Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

УНИВЕРСИТЕТ ИТМО **УЧЕБНЫЙ ЦЕНТР ОБЩЕЙ ФИЗИКИ ФТФ**

Группа: M32101

Студент: Косовец Роман / Дмитрий Надеждин

Преподаватель: Дорофеева Юлия Александровна

Рабочий протокол и отчет по лабораторной работе № 4
«Методы решения СЛАУ»

1. Цель работы:

Реализовать перечисленные ниже методы решения СЛАУ:

- Реализовать метод Гаусса с выбором ведущего элемента для решения СЛАУ
- Реализовать алгоритм LU-разложения с использованием разреженно-строчного формата хранения матрицы
- Реализовать итерационный метод Якоби
- Реализовать метод Гильберта

2. Задачи, решаемые при выполнении работы:

1. Провести исследование реализованных методов на системах с матрицами A^k.
2. Оценить зависимость числа обусловленности и точности полученного решения в зависимости от параметра k.
3. Сравнить между собой прямые и итерационные методы по эффективности методов в зависимости от размеров n матрицы

3. Условие задачи:

Лабораторная работа # 4

Методы решения СЛАУ

Предполагаемый язык выполнения лабораторных работ Python 3. Лабораторные работы выполняются студентами индивидуально или в группах по 2-3 человека (по желанию). По результатам выполнения лабораторной работы необходимо подготовить отчет. Отчет должен содержать описание реализованных вами алгоритмов, ссылку на реализацию, необходимые тесты и таблицы.

Постановка задачи

1. Реализовать метод Гаусса с выбором ведущего элемента для решения СЛАУ.

2. Реализовать алгоритм LU-разложения с использованием разреженно-строчного (разреженно-столбцового) формата хранения матрицы, а также метод решения СЛАУ с использованием LU-разложения.

3. Реализовать итерационный метод решения СЛАУ (метод Зейделя, Якоби или верхней релаксации на выбор).

4. Провести исследование реализованных методов на системах с матрицами $A^{(k)}$, число обусловленности которых регулируется за счет изменения диагонального преобладания. Внедиагональные элементы матрицы $A^{(k)}$ выбираются случайным образом из множества

$$a_{ij} \in \{0, -1, -2, -3, -4\},$$

а диагональные элементы определяются из условия

$$a_{ii} = \begin{cases} -\sum_{i \neq j} a_{ij} & if \quad i > 1 \\ -\sum_{i \neq j} a_{ij} + 10^{-k} & if \quad i = 1, \end{cases}$$

где сумма вычисляется *только по строке*.

Для исследования работы методов рекомендуется решать СЛАУ вида:

$$A^{(k)} x^k = F^k$$

где для определения правой части F^k рассматривается вектор $x^k = (1, 2, \ldots, n)^T$, что позволяет в дальнейшем сравнивать точное и приближенное решение.

5. Оценить зависимость числа обусловленности и точности полученного решения в зависимости от параметра k.

6. Провести аналогичные исследования на матрицах Гильберта, которые строятся согласно формуле

$$a_{ij} = \frac{1}{i + j - 1}, \qquad i, j = 1, .., n$$

где n - размерность матрицы.

7. Сравнить между собой прямые и итерационные методы по эффективности методов в зависимости от размеров n матрицы:

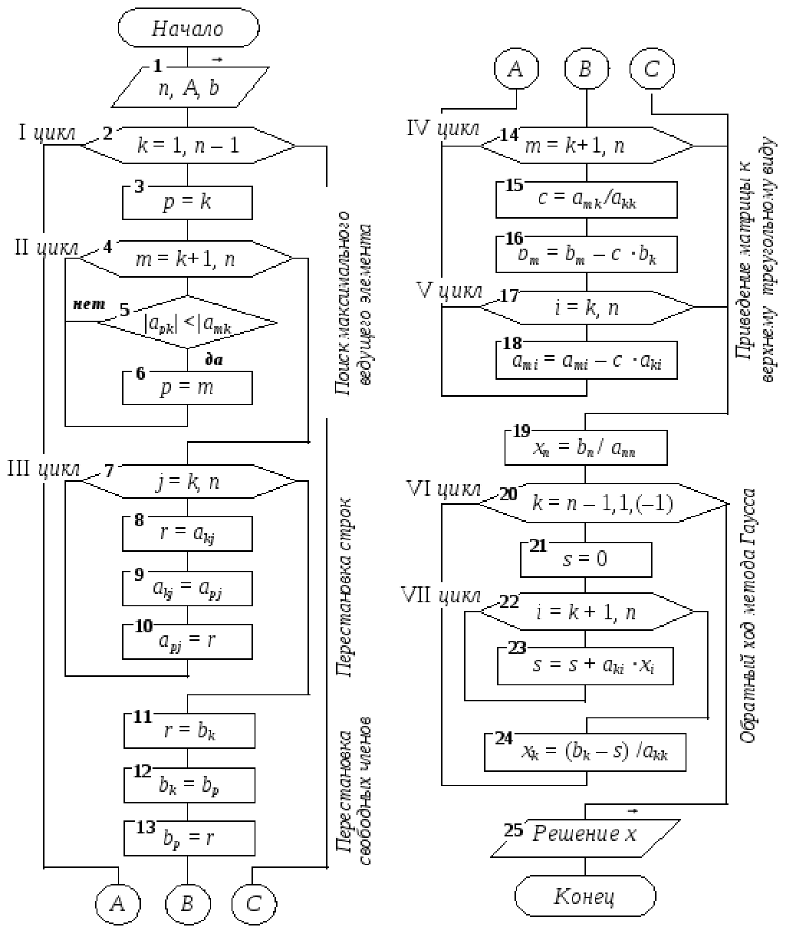
$$n \in \{10, 50, 10^2, 10^3, 10^4, 10^5\}$$

4. Метод Гаусса с выбором ведущего элемента:

Метод гаусса с выбором ведущего элемента — этот метод является улучшенной версией обычного метода Гаусса для решения систем линейных уравнений. Он предназначен для уменьшения накопления ошибок округления и улучшения устойчивости алгоритма.

Основное накопление погрешностей решения в методе Гаусса происходит на этапе приведения системы к треугольному виду. Механизм накопления основной части этой погрешности заключается в привнесении погрешностей вычисления коэффициентов ведущего уравнения в коэффициенты последующих уравнений при исключении каждого очередного неизвестного.

В методе Гаусса с выбором главного элемента на каждом шаге исключения *i*-го неизвестного в качестве ведущего используется уравнение (с *i*-го по *n*-ое), содержащее максимальный по модулю коэффициент – главный элемент. При этом в качестве него может использоваться один из коэффициентов *i*-го столбца, *i*-ой строки или всей непреобразованной части матрицы. Первый подход называется выбором главного элемента по столбцу, второй – по строке, а третий – по всей матрице.



Преимущества метода Гаусса с выбором ведущего элемента:

- Улучшенная устойчивость. Выбор ведущего элемента максимизирует его абсолютное значение, что помогает уменьшить накопление ошибок округления и улучшает устойчивость алгоритма.
- Повышенная точность. Этот метод может обеспечить более точные результаты, особенно при работе с числами с плавающей точкой.

Недостатки метода Гаусса с выбором ведущего элемента:

- Дополнительные вычисления. Для выбора ведущего элемента требуется выполнить дополнительные вычисления, такие как поиск максимального элемента в столбце. То есть небольшое увеличение вычислительной сложности.
- Дополнительная сложность реализации.

Алгоритм решения на Python:
https://github.com/RomanKosovets/PriMat/blob/main/Lab_4/GaussPrincipalComponent.py

5. LU-разложения:

LU-разложения с использованием разреженно-строчного формата хранения матриц – метод LU-разложения является численным методом для решения систем линейных уравнений (СЛАУ). Он основан на представлении матрицы коэффициентов системы в виде произведения двух матриц: верхнетреугольной и нижнетреугольной. Это позволяет разложить исходную матрицу на две более простые матрицы и решить систему последовательно с использованием этих разложений.

Разреженно-строчный формат хранения матрицы (CSR - Compressed Sparse Row) является одним из способов представления разреженных матриц, в котором только ненулевые элементы сохраняются в памяти. Он основан на трех массивах: массив значений ненулевых элементов, массив индексов столбцов и массив индексов начала строк.

Алгоритм 1. $L\bar{U}$-разложение по методу Гаусса с выбором главного элемента

Для $k = 1$ *до* n
  Выбираем главный элемент в $A^{(k-1)}$.
  Нормируем первую строку матрицы $A^{(k-1)}$.
  Для $i = k + 1$ *до* n
    Вычитаем первую строку матрицы $A^{(k-1)}$, *умноженную на* $a_{ik}^{(k-1)}$, *из* i*-й строки.*

Алгоритм для решения СЛАУ с использованием метода LU-разложения:

- Преобразование матрицы системы в разреженно-строчный формат хранения (CSR формат)
- Выполнение LU-разложения над матрицей в CSR формате.
- Решение двух систем линейных уравнений: $L_y = b$ и $U_x = y$, где L и U - матрицы разложения, y - промежуточный вектор, x - вектор решения
- Восстановление вектора решения x.

Преимущества LU-разложения:

- Эффективность использования памяти. Разреженно-строчный формат хранения матрицы позволяет существенно сократить использование памяти по сравнению с полным хранением матрицы.

- Улучшенная производительность. Благодаря разреженной структуре матрицы, метод LU-разложения с использованием разреженно-строчного формата может быть более эффективным с точки зрения вычислительной сложности и времени выполнения операций над матрицами.

Недостатки LU-разложения:

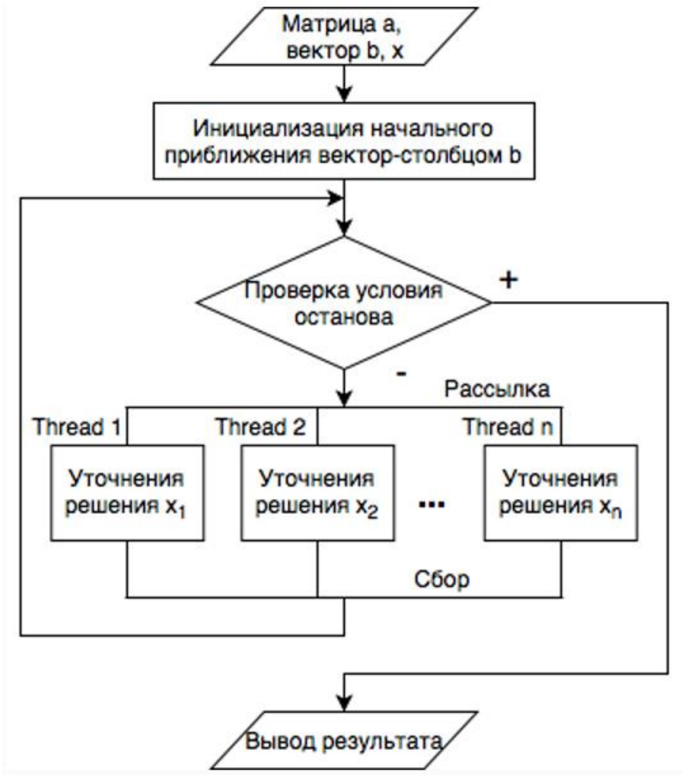
- Сложность реализации. Реализация этого метода сложнее, чем обычных методов работы с матрицами.
- Ограничения на типы матриц. В редких случаях алгоритм может получить на вход матрицы с большим количеством ненулевых элементов в каждой строке, поэтому выигрыша в производительности не будет.

Алгоритм решения на Python:
https://github.com/RomanKosovets/PriMat/blob/main/Lab_4/LU_Decomposition.py

6. Итерационный метод Якоби:

Итерационный метод Якоби – для решения итерационно СЛАУ, мы выбрали метод Якоби, этот метод является одним из итерационных методов решения СЛАУ. Он относится к классу простых итерационных методов и применяется для решения СЛАУ с диагонально-преобладающими матрицами.

Алгоритм решения СЛАУ методом Якоби:
Вводятся матрица A и вектора b, x. Далее в качестве начального приближения берется столбец правых членов — вектор b. По потокам распределяем элементы матрицы A и вектора b, и выполняем первую итерацию. Проверяем выполнение условие остановки, если не выполняется, переходим к следующей итерации, если выполняется, заканчиваем вычисления.



Преимущества метода Якоби:

- Простота реализации. Метод Якоби легко реализовать и понять. Поскольку он основан на простой реализации.
- Универсальность. Метод Якоби может быть применен к широкому классу СЛАУ и не требует особых условий матрицу А.
- Независимость от начальной точки. В отличие от алгоритма с постоянным шагом, метод наискорейшего спуска не требует тщательного выбора начального шага, так как шаг определяется адаптивно на каждой итерации.

Недостатки метода Якоби:

- Медленная сходимость. Метод Якоби может сходится медленно для некоторых типов матриц, особенно для плохо обусловленных систем.
- Ограничение применимости. Метод Якоби может быть неэффективен для больших и разреженных матриц, так как требует хранение и обновление всей матрицы на каждой итерации.

Алгоритм решения на Python:
https://github.com/RomanKosovets/PriMat/blob/main/Lab_4/JacobiMethod.py

7. Исследование реализованных методов:

Теперь проанализируем написанные алгоритмы. Проведем исследование реализованных методов (метода Якоби и метода LU-разложения) на системах с матрицами A^k, число обусловленности которых регулируется за счет изменения диагонального преобладания. Внедиагональные элементы матрицы A^k выбираются случайным образом из множества {0, -1, -2, -3, -4}, а диагональные элементы определяются из условия:

$$a_{ii} = \begin{cases} -\sum_{i \neq j} a_{ij} & if \quad i > 1 \\ -\sum_{i \neq j} a_{ij} + 10^{-k} & if \quad i = 1, \end{cases}$$

где сумма вычисляется только по строке.

После исследования были получены следующие данные:

Метод Якоби:

- Средняя точность: 0.94
- Среднее время выполнения: 0.00145

LU-разложение:

- Средняя точность: 1.0
- Среднее время выполнения: 0.00023

Из проведенного исследования методов Якоби и Гаусса на системах с матрицами A^k, где число обусловленности регулируется изменением диагонального преобладания, можно сделать следующие выводы на счет оценки зависимости числа обусловленности и точности решения от параметра k:

- *Число обусловленности:* Параметр k может использоваться для регулирования числа обусловленности матриц A^k. Чем больше значение k, тем больше внедиагональные элементы матрицы A^(k) и, соответственно, возрастает число обусловленности. Можно ожидать, что <u>при увеличении k числа обусловленности также будет возрастать</u>. Это может привести к ухудшению устойчивости и точности решения, особенно для итерационных методов, таких как метод Якоби.
- *Точность решения:* при увеличении числа обусловленности матриц A^k ожидается снижение точности решения. Высокое число обусловленности может приводить к усилению ошибок округления и

увеличению погрешностей вычислений. Это может отразиться на точности полученных решений СЛАУ.

Алгоритм решения на Python:
https://github.com/RomanKosovets/PriMat/blob/main/Lab_4/Analizer.py

8. Матрицы Гильберта:

Проведем аналогичные исследования на матрицах Гильберта.

Матрица Гильберта - Матрица Гильберта является классическим примером особенной матрицы, которая получается путем заполнения элементов по определенному правилу. Матрица Гильберта H размером $n \times n$ определяется следующим образом:

$$a_{ij} = \frac{1}{i+j-1}, \qquad i, j = 1, .., n$$

Число обусловленности матрицы Гильберта растет очень быстро с увеличением размера матрицы, что делает ее плохо обусловленной для больших значений n. Для тестирования методов Гаусса и Якоби на матрице Гильберта мы будем использовать матрицы разных размеров, начиная от небольших (например, n=5) до более крупных (например, n=50).

Для каждого значения n, мы будем изменять диагональное преобладание матрицы Гильберта и сравнивать точность и скорость сходимости методов Гаусса и Якоби.

Полученные результаты после исследования:

Оба метода решили СЛАУ правильно:

- Решение методом Гаусса: [30, -420, 1680, -2520, 1260]
- Решение методом Якоби: [30, -420, 1680, -2520, 1260]

Несколько полученных результатов с разными матрицами Гильберта и полученные числа обусловленности:

- Число обусловленности: 5039074.44

- Решение методом Гаусса: [15014.99, -87359.98, 185639.95, -171359.97, 58139.98]
- Решение методом Якоби: [15014.99, -87359.97, 185639.95, -171359.95, 58139.98]

- Число обусловленности: 18617742849.2
- Решение методом Гаусса: [4.40920171e+08, -1.85266814e+09, 2.91781605e+09, -2.04141381e+09, 5.35346240e+08]
- Решение методом Якоби: [4.40920171e+08, -1.85266814e+09, 2.91781605e+09, -2.04141381e+09, 5.35346240e+08]

- Число обусловленности: 1.56787843706e+17
- Решение методом Гаусса: [-6.26159659e+10, 1.78621329e+11, 1.59010617e+11, 3.26167777e+10, 1.03884797e+10]
- Решение методом Якоби: [-6.26159659e+10, 1.78621329e+11, 1.59010617e+11, 3.26167777e+10, 1.03884797e+10]

Алгоритм решения на Python:
https://github.com/RomanKosovets/PriMat/blob/main/Lab_4/GilbertMatrixAnalizer.py

9. Вывод:

Для начала сравним прямые и итерационные методы по эффективности и сделаем вывод о них.

В общем случае, можно упомянуть некоторые общие соображения:

- Прямые методы, такие как метод Гаусса, обычно требуют вычисления обратной матрицы, что может быть затратным по времени и памяти для больших матриц. Сложность прямых методов обычно составляет O(n^3), где n - размерность матрицы.
- Итерационные методы, такие как метод Якоби, основаны на итерационном уточнении решения. Они могут быть более эффективными для больших разреженных матриц, так как требуют меньше операций над матрицей. Однако, для сильно несимметричных или плохо обусловленных матриц, итерационные методы могут сходиться медленно или даже расходиться.

Выбор конкретного метода зависит от требуемой точности решения, размерности матрицы, плотности матрицы, структуры матрицы и других факторов. В некоторых случаях, комбинирование прямых и итерационных методов может быть более эффективным

В итоге, в лабораторной работе были рассмотрены и реализовали различные методы решения СЛАУ (прямые и итерационные), такие как метод Гаусса, метод Якоби. Каждый метод имеет свои преимущества и недостатки.

Прямые методы, обеспечивают точное решение системы линейных уравнений. Они могут быть эффективны для небольших и средних размеров матриц, особенно если матрица имеет небольшую разреженность.

Итерационные методы, могут быть могут быть особенно полезны для систем с большими размерами или с особыми структурами матрицы. Однако, итерационные методы могут быть менее точными и требовать большего числа итераций.

Число обусловленности матрицы является важной характеристикой, определяющей устойчивость системы линейных уравнений. Чем выше число обусловленности, тем более чувствительна система к погрешностям в исходных данных. Высокое число обусловленности может привести к ухудшению точности решения и увеличению ошибок, поэтому, при решении систем линейных уравнений, важно учитывать число обусловленности и применять соответствующие методы.