Układy równań liniowych

Metody Numeryczne – projekt nr. 2

sprawozdanie

Maciej Adryan

26 kwietnia 2021

Streszczenie

Celem projektu jest implementacja metod iteracyjnych (metody Jacobiego oraz Gaussa-Seidla) oraz metody bezpośredniej (faktoryzacji LU) rozwiązywania układów równań liniowych.

Do zrealizowania zadania wykorzystany przeze mnie został język Python w wersji 3.9.
Dodatkowo wykorzystane zostały:

- opensource'owa biblioteka **matplotlib**,
- biblioteka standardowa **time** pomocna przy mierzeniu czasu osiągnięcia zadowalającej zbieżności poszczególnych metod
- oraz biblioteka **numpy** ułatwiająca zarządzanie danymi.
- biblioteka **icecream** pomocna przy debugowaniu.

W celu mierzenia czasu wykoniania funkcji stworzyłem wrapper dla funkcji, informujący o czasie wykonania oraz otrzymanej normie residuum (oraz liczbie iteracji w przypadku metod iteracyjnych).

Dla czytelności program został rozbity na dwa pliki: **main**.py oraz **mylibs**.py, kroki potrzebne do rozwiązania poszczególnych zadań zostały przedstawione w pliku **main**.

Rozdział 1

Analiza

1. **Zadanie A**

   $Index = 175854$ otrzymujemy wartości $a1 = 13$ i $a2 = a3 = -1$. Otrzymana macierz współczynników ma wymiary 954×954, wektor danych ma wartośc równą $sin(n * (f + 1))$. Gdzie f jest trzecią cyfrą indeksu a n jest liczbą naturalną z zakresu $\langle 1, N \rangle$.

2. **Zadanie B**

   Dla danych z podpunktu A oraz ustalenia zadowalającej nas normy residuum równej 10^{-6}, po wykonaniu obydwu algorytmów otrzymujemy nastepujące dane:

   Jacobi: 0.07964944839477539 [s]
   iterations: 14
   residuum: $3.531624076185307 \times 10^{-7}$

   GaussSeidl: 1.3045377731323242 [s]
   iterations: 10
   residuum: $6.330398043165935 \times 10^{-7}$

   Metoda Jacobiego zbiega się do zadowalającej normy residuum zdecydowanie szybciej, wymaga jednak większej ilości iteracji niż wolniejsza w tym przypadku metoda Gaussa-Seidla.

3. Zadanie C

Dla otrzymanych wartości: $a1 = 3$, $a2 = a3 = -1$ oraz $N = 954$ oraz dla wektora **b** identycznego jak w Zadaniu A otrzymujemy komunikat informujący, iż dla podanych danych obydwie z metod nie zbiegają się:

Jacobi does not converge for this data!

GaussSeidl does not converge for this data!

Program po natrafieniu na wartośc residuum = **NaN** lub ∞ zwraca **tuple** o wartośiach (∞, ∞) informujący o fakcie niezbiegania się metod iteracyjnych dla podanych danych.

4. Zadanie D

Po zaimplementowaniu metody bezpośredniej rozwiązywania układów równań linowych (w tym przypadku faktoryzacji LU) i zastosowaniu jej na danych identycznych jak w Zadaniu C otrzymujemy nastepujące dane:

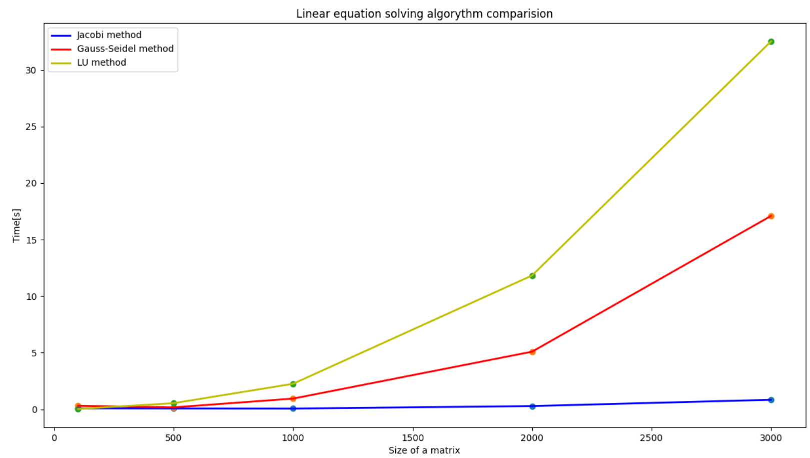
LU_solve:2.1593382358551025 [s]
residuum: $3.836268130024082 \times 10^{-15}$

Osiągnięty wynik ma zadowalającą dokładność, oraz, w przeciwieństwie do metody Gaussa-Seidla i Jacobiego, metoda faktoryzajci LU rozwiązuje zadany problem, jednak porównując czas jej wykonania z czasem wykoniania z Zadania B widać znaczny wzrost czasu wykonania. Wynika to z charakterystyki zastosowanej metody, jest ona bardziej uniwersalna niż metoda G-S czy Jacobiego, jednak znacznie wolniejsza, co zostanie dobrze zobrazowane w następnym podpunkcie.

5. Zadanie E

Po stworzeniu wykresu zależności czasu wykonania się poszczególnych algorytmów potwierdzają się dokonane przeze mnie obserwacje które zawarłem w podpunkcie D.
Wykres zależności czasu wykonania się algorytmów od liczby niewiadomych N $\subset\{100, 500, 1000, 2000, 3000\}$ przedstawia się następująco:

Rysunek 1.1: Wykres zależności czasu wykonania się algorytmów od liczby niewiadomych N $\subset\{100, 500, 1000, 2000, 3000\}$



Łatwo jest zauważyć, iż wykres potwierdza nasze poprzednie podejrzenia.

6. Zadanie F

Dla wszystkich trzech metod, czas wykonywania obliczeń wzrasta wraz z liczbą niewiadomych. Metoda Gaussa-Seidla oraz Jacobiego są znacznie szybsze od metody faktoryzacji LU a ich przewaga rośnie wraz ze wzrostem wymiarów macierzy. Nie wszystkie układy równań można jednak rozwiązać za pomocą metod iteracyjnych, tak jak miało to miejsce w Zadaniu C, w takich przypadkach należy zastosować wolniejszą, ale pewną, metodę bezpośrednia (np. faktoryzacji LU).