0774573   0.01060347]
 [-0.65665282 -0.13547502 -0.31405985 -0.1685556 ]
 [ 1.93770374  0.08486327  0.13832064 -0.37988954]]
 determinant is = -2.037907303302476
```

а)

```
Matrix for type TD with max cond=22786.787123983315 is:
 [[ 0.98757736  0.02650991  0.          0.        ]
 [-0.6061328  -0.16432423 -0.87653618  0.        ]
 [ 0.          0.10337888  0.03121298 -0.5386669 ]
 [ 0.          0.         -0.52993018 -0.49094807]]
 Inverse matrix is:
 [[  113.37320632   183.07013208   261.92336047  -287.38160383]
 [-4185.78652014 -6819.9378084  -9757.4684009  10705.86798128]
 [  706.31073071  1150.7981197   1648.11006399 -1808.3019118 ]
 [ -762.39300024 -1242.17344151 -1778.97279734  1949.84727418]]
 determinant is = 4.3555136448110324e-05
Matrix for type PD with max cond=6738376.674616624 is:
 [[ 0.99957405 -0.07816271 -0.19743534  0.01978697]
 [-0.07816271  0.13579367  0.31092791 -0.01363632]
 [-0.19743534  0.31092791  0.7206352   0.03232295]
 [ 0.01978697 -0.01363632  0.03232295  0.48907231]]
 Inverse matrix is:
 [[ 4.12149721e+02 -4.48869487e+04  1.95949766e+04 -2.56325132e+03]
 [-4.48869487e+04  4.90120731e+06 -2.13954409e+06  2.79874694e+05]
 [ 1.95949766e+04 -2.13954409e+06  9.33985413e+05 -1.22174955e+05]
 [-2.56325132e+03  2.79874694e+05 -1.22174955e+05  1.59837993e+04]]
 determinant is = 6.766683518692716e-08
```

б)

Рис. 5 – Обратные матрицы и значения определителя для матрицы, соответствующей максимальному числу обусловленностей для случайно сгенерированных матриц общего вида и матриц с диагональным преобладанием (а) и для трехдиагональных и положительно определенных матриц (б).

Как видно из результатов рис. 5 для случаев с большим числом обусловленности обратные матрицы содержат сравнительно большие значения, чем исходные, а значение определителя исходной матрицы близко к нулю. Откуда можно сделать вывод, что при значении определителя матрицы, близкой нулю, т.е. при стремлении матрицы к сингулярной – возникают вычислительные погрешности, приводящие к большому числу обусловленности. Это вызвано тем, что функция, реализующая проверку на сингулярность (Листинг 5), использует в своем условии равенство, неприменимое для чисел с плавающей запятой. Во избежание подобной проблемы был введен коэффициент $\epsilon = 0.1$, который делает проверку на сингулярность более строгой, откуда функция проверки на сингулярность принимает вид, представленный в листинге 10.

Листинг 10 – измененная функция проверки матрицы на сингулярность.

```
def is_matrix_singular(A):
    return np.abs(np.linalg.det(A)) < SINGULAR_EPSILON
```

После данного изменения максимальные значения чисел обусловленности значительно уменьшились и их распределение приняло более равномерный вид, уменьшив в дальнейшем разброс погрешностей.

Для данного измененного условия проверки на сингулярность были заново сгенерированы 1000 матриц общего вида и заново вычислены значения спектральных радиусов, распределение которых представлено на рис. 6, и числа обусловленности, распределение которых представлено на рис. 7.

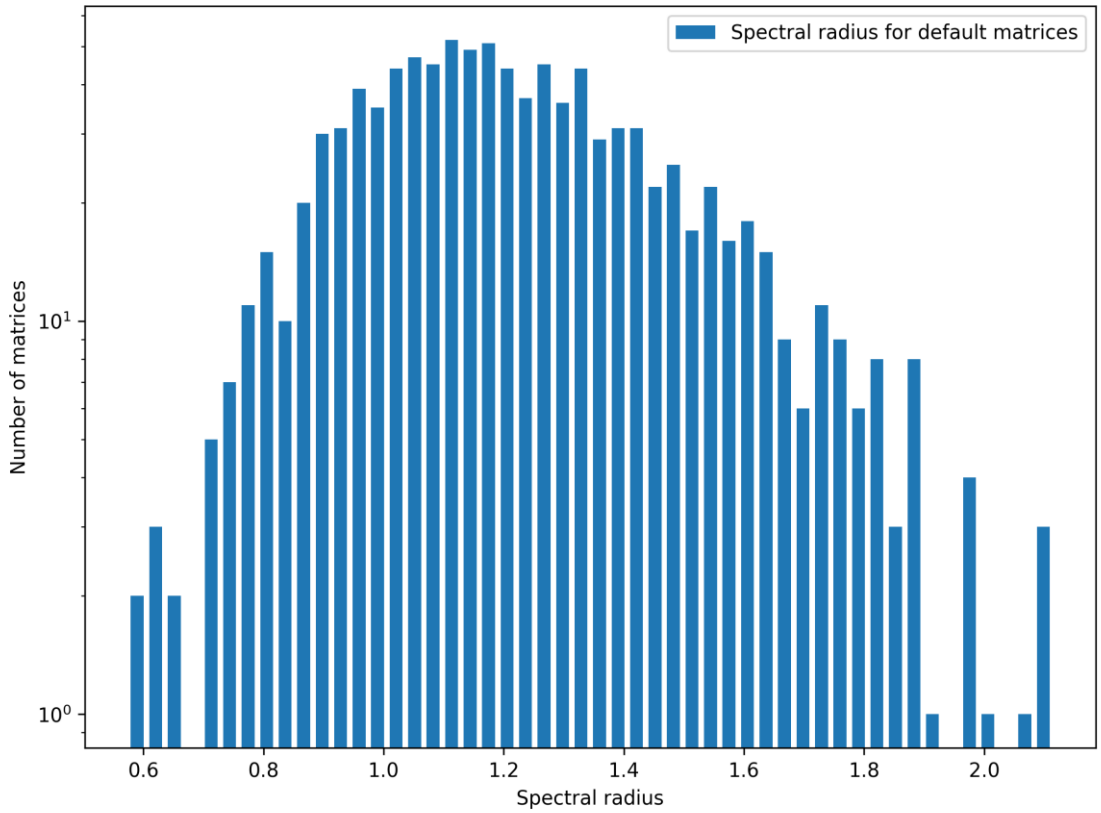

Рис. 6 - Распределение спектральных радиусов для случайно сгенерированных матриц общего вида с измененным условием проверки на сингулярность.

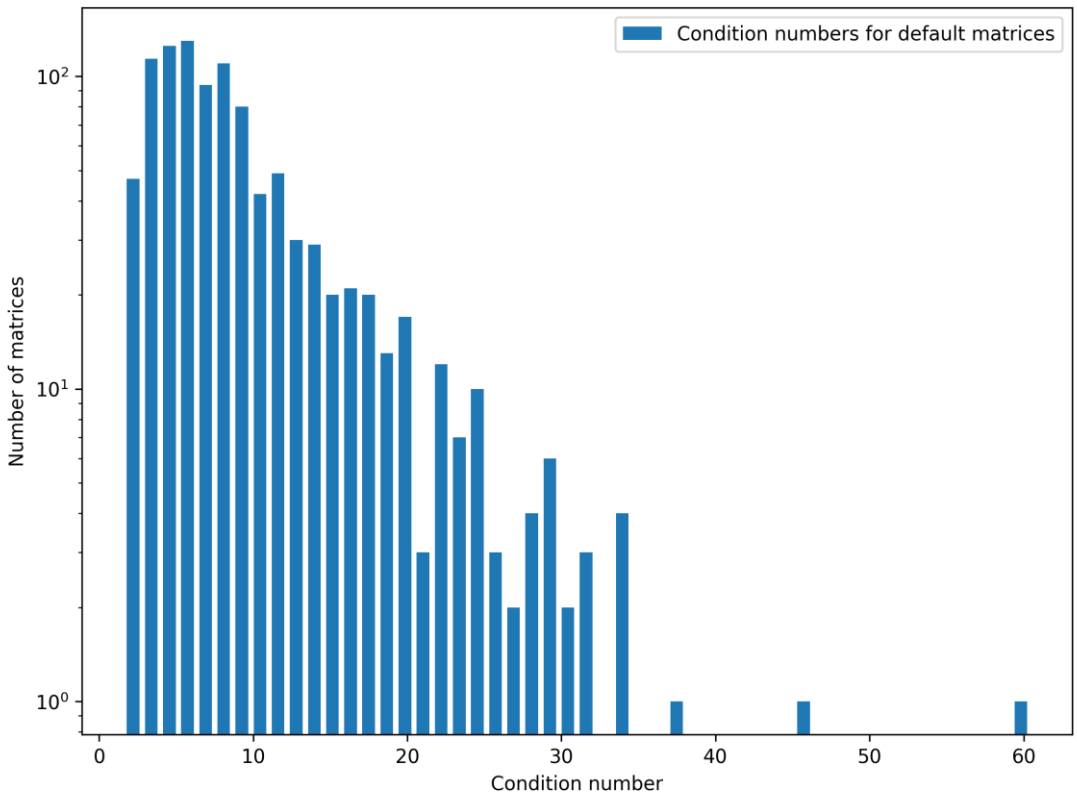

Рис. 7 - Распределение чисел обусловленности для случайно сгенерированных матриц общего вида с измененным условием проверки на сингулярность.

Распределение чисел обусловленности на рис. 7 является более равномерным и не имеет флуктуаций с большими значениями числа обусловленности, что позволяет сделать вывод, что введение точности в проверку матрицы на сингулярность, улучшило результаты исследования, избавив от большой вычислительной погрешности.

Также, для сравнения реализованных методов решения СЛАУ, были получены решения СЛАУ «точным» методом и методом «по умолчанию» и вычислены относительные погрешности с использованием среднеквадратичной и супремум норм. Распределение данных погрешностей представлены в виде гистограмм на рис. 8 и рис. 9.

Для поиска относительной погрешности с помощью среднеквадратичной нормы программно была реализована функция $rel\_err\_rms()$, реализующая вычисление погрешности по формуле:

$$\frac{\|\boldsymbol{x}-\tilde{\boldsymbol{x}}\|_2}{\|\boldsymbol{x}\|_2}=\sqrt{\frac{\sum_{i=1}^{n}(x_i-\tilde{x_i})^2}{\sum_{i=1}^{n}x_i^2}}, \tag{13}$$

где $\boldsymbol{x}$ – решение, полученное при помощи точного метода, а $\tilde{\boldsymbol{x}}$ – решение, полученное при помощи метода «по умолчанию».

Для поиска относительной погрешности с помощью супремум нормы программно была реализована функция $rel\_err\_supremum()$, реализующая вычисление погрешности по формуле:

$$\frac{\|\boldsymbol{x}-\tilde{\boldsymbol{x}}\|_\infty}{\|\boldsymbol{x}\|_\infty}=\frac{\max\limits_{i\in[1;n]}|x_i-\tilde{x_i}|}{\max\limits_{i\in[1;n]}|x_i|}. \tag{14}$$

Помимо погрешностей, были посчитаны отношения максимального по модулю собственного числа к минимальному по модулю собственному числу. Распределение данных отношений представлено на рис. 10.

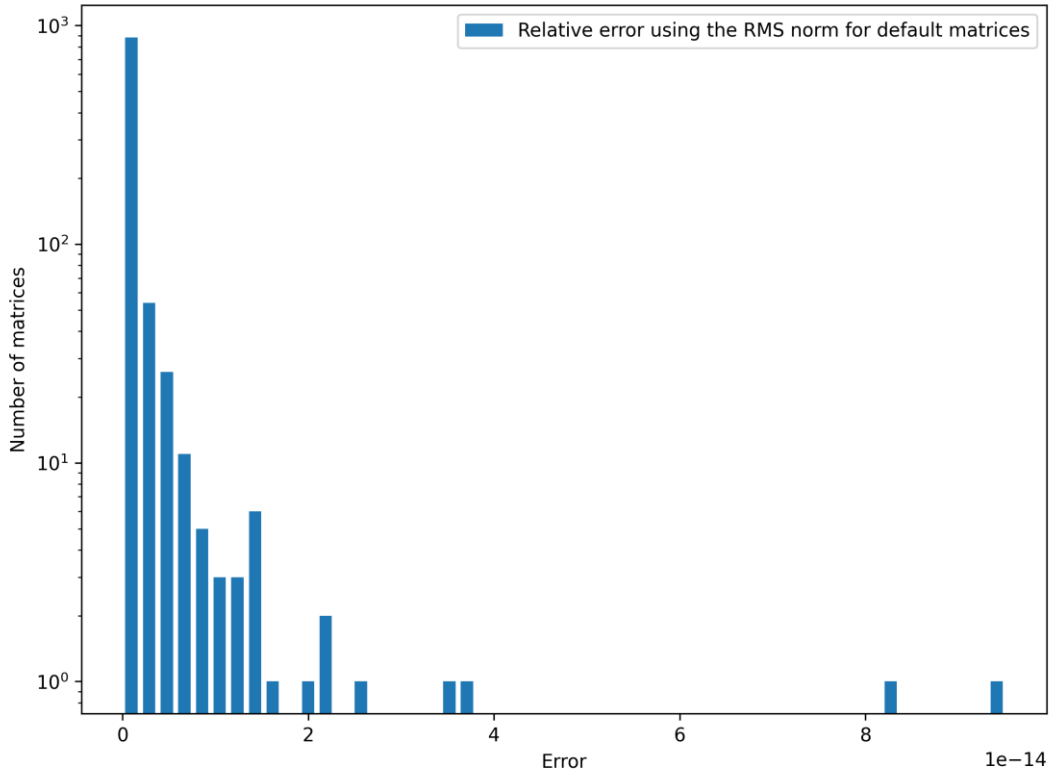

Рис. 8 - Распределение относительных погрешностей, полученных с помощью среднеквадратичной нормы, для случайно сгенерированных матриц общего вида.

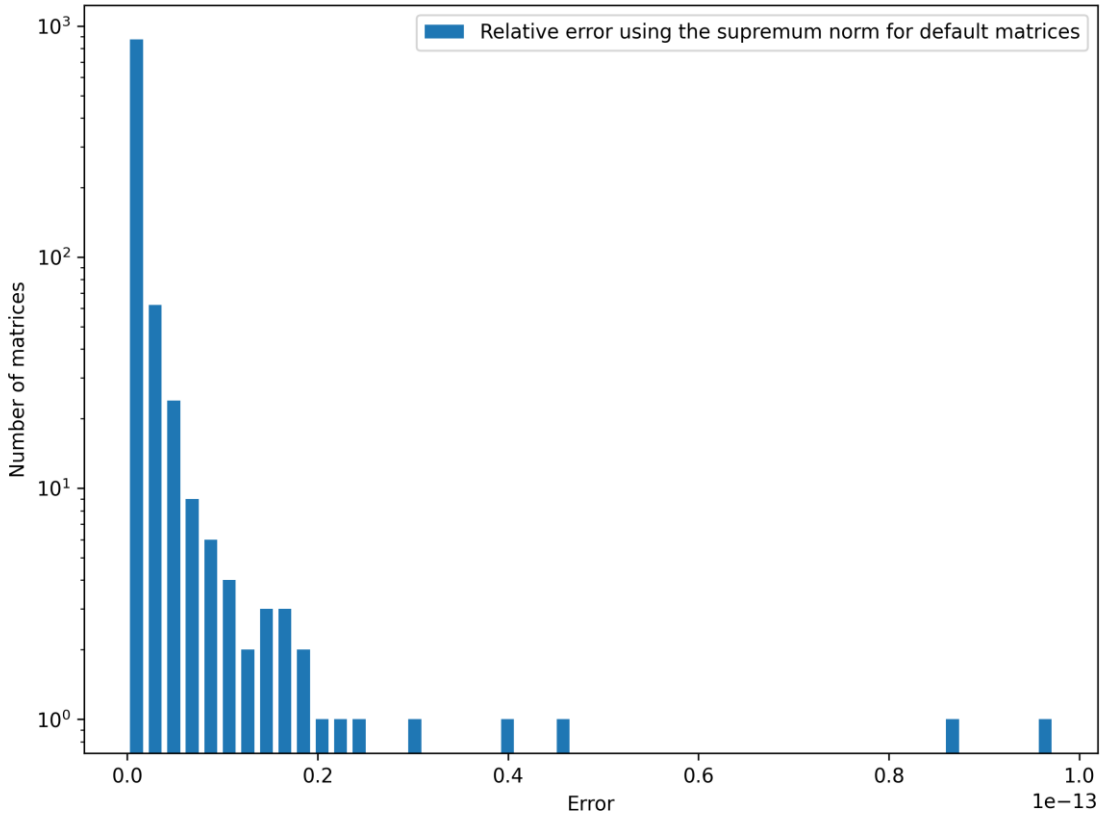

Рис. 9 - Распределение относительных погрешностей, полученных с помощью супремум нормы, для случайно сгенерированных матриц общего вида.

Метод Гаусса можно считать вычислительно неустойчивым, так как в ходе элементарных преобразований может возникнуть деление на очень малое число, что приведет к накоплению вычислительной погрешности. В этом случае, для некоторых матриц, в ходе элементарных преобразований которых возникло подобное деление, вычислительная погрешность будет больше, чем для остальных. Это демонстрирует распределение погрешностей на рис. 8-9. Для данной логарифмической шкалы распределение сконцентрированно в левой части, что говорит о том, что говорит о том, что для некоторых матриц значение погрешности возрастает на порядок и выше. Однако, порядок максимальной погрешности (10^{-14}) достаточно мал, что также можно объяснить равномерным распределением коэффициентов генерируемых матриц. Данный малый порядок максимальной погрешности позволяет сделать вывод, что метод Гаусса без выбора главного элемента является достаточно точным для рассматриваемого типа матриц.

Как будет замечено и далее, распределение чисел обусловленности напрямую влияет на погрешность, что можно аргументировать также и определением самого числа обусловленности, рассмотренным ранее. Для данного случая распределение числа обусловленностей также имеет в большей степени сконцентрированность в левой части, что также наблюдается и в распределении погрешностей. Помимо этого, максимальное значение числа обусловленности не сильно отличается от единицы, поэтому максимальное значение погрешности также не получается большим.

Спектральный радиус может быть связан с числом обусловленности через соотношение:

$$\rho(\boldsymbol{A}) \leq \|\boldsymbol{A}\| \tag{15}$$

и определение числа обусловленности (12). Оба этих выражения содержат норму матрицы A , откуда можно сделать вывод, что при больших значениях спектрального радиуса, а именно при большом отклонении от единицы, число обусловленности также возрастает, что приводит, в свою очередь, к увеличению погрешности.

На распределении спектральных радиусов на рис. 6 можно заметить, что значение спектрального радиуса, равного единицы, превышают около 1/4 всех матриц, что говорит о том, что распределение чисел обусловленности будет в своем большинстве в диапазоне, превышающем значение числа обусловленности, равного единице, что подтверждает распределение на рис.7.

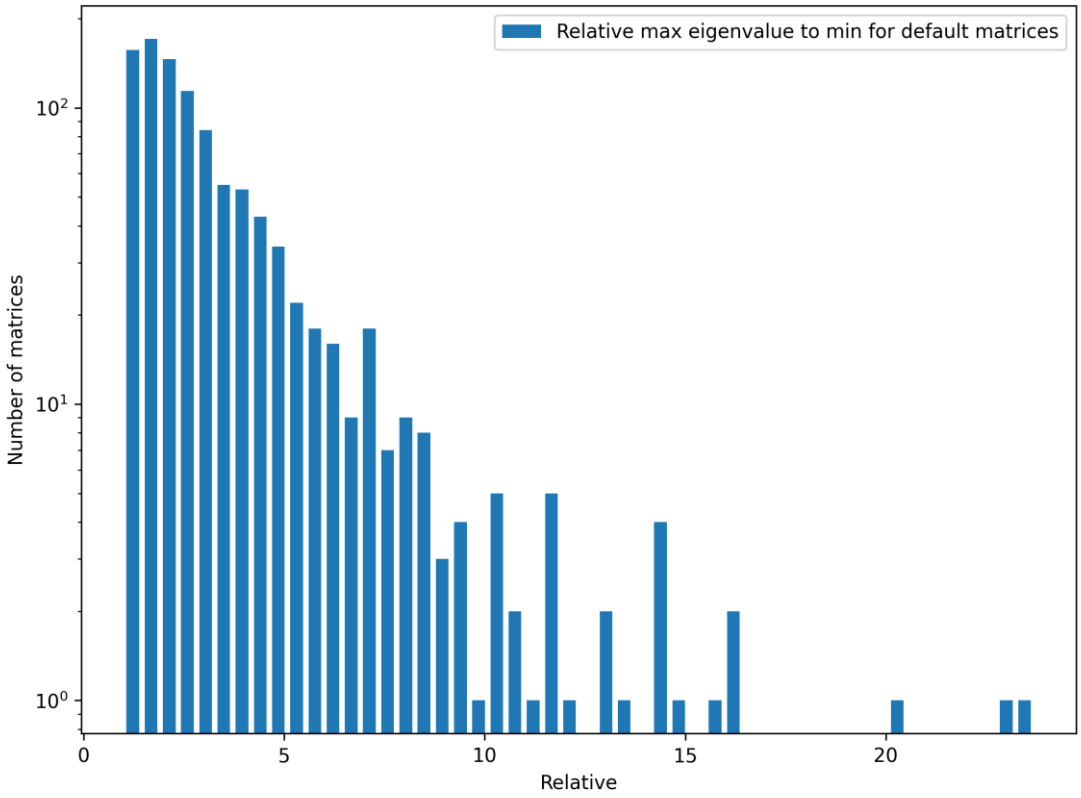

Рис. 10 - Распределение отношений максимального по модулю собственного числа к минимальному для случайно сгенерированных матриц общего вида.

Анализ распределения отношений максимального по модулю собственного числа к минимальному на рис. 10 показывает поведение распределение, схожее с распределением чисел обусловленности на рис. 7, из чего можно сделать вывод о близкой взаимосвязи данных отношений и чисел обусловленности.

7. Анализ решения СЛАУ для матриц со строгим диагональным преобладанием.

В качестве метода, минимизирующего количество арифметических операций (метода «по умолчанию») для матриц со строгим диагональным преобладанием был выбран метод Гаусса без выбора главного элемента. Из рассматриваемых методов для данного типа матриц подходят метод Гаусса и метод Гаусса с полным выбором главного элемента. Поскольку полный выбор главного элемента увеличивает количество арифметических и логических операций, метод Гаусса без выбора главного элемента лучше удовлетворяет условию метода «по умолчанию»

Для анализа решений СЛАУ для матриц общего вида, а также для анализа характеристик таких матриц, с помощью функции *generate_rand_matrix* были

сгенерированы 1000 случайных матриц $\boldsymbol{A}^{(j)}$ со строгим диагональным преобладанием.

Затем, с помощью реализованной функции *spectral_radius* были вычислены значения спектральных радиусов для каждой матрицы.

Распределение спектральных радиусов для сгенерированных матриц представлено в виде гистограммы на рис. 11.

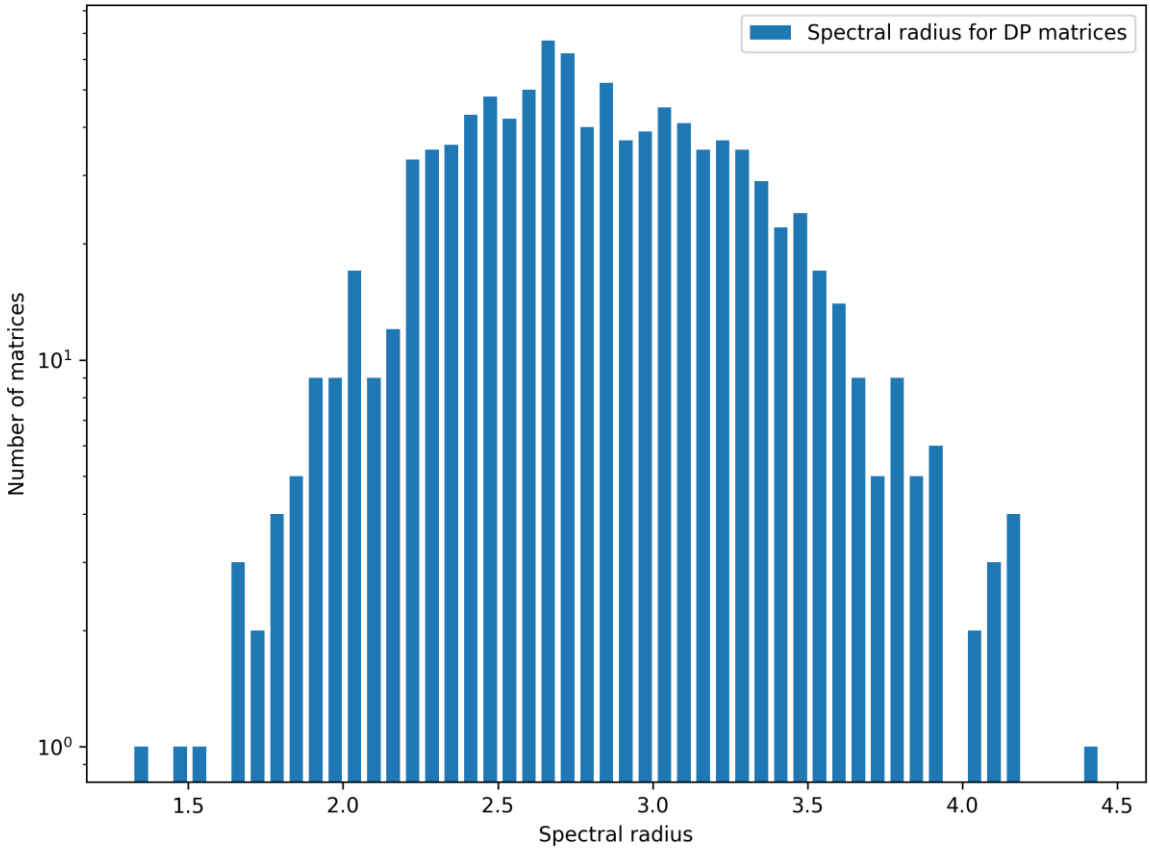

Рис. 11 - Распределение спектральных радиусов для случайно сгенерированных матриц со строгим диагональным преобладанием.

Также для каждой матрицы было получено число обусловленности, вычисленной с помощью библиотечной функции $numpy.linalg.cond()$.

Распределение чисел обусловленности для сгенерированных матриц представлено в виде гистограммы на рис. 12.

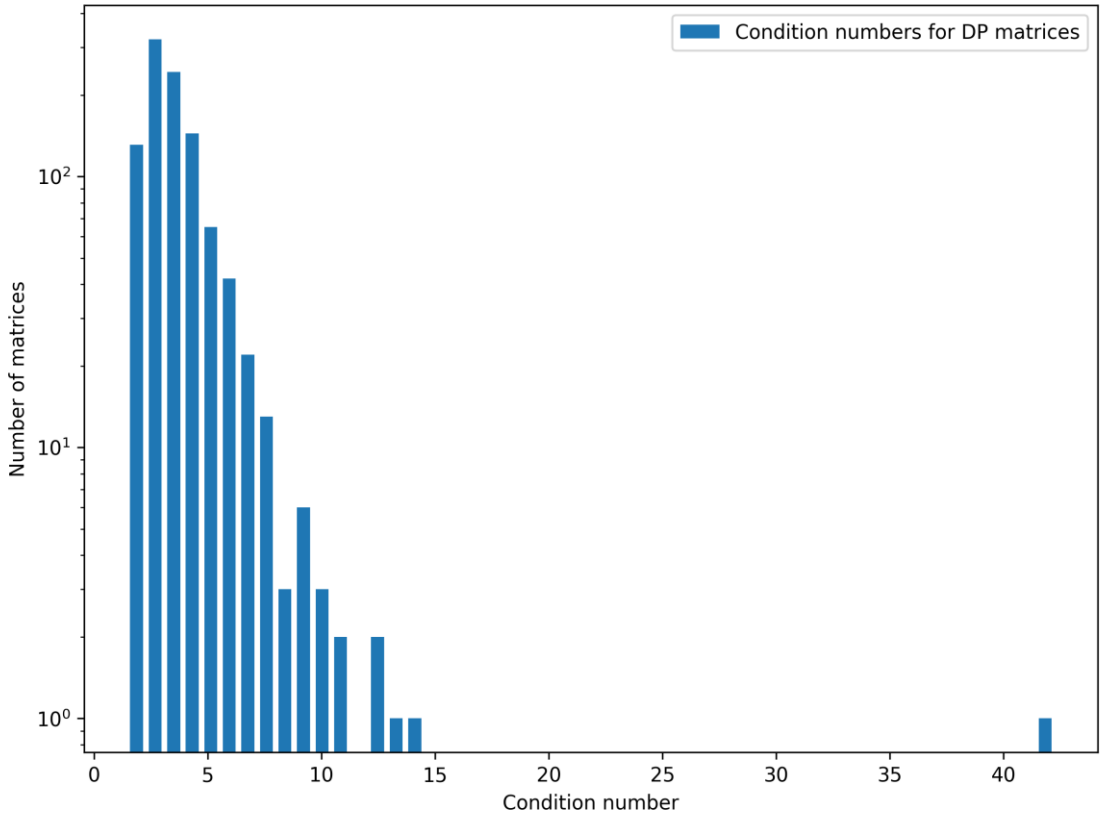

Рис. 12 - Распределение чисел обусловленности для случайно сгенерированных матриц со строгим диагональным преобладанием.

Также, для сравнения реализованных методов решения СЛАУ, были получены решения СЛАУ «точным» методом и методом «по умолчанию» и вычислены относительные погрешности с использованием среднеквадратичной и супремум норм. Распределение данных погрешностей представлены в виде гистограмм на рис. 13 и рис. 14.

Помимо погрешностей, были посчитаны отношения максимального по модулю собственного числа к минимальному по модулю собственному числу. Распределение данных отношений представлено на рис. 15.

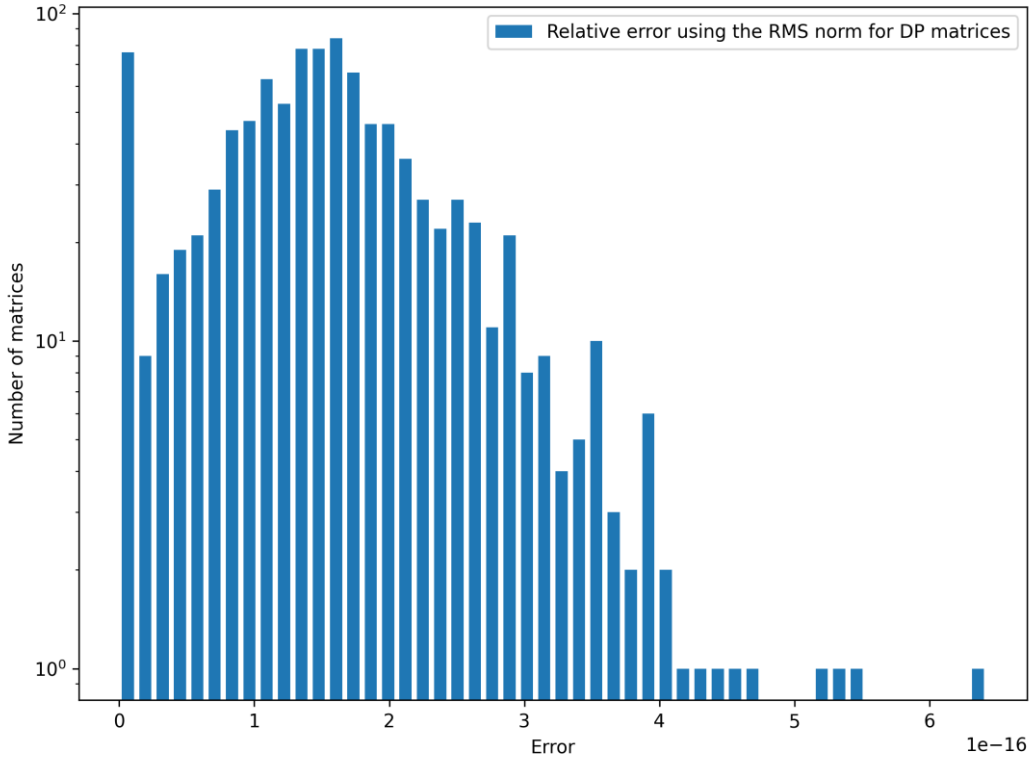

Рис. 13 - Распределение относительных погрешностей, полученных с помощью среднеквадратичной нормы, для случайно сгенерированных матриц со строгим диагональным преобладанием.

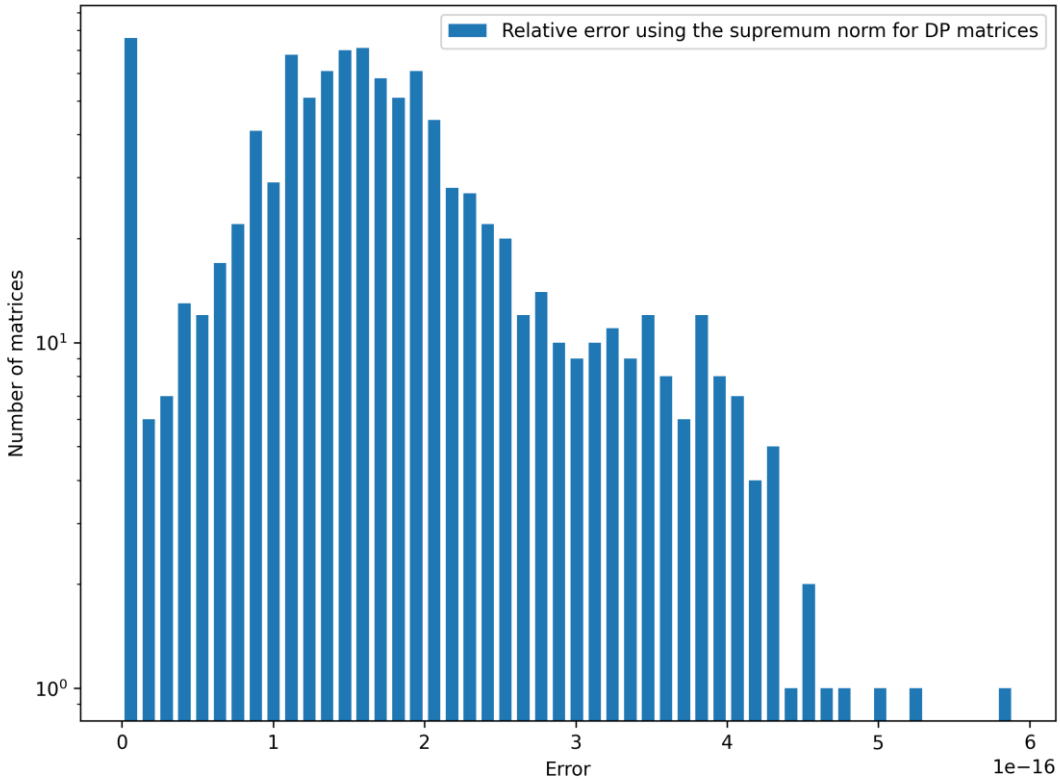

Рис. 14 - Распределение относительных погрешностей, полученных с помощью супремум нормы, для случайно сгенерированных матриц со строгим диагональным преобладанием.

Для матриц со строгим диагональным преобладанием распределение погрешностей (рис 13-14) по сравнению с распределениями для матриц общего вида (рис. 8-9) имеет более равномерный характер, что говорит о большей вычислительной устойчивости метода Гаусса без выбора главного элемента для матриц с диагональным преобладанием, чем для матриц общего вида. Помимо этого, порядок максимальной погрешности также меньше на 3 порядка, чем для случая матриц общего вида, что говорит о большей точности метода Гаусса для матриц со строгим диагональным преобладанием.

Это подтверждается и более узким диапазоном чисел обусловленности для данных матриц, что видно из их распределения на рис. 12. Помимо этого, уменьшается количество скачков данного числа, и их максимальное значение. Это прямым образом отражается на поведении погрешностей (диапазон чисел обусловленности уже – погрешности равномернее – метод устойчивее). И действительно, метод Гаусса не требует перестановок для матриц со строгим диагональным преобладанием, что увеличивает точность.

Однако, распределение спектральных радиусов на рис. 11 имеет более широкий диапазон значений, чем для матриц общего вида. Данное несовпадение поведений спектральных радиусов и чисел обусловленности можно отнести к вычислительной погрешности определения собственных чисел для матриц со строгим диагональным преобладанием, поскольку спектральный радиус напрямую зависит от собственных чисел. Однако само распределение имеет более равномерный характер, что также говорит о большей устойчивости. Также необходимо учитывать, что «точным» методом в контексте данной лабораторной работы был принят метод Гаусса с полным выбором главного элемента и поскольку для трехдиагональных матриц не требуются перестановки, фактически сравниваются решения для двух одинаковых методов, поэтому значения погрешностей являются минимальными.

Распределение отношений максимального по модулю собственного числа к минимальному (рис. 15) имеет более узкий диапазон, чем для случая матриц общего вида, что может влиять на более равномерное распределение спектральных радиусов и также на более равномерное распределение погрешности и ее меньший порядок.

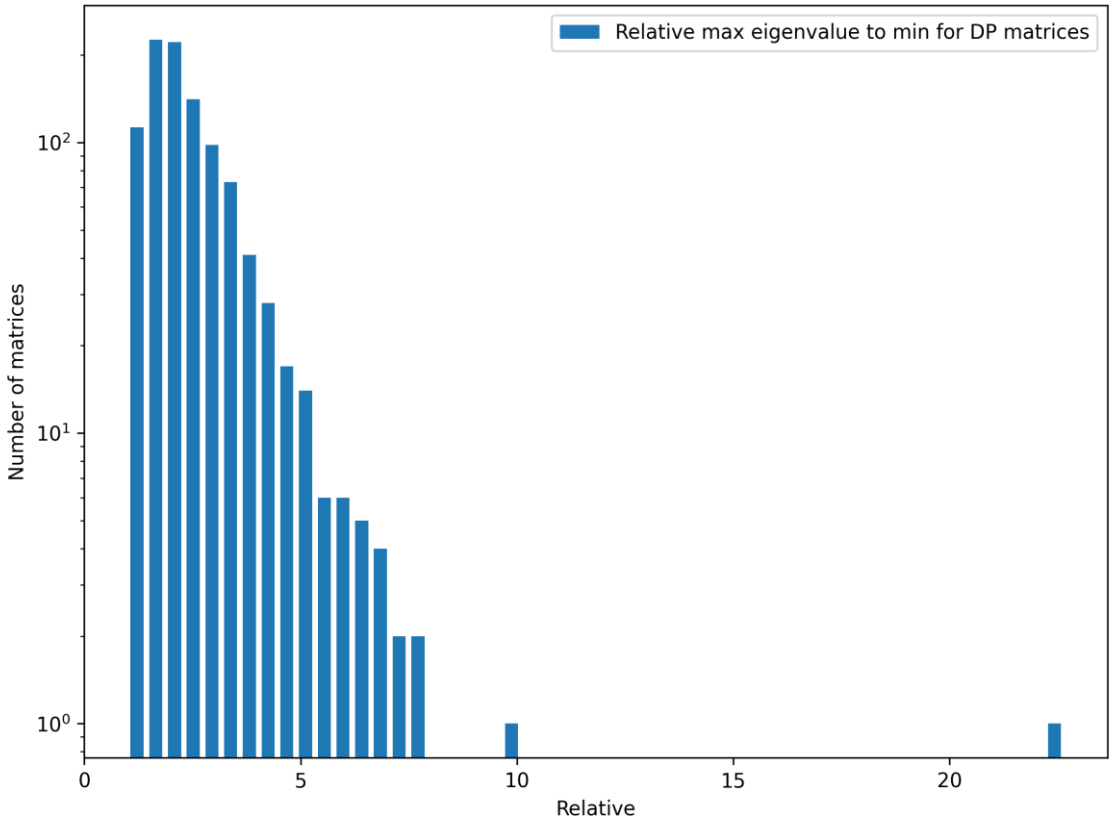

Рис. 15 - Распределение отношений максимального по модулю собственного числа к минимальному для случайно сгенерированных матриц со строгим диагональным преобладанием.

8. Анализ решения СЛАУ для трехдиагональных матриц.

В качестве метода, минимизирующего количество арифметических операций (метода «по умолчанию») для матриц со строгим диагональным преобладанием был выбран метод прогонки.

Из рассматриваемых методов для данного типа матриц подходят метод Гаусса, метод Гаусса с полным выбором главного элемента и метод прогонки. Метод прогонки обладает вычислительной сложностью $O(n)$, в то время как метод Гаусса обладает сложностью $O(n^3)$. Исходя из данных соображений, метод прогонки является самым эффективным для решения СЛАУ с трехдиагональными матрицами

Для анализа решений СЛАУ для матриц общего вида, а также для анализа характеристик таких матриц, с помощью функции *generate_rand_matrix* были сгенерированы 1000 случайных матриц $\boldsymbol{A}^{(j)}$ со строгим диагональным преобладанием.

Затем, с помощью реализованной функции *spectral*_radius были вычислены значения спектральных радиусов для каждой матрицы.

Распределение спектральных радиусов для сгенерированных матриц представлено в виде гистограммы на рис. 16.

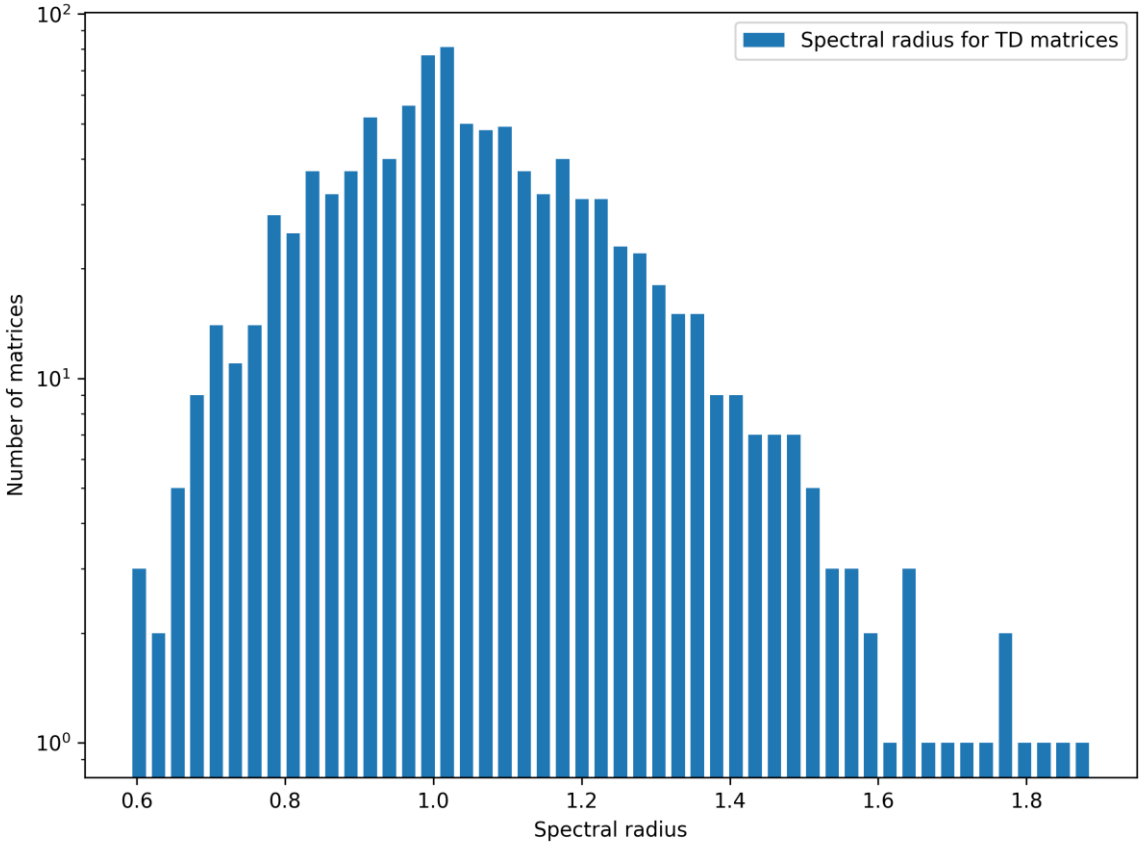

Рис. 16 - Распределение спектральных радиусов для случайно сгенерированных трехдиагональных матриц.

Также для каждой матрицы было получено число обусловленности, вычисленной с помощью библиотечной функции $numpy.linalg.cond()$.

Распределение чисел обусловленности для сгенерированных матриц представлено в виде гистограммы на рис. 17.

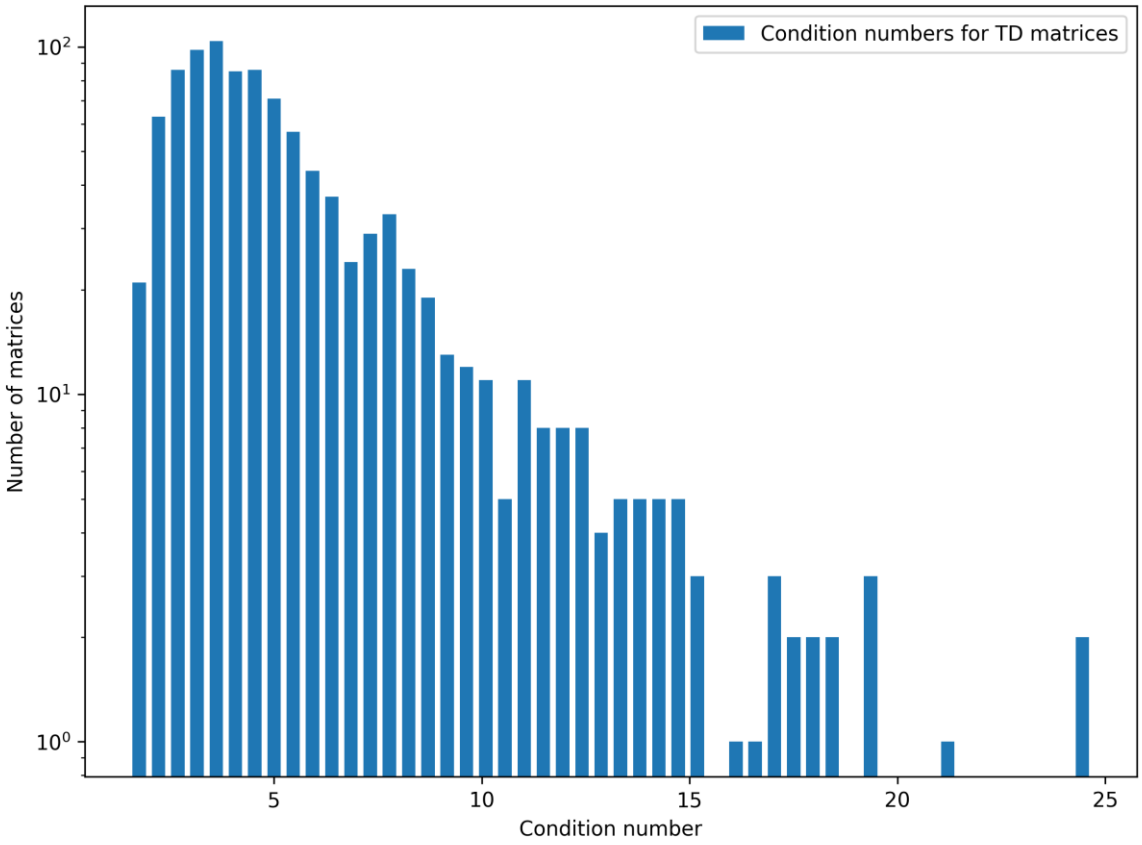

Рис. 17 - Распределение чисел обусловленности для случайно сгенерированных трехдиагональных матриц.

Также, для сравнения реализованных методов решения СЛАУ, были получены решения СЛАУ «точным» методом и методом «по умолчанию» и вычислены относительные погрешности с использованием среднеквадратичной и супремум норм. Распределение данных погрешностей представлены в виде гистограмм на рис. 18 и рис. 19.

Помимо погрешностей, были посчитаны отношения максимального по модулю собственного числа к минимальному по модулю собственному числу. Распределение данных отношений представлено на рис. 20.

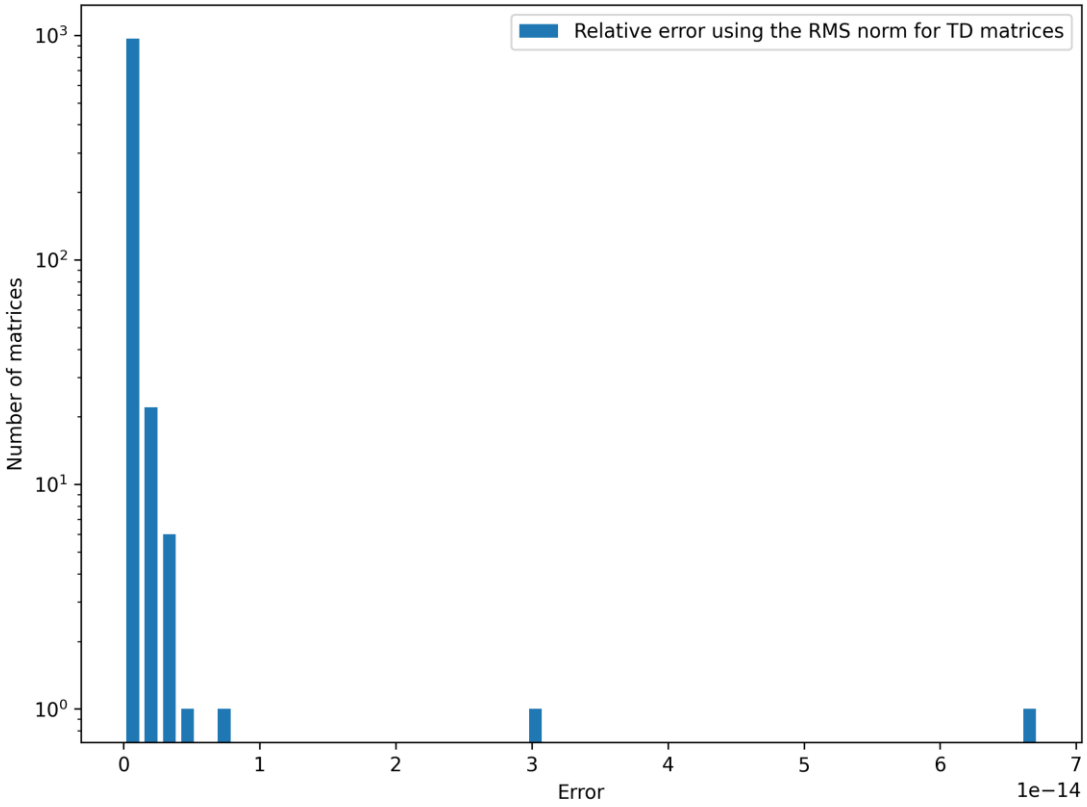

Рис. 18 - Распределение относительных погрешностей, полученных с помощью среднеквадратичной нормы, для случайно сгенерированных трехдиагональных матриц.

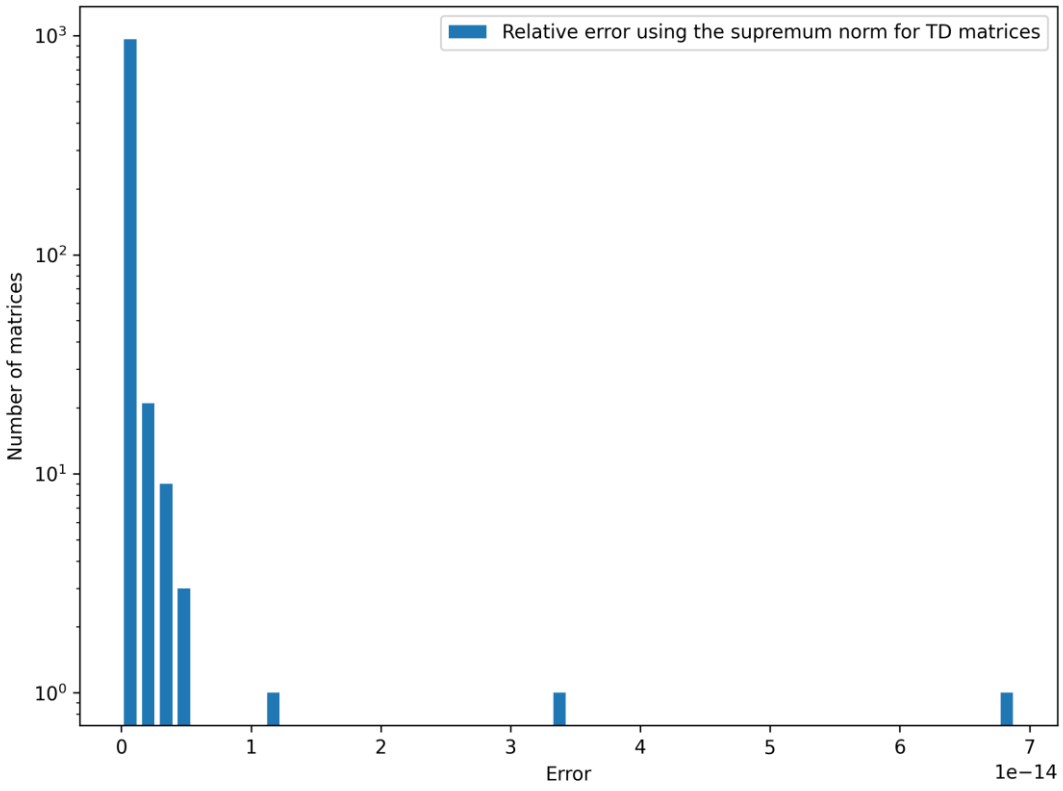

Рис. 19 - Распределение относительных погрешностей, полученных с помощью супремум нормы, для случайно сгенерированных трехдиагональных матриц.

Метод прогонки также, как и метод Гаусса, можно считать вычислительно неустойчивым так как в обоих методах происходит накопление вычислительной погрешности, вызванной делением. В методе прогонки данное накопление возникает в силу рекурентного выражения (4) для определения компонентов вектора решения. В этом случае, для некоторых матриц, вычислительная погрешность будет больше, чем для остальных. Это демонстрирует распределение погрешностей на рис. 18-19. Анализ данных результатов показывает, что возникают отклонения от распределения, которые демонстрируют вычислительную неустойчивость метода. Однако, порядок максимальной погрешности (10^{-15}) достаточно мал и является меньшим, чем для метода Гаусса для матриц общего вида, что говорит о том, что метод прогонки является более точным, поскольку накопленная вычислительная погрешность меньше, так как операций делений для данного метода требуется меньше.

Малый порядок погрешности подтверждается более узким диапазоном чисел обусловленности для данных матриц, что видно из их распределения на рис. 17. Однако, для некоторых матриц присутствуют значительные отклонения числа обусловленности от основного распределения, что выражается в отклонениях погрешности.

На распределении спектральных радиусов на рис. 16 можно заметить, что значение спектрального радиуса, равного единицы, превышают около 1/3 всех матриц, что говорит о том, что распределение чисел обусловленности будет в своем большинстве в диапазоне, превышающем значение числа обусловленности, равного единице, что подтверждает распределение на рис. 17.

Распределение отношений максимального по модулю собственного числа к минимальному (рис. 18) имеет более узкий диапазон, чем для случая матриц общего вида, что влияет на меньшие значения спектральных радиусов и более высокую точность решения. Однако, отклонения от равномерного распределения для данных отношений также присутствует, откуда можно проследить, что для рассматриваемых величин при неравномерном распределении одной и возникновении в ее распределении значительных отклонений – подобные отклонения будут возникать и для всех остальных распределений.

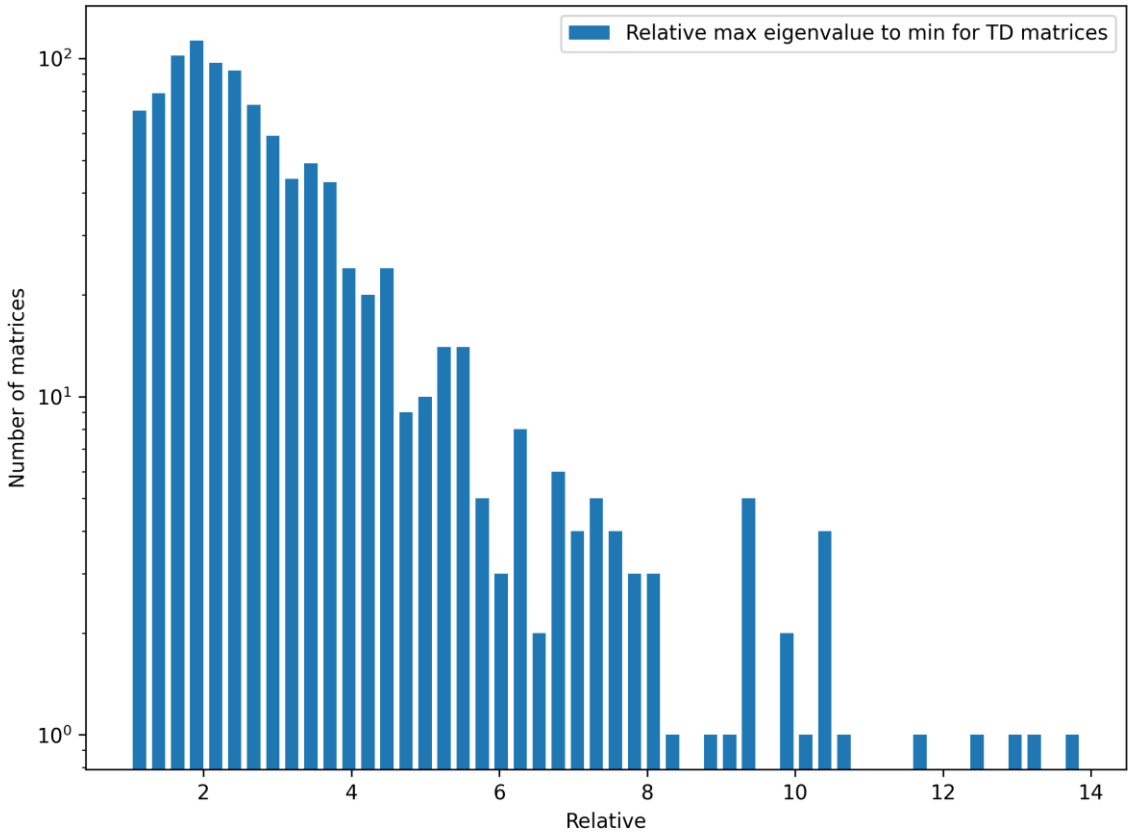

Рис. 20 - Распределение отношений максимального по модулю собственного числа к минимальному для случайно сгенерированных трехдиагональных матриц.

9. Анализ решения СЛАУ для положительно определенных матриц.

В качестве метода, минимизирующего количество арифметических операций (метода «по умолчанию») для матриц со строгим диагональным преобладанием был выбран метод разложения Холецкого.

Из рассматриваемых методов для данного типа матриц подходят метод Гаусса, метод Гаусса с полным выбором главного элемента и метод Холецкого. Метод Холецкого, в силу использования свойств разложения Холецкого требует меньше операций, чем метод Гаусса, поэтому был выбран в качестве метода «по-умолчанию»

Для анализа решений СЛАУ для матриц общего вида, а также для анализа характеристик таких матриц, с помощью функции *generate_rand_matrix* были сгенерированы 1000 случайных матриц $\boldsymbol{A}^{(j)}$ со строгим диагональным преобладанием.

Затем, с помощью реализованной функции *spectral*_radius были вычислены значения спектральных радиусов для каждой матрицы.

Распределение спектральных радиусов для сгенерированных матриц представлено в виде гистограммы на рис. 21.

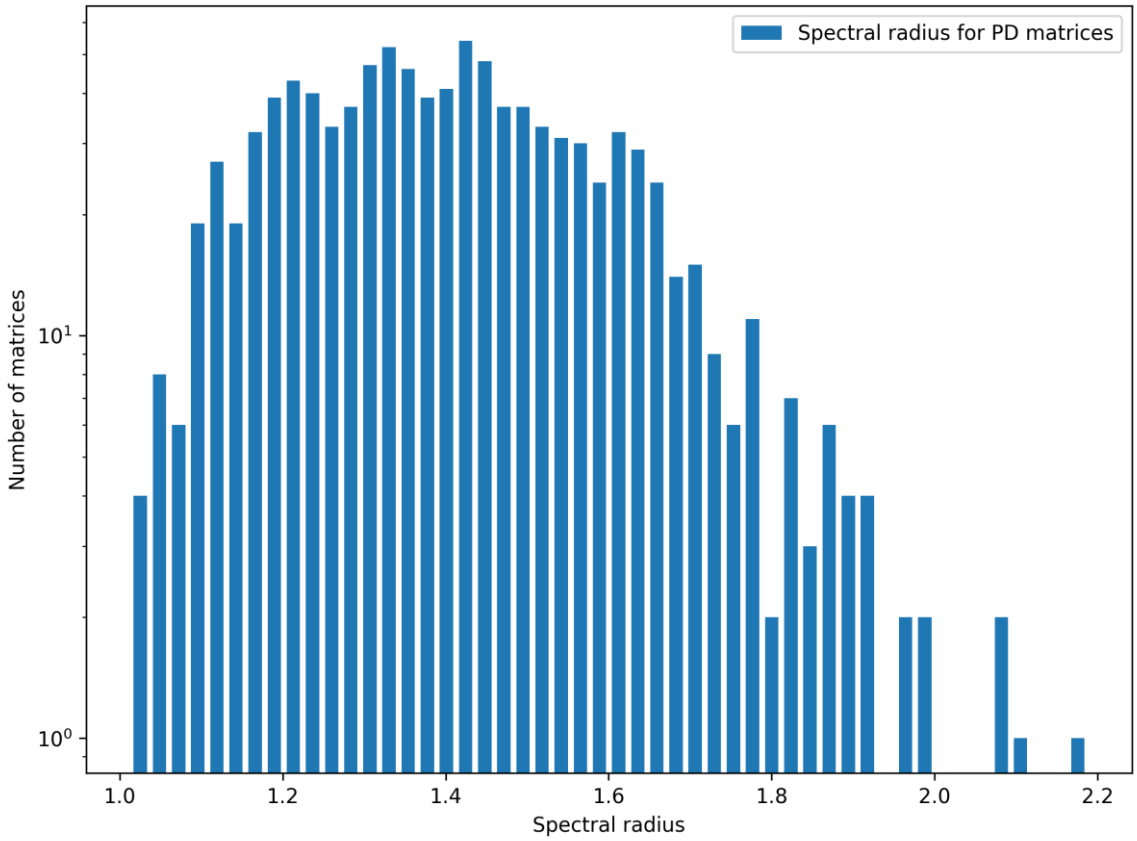

Рис. 21 - Распределение спектральных радиусов для случайно сгенерированных положительно определенных матриц.

Также для каждой матрицы было получено число обусловленности, вычисленной с помощью библиотечной функции $numpy.linalg.cond()$.

Распределение чисел обусловленности для сгенерированных матриц представлено в виде гистограммы на рис. 22.

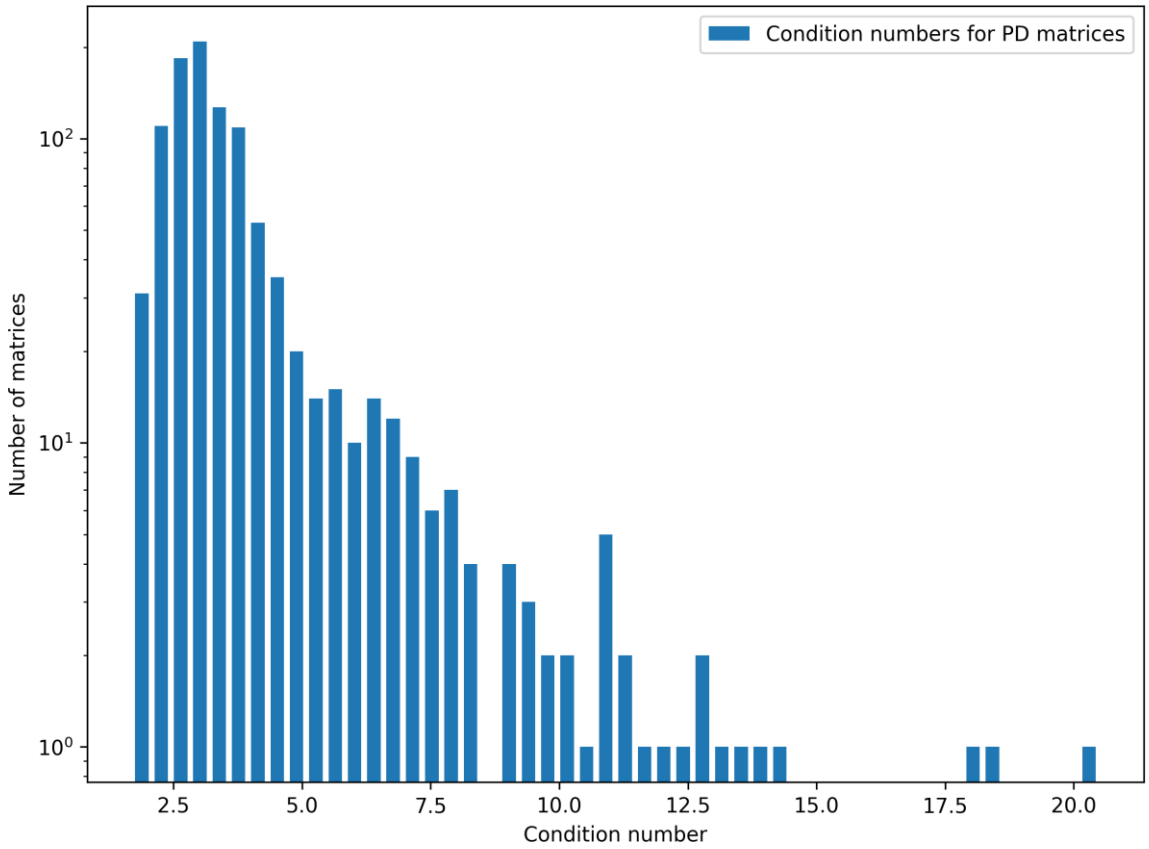

Рис. 22 - Распределение чисел обусловленности для случайно сгенерированных положительно определенных матриц.

Также, для сравнения реализованных методов решения СЛАУ, были получены решения СЛАУ «точным» методом и методом «по умолчанию» и вычислены относительные погрешности с использованием среднеквадратичной и супремум норм. Распределение данных погрешностей представлены в виде гистограмм на рис. 23 и рис. 24.

Помимо погрешностей, были посчитаны отношения максимального по модулю собственного числа к минимальному по модулю собственному числу. Распределение данных отношений представлено на рис. 25.

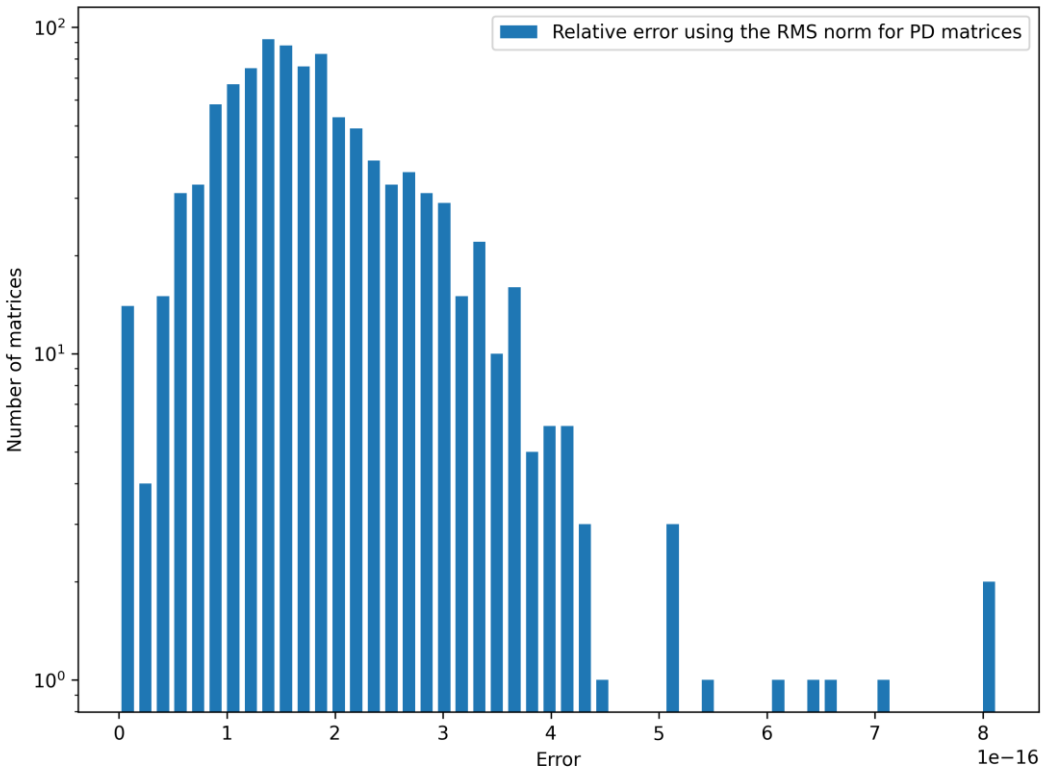

Рис. 23 - Распределение относительных погрешностей, полученных с помощью среднеквадратичной нормы, для случайно сгенерированных положительно определенных матриц.

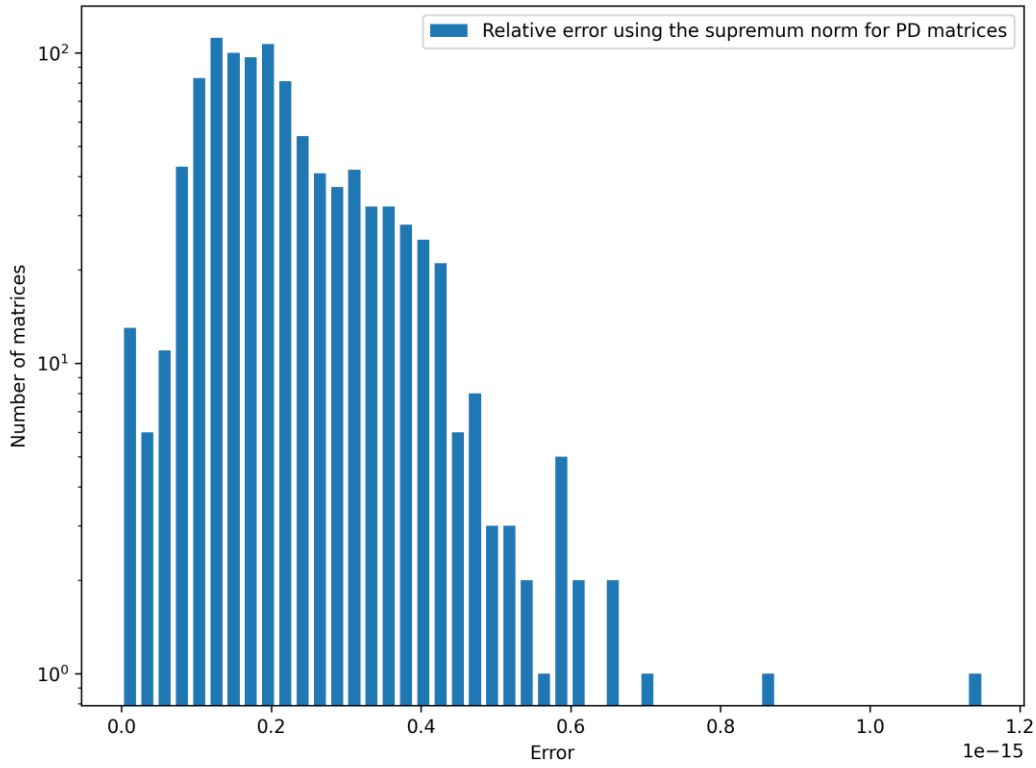

Рис. 24 - Распределение относительных погрешностей, полученных с помощью супремум нормы, для случайно сгенерированных положительно определенных матриц.

Результаты распределения погрешностей на рис. 23-24 имеют достаточно равномерный характер, с незначительными отклонениями от основного распределения. Это говорит о достаточной устойчивости метода для рассматриваемого вида матриц. Порядок максимальной погрешности (10^{-16}) является более меньшим, чем для метода Гаусса, что говорит о том, что метод разложения Холецкого является более точным для положительно определенных матриц. Это можно аргументировать тем, что разложение Холецкого требует меньше операций деления и в нем отсутствует накопление вычислительной погрешности, что делает метод устойчивым.

Малый порядок погрешности подтверждается более узким диапазоном чисел обусловленности для данных матриц с незначительными отклонениями, что видно из их распределения на рис. 22.

Несмотря на то, что распределение спектральных радиусов на рис. 21 начинается со значения, большего единицы, в силу более равномерного распределения спектральных радиусов, возникает более равномерное распределение погрешностей.

Распределение отношений максимального по модулю собственного числа к минимальному (рис. 15) имеет более узкий диапазон, чем для случая матриц общего вида и показывает поведение, близкое к поведению погрешностей.

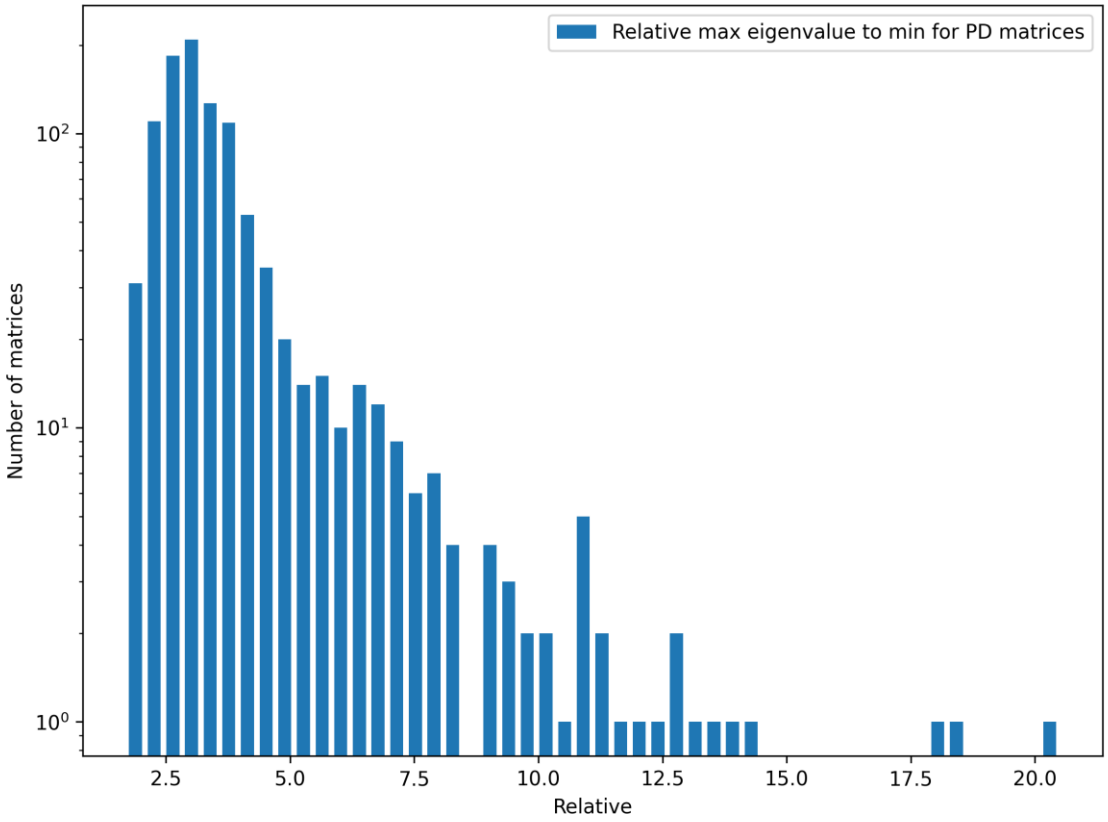

Рис. 25 - Распределение отношений максимального по модулю собственного числа к минимальному для случайно сгенерированных положительно определенных матриц.

Заключение

В ходе лабораторной работы были проанализированы прямые методы решения СЛАУ: метод Гаусса, метод разложения Холецкого и метод прогонки. Данные методы были проанализированы на предмет устойчивости и погрешности для матриц общего вида, матриц с диагональным преобладанием, трехдиагональных матриц и положительно определенных матриц.

Было обнаружено, что для конкретного вида матриц и методов, специализированных на получении решения СЛАУ для данных конкретных матриц, наблюдается увеличение точности решения, которое аргументируется более низкой погрешностью вычислений. Так, метод разложения Холецкого и метод прогонки демонстрируют более низкие порядки погрешности по сравнению с общим методом Гаусса. А матрицы со строгим диагональным преобладанием увеличивают точность метода Гаусса без выбора главного элемента по причине уменьшения накопления вычислительной погрешности, что было рассмотрено в п.7. Однако, также стоит отметить, что в качестве «точного» метода был выбран метод Гаусса с полным выбором главного

элемента, который также имеет свою погрешность. Поэтому в рамках данной лабораторной работы малая погрешность обозначает приближение к эталонному решению, полученному в ходе решения СЛАУ методом Гаусса с полным выбором элемента.

Также было продемонстрировано, что метод прогонки для трехдиагональных матриц и метод Гаусса без выбора главного элемента для матриц общего вида являются неустойчивыми, что выражается в больших отклонениях погрешности для определенных матриц от основного распределения. Метод Холецкого и метод Гаусса без выбора главного элемента для трехдиагональных матриц являются устойчивыми, что демонстрируется более равномерным распределением рассматриваемых величин и погрешностей.

Числа обусловленности напрямую влияют на погрешность решения – чем отклонений в распределении – тем их больше в погрешностях, и тем она более неравномерная; чем меньше значение числа обусловленности – тем меньше порядок погрешности.

На равномерность решения также влияют и спектральные радиусы, и отношение максимального собственного числа к минимальному – при хаотичном или неравномерном распределении данных величин – такая же хаотичность и неравномерность наблюдается для погрешностей и чисел обусловленности.

Из данного исследования можно сделать вывод, что для конкретной задачи решения СЛАУ следует выбирать соответствующий метод, который минимизирует погрешность при таковой возможности.

Список использованных источников

1. **Першин А.Ю.** *Лекции по вычислительной математике (черновик)*. [Электронный ресурс] // Кафедра РК6 (Системы автоматизированного проектирования). МГТУ им. Н.Э. Баумана, Москва, 2020 г., 142 с.
2. *Triangular matrix* [Электронный ресурс] // https://en.wikipedia.org/wiki/Triangular_matrix - (Дата обращения: 03.06.2020)
3. *Положительно определенная матрица*. [Электронный ресурс] // https://ru.wikipedia.org/wiki/Положительно_определенная_матрица - (Дата обращения: 04.06.2020).
4. *Критерий Сильвестра*. [Электронный ресурс] // https://ru.wikipedia.org/wiki/Критерий_Сильвестра - (Дата обращения: 04.06.2020).
5. *Positive semidefinite and positive definite matrice*. // Parameter Estimation for Scientists and Engineers by Adriaan van den Bos - 2007, Appendix C [Электронный ресурс]. URL: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470173862.app3 (Дата обращения: 04.06.2020).