MOwNiT
Laboratorium 3

Kacper Janda

1 Wady klasycznego algorytmu Gaussa

Rozwiązywanie układu równań liniowych podstawową metodą Gaussa zawodzi gdy na przekątnej macierzy A istnieją zera. Algorytm próbuje wtedy podzielić przez 0 co prowadzi do błędu. Algorytm ten nie uwzględnia postaci macierzy A - na przykład gdy jest ona macierzą diagonalną algorytm i tak wykonuje wszystkie operacje. Metoda ta posiada złożoność obliczeniową $O(n^3)$. Kolejnym problemem jest fakt powstawania błędów numerycznych spowodowanych dzieleniem przez liczby o małej wartości. Często ma to znaczny wpływ na końcowy wynik.

2 Optymalizacja algorytmu Gaussa

Aby zmniejszyć czas potrzebny rozwiązywania układu równań można zastosować rozkład LU macierzy A. Innym sposobem na poprawienie działania algorytmu jest zastosowanie wybierania elementów głównych. Pozwala to zredukować błędy numeryczne.

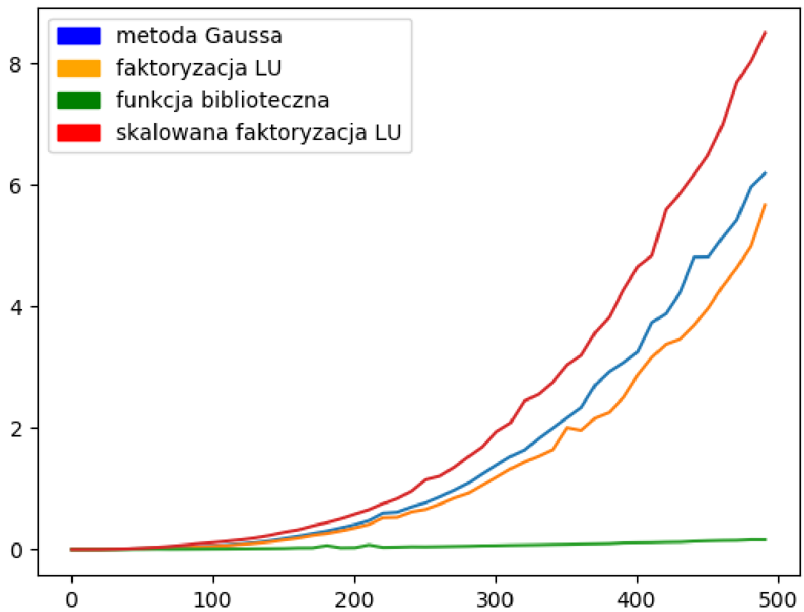
1. Przykład występowania błędów numerycznych

$$A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix} \; b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Metoda	Wynik
Metoda Gaussa	$\begin{bmatrix} 0.0 \\ 1.0 \end{bmatrix}$
Faktoryzacja LU	$\begin{bmatrix} 0.0 \\ 1.0 \end{bmatrix}$
Skalowana faktoryzacja LU	$\begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix}$
Funkcja biblioteczna	$\begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix}$

W metodzie używającej skalowanej faktoryzacji LU nie wystąpiły znaczące błędy nueryczne. W przypadku metody Gaussa oraz zwykłej faktoryzacji błędy spowodowały całkowitą zmianę wyniku.

3 Porównanie wydajności



Wykras pokazuje, że w celu uzyskania jak najlepszej wydajności należy korzystać z funkcji bibliotecznych (w tym przypadku została wykorzystana funkcja 'solve' z biblioteki 'numpy.linalg'). Zgodnie z przewidywaniami wykresy zaimplementowanych funcji przedstawiają złożoność $O(n^3)$.