Metody Numeryczne Projekt 2 - analiza metod rozwiązywania układów równań liniowych

Maksymilian Kempa **197259**

7 maja 2024

1 Wstęp

Celem projektu jest implementacja i analiza dwóch metod iteracyjnych (Jacobiego i Gaussa-Seidla) oraz jednej metody bezpośredniej (faktoryzacja LU) rozwiązywania układów równań liniowych. Wszystkie metody zostały wykorzystane do rozwiązania równania liniowego postaci $\mathbf{Ax} = \mathbf{b}$, gdzie $\mathbf{A}$ jest macierzą systemową (w zależności od problemu może ona reprezentować np. obwód elektroniczny, geometrię sali koncertowej, turbinę itp.), $\mathbf{b}$ jest wektorem pobudzenia (czyli reprezentacją np. fali dźwiękowej, impulsu elektrycznego itp.), natomiast x jest naszą niewiadomą i rozwiązaniem układu równań (czyli rozkładem pola elektromagnetycznego, natężeniem dźwięku itp.).

Do implementacji projektu wykorzystałem środowisko programistyczne PyCharm, a sam kod został napisany w języku Python.

2 Stworzenie układu równań

W celu przeprowadzenia analizy została utworzona macierz $\mathbf{A}$ o wymiarach 959×959, której główna diagonala została wypełniona wartościami 7, natomiast sąsiadujące z nią odpowiednio o 1 i 2 indeksy zostały wypełnione wartościami -1. Pozostałe komórki macierzy zawierają 0. Tak utworzona macierz prezentuje się następująco:

$$\begin{bmatrix} 7 & -1 & -1 & 0 & 0 & 0 & \dots & 0 \\ -1 & 7 & -1 & -1 & 0 & 0 & \dots & 0 \\ -1 & -1 & 7 & 7 & -1 & 0 & \dots & 0 \\ 0 & -1 & -1 & 7 & -1 & -1 & \dots & 0 \\ 0 & 0 & -1 & -1 & 7 & -1 & \dots & 0 \\ 0 & 0 & 0 & -1 & -1 & 7 & \dots & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 7 \end{bmatrix} \tag{1}$$

Jest to macierz pasmowa (ang. *banded matrix*), czyli taka, w której wszystkie elementy niezerowe znajdują się wzdłuż głównej przekątnej oraz pewnej liczby przekątnych sąsiednich. Poza tymi przekątnymi reszta elementów macierzy jest równa 0. Natomiast jako druga część układu równań został utworzony wektor

$\mathbf{b}$, którego n-ty element przyjmuje wartość sin(10n).

3 Opracowanie wykonanych zadań

3.1 Zadanie B

Celem zadania było porównanie działania metod iteracyjnych Jacobiego i Gaussa-Seidela w rozwiązywaniu równania macierzowego z użyciem macierzy A i wektora b, opisanych w poprzednim punkcie.

Zarówno metoda Jacobiego, jak i metoda Gaussa-Seidela polegają na przybliżaniu rozwiązania układu poprzez iteracyjne aktualizowanie wartości wektora wynikowego na podstawie poprzednich iteracji. W każdej iteracji jest także aktualizowana norma błędu rezydualnego, czyli wartość odchylenia wyliczonego rozwiązania od rozwiązania prawdziwego, ponieważ wektor wynikowy zawiera jedynie przybliżone rozwiązanie. Jest ona obliczana w następujący sposób dla k-tej iteracji:

$$res^(k) = Ax^(k) - b$$

Ponieważ osiągnięcie wartości rzeczywistej jest w praktyce nieosiągalne, wartość residuum stanowi granicę, po osiągnięciu której algorytm kończy swoje działanie i zwraca przybliżony wynik. Na potrzeby projektu przyjęto normę w wysokości 10^{-9}.

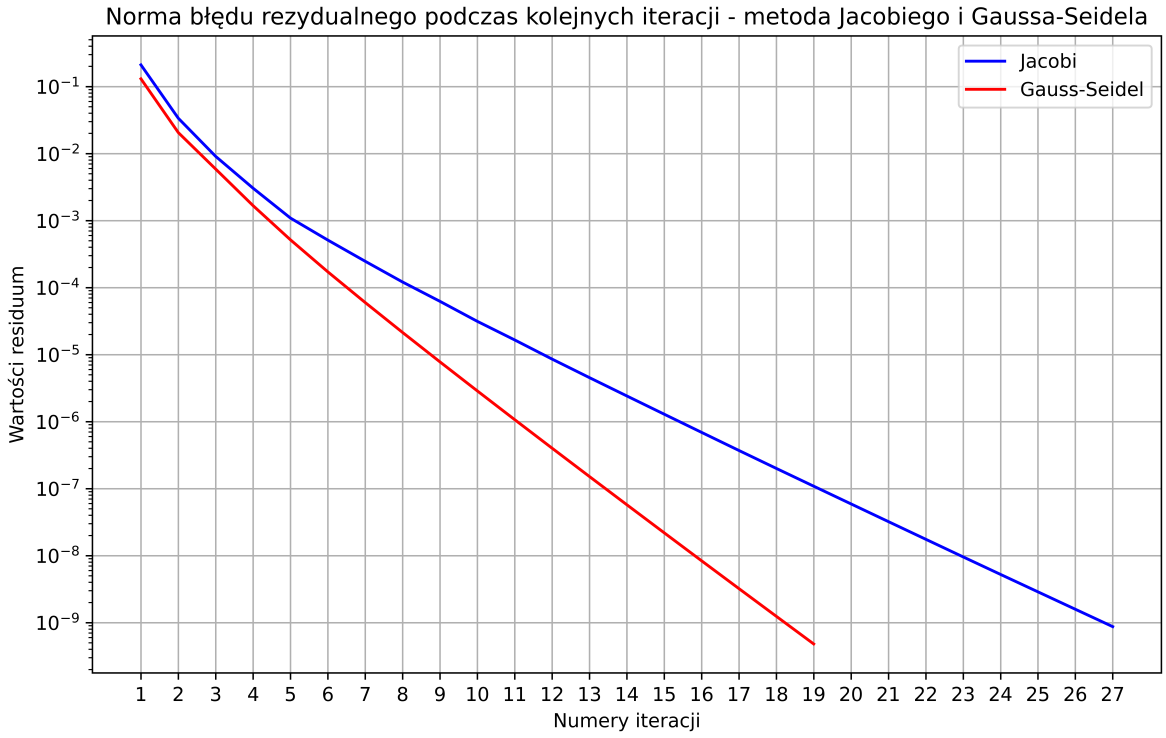
Pierwszą z analizowanych metod była metoda Jacobiego, której wyznaczenie i-tego elementu wektora wynikowego x wygląda następująco:

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

gdzie a i b są kolejnymi wartościami elementów macierzy $\mathbf{A}$ i wektora $\mathbf{b}$ o podanych indeksach i oraz j. Natomiast wzór na i-ty element w k-tej iteracji dla metody Gaussa-Seidela wygląda następująco:

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

Spójrzmy zatem, jak obie metody poradziły sobie z rozwiązywaniem zadanego układu równań.

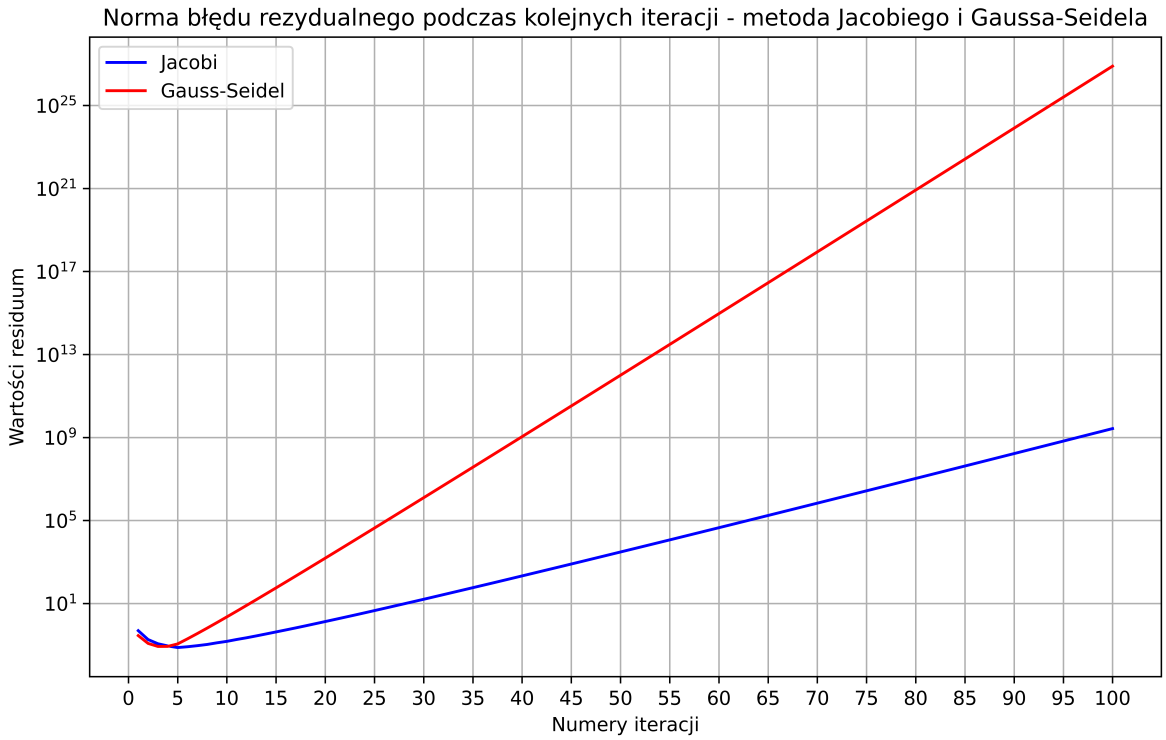


Rysunek 1: Wykres przedstawiający zmianę normy błędu rezydualnego podczas kolejnych iteracji wykonywania algorytmów Jacobiego i Gaussa-Seidela - zadanie B

Analizując powyższy wykres, możemy stwierdzić, że lepszy wynik uzyskała metoda Gaussa-Seidela. Udało jej się uzyskać dokładne przybliżenie już na przestrzeni 19 iteracji, podczas gdy metoda Jacobiego potrzebowała na to 27 iteracji. Różnicę widać także w czasie wykonywania (2,56 sekundy Gaussa-Seidela w stosunku do 3,58 sekundy Jacobiego), aczkolwiek w przypadku tych danych wejściowych obie metody szybko obliczyły wektor wynikowy i różnica czasowa nie ma aż takiego znaczenia. Jak widać na wykresie, norma residuum maleje z każdą kolejną iteracją, co oznacza, że wyczaczamy coraz bardziej dokładne rozwiązania. Jest to pożądane zjawisko, które oznacza, że metody iteracyjne zbiegają się.

3.2 Zadanie C

Cel tego zadania był analogiczny do poprzedniego, jednak tym razem została lekko zmodyfikowana macierz systemowa **A**: jej główna przekątna została wypełniona wartościami 3 zamiast 7. Resultaty zostały przedstawione na wykresie:



Rysunek 2: Wykres przedstawiający zmianę normy błędu rezydualnego podczas kolejnych iteracji wykonywania algorytmów Jacobiego i Gaussa-Seidela - zadanie C

Jak widać, tym razem norma błędu rezydualnego rosła z każdą iteracją, zamiast maleć. Po kilku początkowych iteracjach, na przestrzeni których wartość residuum się zmniejszała, norma zaczęła liniowo wzrastać, dochodząc do wartości rzędu 10^{25} w przypadku metody Gaussa-Seidela i 10^{9} dla metody Jacobiego. Zamiast przybliżać się do rzeczywistego rozwiązania, algorytm zaczął się od niego oddalać, aż zakończył swoją pracę po 100 iteracjach (taki został ustawiony limit, by program nie działał w nieskończonej pętli) w przypadku obu metod.

3.3 Zadanie D

Celem zadania D było sprawdzenie, jak z rozwiązywaniem układu równań z poprzedniego zadania radzi sobie metoda bezpośrednia. W tym celu zaimplementowana została metoda faktoryzacji LU, która polega na rozkładzie macierzy **A** na iloczyn dwóch macierzy trójkątnych: macierzy dolnej (**L**) i macierzy

górnej (**U**), takich, że $A = LU$. Po przeprowadzeniu takiej dekompozycji macierzowej, równanie $Ax = b$ sprowadza się do dwóch łatwiejszych do rozwiązania układów równań:

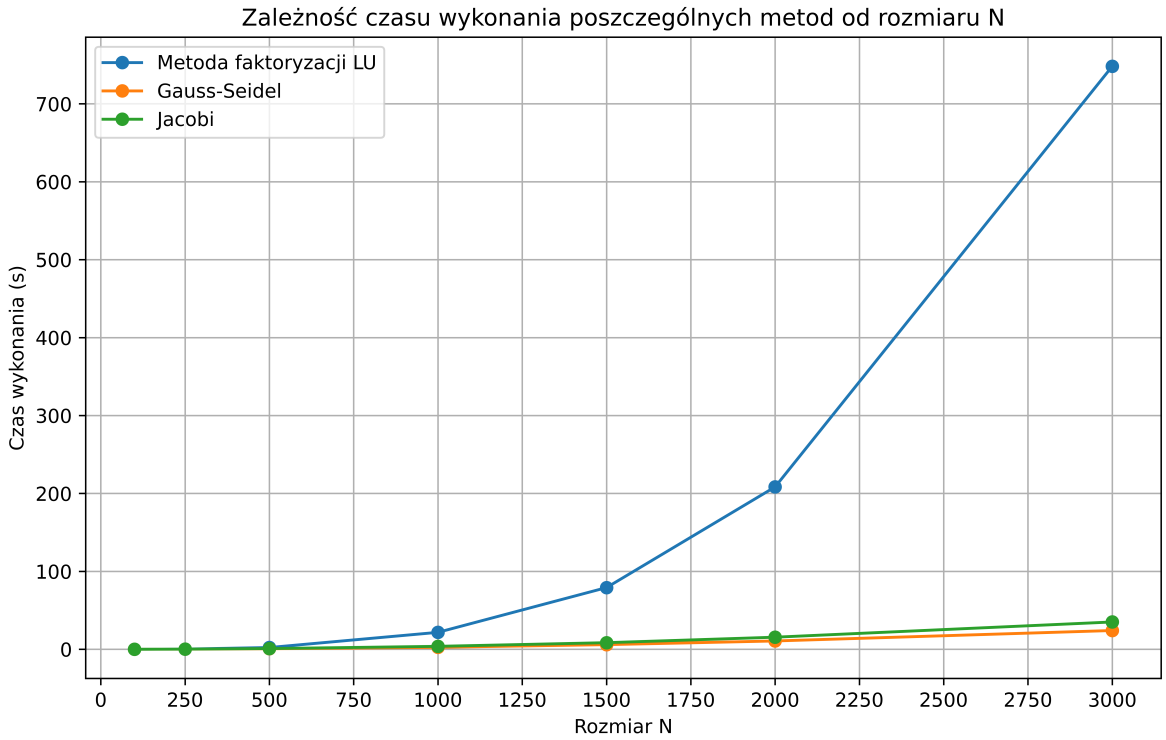
- Układ równań dolnotrójkątnych: $Ly = b$
- Układ równań górnotrójkątnych: $Ux = y$

Pierwszy układ można rozwiązać stosując algorytm podstawiania w przód (*forward substitution*), a drugi - stosując algorytm podstawiania w tył (*backward substitution*).

Zaimplementowana metoda została wykorzystana do rozwiązania układu równań z zadania C. W przeciwieństwie do metod iteracyjnych, wyznaczyła ona przybliżone rozwiązanie w czasie 19,20 s. Końcowa wartość normy błędu rezydualnego wyniosła $9,84 * 10^{-13}$. Jest to nawet niższy rząd błędu, niż poszukiwany w przypadku metod iteracyjnych, co oznacza dokładniejszą wartość rozwiązania.

3.4 Zadanie E

W tym zadaniu możemy zaobserwować zależność zależność czasu wyznaczania rozwiązania równania macierzowego od rozmiaru macierzy wejściowej dla trzech różnych metod użytych w poprzednich zadaniach. Do przeprowadzenia analizy została wykorzystana macierz o wartościach takich jak w zadaniu B (z główną diagonalą o wartościach 7). Rozmiary macierzy N wynoszą kolejno 100, 250, 500, 1000, 1500, 2000 i 3000. Wykres przedstawiający opisaną wyżej zależność wygląda następująco:



Rysunek 3: Wykres przedstawiający zależność czasu rozwiązywania układu równań od rozmiaru danych wejściowych N dla trzech różnych metod

O ile w przypadku mniejszych rozmiarów (100, 250) nie widać jeszcze znaczącej różnicy między metodą bezpośrednią a metodami iteracyjnymi, o tyle z każdym kolejnym N różnica ta znacząco wzrasta. Dla N = 3000 metoda faktoryzacji LU potrzebowała około 750 sekund, by wyznaczyć rozwiązanie. Na jej tle świetnie wypadają za to obie metody iteracyjne, które dla N = 3000 potrzebowały jedynie około 30 sekund. Linie metod iteracyjnych na wykresie prawie się pokrywają, choć minimalnie lepiej wypada metoda Gaussa-Seidela.

4 Wnioski

Analiza wykresów normy błędu rezydualnego oraz czasu wykonywania rozwiązywania układów równań liniowych dla różnych metod pokazała, że metody iteracyjne, takie jak metoda Jacobiego i metoda Gaussa-Seidela, są przeważnie szybsze niż metoda bezpośrednia faktoryzacji LU. Dzieje się tak dlatego, że metoda faktoryzacji LU wymaga wykonania operacji dekompozycji macierzy, co jest bardziej czasochłonne dla dużych macierzy. W szczególności, złożoność obliczeniowa metody faktoryzacji LU to $O(n^3)$, gdzie n to rozmiar macierzy wejściowych. Natomiast złożoności metod Jacobiego i Gaussa-Seidela to $O(n^2)$.

To stwierdzenie sprawdza się jednak tylko w przypadkach, gdy rozwiązanie się zbiega. Metody iteracyjne, takie jak metoda Jacobiego czy Gaussa-Seidela, mogą wymagać odpowiedniego dobrania parametrów. W przypadku zadania C na wyznaczenie rozwiązania wpłynął fakt, że macierz nie jest dominująca diagonalnie, przez co wyznaczenie dokładnego rozwiązania za pomocą tych metod stało się niemożliwe do zrealizowania. Macierz dominująca diagonalnie to taka, w której moduły elementów znajdujące się na głównej przekątnej są większe niż suma modułów pozostałych elementów z danych wierszy. W tym przypadku dobrze sprawdziła się natomiast metoda bezpośrednia faktoryzacji LU, która nie wymaga iteracyjnego poprawiania przybliżenia rozwiązania, lecz opiera się na rozwiązaniu dwóch prostych układów równań trójkątnych, co jest bardziej skuteczne w przypadku układów, dla których metody iteracyjne nie są zbieżne.

Analiza wykresów pokazała również, że metoda Gaussa-Seidela wypada trochę lepiej czasowo niż metoda Jacobiego. Dzieje się tak dlatego, że w metodzie Gaussa-Seidela, nowo obliczone wartości dla poszczególnych elementów wektora rozwiązań x_i są natychmiast wykorzystywane do dalszych obliczeń w tej samej iteracji. Ta natychmiastowa aktualizacja oznacza, że w trakcie pojedynczej iteracji algorytm ma dostęp do najbardziej aktualnych danych, co może przyspieszyć proces zbieżności.