Отчет о выполненнном задании по численным методам линейной алгебры.

Егор Подлесов

14 ноября 2023 г.

Краткий обзор.

Сатья показывает результаты реализации и замеров времени работы алгоритма решения **системы линейных алгебраических уравнений** методом **вращений Гивенса**.

Листинг 1: Схема проекта

```
cmc-numerical-methods-of-linear-algebra
|-- benchmark
|    --- CMakeLists.txt
|    --- givens_rotations.cpp
|-- CMakeLists.txt
|-- src
|    |-- CMakeLists.txt
|    |-- common.h
|    |-- givens_rotations.cpp
|    |-- givens_rotations.h
|    |-- matrix.cpp
|    |-- matrix.h
|    |-- triangular_matrix.cpp
|    |-- triangular_matrix.h
|    |-- vector.cpp
|    |-- vector.h
|-- tests
     |-- CMakeLists.txt
     |-- givens_rotations_ut.cpp
     |-- SLAU_var_5.csv
```

- **src** - реализация классов **TMatrix**, **TVector**, а также решения **СЛАУ** вращениями Гивенса.
- **benchmark** - замер времени выполнения алгоритма.
- **tests** - тестирование алгоритма.

Постановка задачи.

Дано: Невырожденная матрица A размеров 100×100.
Требуется:

1. Реализовать алгоритм решения **СЛАУ** с коэффициентами - элементами данной матрицы, используя метод вращений Гивенса. Вектор-результат b сгенерировать из случайных чисел на отрезке $[-1, 1]$ с равномерным распределением. То есть надо решить систему вида
$$Ax = b,$$
где $A \in \mathbb{R}^{100\times100}$, $b \in \mathbb{R}^{100}$.

2. Протестировать программу - убедиться что найденный вектор x удовлетворяет следующему неравенству $\|Ax - b\| < \varepsilon$, для некоторого достаточно малого ε.

3. Замерить время работы программы на входной матрице.

Метод вращений Гивенса.

Метод заключается в применении последовательных операций умножения на матрицы специального вида - так называемые матрицы вращений. Вследствии чего исходная матрица получает треугольный вид.

Матрица вращения имеет следующий вид:
$$Q_{ij} = \begin{bmatrix} q_{ii} & \cdots & q_{ij} \\ \vdots & \ddots & \vdots \\ q_{ji} & \cdots & q_{jj} \end{bmatrix}$$

То есть у матрицы Q_{ij} на всех позициях кроме (i, i), (i, j), (j, i), (j, j) стоят нулевые элементы. При этом ненулевые элементы подчиняются слеющим соотношениям:
$$\begin{cases} q_{ii} = q_{jj} \\ q_{ji} = -q_{ij} \\ q_{ii}^2 + q_{ij}^2 = 1 \end{cases}$$

Замечание: Можно положить, $q_{ii} = cos(\alpha)$ и $q_{ij} = sin(\alpha)$. Поэтому матрица Q_{ij} называется матрицей поворота на угол α.

Таким образом можно занулять поддиагональные элементы в каждой колонке. Для j-й колонки зануление поддиагонального элемента в строке i подразумевает матрицу поворота Q_{ji} с следующими элементами:

$$\begin{cases} q_{jj} = \frac{a_{jj}}{\sqrt{a_{jj}+a_{ij}}} \\ q_{ji} = \frac{a_{ij}}{\sqrt{a_{jj}+a_{ij}}} \end{cases}$$

Проверив подстановкой, убеждаемся что элемент a_{ij} зануляется.

Сделав это для всех поддиагональных элементов мы получаем разложение:

$$A = QR$$

где Q - унитарная (ортогональная) матрица, R - верхняя треугольная матрица.

Система

$$Ax = b$$

полчает вид

$$Rx = Q^T b$$

Такая система решается просто. Последовательно находим x_n, x_{n-1}, $\ldots$, x_1.

Краткое описание реализации.

Основные функции

- **NTriangularMatrix::SolveSystem** - решение СЛАУ с треугольной матрицей.
- **NGivensRotations::SolveSystem** - решение СЛАУ вращениями Гивенса.
- **NGivensRotations::SystemToTriangular** - приведение матрицы к треугольной форме вращениями Гивенса.

Вся реализация сохранена в репозитории **GitHub: ypodlesov**