Kocjan Nikodem

05.03.2024

Metody numeryczne ćwiczenie 2

Rozkład LU macierzy trójdiagonalnej - rozwiązanie równania $Poisona$ w jednym wymiarze

1. Cel ćwiczenia

Celem ćwiczenia było napisanie programu, który rozwiąże równanie $Poisona$, wykorzystując rozkład macierzy LU.

2. Wstęp teoretyczny

Równanie $Poisona$ jest podstawowym narzędziem w fizyce teoretycznej i inżynierii, pozwalającym na opis rozkładów pola potencjału w odpowiedzi na określony rozkład ładunków lub mas. W kontekście jednowymiarowym, równanie to redukuje się do formy, która pozwala na zastosowanie metody dyskretyzacji w celu przekształcenia problemu ciągłego na problem algebraiczny, który można rozwiązać numerycznie. Wzór $Poisona$ jednowymiarowo:

$$\nabla^2 V(x) = -\rho(x) \qquad (1)$$

$V(x)$ - potencjał w punkcie x

$\rho(x)$ - gęstość źródła

Rozpatrujemy $x \epsilon [-Xb, Xb]$ z warunkiem brzegowym $V(-Xb) = V(Xb) = 0$ dla rozkładu gęstości

$$0 \,, \; x\,\varepsilon\,[-Xb, \; -Xa)$$

$$+1 \,, \; x\,\varepsilon\,[-Xa, \; 0)$$

$$0 \,, \; x = 0 \qquad (2)$$

$$-1 \,, \; x\,\varepsilon\,(0, \; Xa]$$

$$0 \,, \; x\,\varepsilon\,(Xa, \; Xb]$$

Wprowadzamy siatkę z węzłami gdzie:

$$x_i = -Xb + h \cdot (i-1), \; i = 1, \; 2 \; \dots \,, N \qquad (3)$$

$N -$ liczba węzłów

$h -$ Odległość między węzłami:

$$h = \frac{2Xb}{N-1} \qquad (4)$$

Drugą pochodną w równaniu (1) zastępujemy ilorazem różnicowym zdefiniowanym na siatce:

$$\nabla^2 V = \frac{d^2 V}{dx^2} = \frac{V_{i-1} - 2V_i + V_{i+1}}{h^2} = -\rho_i \qquad (5)$$

Równanie (4) generuje układ równań w postaci

$$A \cdot V = P \tag{6}$$

Gdzie A:

$$\begin{matrix} d_1 & c_1 & 0 & 0 & 0 & 0 \\ a_2 & d_2 & c_2 & 0 & 0 & 0 \\ 0 & a_3 & d_3 & c_3 & 0 & 0 \\ & & \dots & \dots & \dots & \\ 0 & 0 & 0 & a_{n-1} & d_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & a_n & d_n \end{matrix}$$

$V -$ Potencjały od $i = 1$ do $i = N$

$P -$ Gęstości źródła z znakiem “– “

Wartości $a,\ c,\ d$ otrzymujemy z wzorów:

$$a = \frac{1}{h^2} \tag{7}$$

$$c = \frac{1}{h^2} \tag{8}$$

$$d = \frac{-2}{h^2} \tag{9}$$

Kluczowym elementem analizy będzie zastosowanie rozkładu LU macierzy. Polega ona na wyznaczeniu macierzy L oraz U , które połączone są relacją $A = L \cdot U$

I) Wyznaczenie macierzy L

Jest to macierz dolno-trójkątna, która na swojej przekątnej zawiera same 1. Za pomocą przekształceń macierzy współczynników A , podobnych do metody $Gaussa$ otrzymujemy macierz

$$\mathrm{L}=\begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ l_{21} & 1 & 0 & \dots & 0 \\ l_{31} & l_{32} & 1 & 0 & 0 \\ \dots & \dots & \dots & 1 & 0 \\ l_{n1} & l_{n2} & l_{n3} & \dots & 1 \end{bmatrix}$$

II) Wyznaczenie macierzy U

Jest to macierz górno - trójkątna. Otrzymujemy ją również za pomocą przekształceń macierzy współczynników A

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ 0 & u_{22} & u_{23} & \dots & u_{2n} \\ 0 & 0 & u_{33} & \dots & u_{nn} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & u_{nn} \end{bmatrix}$$

III) Rozwiązanie układu równań

Dysponując macierzami L i U , możemy rozwiązać układ równań

$$A\vec{x} = \vec{b}$$

$$LU\vec{x} = \vec{b}$$

poprzez rozwiązanie 2 układów równań:

$$L\vec{y} = \vec{b}$$

$$U\vec{x} = \vec{y}$$

Rozwiązanie każdego z równań wiąże się z nakładem obliczeń jak dla układu z macierzą trójkątną $\sim 1n^2$

Rozkład LU to nakład rzędu $\sim \frac{n^3}{2}$

3. Deklaracja zmiennych oraz implementacja bibliotek

Równanie $Poisona$ rozwiązywałem w języku C++, korzystając z $Visual\ Studio\ Code$. Nie implementowałem żadnych dodatkowych bibliotek, oprócz standardowej $< iostrem >$

Deklaracja stałych, podanych w poleceniu zadania:

```
double Xb = 2;      // Zakres przedziału gęstości Xa
double Xa = 0.5;    // Zakres przedziału gęstości Xb
int N = 500;        // Ilość węzłów przyjęta w zadaniu
```

Deklaracja potrzebnych tablic:

```
double tab_a[N];     // Tablica wartości a
double tab_c[N];     // Tablica wartości c
double tab_d[N];     // Tablica wartości d
double tab_p[N];     // Tablica wartości gęstości źródła
double tab_x[N];     // Tablica wartości x ( dyskretyzacji )
double L[N];         // Tablica wartości, która zastąpi macierz L
double U[N];         // Tablica wartości, która zastąpi macierz U
double Y[N];         // Tablica rozwiązań układu Ly = b
double V[N];         // Tablica rozwiązań ( potencjały w punktach x)
```

4. Opis działania

I) Obliczenie odległości między węzłami

Zaczynamy od obliczenia odległości między węzłami h za pomocą wzoru (3). W naszym przypadku wynosi ono 0.00801603.

```
double h = (2 * Xb) / (N - 1);
```

```
0.00801603
```

II) Obliczenie wartości tablicy tab_x za pomocą wzoru (2).

```
for(int i = 0; i < N; i++){
    tab_x[i] = -Xb + h *i ;
}
```

W przypadku działania na kodzie, zamiast $i-1$ używamy i, co jest spowodowane indeksowaniem tablic od 0. W pozostałej części sprawozdania będę pomijał ten komentarz, poniważ jest to trywialne.

III) Inicjujemy wartości graniczne tablic $a,\ d,\ c,\ p$

```
tab_a[0] = 0;
tab_d[0] = 1;
tab_c[0] = 0;
tab_p[0] = 0;

tab_d[N-1] = 1;
tab_a[N-1] = 0;
tab_c[N-1] = 0;
tab_p[N-1] = 0;
```

IV) Uzupełniamy tablice a, d, c, p.

```
for(int i = 1; i < N-1; i++){
    if(tab_x[i] < -Xa){
        tab_p[i] = 0;
    } else if(tab_x[i] >= -Xa && tab_x[i] < 0){
        tab_p[i] = 1;
    }else if(tab_x[i] == 0) {
        tab_p[i] = 0;
    }else if(tab_x[i] > 0 && tab_x[i] <= Xa){
        tab_p[i] = -1;
    } else {
        tab_p[i] = 0;
    }

    tab_d[i] = -2 / (h * h);
    tab_a[i] = 1 / (h * h);
    tab_c[i] = tab_a[i];
}
```

p — *układ równań* (2)

a — *wzór* (7)

c — *wzór* (8)

d — *wzór* (9)

V) Obliczamy wartości tablic L oraz U za pomocą wzorów

```
L[0] = 0;
U[0] = tab_d[0];

for(int i = 1; i < N; i++){
    L[i] = tab_a[i] / U[i-1];
    U[i] = tab_d[i] - L[i] * tab_c[i-1];
};
```

$$u_1 = d_1$$

$$l_i = \frac{a_i}{u_{i-1}},\ i = 2,\ 3\ \dots\ N$$

$$u_i = d_i - l_i \cdot c_{i-1},\ i = 2,3\ \dots\ N$$

VI) Obliczamy wartości rozwiązań układu $L \cdot y = b$

```
Y[0] = -tab_p[0];

for(int i = 1; i < N; i++){
    Y[i] = -tab_p[i] - L[i] * Y[i-1];
}
```

$$y_1 = b_1$$

$$y_i = b_i - l_i \cdot y_{i-1}$$

W moim kodzie nie deklarowałem dodatkowej tablicy B. Zamiast tego używałem -tab_p, ponieważ wynosi ona tyle samo.

VII) Wpisanie rezultatów do tablicy wyników

```
V[N-1] = Y[N-1] / U[N-1];

for(int i = N-2; i >= 0; i--){
    V[i] = (Y[i] - tab_c[i] * V[i+1]) / U[i];
}
```

$$v_n = \frac{y_n}{u_n}$$
$$v_i = \frac{y_i - c_i \cdot v_{i+1}}{u_i}$$

5. Analiza wyników

Wyniki z tablicy V będziemy porównywać z rozwiązaniami dokładnymi, obliczonymi z wzorów:

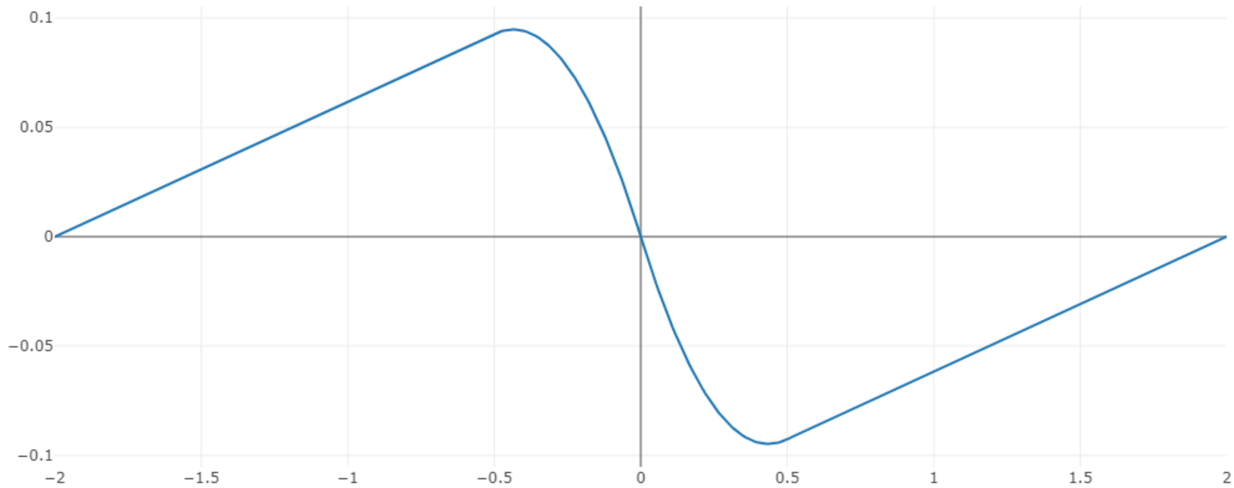
$$\frac{x}{16} + \frac{1}{8}, \; x \epsilon [-Xb, \; -Xa]$$

$$-\frac{x^2}{2} - \frac{7}{16}x, \; x \epsilon [-Xa, \; 0]$$

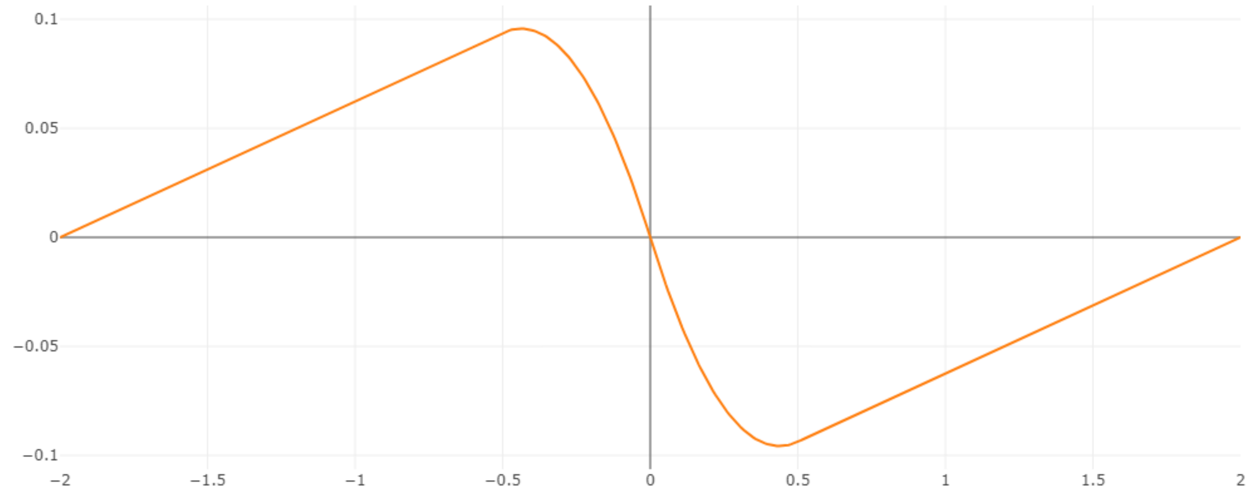
$$\frac{x^2}{2} - \frac{7}{16}x, \; x \epsilon [0, \; Xa]$$

$$\frac{x}{16} - \frac{1}{8}x, \; x \epsilon [Xa, \; Xb]$$

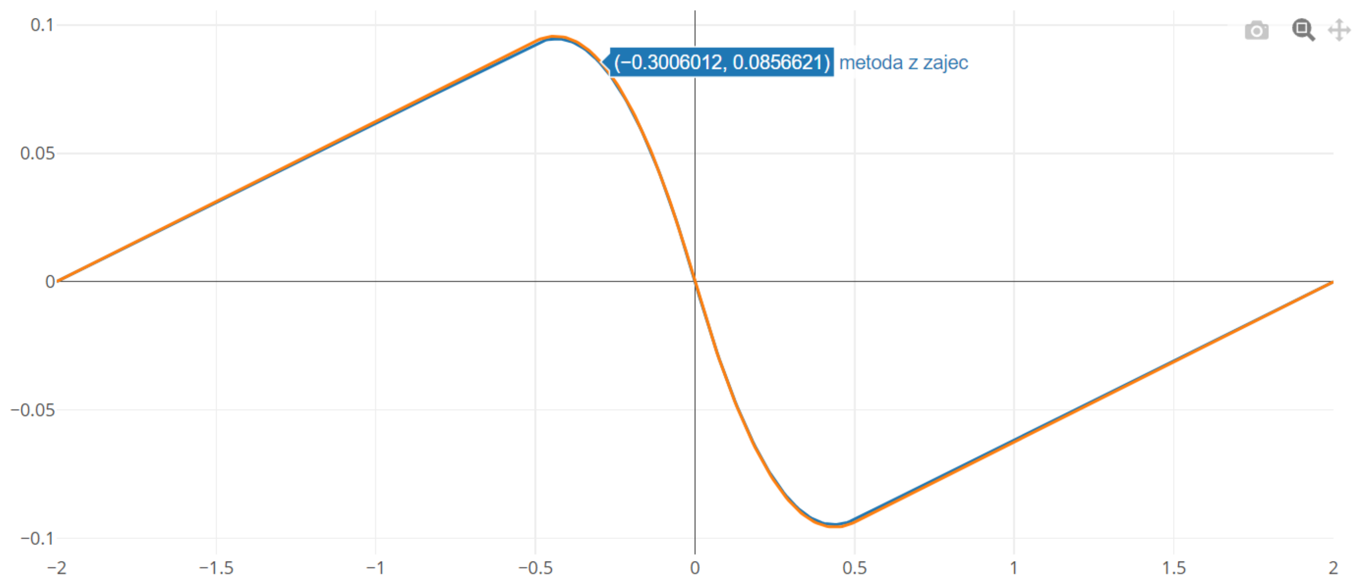
Wykres wyników dla tablicy V, obliczonej za pomocą rozkładu LU:



Wykres wyników dla danych obliczonych z rozwiązania dokładnego:



Dane nałożone na jeden wykres:



Jak widzimy na ostatnim wykresie, dane nachodzą na siebie tak bardzo, że ledwo można rozróżnić niebieską od pomarańczowej linii.

Wykresy wygenerowane zostały za pomocą programu typu open source $plotly$ w języku $java\ script$.

6. Wnioski

Program poprawnie rozwiązuje równanie $Poisona$ w formie jednowymiarowej. Użycie rozkładu macierzy LU usprawnia działanie programu, oraz pomaga z dużą dokładnością obliczyć potencjał V . Porównanie wyników, otrzymanych poprzez rozwiązanie zagadnienia w języku $C++$, z rozwiązaniami analitycznymi pokazało wysoką dokładność metody, co zostało zilustrowane na wykresach, gdzie dane numeryczne i analityczne prawie się pokrywały. Ogromnym plusem programu oraz metody jest to, że wykorzystujemy tylko i wyłącznie bibliotekę $iostream$ do rozwiązania całego równania. Oczywiście został też użyty program $plotly$, lecz służył on tylko do wizualizacji i porównania wyników.