Jakub Adamowicz

165761

Projekt 2 – metody Jacobiego, Gaussa-Seidela oraz faktoryzacji LU sprawozdanie

Wprowadzenie

Celem projektu jest implementacja następujących metod rozwiązywania układów równań liniowych:

- Metody iteracyjne:
  - Jacobiego
  - Gaussa – Seidela
- Metody bezpośrednie:
  - Faktoryzacja LU

oraz porównanie ich działania.

Dla dalszych rozważań ustalmy, że układ równań liniowych ma postać

$$Ax = b$$

gdzie A – macierz współczynników, b – wektor wyrazów wolnych, x – rozwiązanie układu, wektor.
W dalszych wzorach, przykładowo, symbol a_{ij} oznacza element macierzy A na pozycji (i,j), symbol b_i oznacza i-ty element wektora b.

Opis metod iteracyjnych

Metody iteracyjne mają za zadanie obliczać ciąg kolejnych przybliżeń rozwiązania z nadzieją, że będzie zbieżny do rozwiązania dokładnego. Powstaje zatem potrzeba wyznaczenia błędu dla rozwiązania przybliżonego; z pomocą przychodzi wektor residuum:

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

gdzie $x^{(k)}$ – przybliżenie rozwiązania otrzymane w k-tej iteracji metody iteracyjnej; jeżeli $x^{(k)}$ jest rozwiązaniem dokładnym, norma wektora residuum wynosi 0.

W praktyce wystarczy nam pewna z góry ustalona dokładność; zatem warunkiem stopu metod iteracyjnych jest osiągnięcie normy poniżej pewnej zadanej liczby.

Przejdźmy zatem do wzorów metod iteracyjnych użytych w niniejszym ćwiczeniu[1].

[1] W implementacji uwzględniono indeksowanie od 0; poza tym brak istotnych różnic.

- Metoda Jacobiego:

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

- Metoda Gaussa – Seidela:

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

Przewagą metody Gaussa-Seidela jest używanie już znanych elementów aktualnie obliczanego przybliżenia rozwiązania, nie zaś tylko elementów ostatniego przybliżenia.

Opis faktoryzacji LU

Metoda faktoryzacji LU jest wariantem metody Gaussa. Krótki jej opis przedstawia się następująco:

- Tworzymy macierze trójkątne – dolną L i górną U, takie, że LUx = b
- Tworzymy wektor pomocniczy: y = Ux
- Rozwiązujemy układ równań: Ly = b za pomocą podstawiania w przód
- Rozwiązujemy układ równań: Ux = y za pomocą podstawiania wstecz

Metoda faktoryzacji LU zwraca dokładne rozwiązanie (na komputerze może dojść do straty precyzji przy użyciu operacji zmiennoprzecinkowych), lecz potrzebuje więcej czasu na wykonanie; to jest $O(n^3)$ operacji, podczas analogicznie metody iteracyjne wymagają $O(n^2)$ operacji (o ile ciąg przybliżeń jest zbieżny).

Implementacja oraz testy

Na potrzeby projektu zaimplementowałem powyższe metody w języku C++. Zdecydowałem się na reprezentację macierzy poprzez osobną klasę; posiada ona tablicę dwuwymiarową elementów macierzy typu double dynamicznie alokowaną przy konstrukcji obiektu; przeciążyłem także operatory odpowiedzialne za podstawowe operacje na macierzach. Wektor reprezentuję przez klasę pochodną macierzy, to jest przez macierz o wymiarach Nx1, gdzie N – liczba elementów wektora.

Na potrzeby faktoryzacji LU zaimplementowałem podstawianie w przód i w tył.

Implementacja metod iteracyjnych wykorzystuje wzory podane powyżej. Warunkiem stopu dla obu metod jest osiągnięcie normy wektora residuum równej lub mniejszej niż 10^{-9} lub osiągnięcie 1000 iteracji (aby uniknąć nieskończonego wykonywania metod w razie rozbieżności).

Dla obu metod iteracyjnych za początkowe przybliżenie przyjmuję wektor o N elementach, gdzie każdy element jest równy 1/N.

Podpunkty A i B

Test polegał na porównaniu działania metod iteracyjnych Jacobiego i Gaussa-Seidela, tj. wykonywaniu obu metod aż do osiągnięcia normy z wektora residuum mniejszej lub równej 10^{-9} oraz porównaniu obu metod pod względem liczby iteracji i czasu[2] wykonania.

Na potrzeby pierwszego testu stworzyłem macierz pasmową A o rozmiarze NxN (tutaj 961 x 961) zawierającą pięć diagonali – główną z elementami a1 = 5+7 = 12, dwie sąsiednie z elementami a2 = -1 oraz dwie sąsiednie do nich z elementami a3 = -1 oraz wektora b o 961 elementach równych sin(n*(f-1)), gdzie n = 1…961.

Macierz A przedstawia się następująco:

$$\begin{bmatrix} 12 & -1 & -1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 12 & -1 & -1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & -1 & 12 & -1 & -1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & -1 & 12 & -1 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & -1 & -1 & 12 \end{bmatrix}$$

Statystyki przedstawiają się następująco:

- Jacobi – 20 iteracji, osiągnięta norma $8.04363 * 10^{-10}$, czas 0.067 s
- Gauss-Seidel – 15 iteracji, osiągnięta norma $4.3532 * 10^{-10}$, czas 0.048 s

Zgodnie z oczekiwaniami metoda Gaussa-Seidela zbiegła się szybciej.

Podpunkty C i D

Test polegał na przetestowaniu obu metod iteracyjnych i porównaniu ich z faktoryzacją LU dla podobnej macierzy pasmowej, ale dla współczynników a1 = 3, a2 = a3 = −1; rozmiary pozostały te same (961x961), wektor b też pozostał bez zmian.

Macierz A przedstawia się następująco:

$$\begin{bmatrix} 3 & -1 & -1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & -1 & 3 & -1 & -1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & -1 & 3 & -1 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & -1 & -1 & 3 \end{bmatrix}$$

Obie metody iteracyjne się rozbiegły – po granicznym tysiącu iteracji osiągnęły normy:

- Jacobi: $3.14364 * 10^{123}$
- Gauss-Seidel: +nieskończoność (doszło do przekroczenia zakresu typu double w czasie obliczania normy)

Przyczyną rozbiegnięcia się metod iteracyjnych jest fakt, iż w tym przypadku macierz A nie jest diagonalnie dominująca – nie jest spełniony jeden z warunków zbieżności obu metod iteracyjnych.

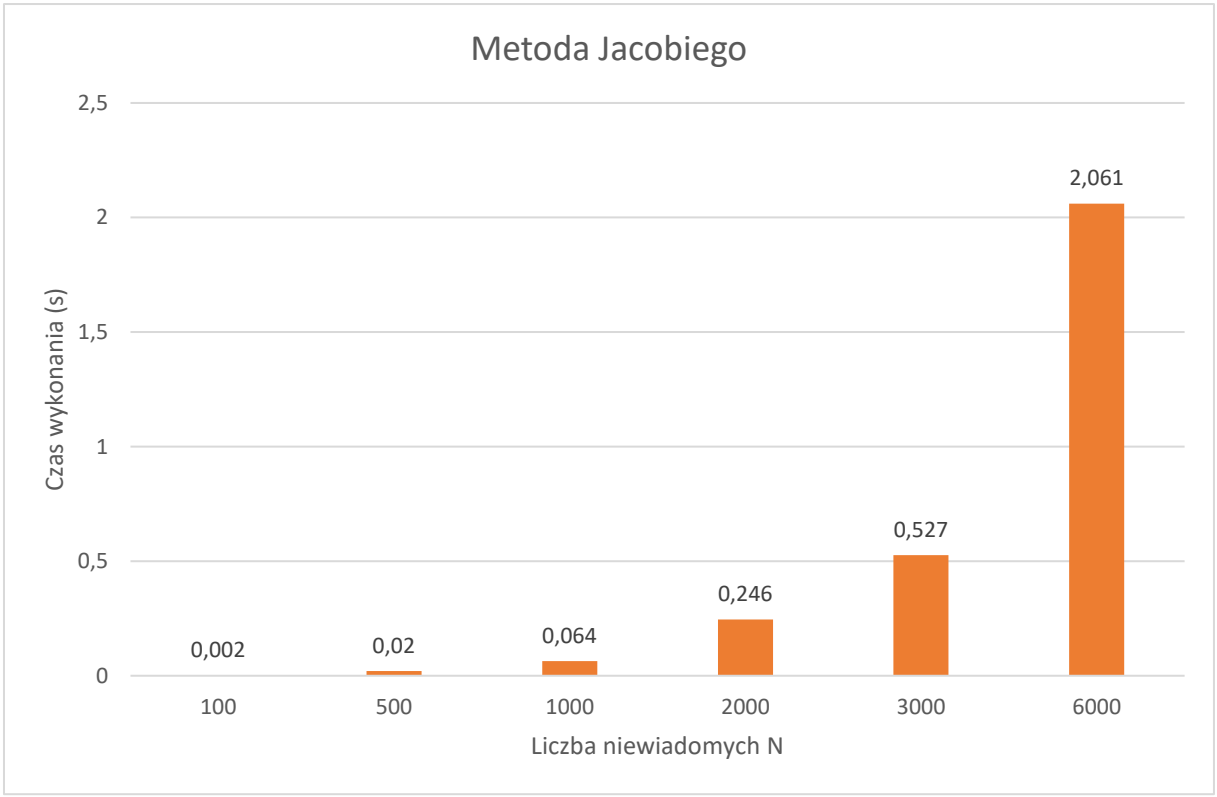
[2] Wykorzystano konfigurację Release, tryb 64-bit, optymalizacja pod względem czasu wykonania (VS 2015)

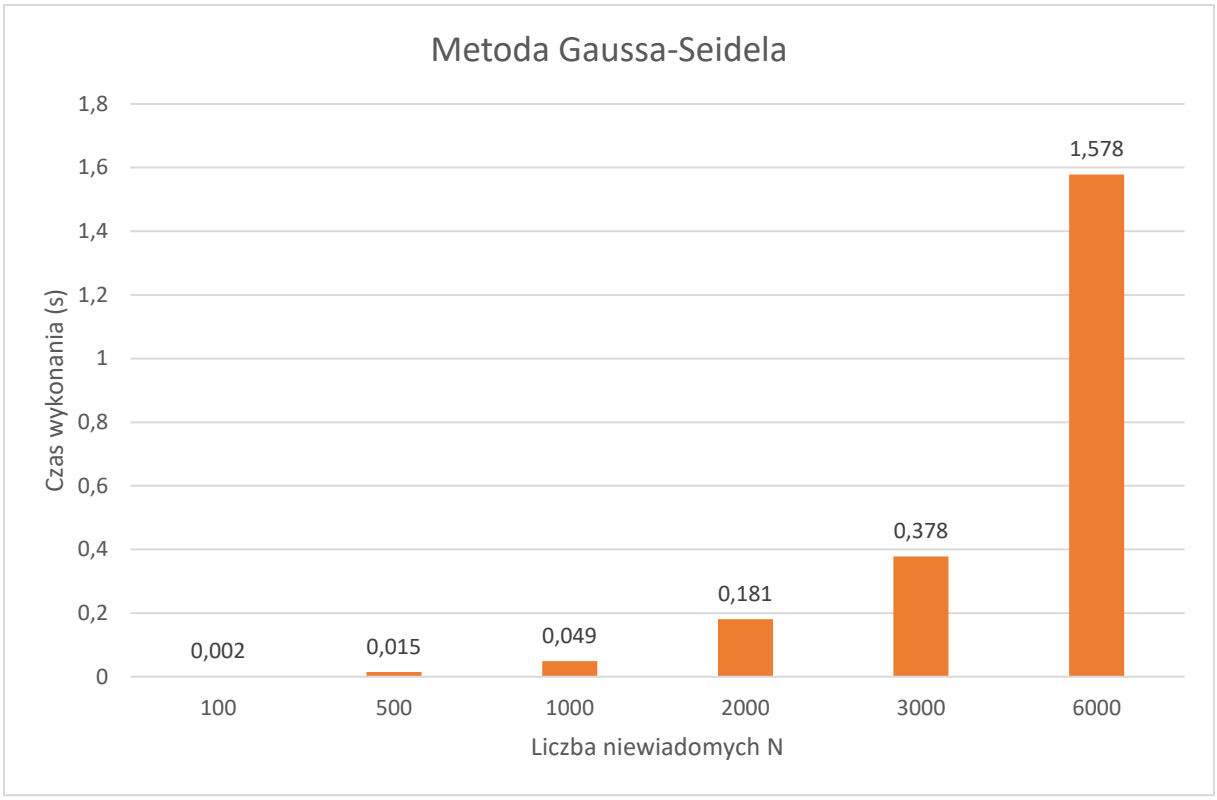
Metoda faktoryzacji LU zaś potrafiła wyliczyć rozwiązanie dla tego układu w czasie 0.32 s, osiągając normę wektora residuum równą $2.0811 * 10^{-13}$. Niepokoi to, że norma nie wynosi 0, jako że z założenia faktoryzacja LU jako metoda bezpośrednia ma zwrócić dokładny wynik, jednakże przyczyną może być nie tyle niepoprawne działanie, co strata precyzji przy obliczeniach zmiennoprzecinkowych – tym bardziej że nie jest to norma o dużym rzędzie.

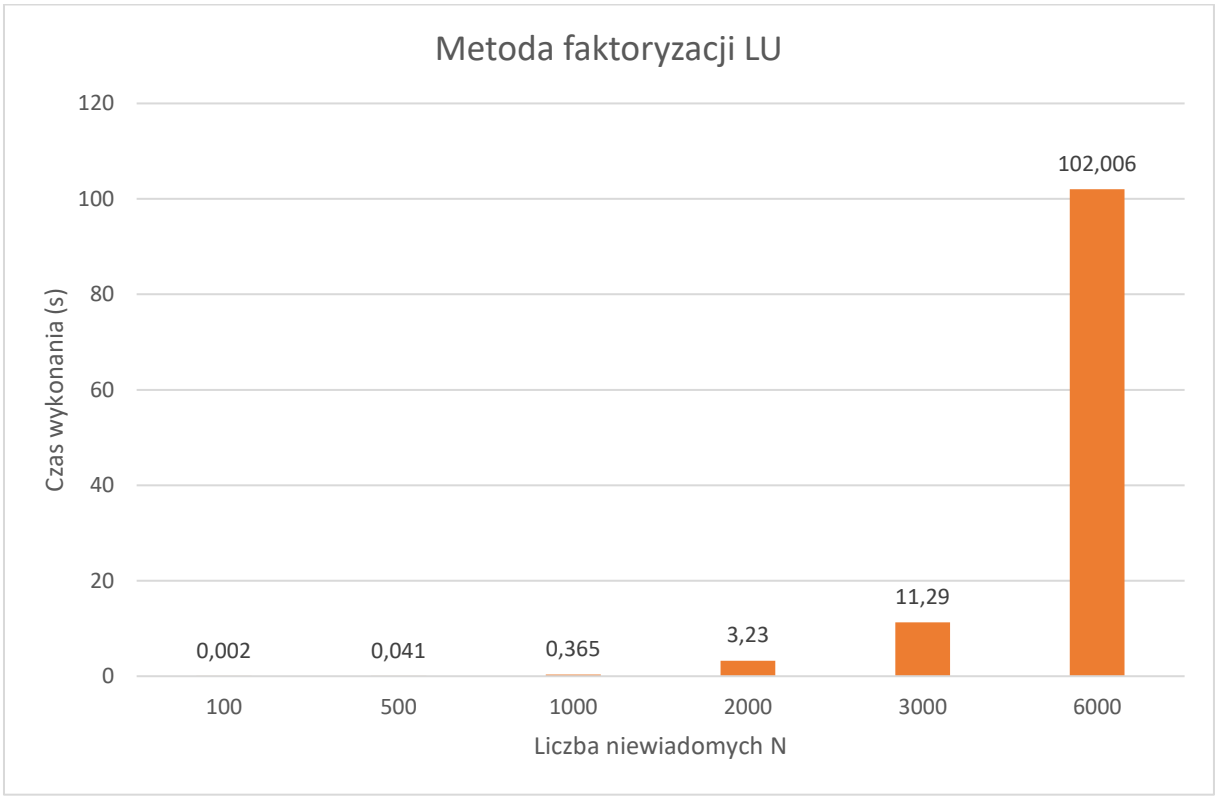
Podpunkt E

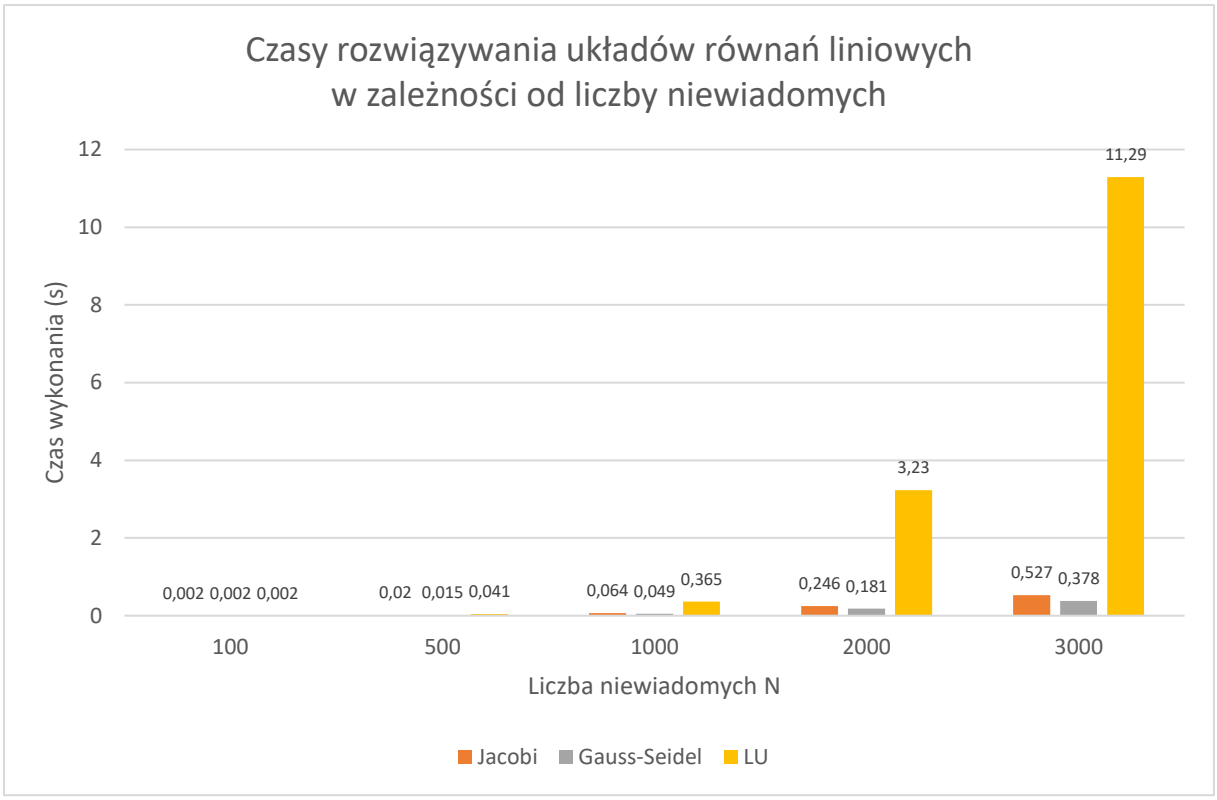
Test polegał na utworzeniu układu równań o założeniach podobnych jak dla podpunktu A, ale dla N równych: 100, 500, 1000, 2000, 3000, 6000 elementów oraz na porównaniu czasów wykonania dla obu metod iteracyjnych oraz faktoryzacji LU.

Wykresy czasów wykonania przedstawiają się następująco:




Metoda Gaussa-Seidela
1,8
1,6
1,4
1,2
1
0,8
0,6
0,4
0,2
0
Czas wykonania (s)
0,002
0,015
0,049
0,181
0,378
1,578
100
500
1000
2000
3000
6000
Liczba niewiadomych N


Metoda faktoryzacji LU
120
100
80
60
40
20
0
Czas wykonania (s)
0,002
0,041
0,365
3,23
11,29
102,006
100
500
1000
2000
3000
6000
Liczba niewiadomych N



W ostatnim wykresie zbiorczym pominąłem przypadek N = 6000 ze względu na wyraźnie większy czas wykonania dla faktoryzacji LU niż dla wszystkich innych przypadków, co spowodowałoby trudności w przedstawieniu wyników.

Obserwacje i wnioski

Czas wykonania metod iteracyjnych dla małych liczb niewiadomych (np. 100,500) nie różni się znacząco od czasu wykonania faktoryzacji LU; różnica staje się znacząca dla dużych wartości niewiadomych - przykładowo dla N=6000: 2,061 s dla metody Jacobiego, 1,578 s dla metody Gaussa-Seidela oraz aż 102,006 s dla faktoryzacji LU.

Jeżeli mówimy o samych metodach iteracyjnych, metoda Gaussa-Seidela osiąga nieco lepsze wyniki niż metoda Jacobiego – potrzebuje nieco mniej iteracji i czasu do osiągnięcia zadanej dokładności.

Z powyższego możemy wywnioskować, że dla małych ilości niewiadomych możemy bez większych problemów stosować metody bezpośrednie takie jak faktoryzacja LU; dla dużych ilości niewiadomych warto używać metod iteracyjnych. Co prawda może to skutkować obniżoną dokładnością rozwiązania, jednakże metody iteracyjne mają olbrzymią przewagę czasu wykonania nad bezpośrednimi (złożoność kwadratowa, nie sześcienna). Zastosowanie metod iteracyjnych może być wręcz koniecznością w rzeczywistych zastosowaniach, gdzie na co dzień rozwiązuje się układy równań liniowych z ilością niewiadomych przekraczającą milion.

Jeżeli z metod iteracyjnych mamy do wyboru metody Jacobiego oraz Gaussa-Seidela, warto wybrać tę drugą – oczywiście o ile się zbiega.