Patryk Miszke 193249

Metody Numeryczne

Projekt 2 – Układy równań liniowych

Wstęp (Zadanie A)

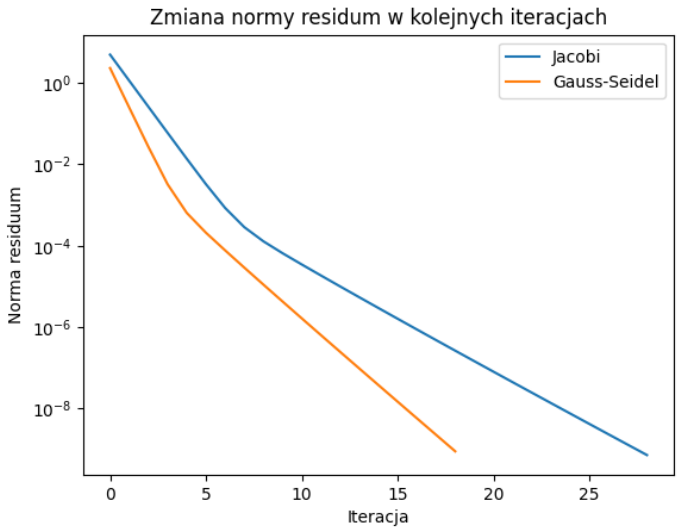
Pierwszy układ równań został policzony dla: a1 = 7, a2 = a3 = -1. Macierz A posiada rozmiary NxN, natomiast b stanowi wektor o długości N, gdzie N = 949. Jego kolejne elementy są liczone według wzoru b[n] = sin(n*4). A1 jest główną diagonalą, a2 stanowi dwie sąsiednie diagonale, a a3 dwie skrajne. Równanie ma postać Ax = b, gdzie x stanowi szukany wektor rozwiązań.

$$\mathbf{A} = \begin{bmatrix} a1 & a2 & a3 & 0 & 0 & 0 & 0 & \dots & 0 \\ a2 & a1 & a2 & a3 & 0 & 0 & 0 & \dots & 0 \\ a3 & a2 & a1 & a2 & a3 & 0 & 0 & \dots & 0 \\ 0 & a3 & a2 & a1 & a2 & a3 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & a3 & a2 & a1 \end{bmatrix}$$

Implementacja metody Jacobiego oraz Gaussa-Seidla (Zadanie B)

Metoda	Jacobi	Gauss-Seidel
Liczba iteracji	28	18
Czas operacji [s]	13,74	8,31

Metoda Jacobiego wymaga większej ilości iteracji, co za tym idzie ma większą złożoność czasową, niż metoda Gaussa-Seidla. Poniższy wykres przedstawia wartość normy residuum dla każdej iteracji. Dla metody Gaussa-Seidla ta wartość maleje szybciej, dzięki czemu algorytm wcześniej otrzymuje pożądany wynik z pewną tolerancją (10^{-9}).

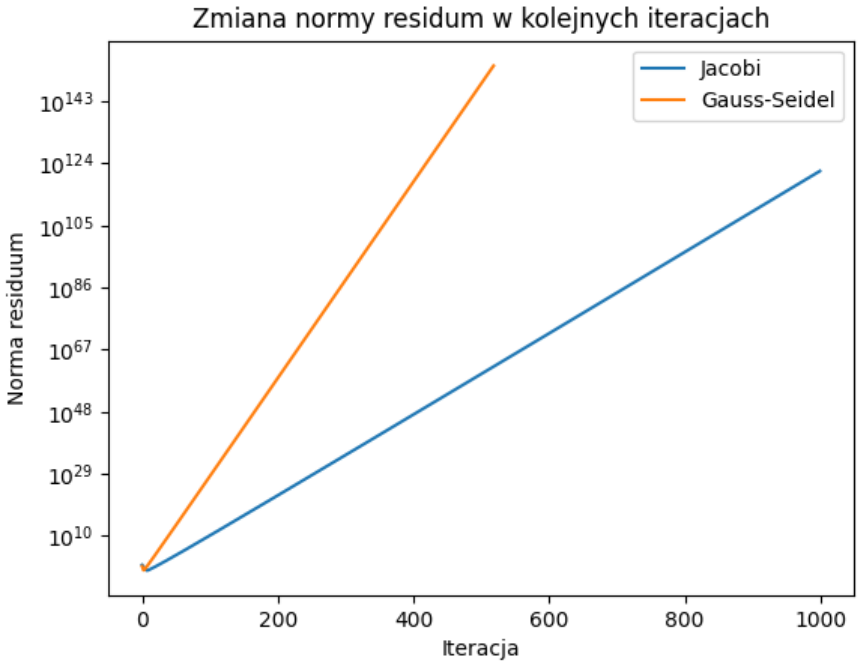


Drugi układ równań (Zadanie C)

Drugi układ równań został policzony dla: a1 = 3, a2 = a3 = -1. Macierz A, wektor b oraz rozmiar N są wyznaczone według formuły z zadania A. Zmieni się zatem jedynie główna diagonala (a1) w macierzy A.

Metoda	Jacobi	Gauss-Seidel
Liczba iteracji	1000	1000
Czas operacji [s]	438,97	438,11

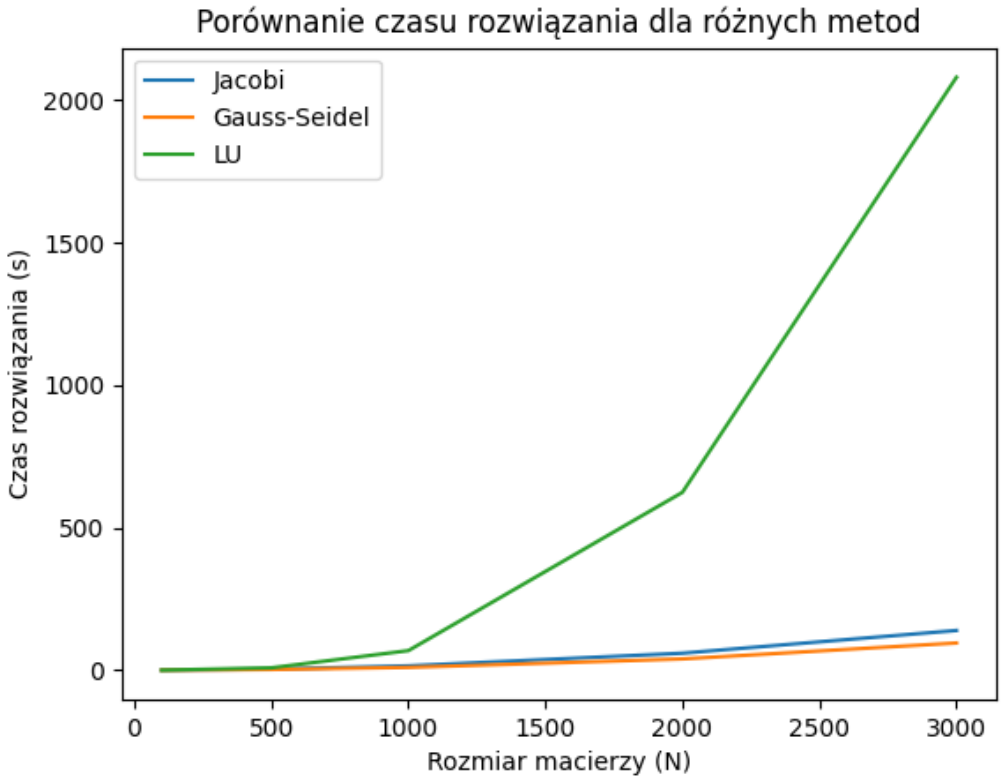
W obu metodach liczba iteracji wynosi 1000, co stanowi ustaloną maksymalną liczbę iteracji (dlatego czas operacji również jest niemalże identyczny). Algorytmy nie znalazły normy residuum mniejszej od założonej tolerancji (10^{-9}), więc wykonywały się, aż do granicy iteracyjnej. Co więcej, na poniższym wykresie prezentującym normy residuum w kolejnych iteracjach łatwo zauważyć, iż wartości te rosną zamiast maleć (rozbiegają się). W przypadku metody Gaussa-Seidla w tysięcznej iteracji jest rzędu ponad 10^{143}, co daje wynik nieprawidłowy - bardzo odległy od rzeczywistego, a także oczekiwanego.



Implementacja metody bezpośredniej (faktoryzacja LU, Zadanie D)

Metoda bezpośrednia została zastosowana do drugiego układu równań z zadania C. W tym przypadku norma residuum wynosi 6,65. Jest to wartość znacznie mniejsza, niż ta uzyskana przy zastosowania metod Jacobiego oraz Gaussa-Seidla. Co więcej, nie jest odległa od zera, a więc śmiało można powiedzieć, że metoda bezpośrednia w przypadku tego układu zbiega się i jest skuteczniejsza. Jej wadą jest jednak złożoność czasowa, która stanowczo przekracza złożoności poprzednich zastosowanych metod.

Wykres zależności czasu w zależności od liczby niewiadomych N (Zadanie E)



Podsumowanie (Zadanie F)

Metody Jacobiego i Gaussa-Seidla są skutecznymi narzędziami do rozwiązywania układów równań liniowych, szczególnie w przypadkach, gdy rozwiązanie można przybliżyć w stosunkowo krótkim czasie i przy niskim obciążeniu obliczeniowym. Są to metody iteracyjne, co oznacza, że działają poprzez stopniowe przybliżanie się do rozwiązania, aż do osiągnięcia określonej tolerancji.

Zaletą tych metod jest ich szybkość oraz prostota implementacji. W wielu przypadkach mogą one prowadzić do dokładnych lub przybliżonych wyników w czasie krótszym niż metody bezpośrednie, takie jak faktoryzacja LU.

Jednak, jak zauważono w projekcie, metody te mają swoje ograniczenia. Dla niektórych układów, szczególnie tych, które nie są diagonalnie dominujące lub gdzie warunki zbieżności nie są spełnione, te metody mogą nie zbiegać się do poprawnego rozwiązania. Przykłady z Zadania C pokazują, że w takich przypadkach iteracje mogą nawet rozbiegać się, prowadząc do błędnych wyników. Dlatego przy stosowaniu metod iteracyjnych ważne jest, aby upewnić się, że układ równań jest odpowiedni dla tej metody.

Metoda bezpośrednia, jak faktoryzacja LU, mimo większej złożoności obliczeniowej, jest bardziej niezawodna i może być używana tam, gdzie wymagane jest dokładne rozwiązanie lub gdzie metody iteracyjne zawodzą. Daje ona również dokładniejsze wyniki, choć kosztem większej ilości zasobów i czasu obliczeniowego.