**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №6**

**по дисциплине «Вычислительная математика»**

Тема: Метод простых итераций для СЛАУ. Метод Якоби

|  |  |  |
| --- | --- | --- |
| Студент гр. 3342 |  | Мохамед М. Х. |
| Преподаватель |  | Пуеров Г. Ю. |

Санкт-Петербург

2025

## Задание

В работе студенты должны найти решение системы линейных уравнений с n неизвестными, заданной матрицей коэффициентов и вектором свободных членов b, методом простых итераций. Выполнение работы включает следующие этапы:

1) с помощью преподавателя определить систему уравнений, которую нужно решить. Привести исходную систему к виду x = Ax + b, пригодному для использования метода простых итераций;

2) задать необходимую точность получения результата (количество знаков мантиссы числа);

3) разработать программу решения задачи на языке С++. Функция имеет следующие параметры:

- - матрица коэффициентов преобразованной к виду x = Ax+b системы уравнений размера n\*n, тип float;

- - вектор свободных членов преобразованной системы размера n, тип float;

- - полученный в результате проведения итераций вектор решения размера n, тип float;

- - размер системы уравнений, тип int ;

- - количество знаков после запятой в мантиссе результата, остающихся после округления, тип int ;

- it - выходной параметр, равный количеству произведенных итераций, тип int.

В качестве значения функции типа возвращается одно из следующих значений:

а) 0 - все нормально, получено решение ;

б) 1 - не выполняются условия сходимости итерационного процесса;

в) 2 - если n < 2 ;

г) 3 - если n > n\_max .

Константа должна быть задана при разработке головной программы аналогично тому, как это делается при выполнении лабораторной работы No 11;

4) произвести вычисления с использованием разработанной программы и построить график зависимости числа итераций от задаваемой точности.

**Теоретическая часть**

Рассмотрим систему линейных уравнений вида **x = Ax + b**, где **A** — заданная квадратная матрица порядка **n**, а **b** — известный вектор свободных членов. Для решения такой системы может применяться **метод простой итерации**.

Алгоритм метода заключается в следующем:

1. Выбирается произвольный начальный вектор **x⁰** (первое приближение).
2. Последующие приближения вычисляются по рекуррентной формуле:  
   **xᵏ = Axᵏ⁻¹ + b**,  
   где **k = 1, 2, 3, ...**

Согласно доказанной теореме, если **норма матрицы ||A|| < 1**, то система имеет **единственное решение**, а итерационный процесс **сходится** к нему со скоростью геометрической прогрессии.

Для оценки погрешности **k-го приближения** часто используется следующее неравенство:  
**||A|| / (1 - ||A||) · ||xᵏ - xᵏ⁻¹|| < ε**, где **ε** — заданная точность вычислений.

Это неравенство позволяет контролировать точность итерационного процесса и принимать решение о его завершении, когда достигнута требуемая погрешность.

**Использования библиотеки Eigen для анализа результатов**

Для анализа и решения систем линейных уравнений эффективно использовать библиотеку **Eigen** в C++. Она предоставляет удобные методы для вычисления числа обусловленности, нахождения точного решения системы, а также работы с нормами матриц.

Число обусловленности матрицы можно определить через её сингулярные числа (сингулярные значения), которые характеризуют степень растяжения или сжатия векторов при линейном преобразовании. Число обусловленности вычисляется в функции computeConditionNumber как отношение наибольшего сингулярного числа к наименьшему (cond(A) = σ\_max / σ\_min), что соответствует спектральной норме матрицы.

Для нахождения точного решения системы **Ax = b** использовались встроенные методы **Eigen.**

Для проверки корректности вычислений вычисляется норма разности между полученным решением и решением библиотеки Eigen.

### Описание алгоритма решения СЛАУ итерационным методом

Представленная реализация использует модифицированный метод Якоби для решения систем линейных алгебраических уравнений. Алгоритм начинается с проверки условий сходимости, включая анализ спектрального радиуса матрицы коэффициентов. В итерационном процессе последовательно уточняется приближенное решение по формуле x^(k+1) = Tx^(k) + c, где T - преобразованная матрица коэффициентов. Вычисления продолжаются до достижения заданной точности ε=10^(-ntch), что обеспечивает контроль погрешности на каждом шаге.

### Генерация тестовых данных

Функция generateRandomMatrix формирует случайную систему уравнений с гарантированной сходимостью метода. Особенностью реализации является искусственное создание диагонального преобладания путем увеличения диагональных элементов. Исходная система AX=B преобразуется в эквивалентную форму X=CX+D, где матрица C строится по правилу c\_ij = -a\_ij/a\_ii (при i≠j), а вектор D содержит нормированные свободные члены d\_i = b\_i/a\_ii. Такое преобразование сохраняет решение системы, но обеспечивает выполнение условий сходимости.

### Реализация метода Якоби

В программной реализации метод Якоби представлен функцией jacobiMethod, которая:

1. Проверяет норму матрицы C на соответствие критерию сходимости
2. Использует векторную операцию Xₖ₊₁ = C\*Xₖ + D для обновления решения
3. Контролирует точность по норме разности приближений
4. Поддерживает ограничение на максимальное число итераций
5. Выводит диагностическую информацию о ходе вычислений

Особенностью данной реализации является использование нормы матрицы для оценки скорости сходимости и автоматический расчет оптимального параметра точности на основе входного параметра precision. Для ускорения вычислений применяются оптимизированные матричные операции из библиотеки Eigen.

Тестирование программы приведено в таблице 1.

Таблица 1 – Результаты тестирования

|  |  |  |
| --- | --- | --- |
| № п/п | Входные данные | Выходные данные |
|  | Enter matrix size (n) and precision: 100 5 | Generating random matrix of size 100...  Matrix generation complete  Jacobi method started...  Target precision: 1e-05  Jacobi method completed successfully  Total iterations: 7  Total execution time: 0 ms  Solving system using Eigen...  Eigen solver completed in 17 ms  Solution comparison (Jacobi | Eigen):  x[0]: 0.0121969 | 0.0121968  x[1]: 0.00581773 | 0.00581773  x[2]: -0.00730181 | -0.0073018  x[3]: 0.00567037 | 0.00567038  x[4]: -0.00662349 | -0.0066235  x[5]: 0.0109838 | 0.0109838  x[6]: -0.012188 | -0.012188  x[7]: -0.011687 | -0.011687  x[8]: -0.00494718 | -0.00494718  x[9]: 0.00543809 | 0.0054381  ... (showing first 10 of 100 elements)  Norm of difference: 7.98184e-08  Computing condition number...  Condition number: 1.44852 |
|  | Enter matrix size (n) and precision: 1000 5 | Generating random matrix of size 1000...  Matrix generation complete  Jacobi method started...  Target precision: 1e-05  Jacobi method completed successfully  Total iterations: 6  Total execution time: 22 ms  Solving system using Eigen...  Eigen solver completed in 3219 ms  Solution comparison (Jacobi | Eigen):  x[0]: -0.00122185 | -0.00122185  x[1]: -0.000301882 | -0.000301881  x[2]: -0.000843715 | -0.000843716  x[3]: 0.00112612 | 0.00112612  x[4]: -8.38443e-05 | -8.38442e-05  x[5]: -0.000318558 | -0.000318557  x[6]: 0.00120953 | 0.00120953  x[7]: -0.000184004 | -0.000184003  x[8]: 0.000582182 | 0.000582182  x[9]: 0.000416188 | 0.000416189  ... (showing first 10 of 1000 elements)  Norm of difference: 2.09297e-08  Computing condition number...  Condition number: 1.13683 |
|  | Enter matrix size (n) and precision: 10 9 | Generating random matrix of size 10...  Matrix generation complete  Jacobi method started...  Target precision: 1e-09  Jacobi method completed successfully  Total iterations: 14  Total execution time: 0 ms  Solving system using Eigen...  Eigen solver completed in 3 ms  Solution comparison (Jacobi | Eigen):  x[0]: 0.0131417 | 0.0131417  x[1]: 0.127013 | 0.127013  x[2]: -0.0571522 | -0.0571522  x[3]: -0.124085 | -0.124085  x[4]: -0.166674 | -0.166674  x[5]: 0.0517071 | 0.0517071  x[6]: 0.0638886 | 0.0638886  x[7]: -0.120915 | -0.120915  x[8]: 0.151291 | 0.151291  x[9]: 0.048158 | 0.048158  Norm of difference: 2.29642e-08  Computing condition number...  Condition number: 1.99244 |

**Исследование зависимости количества итераций от точности**

График зависимости числа итераций метода Якоби от точности представлены на рисунке 1.

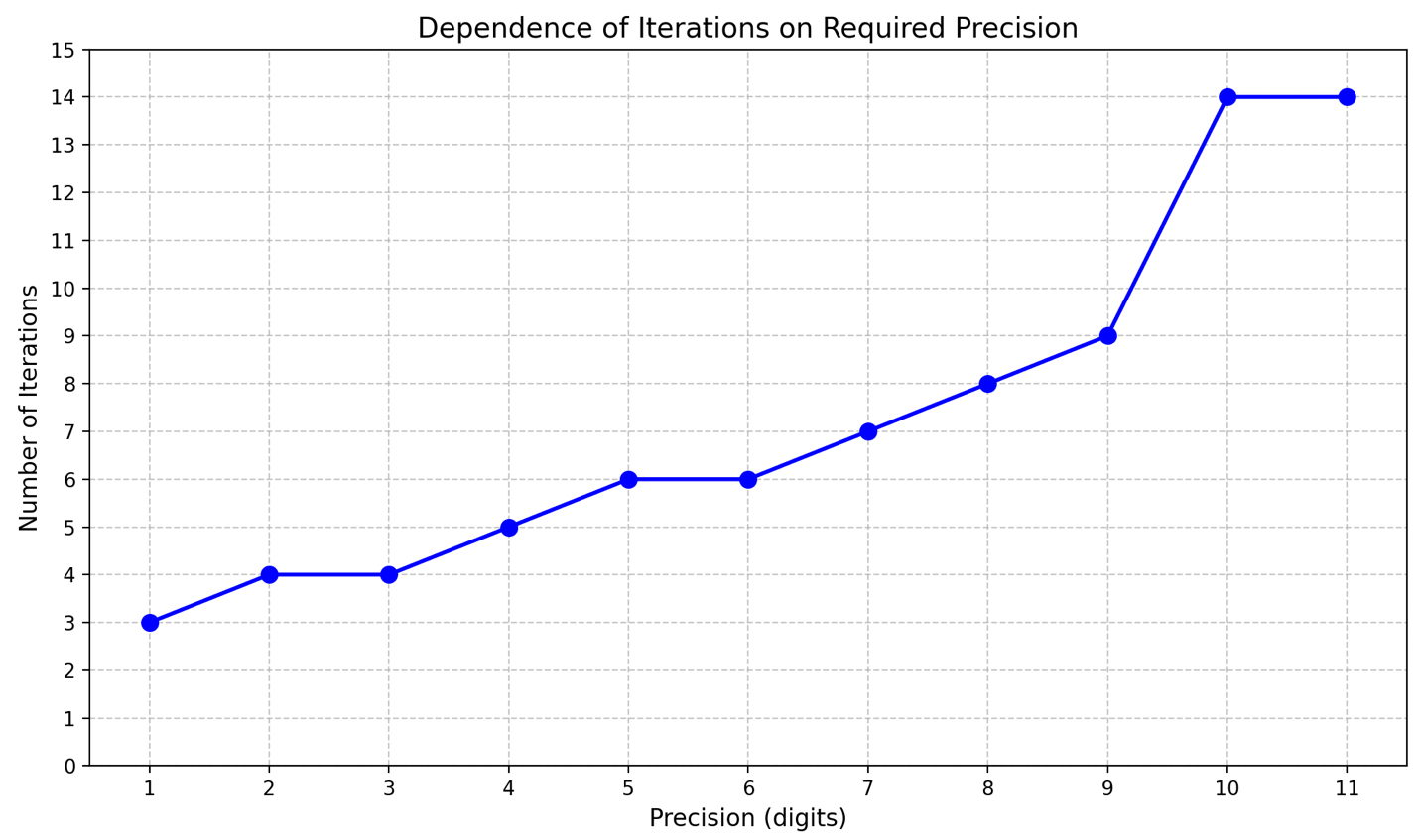


Рисунок 1 – График зависимости числа итераций метода простых итераций от

точности

Из графика видно, что при увеличении требуемой точности решения увеличивается число итераций.

**Вывод**

В ходе лабораторной работы была написана программа на языке C++, реализующая решение систем линейных уравнений методом Якоби. Для верификации результатов проводилось сравнение с эталонными решениями, полученными при помощи библиотеки Eigen. Экспериментальные данные подтвердили корректность работы алгоритма - расхождения с библиотечными решениями оказались в пределах вычислительной погрешности. Особенностью реализации метода Якоби стало использование специального преобразования исходной системы для обеспечения диагонального преобладания, что гарантировало сходимость итерационного процесса. Для контроля точности применялся критерий остановки, учитывающий как норму матрицы системы, так и достигнутую точность текущего приближения.

Было проведено исследование зависимости числа итераций от точности. В результате был сделан вывод, что при увеличении точности количество итераций растет.