Akademia Górniczo-Hutnicza w Krakowie

Wydział Fizyki i Informatyki Stosowanej

Metody Numeryczne

Laboratorium 02: Rozkład LU macierzy trójdiagonalnej

Andrzej Świętek

04.03.2024

Contents

1. Wstęp teoretyczny

1.1. Rozkład macierzy LU

Metoda dekompozycji LU służy do rozwiązywania układów równań liniowych. Polega na zastąpieniu macierzy A dwiema macierzami trójkątnymi:

$$\mathbf{A = LU}$$

Przyjmujemy, że elementy na głównej przekątnej jednej z macierzy są równe 1. Metoda ta znajduje zastosowanie w szybkim rozwiązywaniu układów równań liniowych oraz obliczaniu wyznacznika macierzy.
Jest często stosowana w przypadku wielu obliczeń z stałymi współczynnikami i zmieniającymi się wyrazami wolnymi w układzie.

1.1.1. Opis Algorytmu:

1. Inicjalizacja Zdefiniuj macierze L i U o takich samych wymiarach jak macierz A. Wypełnij macierz L elementami zerowymi na głównej przekątnej.
2. Eliminacja Gaussa
   (a) Dla i = 1, 2, ..., n-1:
      i. Dla j = i+1, ..., n:
         A. Oblicz mnożnik eliminacji: $l_{ij} = a_{ij}/a_{ii}$.

B. Zapisz mnożnik w macierzy L: $L[i,j] = l_{ij}$.

C. Od każdego elementu j-tego wiersza macierzy A odejmij iloczyn mnożnika eliminacji i i-tego wiersza:

$$A[j,k] = A[j,k] - l_{ij} * A[i,k] \quad \text{dla} \quad k = i, i+1, ..., n.$$

3. Utworzenie macierzy U:
Macierz U jest macierzą trójkątną górną, która powstała po zakończeniu eliminacji Gaussa.

1.1.2. Przykład

Krok 1:

$$A = \begin{bmatrix} 2 & 1 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad U = \begin{pmatrix} 2 & 1 & 3 \\ 0 & 4 & 2 \\ 0 & 0 & 0 \end{pmatrix}$$

Krok 2:

1. Dla $i = 1$:
   (a) Dla $j = 2$:
   $l_{21} = \frac{4}{2} = 2$.
   $L[2,1] = 2$.
   $A[2,k] = A[2,k] - 2 \times A[1,k]$ dla $k = 1, 2, 3$.
2. Dla $i = 2$:
   (a) Dla $j = 3$:
   $l_{32} = \frac{7}{4} = 1.75$. $L[3,2] = 1.75$.
   $A[3,k] = A[3,k] - 1.75 \times A[2,k]$ dla $k = 2, 3$.

Krok 3:

$$A = \begin{bmatrix} 2 & 1 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ \frac{7}{2} & \frac{3}{2} & 1 \end{pmatrix} \qquad U = \begin{pmatrix} 2 & 1 & 3 \\ 0 & 3 & 0 \\ 0 & 0 & -\frac{3}{2} \end{pmatrix}$$

1.2. Dyskretyzacja równań

Wprowadzamy siatkę z węzłami tj.

$$x_i = -X_b + h \cdot (i-1), i \in [i, N]$$

$$h = \frac{2X_b}{N-1}$$

Zapis pochodnej zastępujemy iloracem równicowym zdefiniowanym na siatce:

$$\nabla^2 V = \frac{\partial^2 V}{\partial x^2} = \frac{V_{i-1} - 2V_i + V_{i+1}}{h^2} = -\rho(x)$$

gdzie h jest odległością między węzłami siatki a N jest ilością wszystkich węzłów.

1.3. Rozkład LU macierzy trójdiagonalnej

Metoda trójdiagonalna, znana również jako metoda Thomasa, jest jedną z popularnych technik rozwiązywania układów równań liniowych, w których macierz współczynników jest trójdiagonalna. Taka macierz posiada niezerowe elementy jedynie na głównej przekątnej oraz dwóch przekątnych bezpośrednio nad i pod nią.

$$A = \begin{bmatrix} d_1 & c_1 & 0 & 0 & \dots & 0 \\ a_2 & d_2 & c_2 & 0 & \dots & 0 \\ 0 & a_3 & d_3 & c_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & d_{n-1} & c_{n-1} \\ 0 & 0 & 0 & \dots & a_n & d_n \end{bmatrix}$$

Wartości elemetów macierzy otrzymujemy z równania drugiej pochodnej:

$$d_i = \frac{-2}{h^2}$$

$$a_i = c_i = \frac{1}{h^2}$$

$$\begin{bmatrix} d_1 & c_1 & 0 & 0 & 0 & 0 \\ a_2 & d_2 & c_2 & 0 & 0 & 0 \\ 0 & a_3 & d_3 & c_3 & 0 & 0 \\ & \vdots & & \ddots & & \\ 0 & 0 & 0 & a_{n-1} & d_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & a_n & d_n \end{bmatrix} = L \cdot U$$

$$L \cdot U = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ l_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & l_3 & 1 & 0 & 0 & 0 \\ & \vdots & & \ddots & & \\ 0 & 0 & 0 & l_{n-1} & 1 & 0 \\ 0 & 0 & 0 & 0 & l_n & 1 \end{bmatrix} \begin{bmatrix} u_1 & c_1 & 0 & 0 & 0 & 0 \\ 0 & u_2 & c_2 & 0 & 0 & 0 \\ 0 & 0 & u_3 & c_3 & 0 & 0 \\ & & \vdots & & \ddots & \\ 0 & 0 & 0 & 0 & u_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & 0 & u_n \end{bmatrix}$$

Dla macierzy trójdiagonalnej elementy macierzy L i U liczymy następująco:

$$v_1 = d_1, i = 1$$

$$l_i = \frac{a_i}{u_{i-1}}, i \in [2, N]$$

$$u_i = d_i - l_i \cdot c_{i-1}, i \in [2, N]$$

Metoda trójdiagonalna jest szczególnie skuteczna w przypadku układów równań, w których macierz współczynników jest rzadka, co oznacza, że większość elementów jest zerowa, co pozwala na efektywne wykorzystanie pamięci i czasu obliczeń.

2. Problem

2.1. Rozwiązanie równania Poissona

$$\nabla^2 V(x) = -\rho(x)$$

w przedziale $x \in [-X_b, X_b]$ z warunkiem brzegowym $V(-X_b) = V(X_b) = 0$ dla rozkładu gęstości:

$$\rho(x) = \begin{cases} 0, & \text{if } x \in [-X_b, -X_a) \\ +1, & \text{if } x \in [-X_a, 0) \\ 0, & \text{if } x = 0 \\ -1, & \text{if } x \in (0, X_a] \\ 0, & \text{if } x \in (X_a, X_b] \end{cases}$$

Równanie to mówi nam, że laplasjan pola potencjału w punkcie jest równy przeciwności gęstości ładunku w tym punkcie. Warunki brzegowe są często dodawane, aby rozwiązać równanie w konkretnych warunkach granicznych, na przykład gdy potencjał jest znany na pewnych powierzchniach lub w określonych punktach.

W przypadku rozpatrywanego problemu, równanie Poissona jest używane do obliczenia potencjału elektrycznego w danym przedziale $x \in [-X_b, X_b]$, przy założeniu określonego rozkładu gęstości ładunku $\rho(x)$ i warunków brzegowych $V(-X_b) = V(X_b) = 0$. Metoda rozkładu LU dla macierzy trójdiagonalnej jest wykorzystywana do rozwiązania tego równania numerycznie, co pozwala uzyskać przybliżone wartości potencjału w danym przedziale.

Dokładny wynik potencjału elektrycznego do którego dążymy w rozpatrywanym przedziale jest następujący:

$$V(x) = \begin{cases} \frac{x}{16} + \frac{1}{8}, & \text{dla } x \in [-X_b, -X_a] \\ -\frac{x^2}{2} - \frac{7}{16}x, & \text{dla } x \in [-X_a, 0] \\ \frac{x^2}{2} - \frac{7}{16}x, & \text{dla } x \in [0, X_a] \\ \frac{x}{16} - \frac{1}{8}, & \text{dla } x \in [X_a, X_b] \end{cases}$$

3. Implementacja

3.1. Inicjalizacja danymi początkowymi

Implementacja zaczyna się od zdefiniowania stałych będących założeniami zadania jak i deklaracją głównych wektorów. Możliwym jest używanie *constexpr* aby wykonać niektóre operacje już na etapie compile-time'u.

```
    constexpr int N = 500;

    constexpr double Xa = 0.5;
    constexpr double Xb = 2.0;
    constexpr double h = 2.0 * Xb / (double)(N-1);

    double vector_d[N]; // -2/h^2
    double vector_a[N]; // 1/h^2
    double vector_c[N]; // 1/h^2
    double vector_x_i[N];
    double vector_ro[N];
```

Listing 1: Inicjalizacja

Pierwszym krokiem jest wypełnienie wartościami wektrów d, a i c które kolejno stanowią diagonale macierzy, jak i górną i dolna diagonale, które otrzymujemy w wyniku dyskretyzacji równań.
Wypełnianmy kolejno zgodnie z równościami:

$$d_i = \frac{-2}{h^2}$$

```

for(int i = 0; i < N; i++)
    vector_d[i] = -2.0/ (h*h);
```

Listing 2: Wypełnienie wektora d_i

$$a_i = c_i = \frac{1}{h^2}$$

```

for(int i = 0; i < N; i++)
{
    vector_a[i] = 1.0/ (h*h);
    vector_c[i] = 1.0/ (h*h);
}
```

Listing 3: Wypełnienie wektorów a_i i c_i

Następnie uzupełniamy wektor reprezentujący siaktkę z węzłami:

$$x_i = -X_b + h \cdot (i-1), \qquad i = 1, 2, ...N, \qquad h = \frac{2 \cdot X_b}{(N-1)}$$

Natomiast aby wypełnić wektor wyrazów wolnych kożystamy z funkcji gęstości Poissona:

```

double poisson_ro(double x, double Xa, double Xb) {
    if ( x == 0.0 || (x >= -Xb && x < -Xa) || (x > Xa && x <= Xb))
```

```
        return 0.;
    if ( x >= -Xa && x < 0.0 ) return 1.;
    if ( x > 0.0 && x <= Xa ) return -1.;
    else return 0.0;
}

for (int i = 0; i < N; i++)
    vector_ro[i] = -poisson_ro(vector_x_i[i], Xa, Xb);
```

Listing 4: Wypełnienie wektora wyrazów wolnych

Następnie korzystając z warunków brzegowych inicjujemy pierwsze i ostatnie elementy wketorów d,c i ρ:

```

    // Warunki Brzegowe
    vector_d[0]  = 1;   vector_d[N-1]   = 1;
    vector_c[0]  = 0;   vector_c[N-1]   = 0;
    vector_ro[0] = 0;   vector_ro[N-1]  = 0;
```

Listing 5: Uzupełnienie warunków brzegowych

3.2. Rozkład macierzy na LU

Zgodnie z poniższym wzorem przekładamy matematyczny zapis na kod, tj pierwszy wyraz macierzy górnej jest taki sam jak pierwszy element wektora d_i. Następnie, aby policzyć element macierzy U, musimy wyliczyć wartość l_i, którą z kolei liczymy:

$$
\begin{aligned}
u_1 &= d_1 \\
l_i &= \frac{a_i}{u_{i-1}}, \quad i = 2, 3, \ldots, N \\
u_i &= d_i - l_i c_{i-1}, \quad i = 2, 3, \ldots, N
\end{aligned}
$$

```

    double l[N];
    double u[N];
    u[0] = vector_d[0];                 // u_1 = d_1
    for(int i = 1; i< N; i++)           // n = 2,3 ..N
    {
        l[i] = vector_a[i] / u[i-1];
        u[i] = vector_d[i] - l[i] * vector_c[i-1];
    }
```

Listing 6: Uzupełnienie macierzy L i macierzy U zgodnie z wzorami

- **for(int i = 1; i < N; i++)**:
  Pętla iteruje od 2 do N (wykluczając 1 element)
- **l[i] = vector_a[i] / u[i-1]**:
  Obliczenie wartości l_i dla $i = 2, 3, ..., N$. Mianownikiem jest element $u[i-1]$ czyli element o indeksie o jeden mniejszym od i. Dzieje się tak, ponieważ elementy macierzy L są obliczane wierszami, a do obliczenia l_i potrzebujemy wartości u_{i-1} z poprzedniego wiersza.

- **u[i] = vector_d[i] - l[i] * vector_c[i-1]**:
  Obliczenie wartości u_i dla i = 2, 3, ..., N. Wzór uwzględnia iloczyn l_i i c_{i-1}, gdzie c_{i-1} jest elementem o indeksie o jeden mniejszym od i w wektorze c. Wynika to z rozmieszczenia elementów w macierzu L i U.

3.3. Rozwiązanie równania i zapisanie wyniku

```
1
2     /**
3     * Ly = b - b to ro
4     * y1 = b1
5     * yi = bi - l_i*y_i-1
6     */
7     double vector_y[N]; for (int i =0 ; i < N; i++) vector_y[i] =
      0;
8     vector_y[0] = vector_ro[0];
9     for(int i = 1; i < N; i++)
10        vector_y[i] = vector_ro[i] - l[i] * vector_y[i-1];   // b_i
           l_i *y_{ i 1 }
11
12
13    double vector_v[N];
14    vector_v[N-1] = vector_y[N-1]/u[N-1];   // v_n = y_n/u_n
15    for (int i = (N-1)-1; i >= 0; i--)
16            vector_v[i] = (vector_y[i] - vector_c[i]*vector_v[i+1])
       / u[i];
```

Listing 7: Uzupełnienie macierzy L i macierzy U zgodnie z wzorami

Rozwiązanie uzyskujemy dwuetapowo. Najpierw rozwiązujemy układ $Ly = b$:

$$y_1 = b_1$$

$$y_i = b_i - l_i \cdot y_{i1}$$

Gdzie: b_i są elementami wektora wyrazów wolnych. Następnie rozwiązujemy drugi układ $Uv = y$:

$$v_n = \frac{y_n}{u_n}, \text{ n to ostatni element}$$

$$v_i = (y_i - c_i \cdot v_{i+1})/u_i, \qquad i = n-1, n-2, ..., 1$$

4. Wyniki

Po uruchomieniu programu wyexportowane dane poddałem obróbce w języku Python w celu uzyskania informacji o tym jak bardzo wynik otrzymany metodą numeryczną odbiega od rzeczywistego. Wykres poniżej potwierdza że zdanie jest bardzo dobrze uwarunkowane a otrzymane wartości są dokladne gdyż wykresy obu funckji prawie całkowicie się pokrywają.

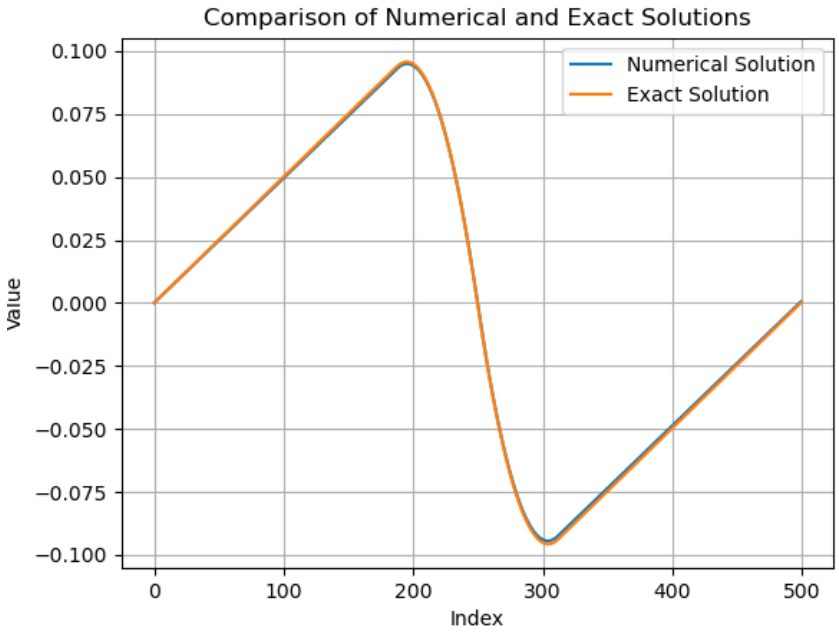

Figure 1: Prównanie wyników podejścia numerycznego z wynikiem dokładnym

Warto zauważyć, że jest to wynik dla N=500. Wraz ze wzrostem N wynik staje się jeszcze dokładniejszy, natomiast po dziesięciokrotnym zminejszeniu wykresy zaczynają sie różnić zwłaszcza w okolicach extemów.

5. Wnioski

Zaimplementowana metoda rozwiązania równania Poissona za pomocą rozkładu LU dla macierzy trójdiagonalnej jest efektywnym sposobem rozwiązania tego problemu numerycznego. Dzięki temu rozwiązaniu możemy wyznaczyć wartości potencjału elektrycznego dla danego rozkładu ładunku elektrycznego. Warto zauważyć, że efektywność tego algorytmu może być kluczowa, szczególnie w przypadku rozwiązywania równań.

- Metoda ta nie tylko jest dobrze uwarunkowana, ale także ma bardzo nieską złożoność pamięciową i obliczeniową:

- Nakład obliczeń: $M = 2n - 2, D = n - 1 \implies O(n)$

- Liczba zajętych komórek: $P = 3n - 2 \implies O(n)$

- Na te szczególne właściwości wpływa fakt że przechowujemy wszystkie ważne informacje tylko w 3 wektorach po których iterujemy.
- Stosując tą metodę nie musimy sie martwić o szczególne sposoby optymalizacji tj. zrównoleglanie spamiętywanie itd. ponieważ nawet dla bardzo dużego układu wykona się bardzo niewiele operacji.
- Można wykonać rozkład LU macierzy T, macierze te mają postać dwudiagonalną
- jeśli macierz jest dominująca kolumnowo to rozkład T=LU jest równoważny rozkładowi z częściowym wyborem elementu podstawowego (niezawodność metody)