Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Лабораторна робота 1

“Алгоритми на матрицях”

Виконали студенти 2-го курсу

Групи К-29

Ільчук Степан

Музика Віктор

Прокопчук Роман

Шевченко Марта

2021

**Розподіл завдань**

Ільчук Степан – 1(Алгоритм Штрассена для множення матриць)

Музика Віктор – 2(Побудова оберненої матриці методом Гауса-Жордана)

Прокопчук Роман – 3(Побудова оберненої матриці методом LU-розкладання)

Шевченко Марта – 4(Побудова оберненої матриці методом Ньютона)

**Тип даних**

Стиль матриць : T\*\*

Тип даних : дійсні числа

**Завдання 1(Ільчук Степан)**

Алгоритм Штрассена для множення матриць

**Теорія**

**Алгоритм**

**Складність**

**Мова програмування**

C++

**Модулі програми**

**Інтерфейс користувача**

Вхідні дані вводяться з текстового файла і виводяться у консоль

**Тестові приклади**

|  |  |
| --- | --- |
| **input** | **output** |
|  |  |

**Висновки**

**Література**

**Завдання 1(Музика Віктор)**

Побудова оберненої матриці методом Гауса-Жордана

**Теорія**

**Алгоритм**

**Складність**

**Мова програмування**

C++

**Модулі програми**

**Інтерфейс користувача**

Вхідні дані вводяться з текстового файла і виводяться у консоль

**Тестові приклади**

|  |  |
| --- | --- |
| **input** | **output** |
|  |  |

**Висновки**

**Література**

**Завдання 3(Прокопчук Роман)**

Побудова оберненої матриці методом LU-розкладання

**Теорія**

LU-розкладом квадратної матриці називається представлення матриці, у якої кожен діагональний елемент не дорівнює 0, у вигляді добутку нижньої трикутної матриці, у якої елементи на головній діагоналі дорівнюють 1, та верхньої трикутної матриці. Наприклад, для квадратної матриці порядку 3 LU-розклад буде виглядати так :

=

Нехай – матриця, для якої потрібно знайти обернену, тобто певну матрицю .

Тоді, за означенням оберненої матриці :

де – одинична матриця

Знаючи LU-розклад , запишемо , як добуток нижньої трикутної матриці та верхньої трикутної матриці :

Введемо матрицю . Отримаємо :

Тепер ми можемо знайти матрицю . Щоб це зробити, потрібно розв’язати систем лінійних рівнянь вигляду

, де – -й стовпчик одиничної матриці.

Оскільки – нижньотрикутна матриця, то ми зможемо легко розв’язати цю систему, підставляючи попередньо знайдені змінні у відповідні рівняння, а першу змінну знайти, як відношення першого елемента та першого елемента першого рядка у матриці

Таким чином, у кожній такій системі ми знайдемо -й стовпчик матриці . Згодом, отримаємо шукану матрицю .

Зробимо зворотню заміну :

Аналогічно пошуку матриці , знаходимо , тільки замість підставляємо -й стовпчик матриці , а змінні починаємо шукати «знизу до верху».

**Алгоритм**

* LU-розклад

1)Ініціалізуємо матрицю L нулями та копіюємо елементи матриці, яку розкладаємо, в матрицю U

2)Для кожного стовпчика матриці L будемо знаходити :

Для кожного рядка матриці U обчислимо :

=

3)Виконуємо крок 2, поки менше, або дорівнює порядку матриці

4)Отримуємо наш розклад

* Знаходження матриці

1)Для кожного стовпчика матриці знаходимо значення першого елемента

2)Для -го стовчика матриці знаходимо -те значення, підставляючи попередні у -те рівняння (рядок)

3)Переходимо до кроку 2, поки не знайдемо повністю наш -ий стовпчик

4)Переходимо до кроку 1, поки менше, або дорівнює порядку матриці

* Знаходження матриці

1)Для кожного стовпчика матриці знаходимо значення останнього елемента

2)Для -го стовчика матриці знаходимо -те значення, підставляючи попередньо знайдені у -те рівняння (рядок)

3)Переходимо до кроку 2, поки не знайдемо повністю наш -ий стовпчик

4)Переходимо до кроку 1, поки менше, або дорівнює порядку матриці

**Складність**

Складність LU-розкладу :), де -порядок матриці

Складність знаходження матриці :) = ), () – складність знаходження кожного стовпчика)

Складність знаходження матриці :) = )

*Загальна складність алгоритму : )*

**Мова програмування**

C++

**Модулі програми**

* void GetLUDecomposition(const Matrix& matrix, Matrix& l\_matrix, Matrix& u\_matrix)

Функція, яка виконує алгоритм LU-розкладання

* void SolveTheMatrixEquationForLower(const Matrix& lower, const Matrix& rhs\_matrix, Matrix& result)

Функція для знаходження матриці, добуток на яку з нижньотрикутною матрицею lower дає матрицю rhs\_matrix

* void SolveTheMatrixEquationForUpper(const Matrix& upper, const Matrix& rhs\_matrix, Matrix& result)

Функція, яка виконує те ж саме, що й попередня, тільки матриця lower замінюється на верхньотрикутну upper

* void GetInvertedMatrix(const Matrix& matrix)

Функція, яка знаходить саме обернену матрицю методом LU-розкладу

* void PrintMatrix(const Matrix& m)

Функція, яка виводить матрицю в консоль

**Інтерфейс користувача**

Вхідні дані вводяться з текстового файла і виводяться у консоль

**Тестові приклади**

|  |  |
| --- | --- |
| **input** | **output** |
| 4  2 3 1 5  6 13 5 19  2 19 10 23  4 10 11 31 | Matrix to inverse  2 3 1 5  6 13 5 19  2 19 10 23  4 10 11 31  L matrix :  1 0 0 0  3 1 0 0  1 4 1 0  2 1 7 1  U matrix :  2 3 1 5  0 4 2 4  0 0 1 2  0 0 0 3  Inverted matrix :  42.375 -14.875 3.75 -0.5  -6.25 2.25 -0.5 0  61.667 -22 5.6667 -0.66667  -25.333 9 -2.3333 0.33333 |

**Висновки**

Метод LU-розкладання є потужним і часто використовується на практиці, адже він дозволяє доволі легко обчислити визначник матриці, розв’язувати системи лінійних рівнянь, та знаходити обернені матриці за типову алгоритмічну складність для матриць : *)*, але його не завжди можна застосувати, наприклад, коли діагональний елемент матриці дорівнює нулю.

**Література**

* Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. Алгоритмы: построение и анализ. 3-е издание. М.: ИД "Вильямс", 2013[ст. 858]

**Завдання 4(Шевченко Марта)**

Побудова оберненої матриці методом Ньютона

**Теорія**

**Алгоритм**

**Складність**

**Мова програмування**

C++

**Модулі програми**

**Інтерфейс користувача**

Вхідні дані вводяться з текстового файла і виводяться у консоль

**Тестові приклади**

|  |  |
| --- | --- |
| **input** | **output** |
|  |  |

**Висновки**

**Література**