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 итерационными методами»

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

Преподаватель Добрецова С.Б.

Санкт-Петербург

2023

Оглавление

Формулировка задачи

Найти решение СЛАУ с помощью метода Зейделя с определённой точностью.

Формализация

- Пусть F(x) : R^n –> R^n = Ax - b.
- Требуется: найти x^{k+1} приближение такое, что $||x^{k+1} - x^k|| < \epsilon$. x^{k+1} будет являться корнем этого уравнения

Метод Зейделя с оптимальным параметром ($\boldsymbol{B_k = E, \alpha_k = \frac{2}{\lambda_1 + \lambda_n}}$):

Условия применимости:

- A – симметричная положительно определённая.

Алгоритм метода:

- Считаем k+1 приближение x из k-го по формуле
  - $x^{k+1} = x^k - \alpha\left(\underline{A}x^{k+1} + \overline{A}x^k - b\right)$
- Делаем это до тех пор, пока вторая норма $\left|x^{k+1} - x^k\right| < \epsilon$
- Полученное k + 1-е приближение – корень уравнения Ax=b с точностью эпсилон.

Предварительный анализ задачи

Построение корня:

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

Построение ортогональной матрицы:

- Для матрицы использовался вектор w = [0.3 0.4 0.5 0.6 0.3 0.2 0.9 0.11 -0.1 -0.3]
- Ортогональная матрица Q строилась с помощью преобразование Хаусхолдера: $Q = E - \frac{2ww^T}{||w||^2}$

Построение СЛАУ:

- Строим диагональную матрицу D с некоторым диапазоном чисел.
- Строим симметричную матрицу по формуле $A = QDQ^T$.
- Находим свободный член по формуле b = Ax.

Проверка условий для методов:

Метод Зейделя с оптимальным параметром:

- A – симметричная положительно определённая по построению.

Тестовый пример к методам

$$A = \begin{pmatrix} 4 & -1 & 1 \\ -1 & 3 & -1 \\ 1 & -1 & 5 \end{pmatrix}, b = \begin{pmatrix} 4 \\ 1 \\ 5 \end{pmatrix}$$

Собственные числа матрицы: $\lambda_1 = 2.325, \lambda_n = 6.214$

Коэффициент $\alpha = \frac{2}{\lambda_1+\lambda_2} = 0.234$

Формула итерации: $x^{k+1} = x^k - \alpha(\overline{A} * x^{k+1} + \underline{A} * x^k - b)$

Начальное приближение: $x^0 = \begin{pmatrix} 4 \\ 1 \\ 5 \end{pmatrix}$

Первое приближение: $x^1 = \begin{pmatrix} 0.253 \\ 0.176 \\ 0.669 \end{pmatrix}$

Второе приближение: $x^2 = \begin{pmatrix} 1.209 \\ 1.198 \\ 1.054 \end{pmatrix}$

Третье приближение: $x^2 = \begin{pmatrix} 1.047 \\ 1.083 \\ 0.999 \end{pmatrix}$

Видно, что x^i приближается с каждой итерацией, причем довольно быстро, к корню $x^* = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

Контрольные тесты

Построим графики зависимостей

- Нормы фактической ошибки и нормы невязки от точности.
- Числа итераций от определителя.

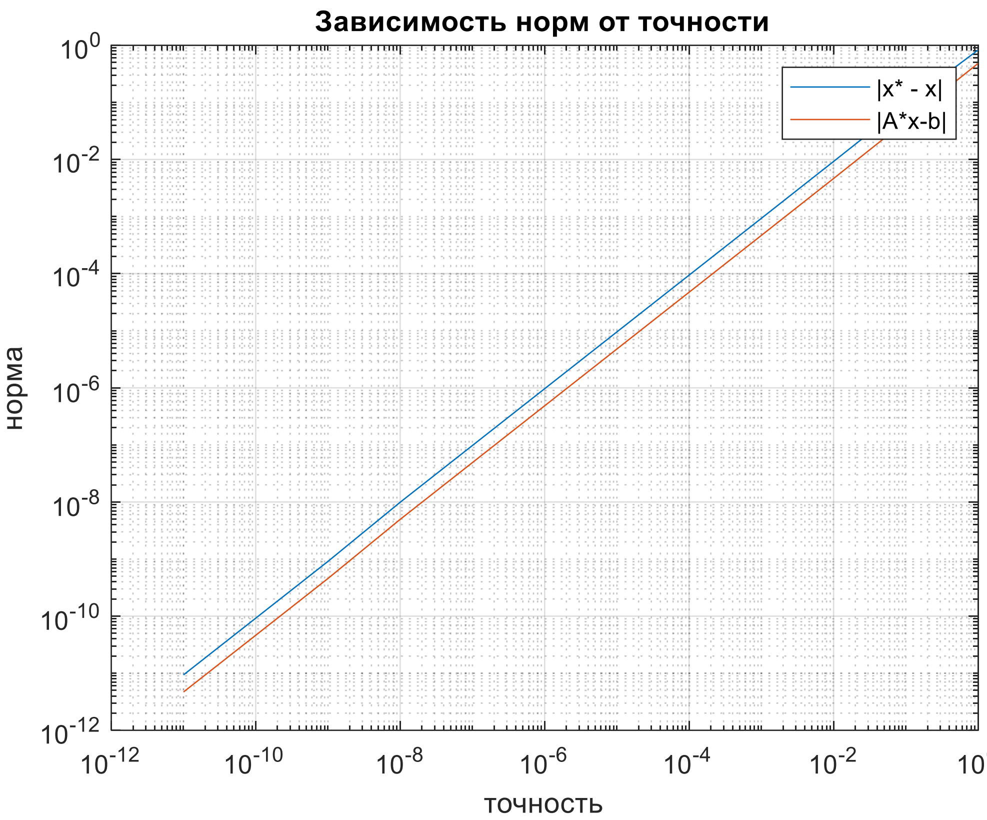
Модульная структура программы

```
// Класс одного линейного уравнения
typedef std::vector<std::vector<long double>> Matrix;
typedef std::vector<long double> LinMatrix;

// Utility функции для произведения операций над матрицами

LinMatrix operator*(Matrix A, LinMatrix x);
LinMatrix operator+(LinMatrix A, LinMatrix B);
LinMatrix operator-(LinMatrix A, LinMatrix B);
LinMatrix operator/(LinMatrix A, long double a);
long double norm(LinMatrix A);

class LinEquation {
	// A * x = b
	Matrix A;
	LinMatrix b;
	LinMatrix x;
	size_t dim;

	// Коэффициент альфа в итерациях
	long double a;

	// Точность
	long double epsilon;

	/**
	 * \brief Функция, считающая новое приближение на основе старого
	 * \param x - старое приближение
	 * \return новое приближение
	 */
	LinMatrix doOneIteration(LinMatrix& x);

	/**
	 * \brief функция, проверяющая, больше ли норма разности между приближениями, чем
эпсилон
	 * \param x1 - новое приближение
	 * \param x0 - старое приближение
	 * \return больше ли норма всё еще, чем эпсилон
	 */
	bool isConditionMet(LinMatrix& x1, LinMatrix& x0);
public:
	/**
	 * \brief Конструктор по умолчанию
	 */
	LinEquation() = default;
	/**
	 * \brief Конструктор класса
	 * \param A - матрица системы
	 * \param b - свободный член
	 * \param l1 - первое собственное число
	 * \param ln - последнее собственное число
	 */
	LinEquation(Matrix& A, LinMatrix& b, long double l1, long double ln);

	/**
	* \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 Фукнция, считающая корень СЛАУ с помощью метода Зейделя
	* \param epsilon - точность нахождения
	*/
	void solve(long double epsilon);
};

// 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()
```

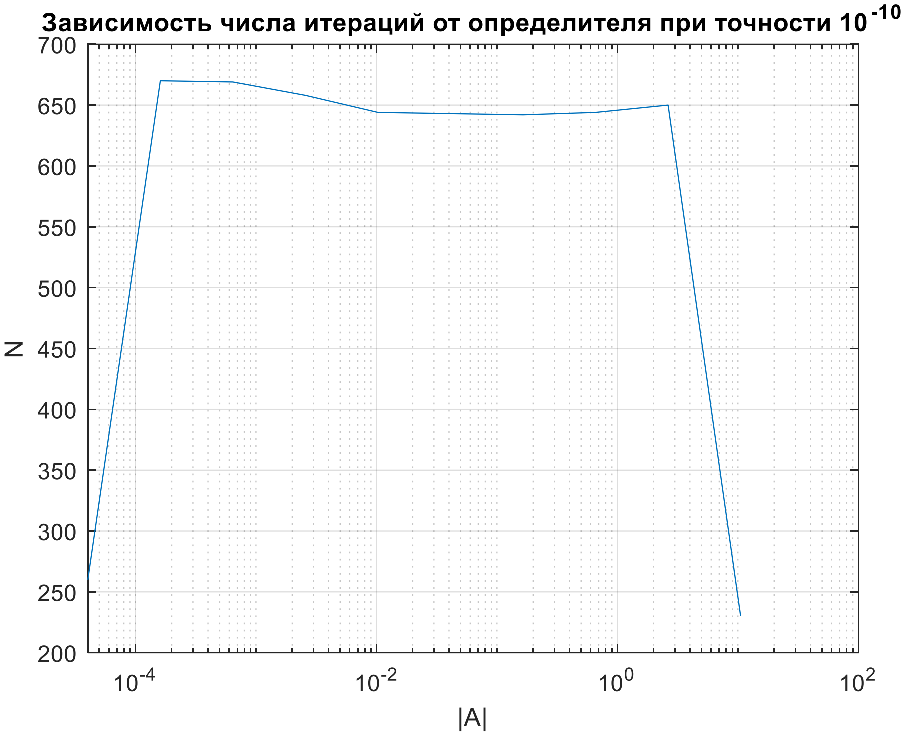
Численный анализ методов

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



На графике видно, что ошибка и норма невязки уменьшаются в соответствии с точностью. Причем зависимость логарифмов растет линейно.

Рассмотрим зависимость числа итераций от определителя.



Из графика видно, что число итераций либо не зависит от определителя (Скачок и падение на графике случились из-за изменения числа обусловленности), либо зависимость нетривиальная и незначительная.

Вывод

В лабораторной работе мне удалось найти корни СЛАУ с заданной точности с помощью итерационного стационарного метода Зейделя.

В ходе исследования была проанализирована зависимость нормы невязки и фактической ошибки от точности, отклонений замечено не было. Также была исследована зависимость числа итераций от определителя матрицы системы, явной зависимости установлено не было.