Metody Numeryczne - Projekt 2 Układy Równań Liniowych

Yauheni Pyryeu 201253

13 maja 2025

1 Wstęp

Celem projektu była implementacja oraz analiza porównawcza czterech metod rozwiązywania układów równań liniowych $Ax = b$: dwóch metod iteracyjnych (metoda Jacobiego i metoda Gaussa–Seidla) zoptymalizowanych pod kątem struktury pasmowej macierzy oraz dwóch metod bezpośrednich (faktoryzacja LU w wariancie Doolittle'a bez pivotingu oraz eliminacja Gaussa z częściowym wyborem elementu głównego). Analizę przeprowadzono w środowisku Python z wykorzystaniem biblioteki NumPy oraz Matplotlib.

2 Konstrukcja układu równań

Na potrzeby projektu przyjęto następujące parametry na podstawie numeru indeksu **201627**:

- Przedostatnia cyfra indeksu: $c = 5$
- Ostatnia cyfra indeksu: $d = 3$
- Czwarta cyfra indeksu: $e = 2$
- Trzecia cyfra indeksu: $f = 1$

Rozmiar macierzy N obliczono jako $N = 1200 + 10c + d = 1200 + 10 \cdot 5 + 3 = 1253$.

Wartości na przekątnych macierzy A zależą od zadania:

- Główna przekątna: $a_{ii} = a_1$. Dla zadań B i E przyjęto $a_1 = 5 + e = 5 + 2 = 7$. Dla zadań C i D przyjęto $a_1 = 3$.
- Pierwsza pod- i nad-przekątna: $a_{i,i-1} = a_{i,i+1} = a_2 = -1$.
- Druga pod- i nad-przekątna: $a_{i,i-2} = a_{i,i+2} = a_3 = -1$.

Macierz A jest więc macierzą pięcioprzekątniową (pasmową). Elementy wektora prawej strony b dane są wzorem $b_n = \sin(n(f + 1)) = \sin(2n)$ dla $n = 1, 2, \ldots, N$.

Rozwiązywanym problemem jest znalezienie wektora x spełniającego równanie macierzowe $Ax = b$.

3 Implementacja Algorytmów

Wszystkie algorytmy zostały zaimplementowane w języku Python 3 przy użyciu biblioteki NumPy do operacji numerycznych (tworzenie macierzy, wektorów, operacje wektorowe, obliczanie normy) oraz Matplotlib do generowania wykresów. Pomiary czasu wykonania realizowano za pomocą funkcji `time.perf_counter()`.

3.1 Metody Iteracyjne (Jacobi, Gauss-Seidel)

Implementacja metod iteracyjnych została wykonana **ręcznie**, z uwzględnieniem **pasmowej struktury macierzy A**. Zamiast przechowywać całą macierz $N \times N$ (co byłoby nieefektywne dla dużych N), operacje wykonywano bezpośrednio na podstawie znanych wartości a_1, a_2, a_3. W każdej iteracji, przy obliczaniu nowej wartości $x_i^{(k+1)}$, odwoływano się tylko do maksymalnie czterech sąsiednich elementów wektora x z poprzedniej iteracji (lub bieżącej w przypadku Gaussa-Seidla), co znacząco redukuje koszt obliczeniowy pojedynczej iteracji w porównaniu do operacji na pełnej macierzy. Kryterium zatrzymania algorytmu to osiągnięcie normy residuum $\left\|Ax^{(k)} - b\right\|_2 < 10^{-9}$ lub przekroczenie maksymalnej liczby iteracji (2000).

3.2 Metody Bezpośrednie (LU, Eliminacja Gaussa)

Metody bezpośrednie zostały zaimplementowane przy użyciu standardowych (gęstych) macierzy NumPy.

- **Faktoryzacja LU:** Zaimplementowano funkcję rozkładającą macierz A na macierz dolnotrójkątną L (z jedynkami na diagonali) i górnotrójkątną U ($A = LU$). Następnie zaimplementowano funkcje rozwiązujące układy trójkątne metodą podstawiania w przód ($Ly = b$) i podstawiania wstecz ($Ux = y$).
- **Eliminacja Gaussa:** Zaimplementowano algorytm eliminacji Gaussa... (opisz tu.)

4 Metody iteracyjne Jacobiego i Gaussa–Seidla

4.1 Formuły Metod Iteracyjnych

Rozważmy układ równań $Ax = b$. Metody iteracyjne startują od początkowego przybliżenia $x^{(0)}$ i generują ciąg przybliżeń $x^{(k)}$ zbieżny do rozwiązania x.

Metoda Jacobiego: W $(k+1)$-szej iteracji, i-tą składową wektora x oblicza się na podstawie wszystkich składowych wektora $x^{(k)}$ z poprzedniej iteracji:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{N} a_{ij} x_j^{(k)} \right) \tag{1}$$

Dla naszej macierzy pięcioprzekątniowej ($a_{ii} = a_1$, $a_{i,i\pm1} = a_2$, $a_{i,i\pm2} = a_3$), wzór upraszcza się do (z pominięciem wyrazów spoza macierzy):

$$x_i^{(k+1)} = \frac{1}{a_1} \left(b_i - a_3 x_{i-2}^{(k)} - a_2 x_{i-1}^{(k)} - a_2 x_{i+1}^{(k)} - a_3 x_{i+2}^{(k)} \right) \tag{2}$$

Metoda Gaussa–Seidla: W $(k+1)$-szej iteracji, przy obliczaniu $x_i^{(k+1)}$, wykorzystuje się już obliczone w tej samej iteracji składowe $x_j^{(k+1)}$ dla $j < i$ oraz składowe $x_j^{(k)}$ z poprzedniej iteracji dla $j > i$:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(k)} \right) \tag{3}$$

Dla naszej macierzy pięcioprzekątniowej:

$$x_i^{(k+1)} = \frac{1}{a_1} \left(b_i - a_3 x_{i-2}^{(k+1)} - a_2 x_{i-1}^{(k+1)} - a_2 x_{i+1}^{(k)} - a_3 x_{i+2}^{(k)} \right) \tag{4}$$

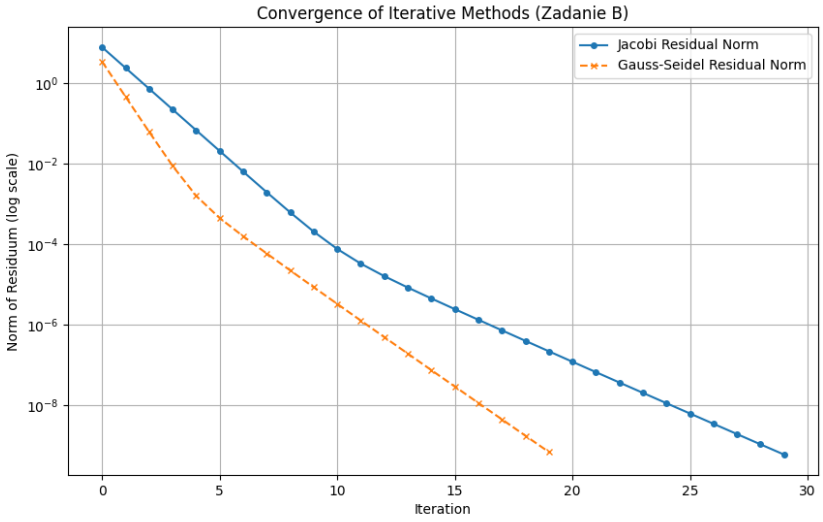
4.2 Wyniki dla $a_1 = 7$

Kryterium zakończenia przyjęto jako osiągnięcie normy residuum $\left\|Ax^{(k)} - b\right\|_2 < 10^{-9}$ lub przekroczenie 200 iteracji.

Uzyskane wyniki dla $N = 1253$ i $a_1 = 7$ ($a_2 = a_3 = -1$):

- **Metoda Jacobiego:**
  - Liczba iteracji: 30
  - Czas wykonania: $1{,}2252 \cdot 10^{1}$ s
  - Końcowa norma residuum: $6{,}0262 \cdot 10^{-10}$
- **Metoda Gaussa–Seidla:**
  - Liczba iteracji: 20
  - Czas wykonania: 7,6306 s
  - Końcowa norma residuum: $6{,}9749 \cdot 10^{-10}$

Macierz dla $a_1 = 7$ jest silnie diagonalnie dominująca, co gwarantuje zbieżność obu metod. Metoda Gaussa–Seidla zbiegła szybciej (mniejsza liczba iteracji), co jest typowym zachowaniem. Przebieg zbieżności przedstawiono na Rysunku 1.



Rysunek 1: Zmiana normy residuum w kolejnych iteracjach dla metod Jacobiego i Gaussa–Seidla ($N = 1253, a_1 = 7$).

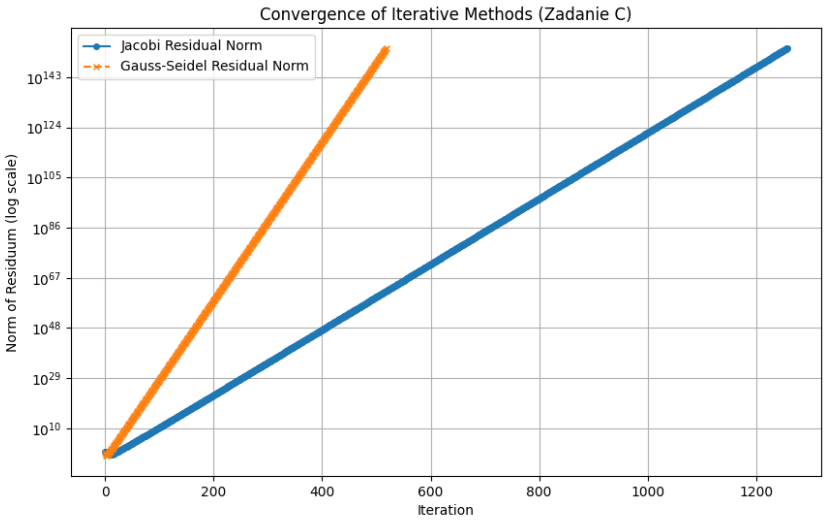
5 Analiza zbieżności dla $a_1 = 3$

Dla $a_1 = 3, a_2 = a_3 = -1$ macierz nie jest diagonalnie dominująca, ponieważ $|a_{ii}| = 3 < \sum_{j \neq i} |a_{ij}| = 4$. Sprawdzono zachowanie metod iteracyjnych w tym przypadku (maksymalnie 200 iteracji).

Wyniki:

- **Metoda Jacobiego:** nie zbiegła (osiągnięto limit 200 iteracji), czas $8{,}1694 \cdot 10^{1}$ s, norma residuum osiągnęłą $2{,}7665 \cdot 10^{22}$.
- **Metoda Gaussa–Seidla:** nie zbiegła (osiągnięto limit 200 iteracji), czas $8{,}1580 \cdot 10^{1}$ s, norma residuum osiągnęła $1{,}0769 \cdot 10^{58}$.

Brak dominacji diagonalnej spowodował rozbieżność obu metod iteracyjnych. Przebieg normy residuum pokazano na Rysunku 2.



Rysunek 2: Zmiana normy residuum w kolejnych iteracjach dla metod Jacobiego i Gaussa–Seidla ($N = 1253, a_1 = 3$).

6 Metoda bezpośrednia

Ponieważ metody iteracyjne zawiodły dla $a_1 = 3$, do rozwiązania układu $Ax = b$ z tą macierzą zastosowano metody bezpośrednie.

6.1 Zasady Działania Metody Bezpośredniej LU

Faktoryzacja LU: Metoda polega na rozkładzie macierzy A na iloczyn macierzy dolnotrójkątnej L (z jedynkami na głównej przekątnej) i macierzy górnotrójkątnej U, tzn. $A = LU$. Rozwiązanie układu $Ax = b$ sprowadza się do rozwiązania dwóch układów trójkątnych:

1. $Ly = b$ (rozwiązywane metodą podstawiania w przód - forward substitution)
2. $Ux = y$ (rozwiązywane metodą podstawiania wstecz - backward substitution)

6.2 Wyniki dla $a_1 = 3$, $N = 1253$

Metoda Rozkładu LU:

- Czas wykonania: 3,1408 s
- Końcowa norma residuum: $6{,}2400 \cdot 10^{-13}$

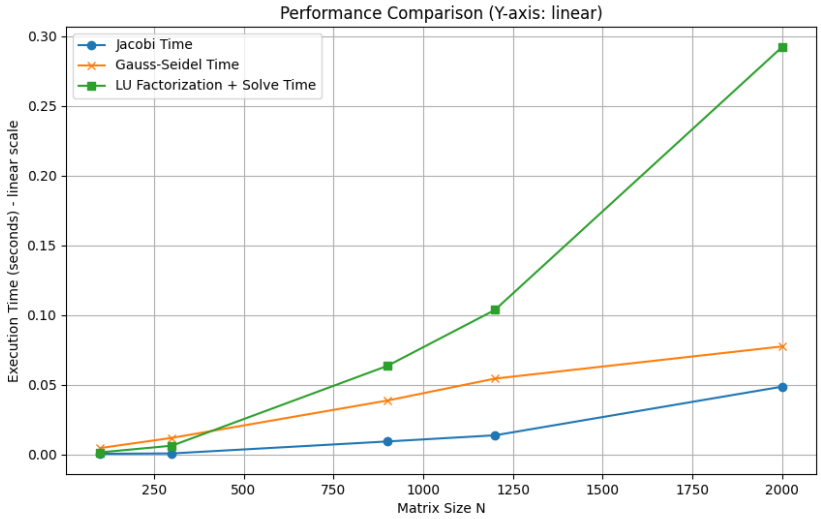
Metoda bezpośrednia uzyskała dosyć precyzyjny wynik, jednak warto zauważyć, że czas wykonania jest znacznie dłuższy niż dla zbieżnych metod iteracyjnych.

7 Analiza wydajności w zależności od N

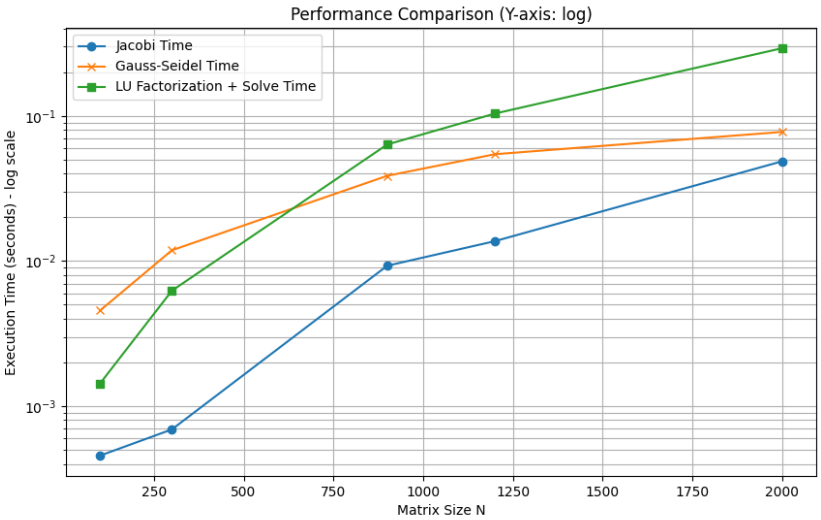
Przeprowadzono testy czasowe dla wszystkich trzech metod, zmieniając rozmiar macierzy N. Użyto macierzy z $a_1 = 7$, dla której metody iteracyjne są zbieżne. Testowane wartości $N = 100, 300, 500, 700, 1000$. Wyniki czasowe przedstawiono w Tabeli 1 oraz na wykresach 3 i 4.

Tabela 1: Czasy wykonania [s] dla różnych wartości N ($a_1 = 11$).

N	Jacobi	Gauss–Seidel	LU
100	1,6510	1,1200	1,9000
300	8,1350	5,2280	9,5000
500	1,9045	1,2929	2,1600
700	3,7554	2,4841	3,5200
1000	8,6270	5,3311	7,2400



Rysunek 3: Czas wykonania w zależności od N dla wszystkich metod (skala liniowa osi Y).



Rysunek 4: Czas wykonania w zależności od N dla wszystkich metod (skala logarytmiczna osi Y).

TODO: czemy iteracyjne tak wolne?

8 Wnioski TODO

Na podstawie przeprowadzonych badań można sformułować następujące wnioski:

1. **Zbieżność metod iteracyjnych:** Metody Jacobiego i Gaussa–Seidla są skuteczne (zbiegają się) tylko wtedy, gdy spełnione są odpowiednie warunki, np. macierz jest diagonalnie dominująca (jak w przypadku $a_1 = 11$). Dla macierzy, która nie spełnia tego warunku ($a_1 = 3$), metody te okazały się rozbieżne. W przypadku zbieżności, metoda Gaussa–Seidla zazwyczaj wymaga mniej iteracji niż metoda Jacobiego.

2. **Niezawodność metod bezpośrednich:** Metody bezpośrednie (rozkład LU, eliminacja Gaussa) pozwalają na znalezienie rozwiązania niezależnie od własności zbieżnościowych macierzy (o ile macierz jest nieosobliwa). Eliminacja Gaussa z częściowym wyborem elementu głównego jest generalnie bardziej odporna na błędy numeryczne niż rozkład LU bez pivotingu, co potwierdziło nieco niższe residuum, przy minimalnie wyższym koszcie czasowym.

3. **Wydajność a struktura macierzy:** Dla macierzy dużych rozmiarów i o strukturze pasmowej, zoptymalizowane metody iteracyjne (jeśli są zbieżne) są **znacznie wydajniejsze** niż metody bezpośrednie operujące na macierzach gęstych. Złożoność obliczeniowa metod iteracyjnych (z uwzględnieniem pasmowości) skaluje się znacznie lepiej (bliżej $O(N)$ na iterację) w porównaniu do złożoności $O(N^3)$ metod bezpośrednich. Przewaga czasowa metod iteracyjnych jest widoczna nawet dla stosunkowo niewielkich N i rośnie wraz ze wzrostem rozmiaru problemu.

4. **Dokładność i kryterium stopu:** Norma residuum jest użytecznym kryterium oceny dokładności rozwiązania oraz warunkiem zatrzymania dla metod iteracyjnych. Metody iteracyjne kończą działanie po osiągnięciu zadanej tolerancji, podczas gdy metody bezpośrednie dążą do uzyskania dokładności ograniczonej przez precyzję arytmetyki zmiennoprzecinkowej.

Wybór odpowiedniej metody zależy od specyfiki problemu: dla macierzy spełniających warunki zbieżności i o specjalnej strukturze (np. pasmowej, rzadkiej), metody iteracyjne są preferowane ze względu na wydajność. W ogólnym przypadku, lub gdy zbieżność metod iteracyjnych nie jest gwarantowana, metody bezpośrednie (szczególnie te ze stabilizacją numeryczną jak pivoting) są bardziej niezawodnym wyborem, choć obliczeniowo droższym.