Obliczenia Naukowe

Karol Janic

30 grudnia 2023

Spis treści

1 Wstęp

Rozwiązanie układu równań liniowych $Ax = b$ jest jednym z podstawowych problemów. Polega na znalezieniu wektora x spełniającego równanie $Ax = b$, gdzie A jest macierzą kwadratową stopnia n, a b jest wektorem długości n. W przypadku, gdy A jest macierzą nieosobliwą, to układ ma jednoznaczne rozwiązanie. W przypadku, gdy A jest macierzą osobliwą, to układ może mieć nieskończenie wiele rozwiązań lub nie mieć ich wcale.

2 Metoda eliminacji Gaussa

W celu znalezienia rozwiązania równania $Ax = b$ można wykorzystać metodę eliminacji Gaussa, która polega na przekształceniu macierzy A do macierzy trójkątnej górnej, a następnie wyznaczeniu rozwiązania poprzez podstawienia od końca.

2.1 Przekształcenie macierzy do postaci trójkątnej górnej

2.1.1 Opis metody

Niech $A \in \mathbb{R}^{n\times n}$ będzie macierzą nieosobliwą, czyli $\det(A) \neq 0$ oraz niech $b \in \mathbb{R}^n$ będzie wektorem. Pierwszym krokiem jest przekształcenie macierzy A do macierzy trójkątnej górnej poprzez wykonywanie operacji elementarnych na wierszach, czyli:

- Dodanie do wiersza i wiersza j pomnożonego przez skalar $\alpha \in \mathbb{R}$.
- Pomnożenie wiersza i przez skalar $\alpha \in \mathbb{R}$.
- Zamiana wierszy i i j.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1k} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2k} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & \dots & a_{kk} & \dots & a_{kn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nk} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \\ \vdots \\ b_n \end{bmatrix} \longrightarrow \begin{bmatrix} a'_{11} & a'_{12} & \dots & a'_{1k} & \dots & a'_{1n} \\ 0 & a'_{22} & \dots & a'_{2k} & \dots & a'_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a'_{kk} & \dots & a'_{kn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & a'_{nn} \end{bmatrix} \begin{bmatrix} b'_1 \\ b'_2 \\ \vdots \\ b'_k \\ \vdots \\ b'_n \end{bmatrix}$$

Taką formę macierzy można uzyskać poprzez zerowanie elementów pod przekątną, czyli takich a_{ij}, że $i > j$. Jedną z operacji elementarnych jest dodanie do wiersza i wiersza j pomnożonego przez skalar $\alpha \in \mathbb{R}$. Wówczas rozwiązania układu równań nie ulegają zmianie. Można zatem dla każdego wiersza $i > j$ wykonać operację dodania do wiersza i wiersza j pomnożonego przez:

$$\lambda_{ij} = -\frac{a_{ij}^{(k)}}{a_{jj}^{(k)}},$$

gdzie j jest aktualnie przetwarzanym wierszem a k jest numerem iteracji. W k-tej iteracji eliminowane są elementy a_{ij}, gdzie $i > k$ a $j = k$. Równocześnie należy zaaktualizować wektor b poprzez dodanie do i-tego elementu b j-tego elementu b pomnożonego przez λ_{ij}.

Ponieważ macierz jest nieosobliwa, to $a_{jj}^{(k)} \neq 0$. Jeżeli jednak macierz jest osobliwa to może się zdarzyć, że dla pewnego k i j zachodzi $a_{jj}^{(k)} = 0$. Wówczas na mocy innego przekształcenia elementarnego, zamiany dwóch wierszy, można przekształcić macierz tak, aby $a_{jj}^{(k)} \neq 0$ - jest to tzw. wybór elementu głównego.

Warto zauważyć, że w implementacji algorytmu ważne jest nie tylko aby element $a_{jj}^{(k)}$ był różny od zera, ale również nie był zbyt mały co do wartości bezwzględnej, ponieważ może to prowadzić do powiększenia błędów numerycznych. Zatem w implementacji algorytmu można zawsze wybierać takie elementy $a_{j'j}^{(k)}$, że

$$|a_{j'j}^{(k)}| = \max_{i\geq j} |a_{ij}^{(k)}|$$

i odpowiednio zamieniać wiersze. Równocześnie należy zamieniać elementy wektora b.

2.1.2 Wizualiacja działania algorytmu bez wyboru elementu głównego

Macierz początkowa i początkowy wektor prawych stron:

$$
\begin{bmatrix}
a_{11}^{(1)} & \dots & a_{1k}^{(1)} & \dots & a_{1n}^{(1)} \\
a_{21}^{(1)} & \dots & a_{2k}^{(1)} & \dots & a_{2n}^{(1)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
a_{k1}^{(1)} & \dots & a_{kk}^{(1)} & \dots & a_{kn}^{(1)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
a_{n1}^{(1)} & \dots & a_{nk}^{(1)} & \dots & a_{nn}^{(1)}
\end{bmatrix}
\begin{bmatrix}
b_1^{(1)} \\ b_2^{(1)} \\ \vdots \\ b_k^{(1)} \\ \vdots \\ b_n^{(1)}
\end{bmatrix}
$$

Pierwsza iteracja:

$$
\begin{bmatrix}
a_{11}^{(1)} & \dots & a_{1k}^{(1)} & \dots & a_{1n}^{(1)} \\
a_{21}^{(1)} - \frac{a_{21}^{(1)}}{a_{11}^{(1)}} \cdot a_{11}^{(1)} & \dots & a_{2k}^{(1)} - \frac{a_{21}^{(1)}}{a_{11}^{(1)}} \cdot a_{1k}^{(1)} & \dots & a_{2n}^{(1)} - \frac{a_{21}^{(1)}}{a_{11}^{(1)}} \cdot a_{1n}^{(1)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
a_{k1}^{(1)} - \frac{a_{k1}^{(1)}}{a_{11}^{(1)}} \cdot a_{11}^{(1)} & \dots & a_{kk}^{(1)} - \frac{a_{k1}^{(1)}}{a_{11}^{(1)}} \cdot a_{1k}^{(1)} & \dots & a_{kn}^{(1)} - \frac{a_{k1}^{(1)}}{a_{11}^{(1)}} \cdot a_{1n}^{(1)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
a_{n1}^{(1)} - \frac{a_{n1}^{(1)}}{a_{11}^{(1)}} \cdot a_{11}^{(1)} & \dots & a_{nk}^{(1)} - \frac{a_{n1}^{(1)}}{a_{11}^{(1)}} \cdot a_{1k}^{(1)} & \dots & a_{nn}^{(1)} - \frac{a_{n1}^{(1)}}{a_{11}^{(1)}} \cdot a_{1n}^{(1)}
\end{bmatrix}
\begin{bmatrix}
b_1^{(1)} \\
b_2^{(1)} - \frac{a_{21}^{(1)}}{a_{11}^{(1)}} \cdot b_1^{(1)} \\
\vdots \\
b_k^{(1)} - \frac{a_{k1}^{(1)}}{a_{11}^{(1)}} \cdot b_1^{(1)} \\
\vdots \\
b_n^{(1)} - \frac{a_{n1}^{(1)}}{a_{11}^{(1)}} \cdot b_1^{(1)}
\end{bmatrix}
=
$$

$$
=
\begin{bmatrix}
a_{11}^{(1)} & \dots & a_{1k}^{(1)} & \dots & a_{1n}^{(1)} \\
0 & \dots & a_{2k}^{(2)} & \dots & a_{2n}^{(2)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \dots & a_{kk}^{(2)} & \dots & a_{kn}^{(2)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \dots & a_{nk}^{(2)} & \dots & a_{nn}^{(2)}
\end{bmatrix}
\begin{bmatrix}
b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_k^{(2)} \\ \vdots \\ b_n^{(2)}
\end{bmatrix}
$$

k-ta iteracja:

$$
\begin{bmatrix}
a_{11}^{(1)} & \dots & a_{1k}^{(1)} & \dots & a_{1n}^{(1)} \\
0 & \dots & a_{2k}^{(2)} & \dots & a_{2n}^{(2)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \dots & a_{kk}^{(k)} & \dots & a_{kn}^{(k)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \dots & a_{nk}^{(k)} - \frac{a_{nk}^{(k)}}{a_{kk}^{(k)}} \cdot a_{kk}^{(k)} & \dots & a_{nn}^{(k)} - \frac{a_{nk}^{(k)}}{a_{kk}^{(k)}} \cdot a_{kn}^{(k)}
\end{bmatrix}
\begin{bmatrix}
b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_k^{(k)} \\ \vdots \\ b_n^{(k)} - \frac{a_{nk}^{(k)}}{a_{kk}^{(k)}} \cdot b_k^{(k)}
\end{bmatrix}
=
$$

$$
=
\begin{bmatrix}
a_{11}^{(1)} & \dots & a_{1k}^{(1)} & \dots & a_{1n}^{(1)} \\
0 & \dots & a_{2k}^{(2)} & \dots & a_{2n}^{(2)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \dots & a_{kk}^{(2)} & \dots & a_{kn}^{(2)} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
0 & \dots & 0 & \dots & a_{nn}^{(k+1)}
\end{bmatrix}
\begin{bmatrix}
b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_k^{(k)} \\ \vdots \\ b_n^{(k+1)}
\end{bmatrix}
$$

Łącznie wykonuje się $n-1$ iteracji.

2.1.3 Pseudokod eliminacji bez wyboru elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$, wektor $b \in \mathbb{R}^n$
Wynik: Macierz A w postaci trójkątnej górnej, wektor b w postaci zmodyfikowanej

```
for k ← 1 to n − 1 do
    for i ← k + 1 to n do
        λ_ik ← −a_ik^(k) / a_kk^(k);
        for j ← k + 1 to n do
            a_ij^(k+1) ← a_ij^(k) + λ_ik · a_kj^(k);
        end
        b_i^(k+1) ← b_i^(k) + λ_ik · b_k^(k);
    end
end
```

2.1.4 Pseudokod eliminacji z wyborem elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$, wektor $b \in \mathbb{R}^n$
Wynik: Macierz A w postaci trójkątnej górnej, wektor b w postaci zmodyfikowanej

```
for k ← 1 to n − 1 do
    for i ← k + 1 to n do
        // Wyszukiwanie elementu głównego
        r ← k;
        for j ← k + 1 to n do
            if |a_jk^(k)| > |a_rk^(k)| then
                r ← j;
            end
        end
        if r ≠ k then
            // Zamiana wierszy w macierzy
            for j ← k to n do
                t ← a_kj^(k);
                a_kj^(k) ← a_rj^(k);
                a_rj^(k) ← t;
            end
            // Zamiana elementów w wektorze
            t ← b_k^(k);
            b_k^(k) ← b_r^(k);
            b_r^(k) ← t;
        end
        λ_ik ← −a_ik^(k) / a_kk^(k);
        for j ← k + 1 to n do
            a_ij^(k+1) ← a_ij^(k) + λ_ik · a_kj^(k);
        end
        b_i^(k+1) ← b_i^(k) + λ_ik · b_k^(k);
    end
end
```

2.2 Rozwiązanie układu równań na podstawie macierzy trójkątnej górnej

2.2.1 Opis metody

Po przekształceniu macierzy A do postaci trójkątnej górnej, otrzymujemy układ równań w postaci przedstawionej poniżej:

$$\begin{bmatrix} a'_{11} & a'_{12} & \dots & a'_{1k} & \dots & a'_{1n} \\ 0 & a'_{22} & \dots & a'_{2k} & \dots & a'_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a'_{kk} & \dots & a'_{kn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & a'_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b'_1 \\ b'_2 \\ \vdots \\ b'_k \\ \vdots \\ b'_n \end{bmatrix}$$

Rozwiązanie układu równań można znaleźć poprzez podstawienia od końca, czyli:

$$x_n = \frac{b'_n}{a'_{nn}} \qquad \text{oraz} \qquad x_k = \frac{b'_k - \sum_{j=k+1}^{n} a'_{kj} \cdot x_j}{a'_{kk}} \qquad \text{dla} \qquad k = n-1, n-2, \dots, 1$$

2.2.2 Pseudokod

Dane: Macierz $A \in \mathbb{R}^{n \times n}$ w postaci trójkątnej górnej, wektor $b \in \mathbb{R}^n$ w postaci zmodyfikowanej
Wynik: Wektor $x \in \mathbb{R}^n$ będący rozwiązaniem układu równań $Ax = b$

```
x_n ← b'_n / a'_nn;
for k ← n − 1 to 1 do
    s ← 0;
    for j ← k + 1 to n do
        s ← s + a'_kj · x_j;
    end
    x_k ← (b'_k − s) / a'_kk;
end
```

2.3 Złożoność obliczeniowa

Złożoność obliczeniowa eliminacji wynosi $O(n^3)$, ponieważ w każdej z $n-1$ iteracji należy wykonać $O(n^2)$ operacji - zarówno w wersji z wyborem i bez wybory elementu głównego. Złożoność obliczeniowa rozwiązania układu równań na podstawie macierzy trójkątnej górnej wynosi $O(n^2)$, ponieważ w każdej z n iteracji należy wykonać $O(n)$ operacji. Zatem łączna złożoność obliczeniowa wynosi $O(n^3)$.

2.4 Złożoność pamięciowa

Złożoność pamięciowa wynosi $O(n^2)$, ponieważ należy przechowywać macierz A, która ma n^2 wartości oraz wektor b, który ma n wartości.

3 Rozkład LU

Rozkład LU macierzy A jest to rozkład macierzy A na iloczyn macierzy dolnotrójkątnej L oraz macierzy górnotrójkątnej U tak aby $A = LU$.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1k} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2k} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & \dots & a_{kk} & \dots & a_{kn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nk} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & \dots & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ l_{k1} & l_{k2} & \dots & l_{kk} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \dots & l_{nk} & \dots & l_{nn} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1k} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2k} & \dots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & u_{kk} & \dots & u_{kn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & u_{nn} \end{bmatrix}$$

3.1 Metoda eliminacji Gaussa a rozkład LU

Jeśli przez $L^{(k)}$ oznaczymy macierz, której elementy l_{ij} są współczynnikami używanymi do wyzerowania elementów pod przekątną, tj. λ_{ij}, to kolejne iteracje algorytmu eliminacji Gaussa można zapisać jako:

$$A^{(k)} = L^{(k)}A^{(k-1)}$$

Wynikiem działania algorytmu eliminacji Gaussa jest macierz trójkątna górna U, zatem:

$$U = L^{(n-1)}L^{(n-2)}\dots L^{(1)}A$$

Wynika z tego, że

$$A = (L^{(n-1)}L^{(n-2)}\dots L^{(1)})^{-1}U = L^{(1)^{-1}}L^{(2)^{-1}}\dots L^{(n-1)^{-1}}U = LU,$$

gdzie $L = L^{(1)^{-1}}L^{(2)^{-1}}\dots L^{(n-1)^{-1}}$.

Dodatkowo $L^{(k)}$ ma postać:

$$\begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \lambda_{k+1,k} & & \\ & & \lambda_{k+2,k} & & \\ & & \vdots & \ddots & \\ & & \lambda_{n,k} & & 1 \end{bmatrix}$$

Z powyższych obserwacji wynika, że:

- $l_{ij} = \lambda_{i,j}$ dla $i < j$,
- $l_{ij} = 1$ dla $i = j$,
- $u_{ij} = a'_{ij}$ dla $i \geq j$.

3.2 Trzymanie rozkładu LU w pamięci

Rozkład LU można efektywnie trzymać w pamięci poprzez zapisanie macierzy L oraz U w miejscu macierzy A. Jest to możliwe, ponieważ macierz L jest macierzą dolnotrójkątną, a macierz U jest macierzą górnotrójkątną oraz na przekątnej macierzy L znajdują się same jedynki($l_{ii} = 1$) - nie trzeba ich zatem przechowywać. Zatem macierz LU można trzymać w pamięci(jako wynik działania algorytmu eliminacji Gaussa) w postaci:

$$\begin{bmatrix} a'_{11} & a'_{12} & \dots & a'_{1k} & \dots & a'_{1n} \\ \lambda_{21} & a'_{22} & \dots & a'_{2k} & \dots & a'_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \lambda_{k1} & \lambda_{k2} & \dots & a'_{kk} & \dots & a'_{kn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \lambda_{n1} & \lambda_{n2} & \dots & \lambda_{nk} & \dots & a'_{nn} \end{bmatrix}$$

3.3 Algorytm rozkładu LU

Algorytm rozkładu LU jest analogiczny do algorytmu eliminacji Gaussa przedstawionego wcześniej, z tą różnicą, że w każdej iteracji nie wykonuje się operacji na wektorze b oraz zamiast zapisywania zer pod przekątną w macierzy A zapisuje się współczynniki λ_{ij}.

3.4 Złożoność obliczeniowa i pamięciowa wyznaczenia rozkładu LU

Złożoność obliczeniowa rozkładu LU jak i złożoność pamięciowa jest taka sama jak w przypadku eliminacji Gaussa, czyli odpowiednio $O(n^3)$ i $O(n^2)$.

3.5 Rozwiązywanie układu równań na podstawie rozkładu LU

Mając rozkład LU macierzy A rozwiązanie układu sprowadza się do rozwiązania dwóch trójkątnych układów równań:

$$Ax = b \iff LUx = b \iff Ly = b \wedge Ux = y$$

$$\begin{bmatrix} 1 & 0 & \dots & 0 & \dots & 0 \\ l_{21} & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ l_{k1} & l_{k2} & \dots & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \dots & l_{nk} & \dots & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \\ \vdots \\ b_n \end{bmatrix}$$

$$\wedge$$

$$\begin{bmatrix} u_{11} & u_{12} & \dots & u_{1k} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2k} & \dots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & u_{kk} & \dots & u_{kn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & u_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \\ \vdots \\ y_n \end{bmatrix}$$

W tym przypadku w wariancie z wyborem elementu głównego funkcja rozkładająca macierz A na macierze L i U nie permutuje wektora b. Jedynie zwraca wektor permutacji p, który musi zostać uwzględniony w wyznaczeniu rozwiązania.

3.5.1 Złożoność obliczeniowa rozwiązywania układu równań na podstawie rozkładu LU

Złożoność obliczeniowa rozwiązywania układu równań na podstawie rozkładu LU wynosi $O(n^2)$, ponieważ należy rozwiązać dwa układy równań trójkątnych, które mają n równań i n niewiadomych.

3.5.2 Złożoność pamięciowa rozwiązywania układu równań na podstawie rozkładu LU

Złożoność pamięciowa rozwiązywania układu równań na podstawie rozkładu LU wynosi $O(n)$, ponieważ należy przechowywać dodatkowy wektor pomocniczy y.

3.5.3 Pseudokod rozwiązywania układu równań na podstawie rozkładu LU bez wyboru elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$ w postaci rozkładu LU, wektor $b \in \mathbb{R}^n$
Wynik: Wektor $x \in \mathbb{R}^n$ będący rozwiązaniem układu równań $Ax = b$

```
for k ← 1 to n do
    s ← 0;
    for j ← 1 to k − 1 do
        s ← s + l_kj · y_j;
    end
    y_k ← b_k − s;
end
for k ← n to 1 do
    s ← 0;
    for j ← k + 1 to n do
        s ← s + u_kj · x_j;
    end
    x_k ← (y_k − s) / u_kk;
end
```

3.5.4 Pseudokod rozwiązywania układu równań na podstawie rozkładu LU z wyborem elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$ w postaci rozkładu LU, wektor $b \in \mathbb{R}^n$, wektor permutacji $p \in \mathbb{N}^n$
Wynik: Wektor $x \in \mathbb{R}^n$ będący rozwiązaniem układu równań $Ax = b$

```
for k ← 1 to n do
    s ← 0;
    for j ← 1 to k − 1 do
        s ← s + l_kj · y_j;
    end
    k' ← p[k];
    y_k ← b_k' − s;
end
for k ← n to 1 do
    s ← 0;
    for j ← k + 1 to n do
        s ← s + u_kj · x_j;
    end
    x_k ← (y_k − s) / u_kk;
end
```

3.6 Zalety użycia rozkładu LU

Rozkład LU można wykorzystać do rozwiązywania układów równań z macierzą A o tej samej postaci, ale innym wektorem prawych stron b. Wówczas mając p różnych wektorów prawych stron złożoność obliczeniowa spada z $O(pn^3)$ do $O(n^3)$, ponieważ rozkład macierzy A na macierze L i U wykonuje się tylko raz.

4 Dostosowywanie algorytmu eliminacji Gaussa i rozkładu LU

Jeśli macierz A jest rzadka i ma specjaną strukturę, to można dostosować algorytm eliminacji Gaussa i rozkładu LU do tej struktury aby zmniejszyć złożoność obliczeniową i pamięciową.

4.1 Struktura macierzy

Rozpatrywana macierz A ma postać:

$$\begin{bmatrix} A_1 & C_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ B_2 & A_2 & C_2 & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & B_3 & A_3 & C_3 & \mathbf{0} & \dots & \mathbf{0} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \mathbf{0} & \dots & \mathbf{0} & B_{v-2} & A_{v-2} & C_{v-2} & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & B_{v-1} & A_{v-1} & C_{v-1} \\ \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & B_v & A_v \end{bmatrix},$$

gdzie $n, l, v \in \mathbb{N}$, $v = \frac{n}{l}$, $A_k \in \mathbb{R}^{l\times l}$ dla $k = 1, ..., v$, $B_k \in \mathbb{R}^{l\times l}$ dla $k = 2, ..., v$, $C_k \in \mathbb{R}^{l\times l}$ dla $k = 1, ..., v-1$.

Postacie macierzy składowych:

$$A_k = \begin{bmatrix} a_{11}^{(k)} & a_{12}^{(k)} & \dots & a_{1l}^{(k)} \\ a_{21}^{(k)} & a_{22}^{(k)} & \dots & a_{2l}^{(k)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{l1}^{(k)} & a_{l2}^{(k)} & \dots & a_{ll}^{(k)} \end{bmatrix}, \quad B_k = \begin{bmatrix} 0 & \dots & \dots & b_{1l}^{(k)} \\ 0 & \dots & \dots & b_{2l}^{(k)} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & b_{ll}^{(k)} \end{bmatrix}, \quad C_k = \begin{bmatrix} c_{11}^{(k)} & 0 & \dots & 0 \\ 0 & c_{22}^{(k)} & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & c_{ll}^{(k)} \end{bmatrix}$$

4.1.1 Realizacja struktury macierzy dla wersji bez wyboru elementu głównego

Można łatwo zauważyć, że większość macierzy pod przekątną jest wyzerowana. Zatem można pamiętać mniej wartości co przełoży się na mniejszą złożoność pamięciową. Na rysunku poniżej zaznaczono elementy macierzy, które są niezerowe lub staną się niezerowe w trakcie działania algorytmu eliminacji Gaussa. Zera pod przekątnymi macierzy C_k staną się niezerowe, ponieważ w trakcie eliminacji Gaussa od wiersza i odejmowany jest poprzedni wiersz, który w kolumnie gdzie wiersz i ma zera ma niezerową wartość. Zerowe elementy w macierzy B_k zostaną zerowe, ponieważ odejmowany wiersz ma już wyzerowane kolumny o indeksach mniejszych niż i.

$$\begin{bmatrix}
a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & a_{14}^{(1)} & c_{11}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{21}^{(1)} & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} & 0 & c_{22}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{31}^{(1)} & a_{32}^{(1)} & a_{33}^{(1)} & a_{34}^{(1)} & 0 & 0 & c_{33}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{41}^{(1)} & a_{42}^{(1)} & a_{43}^{(1)} & a_{44}^{(1)} & 0 & 0 & 0 & c_{44}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{14}^{(2)} & a_{11}^{(2)} & a_{12}^{(2)} & a_{13}^{(2)} & a_{14}^{(2)} & c_{11}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{24}^{(2)} & a_{21}^{(2)} & a_{22}^{(2)} & a_{23}^{(2)} & a_{24}^{(2)} & 0 & c_{22}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{34}^{(2)} & a_{31}^{(2)} & a_{32}^{(2)} & a_{33}^{(2)} & a_{34}^{(2)} & 0 & 0 & c_{33}^{(2)} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{44}^{(2)} & a_{41}^{(2)} & a_{42}^{(2)} & a_{43}^{(2)} & a_{44}^{(2)} & 0 & 0 & 0 & c_{44}^{(2)} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{14}^{(3)} & a_{11}^{(3)} & a_{12}^{(3)} & a_{13}^{(3)} & a_{14}^{(3)} & c_{11}^{(3)} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{24}^{(3)} & a_{21}^{(3)} & a_{22}^{(3)} & a_{23}^{(3)} & a_{24}^{(3)} & 0 & c_{22}^{(3)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{34}^{(3)} & a_{31}^{(3)} & a_{32}^{(3)} & a_{33}^{(3)} & a_{34}^{(3)} & 0 & 0 & c_{33}^{(3)} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44}^{(3)} & a_{41}^{(3)} & a_{42}^{(3)} & a_{43}^{(3)} & a_{44}^{(3)} & 0 & 0 & 0 & c_{44}^{(3)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{14}^{(4)} & a_{11}^{(4)} & a_{12}^{(4)} & a_{13}^{(4)} & a_{14}^{(4)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{24}^{(4)} & a_{21}^{(4)} & a_{22}^{(4)} & a_{23}^{(4)} & a_{24}^{(4)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{34}^{(4)} & a_{31}^{(4)} & a_{32}^{(4)} & a_{33}^{(4)} & a_{34}^{(4)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44}^{(4)} & a_{41}^{(4)} & a_{42}^{(4)} & a_{43}^{(4)} & a_{44}^{(4)}
\end{bmatrix}$$

4.1.2 Realizacja struktury macierzy dla wersji z wyborem elementu głównego

W tym przypadku liczba niezerowych elementów jest większa, ponieważ w każdej iteracji algorytmu eliminacji Gaussa wybierany jest element główny, który może znajdować się w jednym z wierszy poniżej. Z tego powodu koniec wiersza z niezerowymi elementami równy jest najdalszemu końcowi jednego z wierszy poniżej z niezerowymi wartościami w danej kolumnie. W macierzy B_k nie pojawią się żadne nowe niezerowe elementy, ponieważ odpowiednie kolumny w wierszach powyżej są już wyzerowane. Przedstawia to rysunek poniżej.

$$\begin{bmatrix}
a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & a_{14}^{(1)} & c_{11}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{21}^{(1)} & a_{22}^{(1)} & a_{23}^{(1)} & a_{24}^{(1)} & 0 & c_{22}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{31}^{(1)} & a_{32}^{(1)} & a_{33}^{(1)} & a_{34}^{(1)} & 0 & 0 & c_{33}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{41}^{(1)} & a_{42}^{(1)} & a_{43}^{(1)} & a_{44}^{(1)} & 0 & 0 & 0 & c_{44}^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{14}^{(2)} & a_{11}^{(2)} & a_{12}^{(2)} & a_{13}^{(2)} & a_{14}^{(2)} & c_{11}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{24}^{(2)} & a_{21}^{(2)} & a_{22}^{(2)} & a_{23}^{(2)} & a_{24}^{(2)} & 0 & c_{22}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{34}^{(2)} & a_{31}^{(2)} & a_{32}^{(2)} & a_{33}^{(2)} & a_{34}^{(2)} & 0 & 0 & c_{33}^{(2)} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & b_{44}^{(2)} & a_{41}^{(2)} & a_{42}^{(2)} & a_{43}^{(2)} & a_{44}^{(2)} & 0 & 0 & 0 & c_{44}^{(2)} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{14}^{(3)} & a_{11}^{(3)} & a_{12}^{(3)} & a_{13}^{(3)} & a_{14}^{(3)} & c_{11}^{(3)} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{24}^{(3)} & a_{21}^{(3)} & a_{22}^{(3)} & a_{23}^{(3)} & a_{24}^{(3)} & 0 & c_{22}^{(3)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{34}^{(3)} & a_{31}^{(3)} & a_{32}^{(3)} & a_{33}^{(3)} & a_{34}^{(3)} & 0 & 0 & c_{33}^{(3)} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44}^{(3)} & a_{41}^{(3)} & a_{42}^{(3)} & a_{43}^{(3)} & a_{44}^{(3)} & 0 & 0 & 0 & c_{44}^{(3)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{14}^{(4)} & a_{11}^{(4)} & a_{12}^{(4)} & a_{13}^{(4)} & a_{14}^{(4)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{24}^{(4)} & a_{21}^{(4)} & a_{22}^{(4)} & a_{23}^{(4)} & a_{24}^{(4)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{34}^{(4)} & a_{31}^{(4)} & a_{32}^{(4)} & a_{33}^{(4)} & a_{34}^{(4)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44}^{(4)} & a_{41}^{(4)} & a_{42}^{(4)} & a_{43}^{(4)} & a_{44}^{(4)}
\end{bmatrix}$$

4.2 Optymalizacja algorytmu eliminacji Gaussa i rozkładu LU

4.2.1 Algorytm eliminacji Gaussa

Na podstawie opisu algorytmu w sekcji 2.1 można łatwo zaadaptować algorytm eliminacji Gaussa do specyficznej struktury macierzy A:

1. Całkowita liczba iteracji algorytmu się nie zmieni, ponieważ liczba równań jest taka sama.
2. W każdej iteracji k eliminowana była zmienna z wierszy $i = k+1, k+2, ..., n$. W tym przypadku wystarczy zrobić to dla wierszy $i = k+1, k+2, ..., k'$, gdzie k' jest liczbą potencjalnie niezerowych elementów w kolumnie k.
3. W celu eliminacji zmiennej z wierszy $i = k+1, k+2, ..., k'$ należy odjąć odpowiednie przeskalowany wiersz k od wierszy o indeksach i. W wersji podstawowej odejmowane były kolumny od $k+1$ do n, a w tym przypadku wystarczy odjąć kolumny od $k+1$ do k'', gdzie k'' jest indeksem najdalszego potencjalnie niezerowego elementu w wierszu k.
4. W trakcie szukania elementu głównego można ograniczyć się do wierszy $i = k, k+1, ..., k'$.
5. Podczas zamiany wierszy można się ograniczyć do kolumn $j = k, k+1, ..., k''$.

4.2.2 Algorytm wyznaczania rozwiązania układu równań

Na podstawie opisu algorytmu w sekcji 2.2 można łatwo zaadaptować algorytm wyznaczania rozwiązania układu równań trójkątnych do specyficznej struktury macierzy A:

1. Całkowita liczba iteracji algorytmu się nie zmieni, ponieważ liczba równań jest taka sama.
2. W każdej iteracji można ograniczyć liczbę kolumn, które są sumowane korzystając z ograniczenia na indeks najdalszego potencjalnie niezerowego elementu w wierszu - k''.

4.3 Złożoność dostosowanego algorytmu eliminacji Gaussa i rozkładu LU

4.3.1 Złożoność obliczeniowa

W każdej z $n-1$ iteracji algorytmu eliminowania kolejnych zmiennych każda zmienna jest eliminowana z maksymalnie l wierszy. W celu jej eliminacji należy odjąć wartości od maksymalnie l kolumn w wersji bez wyboru elementu głównego oraz maksymalnie $2l$ kolumn w wersji z wyborem elementu głównego. Wynika z tego, że złożoność tego etapu wynosi $O(l^2 n)$. W celu wyznaczenia rozwiązania układu równań trójkątnych w każdej z n iteracji należy wykonać maksymalnie $O(l)$ operacji. Ponieważ l jest ustaloną stałą, to złożoność obliczeniowa obu powyższych etapów w obu wersjach wynosi $O(n)$.

4.3.2 Złożoność pamięciowa bez wyboru elementu głównego

1. Każda z v reprezentacji macierzy A_k potrzebuje l^2 liczb do jej zapisania.
2. Każda z $v-1$ reprezentacji macierzy B_k potrzebuje l liczb do jej zapisania.
3. Każda z $v-1$ reprezentacji macierzy C_k potrzebuje $\frac{l \cdot (l+1)}{2}$ liczb do jej zapisania.

Sumarycznie złożoność pamięciowa wynosi $O(ln)$, a ponieważ l jest ustaloną stałą, to mamy $O(n)$.

4.3.3 Złożoność pamięciowa z wyborem elementu głównego

1. Każda z v reprezentacji macierzy A_k potrzebuje l^2 liczb do jej zapisania.
2. Każda z $v-1$ reprezentacji macierzy B_k potrzebuje l liczb do jej zapisania.
3. Każda z $v-1$ reprezentacji macierzy C_k potrzebuje l^2 liczb do jej zapisania.
4. Dodatkowe niezerowe elementy powstałe w wyniku zamiany wierszy potrzebują $O(l)$ liczb do ich zapisania.
5. Wektor permutacji p potrzebuje n liczb do jego zapisania.

Sumarycznie złożoność pamięciowa wynosi $O(ln)$, a ponieważ l jest ustaloną stałą, to mamy $O(n)$.

4.4 Szczegóły implementacyjne

1. Dla metod bez wyboru i z wyborem elementu głównego zapewniono osobne implementacje struktury macierzy A.
2. Macierz A zapamiętywana jest w postaci wektora n wektorów o długościach równych liczbie potencjalnie niezerowych elementów wiersza.
3. Funkcja znajdująca rozkład LU macierzy A nadpisuje macierz A macierzami L i U.
4. Funkcja znajdująca rozład LU macierzy A z wyborem elementu głównego zwraca wektor permutacji p, który musi zostać uwzględniony podczas wyznaczania rozwiązania układu równań.
5. W implementacji zapewnio możliwość wybory precyzji liczb zmiennoprzecinkowych.
6. Zaimplementowano funckję rozwiązującą układ równań na podstawie danych A i b, funkcję która rozkłada macierz A na macierze L i U oraz funkcję która na podstawie rozkładu LU oraz wektora b wyznacza rozwiązanie układu równań. Każda z tych funkcji wspiera wersję bez wyboru elementu głównego oraz z wyborem elementu głównego.

4.5 Pseudokod

4.5.1 Jak wyznaczyć zakres wierszy z niezerowymi elementami w danej kolumnie?

Dane: Parametry macierzy(bez wyboru elementu głównego) $n, l \in \mathbb{N}$, numer kolumny $col \in \mathbb{N}$
Wynik: Zakres wierszy k'_{1B}, k'_{1T} z niezerowymi elementami w kolumnie k
$k \leftarrow div(col - 1, l)$;
return $\max(1, col - l), \min(n, (k + 1) * l)$;

Dane: Parametry macierzy(z wyborem elementu głównego) $n, l \in \mathbb{N}$, numer kolumny $col \in \mathbb{N}$
Wynik: Zakres wierszy k'_{2B}, k'_{2T} z niezerowymi elementami w kolumnie k
$k_1 \leftarrow div(col - l - 1, l)$;
$k_2 \leftarrow div(col, l)$;
return $\max(1, k_1 * l), min(n, (k_2 + 1) * l)$;

4.5.2 Jak wyznaczyć zakres kolumn z niezerowymi elementami w danym wierszu?

Dane: Parametry macierzy(bez wyboru elementu głównego) $n, l \in \mathbb{N}$, numer wiersza $row \in \mathbb{N}$
Wynik: Zakres kolumn k''_{1B}, k''_{1T} z niezerowymi elementami w wierszu row
$k \leftarrow div(row - 1, l)$;
return $\max(1, k * l), \min(n, row + l)$;

Dane: Parametry macierzy(z wyborem elementu głównego) $n, l \in \mathbb{N}$, numer wiersza $row \in \mathbb{N}$
Wynik: Zakres kolumn k''_{2B}, k''_{2T} z niezerowymi elementami w wierszu row
$k_1 \leftarrow div(row - 1, l)$;
$k_2 \leftarrow div(row, l)$;
return $\max(1, k_1 * l), min(n, (k_2 + 2) * l)$;

4.5.3 Rozkład LU macierzy A bez wyboru elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$, wektor $b \in \mathbb{R}^n$
Wynik: Macierz A przekształcona w rozkład LU
for $k \leftarrow 1$ **to** $n-1$ **do**
  for $i \leftarrow k+1$ **to** $k_{1T}^{''}$ **do**
    $\lambda_{ik} \leftarrow -\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$;
    for $j \leftarrow k+1$ **to** $k_{1T}^{'}$ **do**
      $a_{ij}^{(k+1)} \leftarrow a_{ij}^{(k)} + \lambda_{ik} \cdot a_{kj}^{(k)}$;
    end
    $b_i^{(k+1)} \leftarrow b_i^{(k)} + \lambda_{ik} \cdot b_k^{(k)}$;
  end
end

4.5.4 Rozkład LU macierzy A z wyborem elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$, wektor $b \in \mathbb{R}^n$
Wynik: Macierz A przekształcona w rozkład LU, wektor permutacji p
$p \leftarrow [1, 2, ..., n]$;
for $k \leftarrow 1$ **to** $n-1$ **do**
  for $i \leftarrow k+1$ **to** k''_{2T} **do**
    `// Wyszukiwanie elementu głównego`
    $r \leftarrow k$;
    for $j \leftarrow k+1$ **to** $k_{2T}^{'}$ **do**
      if $|a_{jk}^{(k)}| > |a_{rk}^{(k)}|$ **then**
        $r \leftarrow j$;
      end
    end
    if $r \neq k$ **then**
      `// Zamiana wektora permutacji`
      $t \leftarrow p_k$;
      $p_k \leftarrow p_r$;
      $p_r \leftarrow t$;
      `// Zamiana wierszy w macierzy`
      for $j \leftarrow k$ **to** $k_{2T}^{'}$ **do**
        $t \leftarrow a_{kj}^{(k)}$;
        $a_{kj}^{(k)} \leftarrow a_{rj}^{(k)}$;
        $a_{rj}^{(k)} \leftarrow t$;
      end
    end
    $\lambda_{ik} \leftarrow -\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$;
    for $j \leftarrow k+1$ **to** $k_{2T}^{'}$ **do**
      $a_{ij}^{(k+1)} \leftarrow a_{ij}^{(k)} + \lambda_{ik} \cdot a_{kj}^{(k)}$;
    end
    $a_{ik}^{(k+1)} \leftarrow \lambda_{ik}$;
  end
end

4.5.5 Rozwiązanie układu równań na podstawie rozkładu LU macierzy A bez wyboru elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$ w postaci rozkładu LU, wektor $b \in \mathbb{R}^n$
Wynik: Wektor $x \in \mathbb{R}^n$ będący rozwiązaniem układu równań $Ax = b$

```
for k ← 1 to n do
    s ← 0;
    for j ← k'_{1B} to k − 1 do
        s ← s + l_{kj} · y_j;
    end
    y_k ← b_k − s;
end
for k ← n to 1 do
    s ← 0;
    for j ← k + 1 to k'_{1T} do
        s ← s + u_{kj} · x_j;
    end
    x_k ← (y_k − s) / u_{kk};
end
```

4.5.6 Rozwiązanie układu równań na podstawie rozkładu LU macierzy A z wyborem elementu głównego

Dane: Macierz $A \in \mathbb{R}^{n\times n}$ w postaci rozkładu LU, wektor $b \in \mathbb{R}^n$, wektor permutacji $p \in \mathbb{N}^n$
Wynik: Wektor $x \in \mathbb{R}^n$ będący rozwiązaniem układu równań $Ax = b$

```
for k ← 1 to n do
    s ← 0;
    for j ← k'_{2B} to k − 1 do
        s ← s + l_{kj} · y_j;
    end
    k' ← p[k];
    y_k ← b_{k'} − s;
end
for k ← n to 1 do
    s ← 0;
    for j ← k + 1 to k'_{2T} do
        s ← s + u_{kj} · x_j;
    end
    x_k ← (y_k − s) / u_{kk};
end
```

5 Wyniki eksperymentalne