Algorytmy Numeryczne - Faza 3

Bartłomiej Kruk, Mateusz Motyl, Szymon Lichota

26 maja 2021

1 Treść zadania

Zaimplementować znajdowanie macierzy odwrotnej metodą Gaussa-Jordana (powinna działać przynajmniej dla macierzy do stopnia 6).

2 Teoretyczny opis metody

2.1 Metoda eliminacji Gaussa-Jordana

Metoda eliminacji Gaussa-Jordana przekształca macierz rozszerzoną $[A|f]$ układu

$$Ax = f, A = [a_{ij}]_{i,j=1,...,n}, f = f[f_1, ..., f_n]^T,$$

w następujący sposób.

I krok - pierwszy wiersz dzielimy przez a_{11} i zerujemy nim elementy pod przekątną w pierwszej kolumnie, przez co otrzymamy nowy równoważny układ o macierzy

$$\left[\begin{array}{cccc|c} 1 & a_{12}^{(1)} & ... & a_{1n}^{(1)} & f_1^{(1)} \\ 0 & a_{22}^{(1)} & ... & a_{2n}^{(1)} & f_2^{(1)} \\ ... & & & & \\ 0 & a_{n2}^{(1)} & ... & a_{nn}^{(1)} & f_n^{(1)} \end{array}\right] \left(\begin{array}{cc} a_{1j}^{(1)} = \frac{a_{ij}}{a_{11}}, & f_1^{(1)} = \frac{f_1}{a_{11}} \\ a_{ij}^{(1)} = a_{ij} - a_{1j}^{(1)}, i > 1, & f_i^{(1)} = f_i - f_1^{(1)} a_{i1}, i > 1 \end{array}\right)$$

II krok - drugi wiersz dzielimy przez $a_{22}^{(1)}$ i zerujemy nim elementy w drugiej kolumnie nad i pod przekątną:

$$\left[\begin{array}{ccccc|c} 1 & 0 & a_{13}^{(2)} & ... & a_{1n}^{(2)} & f_1^{(2)} \\ 0 & 1 & a_{23}^{(2)} & ... & a_{2n}^{(2)} & f_2^{(2)} \\ 0 & 0 & a_{33}^{(2)} & ... & a_{3n}^{(2)} & f_3^{(2)} \\ ... & & & & & \\ 0 & 0 & a_{n3}^{(2)} & ... & a_{nn}^{(2)} & f_n^{(2)} \end{array}\right]$$

itd. Po n **krokach** otrzymamy

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & ... & 0 & f_1^{(n)} \\ 0 & 1 & 0 & ... & 0 & f_2^{(n)} \\ ... & & & & & \\ 0 & 0 & 0 & ... & 1 & f_n^{(n)} \end{array}\right]$$

czyli układ, którego rozwiązaniem jest $x_j = f_j^{(n)}, 1 \leq j \leq n$.

2.2 Wyznaczanie macierzy odwrotnej metodą eliminacji Gaussa-Jordana

Wykorzystamy metodę Gaussa-Jordana do wyznaczenia macierzy odwrotnej A^{-1} dla danej macierzy A wymiaru $n \times n$, tzn. takiej, że:

$$AA^{-1} = I, \text{ gdzie } I = \begin{bmatrix} 1 & 0 ... & 0 \\ 0 & 1 ... & 0 \\ ... & & \\ 0 & 0 ... & 1 \end{bmatrix}$$

Jeśli oznaczymy przez $X^{(1)}$, $X^{(2)}$, ..., $X^{(n)}$ kolumny macierzy A^{-1}, to równość $AA^{-1} = I$ daje n układów z jednakową macierzą A:
$AX^{(1)} = [1, 0, ..., 0]^T$, $AX^{(2)} = [0, 1, ..., 0]^T$, ..., $AX^{(n)} = [0, 0, ..., 1]^T$. Metodą eliminacji Gaussa-Jordana przekształcimy macierz $[A|I]$ do macierzy $[I|B]$, czyli:

$$[A|I] \sim [I|B]$$

Wtedy B jest szukaną macierzą odwrotną, tzn. $A^{-1} = B$. Metoda Gaussa-Jordana oparta jest na następujących równaniach

$$A^{-1}A = I$$
$$A^{-1}I = A^{-1}$$

Macierz A oraz macierz jednostkową I poddajemy tej samej sekwencji operacji: z symetrii powyższych równań wynika, że jeżeli ta sekwencja doprowadzi do przekształcenia macierzy A w macierz I to przekształcana równolegle macierz jednostkowa powinna stać się macierzą A^{-1}. Aby sprawdzić, czy macierz A^{-1} rzeczywiście jest poszukiwaną macierzą odwrotną, możemy sprawdzić czy iloczyn $A \cdot A^{-1}$ równa się macierzy jednostkowej I lub odwrócić macierz wynikową otrzymując wejściową.

2.3 Przykład

Niech $A = \begin{bmatrix} 3 & 0 & 2 \\ 2 & 0 & -2 \\ 0 & 1 & 1 \end{bmatrix}$. Wtedy

$$\left[\begin{array}{ccc|ccc} 3 & 0 & 2 & 1 & 0 & 0 \\ 2 & 0 & -2 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right] \overset{w_1+w_2}{\longrightarrow} \left[\begin{array}{ccc|ccc} 5 & 0 & 0 & 1 & 1 & 0 \\ 2 & 0 & -2 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right] \overset{\frac{1}{5}w_1}{\longrightarrow}$$

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0.2 & 0.2 & 0 \\ 2 & 0 & -2 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right] \overset{w_2-2\cdot w_1}{\longrightarrow} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0.2 & 0.2 & 0 \\ 0 & 0 & -2 & -0.4 & 0.6 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right] \overset{-\frac{1}{2}\cdot w_2}{\longrightarrow}$$

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0.2 & 0.2 & 0 \\ 0 & 0 & 1 & 0.2 & -0.3 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right] \overset{w_2\leftrightarrow w_3}{\longrightarrow} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0.2 & 0.2 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0.2 & -0.3 & 0 \end{array}\right] \overset{w_2-w_3}{\longrightarrow} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0.2 & 0.2 & 0 \\ 0 & 1 & 0 & -0.2 & 0.3 & 1 \\ 0 & 0 & 1 & 0.2 & -0.3 & 0 \end{array}\right]$$

Stąd $A^{-1} = \begin{bmatrix} 0.2 & 0.2 & 0 \\ -0.2 & 0.3 & 1 \\ 0.2 & -0.3 & 0 \end{bmatrix}$.

Sprawdzenie:

$$\begin{bmatrix} 0.2 & 0.2 & 0 \\ -0.2 & 0.3 & 1 \\ 0.2 & -0.3 & 0 \end{bmatrix} \cdot \begin{bmatrix} 3 & 0 & 2 \\ 2 & 0 & -2 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

3 Opis implementacji

A - wejściowa macierz o wymiarach $n \times n$.

1. Tworzymy macierz rozszerzoną $[A|I]$, gdzie I jest macierzą jednostkową rzędu n.
2. Jeżeli na przekątnej macierzy występuje 0 zamieniamy wiersze miejscami.
3. Stosujemy metodę Gaussa-Jordana na rozszerzonej macierzy A.
4. Wykonujemy operacje na wierszach w celu przerobienia początkowej macierzy A na macierz tożsamościową I.

Algorithm 1: Wyznaczanie macierzy odwrotnej metodą eliminacji Gaussa-Jordana

```
for i = 1, ..., n do
    for j = 1, ..., n do
        if i = j then
            A_{i,j+n} = 1
        else
            A_{i,j+n} = 0
for i = 1, ..., n do
    if A_{i,i} then
        for j = 1, ..., n do
            if i ≠ j then
                if A_{j,i} ≠ 0 and A_{i,j} ≠ 0 then
                    Swap(i, j, n, A)
                    Change(i, j)
for i = 1, ..., n do
    if A_{i,i} = 0 then
        Print "Mathematical Error!"
        Stop
    for j = 1, ..., n do
        if i ≠ j then
            r = A_{i,j} / A_{i,i}
            for k = 1, ..., 2 · n do
                A_{j,k} = A_{j,k} − r · A_{i,k}
for i = 1, ..., n do
    for j = n + 1, ..., 2 · n do
        A_{i,j} = A_{i,j} / A_{i,i}
```

4 Demonstracja

4.1 Opis programu

Specyfikacja wejścia:
W pierwszym wierszu standardowego wejścia podajemy stopień n macierzy, ktorą będziemy odwracać. W następnych n wierszach wstawiamy n wartości tworząc macierz kwadratową.
Specyfikacja wyjścia:
Program symuluje wyznaczanie macierzy odwrotnej metodą eliminacji Gaussa-Jordana i zwraca macierz odwrotną, jeśli jest to możliwe.

4.2 Przykłady

$$A = \begin{bmatrix} 3 & 0 & 2 \\ 2 & 0 & -2 \\ 0 & 1 & 1 \end{bmatrix}$$

Wejście:

3
3 0 2
2 0 −2
0 1 1

Wyjście:

$$\begin{bmatrix} 0,2 & 0,2 & 0,0 \\ -0,2 & 0,3 & 1 \\ 0,2 & -0,3 & 0,0 \end{bmatrix}$$

Inne przykłady:

- $A = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 3 & 1 & 2 \\ 2 & 3 & 1 & 0 \\ 1 & 0 & 2 & 1 \end{bmatrix}$

  Wyjście:

  $$A^{-1} = \begin{bmatrix} -3 & -0,5 & 1,5 & 1 \\ 1 & 0,25 & -0,25 & -0,5 \\ 3 & 0,25 & -1,25 & -0,5 \\ -3 & 0 & 1 & 1 \end{bmatrix}$$

- $A = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 3 & 1 & 2 & 3 & 0 \\ 2 & 3 & 1 & 0 & 1 & 1 \\ 1 & 0 & 2 & 1 & 3 & 1 \\ 1 & 0 & 1 & 2 & 2 & 1 \\ 1 & 2 & 1 & 1 & 1 & 1 \end{bmatrix}$

  Wyjście:

  $$A^{-1} = \begin{bmatrix} -1,33 & -0,33 & 1 & 0,33 & 0,33 & -0,33 \\ 0,166 & 0,166 & 0 & -0,166 & -0,166 & 0,166 \\ -2,66 & -0,66 & 0 & 1,66 & -1,33 & 2,33 \\ -1,166 & -0,166 & 0 & 0,166 & 0,166 & 0,833 \\ 1,5 & 0,5 & 0 & -0,5 & -0,5 & -1,5 \\ 3,33 & 0,33 & -1 & -1,33 & 0,66 & -0,66 \end{bmatrix}$$

5 Kod źródłowy

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

        ArrayList<Integer> change = new ArrayList<>();
        double r;
        int n;
        System.out.println("Podaj stopień n macierzy");
        Scanner scanner = new Scanner(System.in);
        n = scanner.nextInt();
        double[][] A = new double[n][n];
        double[][] I = new double[n][n];
        for (int i=0; i<n; i++){
            System.out.println("Podaj " + (i+1) +" wiersz");
            for(int j=0; j<n; j++){
                A[i][j]=scanner.nextDouble();
            }
        }
        for(int i=0; i<n; i++){
            System.out.print(Arrays.toString(A[i]));
            Arrays.fill(I[i], 0);
            I[i][i] = 1;
            System.out.println(Arrays.toString(I[i]));
        }
        for(int i=0; i<n; i++){
            if(A[i][i] == 0){
                for(int j=0; j<n; j++){
                    if(j!=i){
                        if(A[j][i] != 0 && A[i][j] != 0){
                            swap(i,j,n,A);
                            change.add(i);
                            change.add(j);
                            break;
                        }
                    }
                }
            }
        }
        for(int i=0; i<n; i++){
            if(A[i][i] == 0){
                System.out.println("Macierz odwrotna nie istnieje");
                System.exit(0);
            }
            for(int j=0; j<n; j++){
                if(i!=j){
                    r = A[j][i]/A[i][i];
                    for(int k=0; k<n; k++){
                        A[j][k] = A[j][k] - r * A[i][k];
                        I[j][k] = I[j][k] - r * I[i][k];
                    }
                }
            }
        }
```

```
        for(int i=0; i<n; i++){
            for(int j=0; j<n; j++){
                I[i][j] = I[i][j]/A[i][i];
                I[i][j] = Math.round(I[i][j]*100000)/100;
                I[i][j] /= 1000;
            }
        }

        double buf;
        while(change.size() != 0){
            for(int k=0; k<n; k++){
                buf = I[k][change.get(change.size()-1)];
                I[k][change.get(change.size()-1)] = I[k][change.get(change.size()-2)];
                I[k][change.get(change.size()-2)] = buf;
            }
            change.remove(change.size()-1);
            change.remove(change.size()-1);
        }

        System.out.println();

        for(int i=0; i<n; i++) {
            System.out.println(Arrays.toString(I[i]));
        }
    }

    public static void swap(int i, int j, int n, double[][] A) {
        double buf;
        for(int k=0; k<n; k++){
            buf = A[i][k];
            A[i][k] = A[j][k];
            A[j][k] = buf;
        }
    }
}
```

Literatura

[1] Jaruszewska-Walczak Danuta. wykład nr 8: metoda eliminacji Gaussa-Jordana, odwracanie macierzy, metoda rozkładu LUPlik

[2] Przykład wyznaczania macierzy odwrotnej metodą eliminacji Gaussa-Jordana

https://www.mathsisfun.com/algebra/matrix-inverse-row-operations-gauss-jordan.html

[3] Matrix Inverse Using Gauss Jordan Method Pseudocode

https://www.codesansar.com/numerical-methods/matrix-inverse-using-gauss-jordan-method-pseudocode.htm