Министерство образования и науки Российской Федерации

Санкт-Петербургский политехнический университет Петра Великого

Институт прикладной математики и информатики

Лабораторная работа по дисциплине

Численные методы

«Решение систем линейных алгебраических уравнений прямыми методами»

Выполнил
студент гр.5030102/20001 Дрекалов Н.С.

Преподаватель Добрецова С.Б.

Санкт-Петербург

2023

Оглавление

Формулировка задачи

Найти решение СЛАУ с помощью LDL^T разложения.

Формализация

- Пусть $F(x) : R^n \to R^n = Ax - b$.
- Требуется: найти $x*$ такое, что $F(x*) = 0$. Корень $x*$ ищется для заданного числа обусловленности

Построение LDL^T разложения:

Условия применимости:

- A – симметричная матрица.

Алгоритм метода:

- Необходимо представить матрицу $A = LDL^T$, где L – нижняя треугольная матрица с единицами на главной диагонали, а D – диагональная матрица. Просто решаем уравнения для каждого столбца, идя слева направо, а по каждому столбцу – сверху вниз.

$$A = LDL^T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & l_{32} & 1 & 0 \\ l_{41} & l_{42} & l_{43} & 1 \end{pmatrix} \begin{pmatrix} d_1 & 0 & 0 & 0 \\ 0 & d_2 & 0 & 0 \\ 0 & 0 & d_3 & 0 \\ 0 & 0 & 0 & d_4 \end{pmatrix} \begin{pmatrix} 1 & l_{21} & l_{31} & l_{41} \\ 0 & 1 & l_{32} & l_{42} \\ 0 & 0 & 1 & l_{43} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Идем по столбцам слева направо и сверху вниз.

1. Для $0 < i <= dimA$:
   1.1. Для $0 < j <= dimA$:
      1.1.1. Считаем сумму:
         1.1.1.1. $sum = a_{ij}$.,
         1.1.1.2. для $1 <= k < j$:
            1.1.1.2.1. $sum -= l_{jk} * d_k * l_{ik}$.
      1.1.2. Если $i = j$:
         1.1.2.1. $d_i = sum$,
         1.1.2.2. $l_{ij} = 1$.

1.1.3. Иначе:

1.1.3.1. l_{ij} = sum / d_j.

2. Полученные L, D такие, что A = LDL^T.

Решение СЛАУ с помощью LDL^T разложения:

Условия применимости:

- A – симметричная матрица.

Алгоритм метода:

1. Представим матрицу A = LDL^T.

   Решим уравнение L * c = b, где c = DL^Tx

2. Для 0 < i <= dimA:

   2.1. c_i = b_i

   2.2. Для 0 < j < i:

   2.2.1. c_i -= l_{ij} * c_j.

   Построим c' = D^{-1} * c:

3. Для 0 < i <= dimA:

   3.1. c_i /= d_i.

   Идем по строкам снизу вверх справа налево.

4. Для dimA >= i > 0:

   4.1. x_i = c_i,

   4.2. Для dimA >= j > i:

   4.2.1. x_i -= l_{ji} * x_j.

5. Полученный X – решение СЛАУ.

Предварительный анализ задачи

Построение корня:

- Корень X находился по формуле: $x_i = \ln(i + 1) * i / 2 * \mathrm{sqrt}(i) * \sin(i)$.

Построение ортогональной матрицы:

- Для матрицы использовался вектор w = [3 4 5 6 3 2 9 11 -1 -3]
- Ортогональная матрица Q строилась с помощью преобразование Хаусхолдера: $Q = E - 2ww^T / ||w||^2$

Построение СЛАУ:

- Берем единичную матрицу и делаем преобразование $e_{11} = 10^i$ для $1 <= i < 15$. Это будет примерно равно числу обусловленности матрицы.
- Строим симметричную матрицу по формуле $A = QE'Q^T$.
- Находим свободный член по формуле $b = Ax$.

Проверка условий для методов:

LDL^T разложение:

- A – симметричная по построению

Нахождения корня с помощью LDL^T разложения.

- A – симметричная по построению

Тестовый пример к методам

Решим тестовый пример для уравнения Ax = b, где

$$A = \begin{pmatrix} -1 & -4 & 5 & 4 \\ -4 & 3 & -9 & 5 \\ 5 & -9 & 7 & -2 \\ 4 & 5 & -2 & 0 \end{pmatrix}, b = \begin{pmatrix} 22 \\ -5 \\ 0 \\ 8 \end{pmatrix}$$

1. Построим LDL^T разложение:

$$A = LDL^T = \begin{pmatrix} d_1 & - & - & - \\ d_1 l_{21} & d_1 l_{21}^2 + d_2 & - & - \\ d_1 l_{31} & d_1 l_{21} l_{31} + d_2 l_{32} & d_1 l_{31}^2 + d_2 l_{32}^2 + d_3 & - \\ d_1 l_{41} & d_1 l_{21} l_{41} + d_2 l_{42} & d_1 l_{31} l_{41} + d_2 l_{32} l_{42} + d_3 l_{43} & d_1 l_{41}^2 + d_2 l_{42}^2 + d_3 l_{43}^2 + d_4 \end{pmatrix}$$

Идем по столбцам:

Первый:

$d_1 = a_{11} = -1$

$l_{21} = a_{21} / d_1 = 4$

$l_{31} = a_{31} / d_1 = -5$

$l_{41} = a_{41} / d_1 = -4$

Второй:

$d_2 = a_{22} – d_1 l^2_{21} = 3 + 1 * 16 = 19$

$l_{32} = (a_{32} – d_1 l_{21} l_{31})/d_2 = (-9 – 20) / 19 = -29/19$

$l_{42} = (a_{42} – d_1 l_{21} l_{41})/d_2 = (5 - 16) / 19 = -11/19$

Третий:

$d_3 = a_{33} – d_1 l^2_{31} – d_2 l^2_{32} = 7 + 25 – 29 * 29 / 19 = -233/19$

$l_{43} = (a_{43} – d_1 l_{31} l_{41} – d_2 l_{32} l_{42}) / d_3 = (-2 * 19 + 20 * 19 - 29 * 11) / -233 = 23/-233$

Четвёртый

$d_4 = a_{44} – d_1 l^2_{41} – d_2 l^2_{42} – d_3 l^2_{43} = 0 + 16 – 121/19 + 23 * 23 / (233 * 19) =$
$= 43168/4427 = 2272/233$

2. Найдем c из уравнения Lc = b, где $c = DL^T x$:

$c_1 = b_1 = 22$

$c_2 = b_2 – l_{21}c_1 = -5 – 4 * 22 = -93$

$c_3 = b_3 – l_{31}c_1 – l_{32}c_2 = 0 + 5 * 22 – 29/19 * 93 = -607/19$

$c_4 = b_4 – l_{41}c_1 – l_{42}c_2 – l_{43}c_3 = 8 + 4 * 22 – 11/19 * 93 – 23/233 * 607/19 =$

$= 172672 / 4427 = 9088 / 233$

3. Умножим обе части слева на D^{-1}:

$$L^T x = D^{-1} c = \begin{pmatrix} c_1/d_1 \\ c_2/d_2 \\ c_3/d_3 \\ c_4/d_4 \end{pmatrix}$$

$c_1' = c_1 / d_1 = -22$

$c_2' = c_2 / d_2 = -93/19$

$c_3' = c_3 / d_3 = 607/233$

$c_4' = c_4 / d_4 = 9088/2272 = 4$

4. Решаем уравнение

$$L^T \mathbf{x} = \mathbf{c}'$$

или

$$\begin{pmatrix} 1 & l_{21} & l_{31} & l_{41} \\ 0 & 1 & l_{32} & l_{42} \\ 0 & 0 & 1 & l_{43} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} c_1' \\ c_2' \\ c_3' \\ c_4' \end{pmatrix}$$

$x_4 = c_4' = 4$

$x_3 = c_3' – l_{43}x_4 = 607/233 + 4 * 23/233 = 699/233 = 3$

$x_2 = c_2' – l_{32}x_3 – l_{42}x_4 = -93/19 + 29 * 3 / 19 + 11 * 4 / 19 = 38/19 = 2$

$x_1 = c_1' – l_{21}x_2 – l_{31}x_3 – l_{41}x_4 = -22 – 8 + 15 + 16 = 1$

Получаем $x = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$

Проверим ответ:

$$A * x = \begin{pmatrix} -1 & -4 & 5 & 4 \\ -4 & 3 & -9 & 5 \\ 5 & -9 & 7 & -2 \\ 4 & 5 & -2 & 0 \end{pmatrix} * \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 22 \\ -5 \\ 0 \\ 8 \end{pmatrix} = b$$

Контрольные тесты

Построим графики зависимостей

- Нормы фактической ошибки от числа обусловленности.
- Нормы невязки от числа обусловленности.

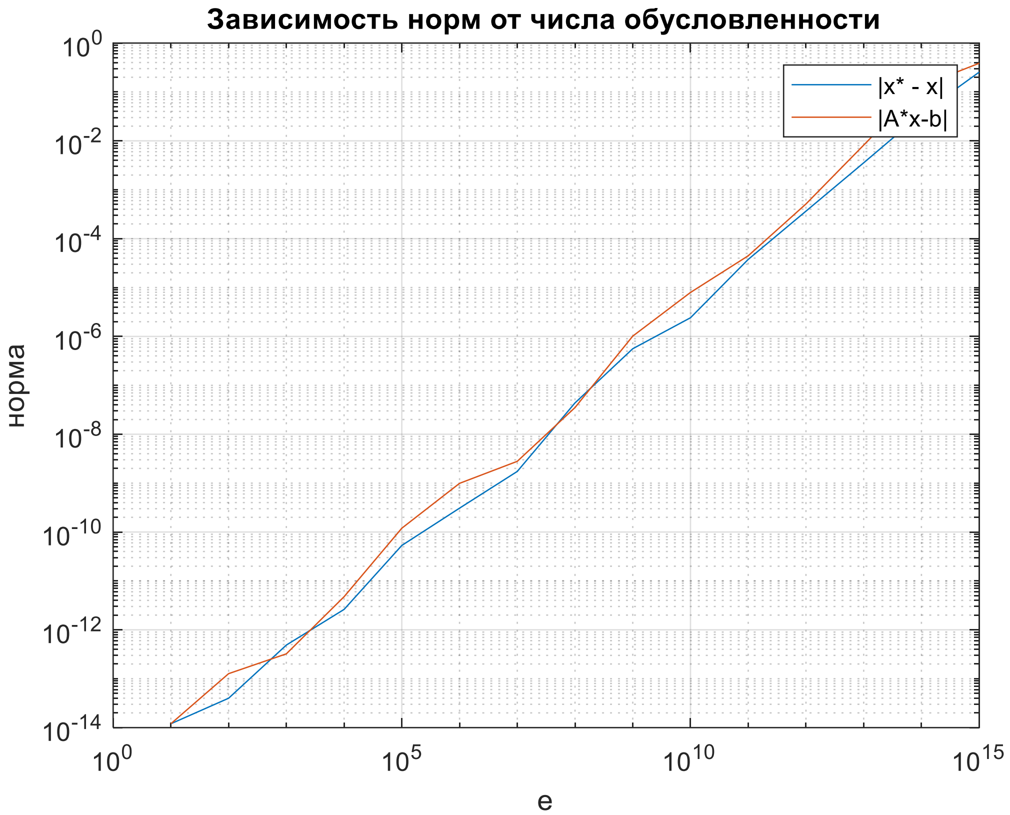
Модульная структура программы

```
// Класс одного линейного уравнения
typedef std::vector<std::vector<long double>> Matrix;
typedef std::vector<long double> LinMatrix;
class LinEquation {
	// A * x = b
	Matrix A;
	LinMatrix b;
	long double conditionalityNumber;
	LinMatrix x;
	// A = LDLt
	Matrix L;
	LinMatrix D;

	/**
	 * \brief Функция находит корень в заданном промежутке методом ПД с точностью
	эпсилон
	 */
	void makeLDLt();
public:
	/**
	 * \brief Конструктор по умолчанию
	 */
	LinEquation() = default;
	/**
	 * \brief Конструктор класса
	 * \param A - матрица системы
	 * \param b - свободный член
	 * \param conditionalityNumber - число обуслвленности
	 */
	LinEquation(Matrix& A, LinMatrix& b, long double conditionalityNumber);

	/**
	* \brief Функция, возвращающая матрицу A
	* \return матрица A
	*/
	const Matrix& getA() const;
	/**
	* \brief Функция, возвращающая матрицу b
	* \return матрица b
	*/
	const LinMatrix& getb() const;
	/**
	* \brief Функция, возвращающая корень
	* \return корень уравнения
	*/
```

```cpp
	const LinMatrix& getx() const;
	/**
	* \brief Функция, возвращающая число обуцсловленности
	* \return число обусловленности
	*/
	const long double getConditionalityNumber() const;

	/**
	* \brief Фукнция, считающая корень СЛАУ с помощью LDLt разложения
	*/
	void solve();
};

// Класс решения
class Solution {
	// Массив всех СЛАУ
	std::vector<LinEquation> linEquations;
	// Файл, в из которого сичтываются СЛАУ
	const std::string inFilename;
	// Файл, в который записываются СЛАУ
	const std::string outFilename;

	bool initialized = false;
	int eMin = 0, eMax = 0;
	int equationsCount = 0;

	/**
	 * \brief Функция, сичтывающая матрицы из заданного файла
	 */
	void readEquationsFromFile();
	/**
	 * \brief Функция, записывающая матрицы в заданный файл
	 */
	void writeMatrices();
	/**
	 * \brief Функция обработки ошибок
	 * \param error - ошибка
	 */
	void parseError(const std::string& error);
public:
	/**
	 * \brief Конструктор класса
	 * \param inFilename - файл, из которого считываются матрицы
	 * \param outFilename - файл, в который записываются матрицы
	 */
	explicit Solution(const std::string& inFilename, const std::string& outFilename);

	/**
	 * \brief Функция, считающая корни у всех СЛАУ
	 */
	void begin();

	/**
	 * \brief Функция, вызываемая при завершении вычислений
	 */
	void end();

	/**
	 * \brief Деструктор класса
	 */
	~Solution();
};

/**
* \brief Точка входа
*/
int main()
```

Численный анализ методов

Рассмотрим зависимость нормы фактической ошибки и нормы невязки от числа обуловленности.



На графике видно, что ошибка и норма невязки растут с увеличением числа обусловленности. Причем зависимость логарифмов растет линейно.

Также было проверено выполнение неравенства $\|\delta x\| / \|x\| \le cond(A)\|\delta b\| / \|b\|$

Приведу ниже код проверки.

Для iй СЛАУ:

```
    db = 2 * (0.5 - rand(MATRIX_SIZE,1)) .* b;
    condNum = norm(A) * norm(inv(A));

    if ~(norm1(i) / norm(roots(:, i)) <= condNum * norm(db) * norm(b))
        inequalityTrue = false;
    end
...
if inequalityTrue
    fprintf("Неравество выполняется!\n");
else
    fprintf("Неравенство не выполняется!\n");
end
```

```
>> task2_analyze_roots
Неравество выполняется!
>>
```

Проверка на матрице с нулевым определителем

Построение матрицы:

В первой построенной ранее симметричной матрице поменяли строку и столбец на другие (в итоге матрица нетривиальная и симметричная, но с нулевым определителем)

```
A =

    8.9883   -0.6543   -0.8179   -0.9815   -0.4908   -0.3272   -1.4723   -1.7994    0.1636    0.4908
   -0.8179    0.0670    1.0837    0.1005    0.0502    0.0335    0.1507    0.1842   -0.0167   -0.0502
   -0.8179    0.0670    1.0837    0.1005    0.0502    0.0335    0.1507    0.1842   -0.0167   -0.0502
   -0.9815    0.0804    0.1005    1.1206    0.0603    0.0402    0.1809    0.2211   -0.0201   -0.0603
   -0.4908    0.0402    0.0502    0.0603    1.0301    0.0201    0.0904    0.1105   -0.0100   -0.0301
   -0.3272    0.0268    0.0335    0.0402    0.0201    1.0134    0.0603    0.0737   -0.0067   -0.0201
   -1.4723    0.1206    0.1507    0.1809    0.0904    0.0603    1.2713    0.3316   -0.0301   -0.0904
   -1.7994    0.1474    0.1842    0.2211    0.1105    0.0737    0.3316    1.4053   -0.0368   -0.1105
    0.1636   -0.0134   -0.0167   -0.0201   -0.0100   -0.0067   -0.0301   -0.0368    1.0033    0.0100
    0.4908   -0.0402   -0.0502   -0.0603   -0.0301   -0.0201   -0.0904   -0.1105    0.0100    1.0301
```

К ней и к матрице B = 10A считался свободный член, и затем матрицы «отправлялись» в программу, считающую корень.

Корни удалось получить написанным алгоритмом. Нормы фактической ошибки равны 1.3846 и 34.7297 соответственно, то есть ошибка примерно так-же, как и значения матрицы (на порядок), в то время как норма невязки равны 0.0458 и 71.4629, то есть норма увеличилась на два порядка.

То есть получилось, что в целом алгоритм может посчитать корни, но у нас нет способа проанализировать их точность (так как не определено число обусловленности из-за не определенной обратной матрицы).

Вывод

В лабораторной работе мне удалось найти корни СЛАУ с помощью LDL^T разложения

В ходе исследования была проанализирована зависимость нормы невязки и фактической ошибки от числа обусловленности. Выяснилось, что ошибка растёт вместе с числом обусловленности, т. е. число обусловленности можно считать величиной, обратной к точности.

Также было проверено на истинность неравенство $\|\delta x\| / \|x\| \leq cond(A) \|\delta b\| / \|b\|$. Оно оказалось истинным для всех составленных СЛАУ