СПбПУ Петра Великого
Высшая школа прикладной математики и вычислительной физики, ФизМех

Направление подготовки
«01.03.02 Прикладная математика и информатика»

Отчёт по лабораторной работе №2

Тема

Решение систем алгебраических уравнений методом LU-разложения

Дисциплина

Численные методы

Выполнил студент группы 5030102/00002 Димитрюк Н. С.
Преподаватель

Санкт-Петербург

1. Формулировка и формализация задачи

1.1 Формулировка задачи

Решить несколько СЛАУ $Ax = B$, где A — невырожденная матрица размером 12×12, используя метод LU-разложения. В ходе работы необходимо исследовать зависимости нормы фактической ошибки и нормы невязки от числа обусловленности матрицы A.

1.2 Формализация задачи

Дано: $Ax = B$, где $A \in R^{n \times n}$ — матрица коэффициентов
$x \in R^n$ — вектор неизвестных
$B \in R^n$ — вектор свободных членов

Найти: x — вектор неизвестных.

2. Алгоритм метода и условия его применимости

2.1 Условия применимости

Матрица коэффициентов является невырожденной, то есть $det(A) \neq 0$.

2.2 Алгоритм

Дано: $Ax = B$, где $A = (a_{ij})$ — матрица коэффициентов
$x = (x_i)$ — вектор неизвестных
$B = (b_i)$ — вектор свободных членов

Алгоритм следующий:

1. Находим матрицы L и U, нижнюю и верхнюю треугольные матрицы соответственно. L и U такие, что $LU = A$. Получаем уравнение $LUx = B$.
2. Делаем замену $Ux = y$. И решаем СЛАУ $Ly = B$ методом прямой подстановки.
3. Теперь решаем СЛАУ $Ux = y$ методом обратной подстановки и получаем искомый вектор неизвестных.

Элементы матрицы U находим по формуле: $u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} \cdot u_{kj}$, где $i \leq j$

Элементы матрицы L находим по формуле: $l_{ij} = \frac{1}{u_{jj}}(a_{ij} - \sum_{k=1}^{j-1} l_{ik} \cdot u_{kj})$, где $i > j$

3. Предварительный анализ задачи

Определитель матрицы коэффициентов не равен нулю, то есть $det(A) \neq 0$, значит СЛАУ имеет единственное решение.

4. Проверка условий применимости

Так как матрица коэфициентов создаётся с помощью функции gallery() с параметром "randsvd"в программе MATLAB, она имеет отличный от нуля определитель, потому что в функции задаётся число обусловленности.

5. Тестовый пример

Дано: матрица коэффициентов $A = \begin{pmatrix} 5 & 8 & 4 \\ 10 & 22 & 11 \\ 15 & 48 & 26 \end{pmatrix}$, матрица свободных членов $B = \begin{pmatrix} 33 \\ 87 \\ 189 \end{pmatrix}$.

Корень для проверки: $x^* = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$.

Найдём решение x, применив метод LU-разложения:

1. Разложим матрицу A: $L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{pmatrix}$, $U = \begin{pmatrix} 5 & 8 & 4 \\ 0 & 6 & 3 \\ 0 & 0 & 2 \end{pmatrix}$.

   Проверим перемножив: $LU = \begin{pmatrix} 5 & 8 & 4 \\ 10 & 22 & 11 \\ 15 & 48 & 26 \end{pmatrix} = A$, всё верно.

   Теперь имеем уравнение $LUx = B$

2. Сделаем замену $Ux = y$ и решим СЛАУ $Ly = B$ методом прямой подстановки:

   $y_1 = \frac{b_1}{l_{11}} = \frac{33}{1} = 33$

   $y_2 = \frac{1}{l_{22}}(b_2 - \sum_{j=1}^{1} l_{2j} \cdot y_j) = \frac{1}{1}(87 - 2 \cdot 33) = 21$

   $y_3 = \frac{1}{l_{33}}(b_3 - \sum_{j=1}^{2} l_{3j} \cdot y_j) = \frac{1}{1}(189 - 3 \cdot 33 - 4 \cdot 21) = 6$

   Получили $y = \begin{pmatrix} 33 \\ 21 \\ 6 \end{pmatrix}$

3. Теперь решаем СЛАУ $Ux = y$ методом обратной подстановки:

   $x_3 = \frac{y_3}{u_{33}} = \frac{6}{2} = 3$

   $x_2 = \frac{1}{u_{22}}(y_2 - \sum_{j=3}^{3} u_{2j} \cdot x_j) = \frac{1}{6}(21 - 3 \cdot 3) = 2$

   $x_1 = \frac{1}{u_{11}}(y_1 - \sum_{j=2}^{3} u_{1j} \cdot x_j) = \frac{1}{5}(33 - 8 \cdot 2 - 4 \cdot 3) = 1$

   Получили $x = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = x^*$, нашли верное решение.

6. Контрольные тесты

1. Размерность матриц 12×12, столбцов — 12×1.
2. Точный корень: $x^* = 1 : 1 : 12$.
3. Число обусловленности матриц: $cond(A) = 100 : 100 : 10000$.
4. Возмущение матрицы свободных членов: $\delta B = 10^{-3} \cdot E_{12\times1}$.

$E_{12\times1}$ — единичная матрица размера 12×1.

7. Модульная структура программы

1. vector< vector<double> > LoadMatrixFromFile(string fileName, int skip) — считывает матрицу из файла.
   Параметры:
   $fileName$ — имя файла
   $skip$ — номер матрицы в файле
   Возвращаемое значение: считанную матрицу.
2. vector<double> LoadVectorFromFile(string fileName, int skip) — считывает матрицу из файла.
   Параметры:
   $fileName$ — имя файла
   $skip$ — номер вектора в файле
   Возвращаемое значение: считанный вектор.
3. vector< vector<double> > MMMultiplication(vector< vector<double> > x, vector< vector<double> > y) — перемножает две матрицы.
   Параметры:
   x — первая матрица
   y — вторая матрица
   Возвращаемое значение: результат перемножения матриц x и y.
4. vector<double> MVMultiplication(vector< vector<double> > x, vector<double> y) — умножает матрицу на столбец.
   Параметры:
   x — матрица
   y — столбец
   Возвращаемое значение: результат перемножения матрицы x на столбец y.
5. void PrintMatrix(vector< vector<double> > m — выводит матрицу в консоль.
   Параметры:
   m — матрица
   Возвращаемое значение: ничего не возвращает.
6. void PrintVector(vector<double> v — выводит вектор в консоль.
   Параметры:
   v — вектор
   Возвращаемое значение: ничего не возвращает.

7. vector<double> LYB(vector< vector<double> > L, vector<double> B) — находит решение СЛАУ $Ly = B$.
   Параметры:
   L — матрица коэффициентов
   B — столбец свободных членов
   Возвращаемое значение: решение СЛАУ — столбец y.

8. vector<double> UXY(vector< vector<double> > U, vector<double> Y) — находит решение СЛАУ $Ux = y$.
   Параметры:
   U — матрица коэффициентов
   y — столбец свободных членов
   Возвращаемое значение: решение СЛАУ — столбец x.

9. vector< vector<double> > LU(vector< vector<double> > A) — находит матрцы L и U LU-разложения матрицы A.
   Параметры:
   A — матрица
   Возвращаемое значение: нижняя треугольная матрица L и верхняя треугольная матрица U.

10. vector<double> AFindXLU(vector< vector<double> > A, vector<double> B) — находит решение СЛАУ $Ax = B$ с помощью LU-разложения матрицы A.
    Параметры:
    A — матрица коэффициентов
    B — столбец свободных членов
    Возвращаемое значение: решение СЛАУ — столбец x.

11. double ActualErrorRate(vector<double> X, vector<double> exactX) — находит норму фактической ошибки для решения X.
    Параметры:
    X — найденное решение
    exact$X$ — точное решение
    Возвращаемое значение: норма фактической ошибки.

12. double DiscrepancyRate(vector< vector<double> > A, vector<double> B, vector<double> X) — находит норму невязки для решения X.
    Параметры:
    A — матрица коэффициентов
    B — столбец свободных членов
    X — найденное решение
    Возвращаемое значение: норма невязки.

15. bool Check(vector< vector<double> > A, vector<double> B, vector<double> X, int condA) — проверяет верность неравенства $\frac{||\delta x||}{||x||} \leq cond(A)\frac{||\delta B||}{||B||}$.
Параметры:
A — матрица коэффициентов
B — столбец свободных членов
X — найденное решение
$condA$ — число обусловленности матрицы A
Возвращаемое значение: true или false, если неравенство выполняется или нет соответственно.

8. Численный анализ

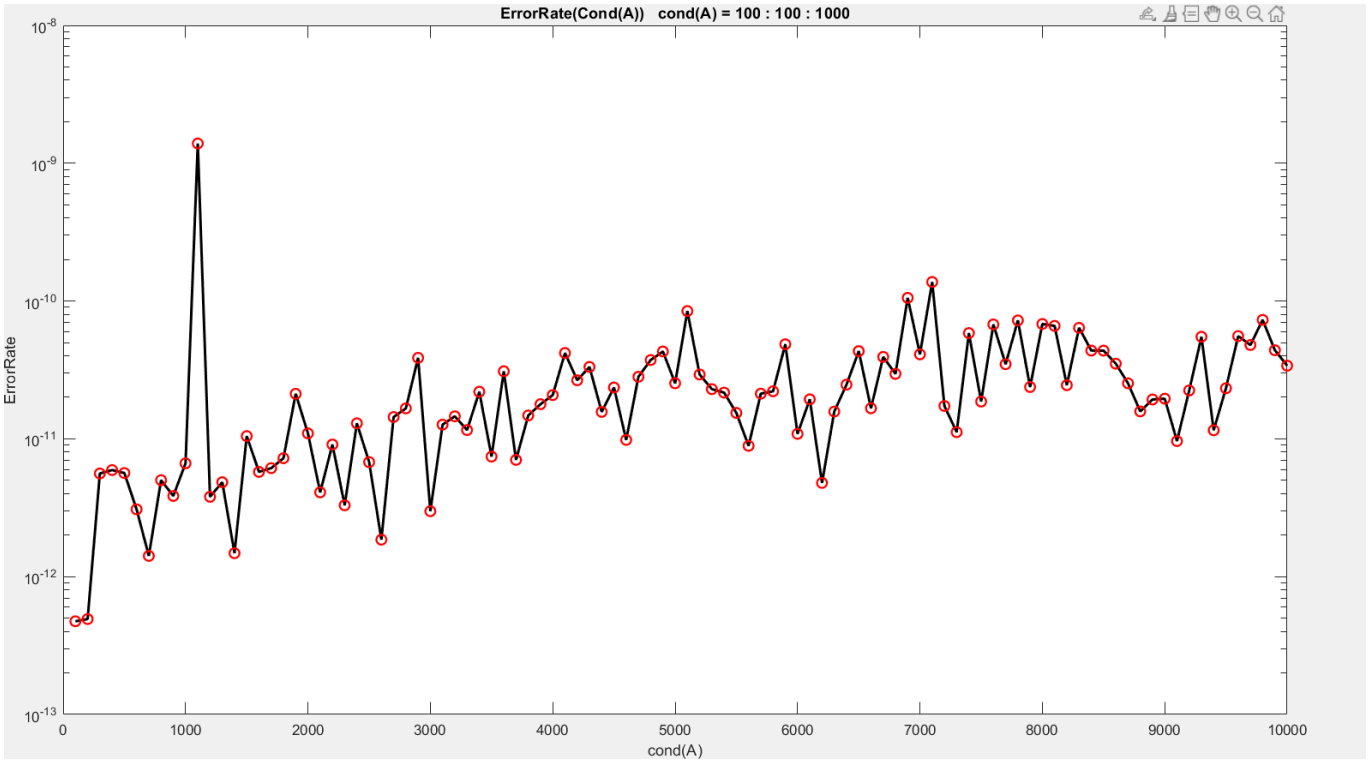

График 1. Норма фактической ошибки

На графике (График 1) показана зависимость нормы фактической ошибки от числа обусловленности матрицы. График похож на пилообразную функцию, но видно, что в общем график растёт, то есть с ростом числа обусловленности растёт норма фактической ошибки. При росте числа обусловленности от 100 до 10000, то есть на два порядка, норма фактической ошибки выросла так же на два порядка.

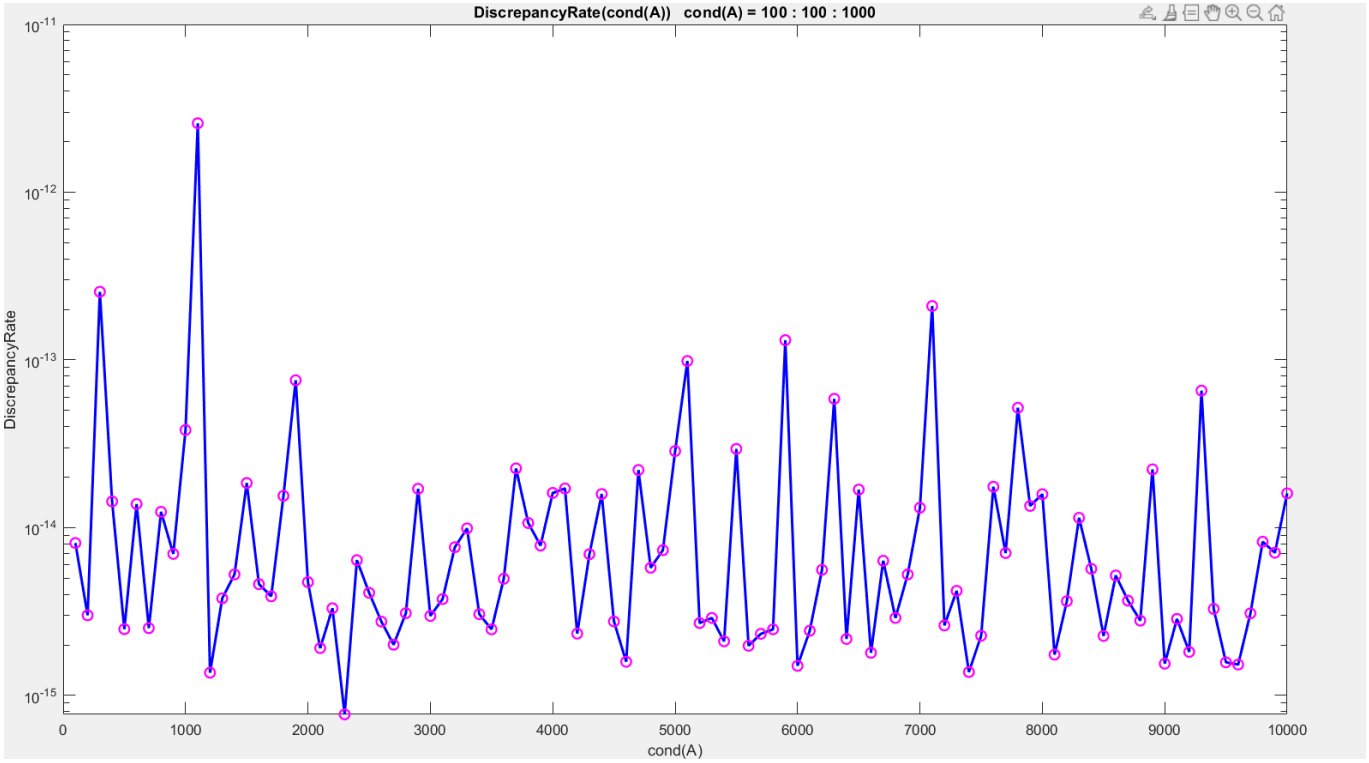

График 2. Норма невязки

На графике (График 2) показана зависимость нормы невязки от числа обусловленности матрицы. Этот график тоже похож на пилообразную функцию, но здесь видно, что в общем график идёт по константе, то есть норма невязки сохраняет порядок своей величины.

9. Вывод

В ходе работы была решена задача о нахождении корня системы линейных алгебраических уравнений с помощью метода LU-разложения. Были выявлены и исследованы зависимости нормы фактической ошибки и нормы невязки от числа обусловленности. По итогам этих исследований, было обнаружено, что норма фактической ошибки растёт почти линейно (по пилообразной функции) с увеличением числа обусловленности, а норма невязки почти не изменяется.

10. Исправления

2.2 Алгоритм

Метод прямой подстановки: $Ly = B$, L — нижняя унитреугольная матрица. Поиск решения: $y_1 = \frac{b_1}{l_{11}}; y_i = \frac{1}{l_{ii}}(b_i - \sum_{j=1}^{i-1} l_{ij}x_j)$, где $i = 2, ..., n$.
Метод обратной подстановки: $Ux = y$, U — верхняя треугольная матрица. Поиск решения: $x_n = \frac{b_n}{u_{nn}}; x_i = \frac{1}{u_{ii}}(b_i - \sum_{j=i+1}^{n} u_{ij}x_j)$, где $i = n-1, ..., 1$..
Нахождение LU-разложения матрицы A: вычисляем по строкам, то есть сначала в первой строке вычисляем j-е элементы ($j = 1, ..., n$), затем во второй и так во всех i-х строках ($i = 1, ..., n$) по приведённым формулам.

4. Проверка условий применимости

Определители матрицы A для числа обусловленности большего 10^3:

$cond(A)$	10^3	$2.5 \cdot 10^3$	$5 \cdot 10^3$
$det(A)$	-10^{-18}	$-4 \cdot 10^{-21}$	$-6 \cdot 10^{-23}$

5. Тестовый пример

Нахождение матриц L и U по алгоритму:

Начинаем с первой строчки ($i = 1$):

Первый элемент ($j = 1$): $i \le j$, значит считаем $u_{11} = a_{11} - \sum_{k=1}^{1-1} l_{1k} \cdot u_{k1} = 5 + 0 = 5$

Второй элемент ($j = 2$): $i \le j$, значит считаем по той же формуле $u_{12} = 8$

Третий элемент ($j = 3$): $i \le j$, снова считаем по той же формуле $u_{13} = 4$

Далее вторая строка ($i = 2$):

Первый элемент ($j = 1$): $i > j$, значит считаем $l_{21} = \frac{1}{u_{11}}(a_{21} - \sum_{k=1}^{1-1} l_{2k} \cdot u_{k1}) = \frac{10+0}{5} = 2$

Для второго и третьего элементов выполняет ($i \le j$), значит считаем $u_{22} = 6$ и $u_{23} = 3$

Третья строка ($i = 3$):

Первый и второй элементы удовлетворяют условию ($i > j$), значит считаем $l_{31} = 3$ и $l_{32} = 4$

Третий элемент ($j = 3$): ($i \le j$), $u_{33} = 2$

Получили матрицы: $L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{pmatrix}$ и $U = \begin{pmatrix} 5 & 8 & 4 \\ 0 & 6 & 3 \\ 0 & 0 & 2 \end{pmatrix}$.

6. Контрольные тесты

2. Точный корень: $x^* = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12)^T$

7. Модульная структура программы

double ActualErrorRate(vector<double> X, vector<double> exactX) — находит норму фактической ошибки для решения X в виде: $||x^* - x|| = \sqrt{\sum x'^2}$.
Параметры:
X — найденное решение
exact$X$ — точное решение
Возвращаемое значение: норма фактической ошибки.

8. Численные анализ

Числа нормы фактической ошибки, нормы невязки и для неравентсва для случая $cond(A) = 100$:
$||x^* - x|| = 4.74264 \cdot 10^{-13}$
$||Ax - B|| = 8.09909 \cdot 10^{-15}$
$||\delta x|| = 1.05018$
$||x^*|| = 25.49509$
$||\delta B|| = 0.00346$
$||B|| = 10.02324$

9. Вывод

В ходе работы была решена задача о нахождении корня системы линейных алгебраических уравнений с помощью метода LU-разложения. Были выявлены и исследованы зависимости нормы фактической ошибки и нормы невязки от числа обусловленности. По итогам этих исследований, было обнаружено, что норма фактической ошибки растёт почти линейно (по пилообразной функции) с увеличением числа обусловленности, а норма невязки почти не изменяется. Это говорит о том, что при решении задач с матрицей коэффициентов, у которой будет большое число обусловленности, будет сложно по норме невязки судить о норме фактической ошибки и использовать норму невязки.