Jakub Kobyliński, 193618

Metody Numeryczne - Układy równań liniowych

Wstęp

Celem projektu była implementacja i analiza dwóch metod iteracyjnych - Jacobiego i Gaussa–Seidla oraz jednej metody bezpośredniej - metodę faktoryzacji LU. Do realizacji zastosowałem język Python 3.12 z wykorzystaniem biblioteki matplotlib, math, time oraz copy przy użyciu IDE - PyCharm.
Rozwiązywany układ równań liniowych miał następującą postać:

$$Ax = b$$

w której to:
A - jest macierzą systemową, pasmową o rozmiarze 918 x 918. Zawiera ona pięć diagonali - główna z elementami a1, dwie sąsiednie z elementami a2 i dwie skrajne diagonale z elementami a3.

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

W zadaniach A i B diagonale macierzy A przyjmują wartości: a1 = 5 + e (dla e wynoszącego 6), a2 = a3 = -1.
Natomiast w zadaniu C wartości a2 i a3 pozostają takie same, z wyjątkiem a1 = 3.

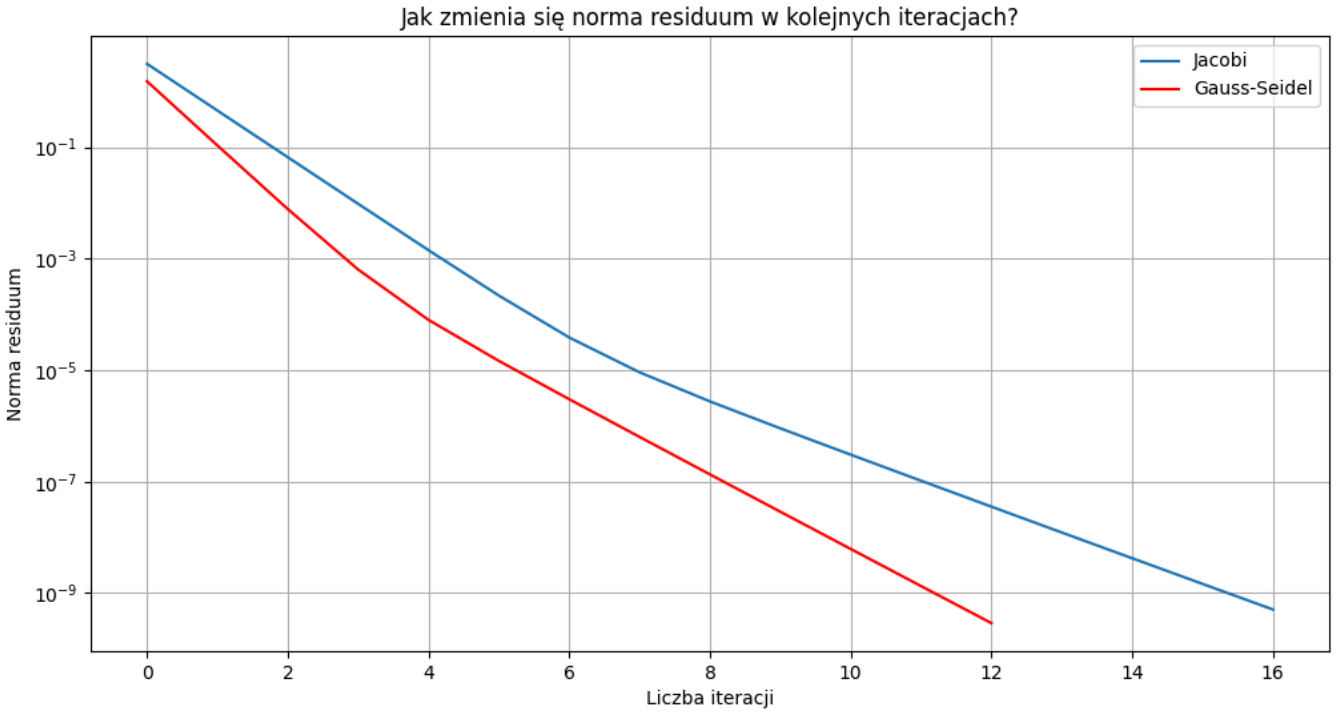
b - jest wektorem pobudzenia o długości 918, którego n−ty element ma wartość sin(n*(f + 1)), dla f przyjmujemy 3.

x - jest wektorem rozwiązań.

Analiza zadania B

Naszym zadaniem było zaimplementowanie dwóch algorytmów iteracyjnych do rozwiązywania równań liniowych - Jacobiego i Gaussa-Seidla, a następnie porównanie ich pod względem ilości iteracji oraz czasu potrzebnych do wyznaczenia rozwiązania układu równań dla macierzy z zadania A, którego norma residuum jest mniejsza niż 10^−9. Tak prezentują się wyniki:

```
Czas trwania algorytmu Jacobiego: 2.015671491622925 sekund
Liczba iteracji algorytmu Jacobiego: 17
Czas trwania algorytmu Gaussa-Seidla: 1.3917930126190186 sekund
Liczba iteracji algorytmu Gaussa-Seidla: 13
```



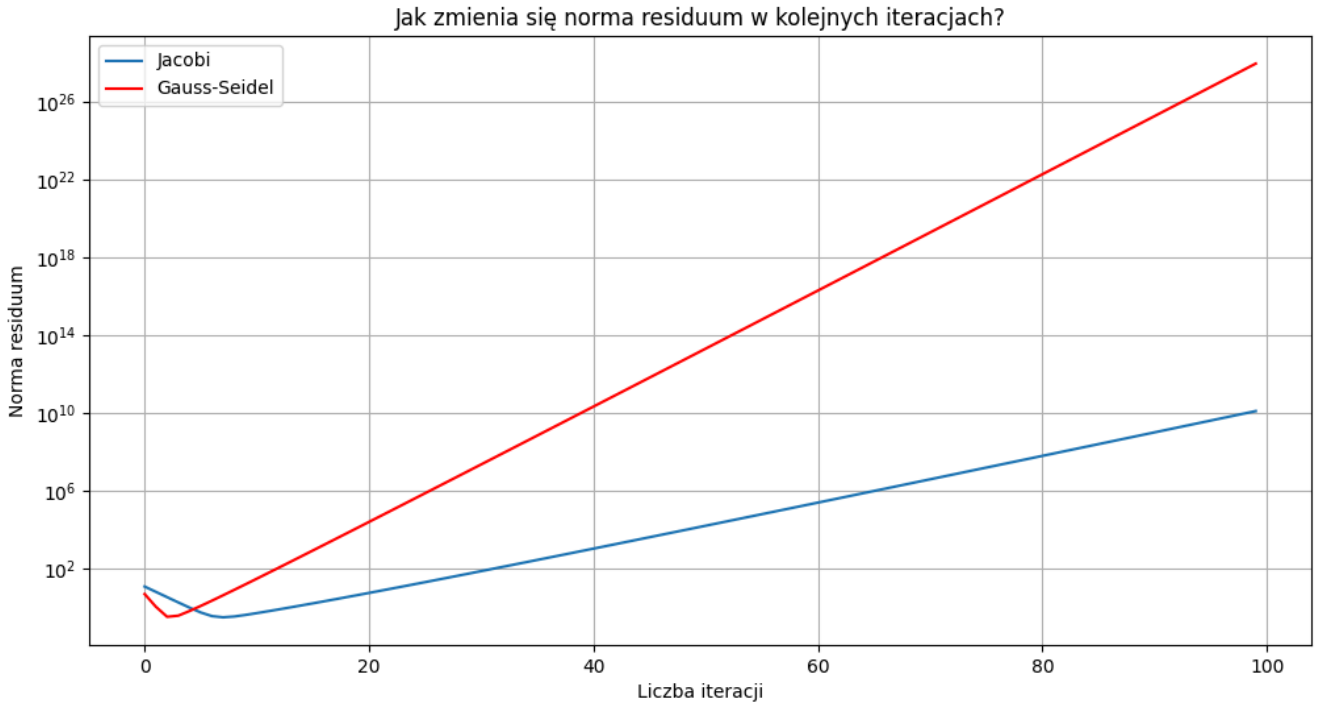
Z przeprowadzonej analizy widzimy że metoda Gaussa–Seidla okazała się być nieznacznie szybsza niż metoda Jacobiego. Swoje zadanie wykonała o około 22 sekundy szybciej oraz potrzebowała do tego o 4 iteracje mniej. Można zauważyć, że dla obu przypadków normy residuum zbiegają się do oczekiwanej wartości w prawie że sposób wykładniczy. Tutaj warto zauważyć, że w tym przypadku macierz A była macierzą diagonalnie dominującą, czyli taką, w której wartości bezwzględne elementów na głównej przekątnej są większe od sumy wartości bezwzględnych pozostałych elementów w wierszach. Jest to własność potrzebna, aby nasze metody iteracyjne się zbiegały.

Analiza zadania C

W tym zadaniu zasady pozostają takie same jak B, lecz tym razem do obliczeń wykorzystujemy tą samą macierz A, lecz zmienioną o diagonale a1 = 3. Tak prezentuje się wyniki:

```
Traceback (most recent call last):
  File "C:\Users\User\Desktop\Studia\Semestr 4\Metody Numeryczne\Projekt\Projekt2-Układy-równań-liniowych\main.py", line 229, in <module>
    plot_test(A, b)
  File "C:\Users\User\Desktop\Studia\Semestr 4\Metody Numeryczne\Projekt\Projekt2-Układy-równań-liniowych\main.py", line 162, in plot_test
    x_jacobi, iterations_jacobi, residuals_jacobi = jacobi(A, b)
                                                    ^^^^^^^^^^^
  File "C:\Users\User\Desktop\Studia\Semestr 4\Metody Numeryczne\Projekt\Projekt2-Układy-równań-liniowych\main.py", line 78, in jacobi
    residuum_norm = norm(residual(A, b, x_new))
                    ^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\User\Desktop\Studia\Semestr 4\Metody Numeryczne\Projekt\Projekt2-Układy-równań-liniowych\main.py", line 59, in norm
    return math.sqrt(sum(x[0] ** 2 for x in residual))
                     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\User\Desktop\Studia\Semestr 4\Metody Numeryczne\Projekt\Projekt2-Układy-równań-liniowych\main.py", line 59, in <genexpr>
    return math.sqrt(sum(x[0] ** 2 for x in residual))
                         ~~~~~^^~~
OverflowError: (34, 'Result too large')
```

Jak widać program wygenerował nam błąd z kategorii OverflowError, może nam to sugerować, że metody nie zbiegają się. Analizę przeprowadzimy jeszcze raz, ale teraz oprócz ograniczenia normy residuum, aby była mniejsza niż 10^-9, dodamy ograniczenie iteracji przykładowo do 100. Oto wynik naszej operacji:



Jak możemy zauważyć po wykresie nasze obie metody nie zbiegają się. Mimo tego, że nasza norma residuum początkowo maleje, to potem zaczyna rosnąć w sposób wykładniczy. Powodem, dla którego może się tak dziać jest modyfikacja jakiej uległa nasza macierz A, przez co nie jest ona już macierzą diagonalnie dominującą. Warte uwagi jest również to, że dla metody Gaussa–Seidla residuum rośnie znacznie szybciej niż dla metody Jacobiego. Przykład pokazuje, że nasze obie metody Gaussa-Seidla oraz Jacobiego nie zawsze muszą działać i prowadzić nas do oczekiwanego wyniku.

Analiza zadania D

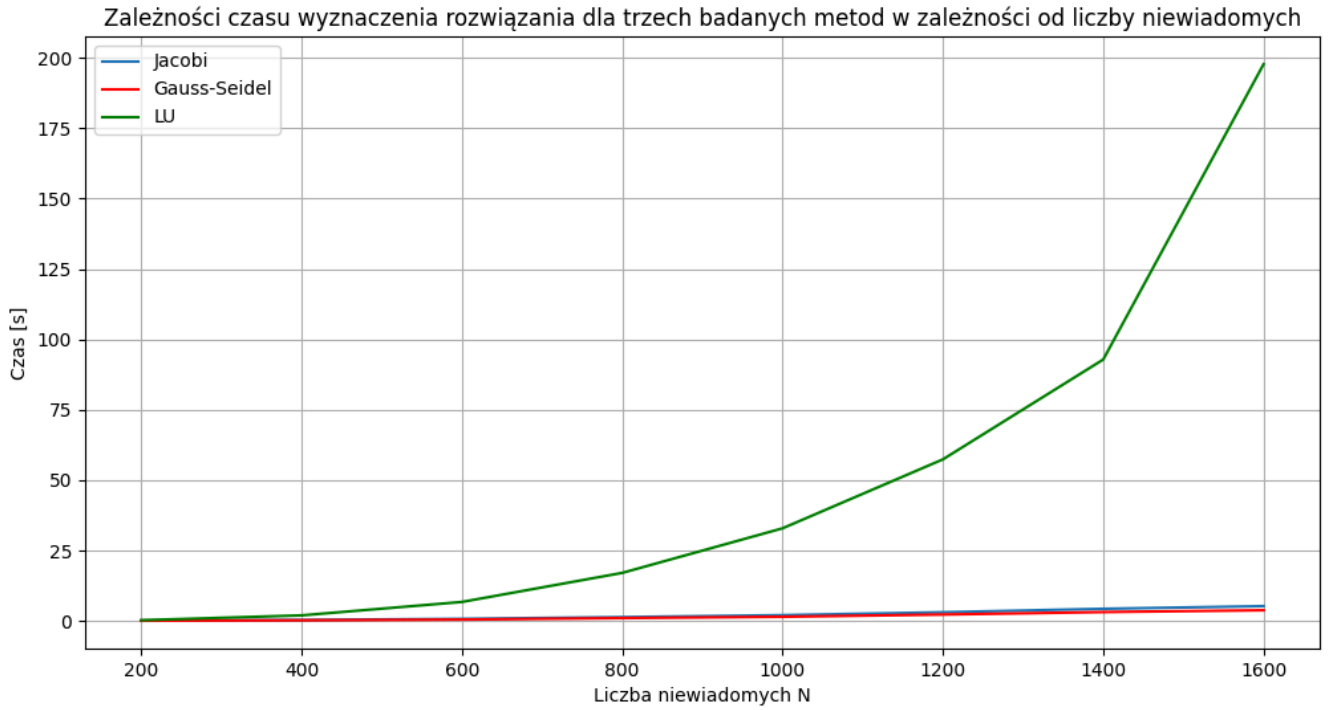
Celem tego zadania była implementacja metody bezpośredniej rozwiązywania układów równań liniowych, czyli metody faktoryzacji LU i zastosowania jej do policzenia równania z podpunktu C, czyli na naszej zmodyfikowanej macierzy A. Tak prezentuje się wyniki:

```
Norma residuum dla metody faktoryzacji LU wynosi: 1.0256888358728 97e-13
Czas trwania algorytmu faktoryzacji LU: 42.92463755607605 sekund
```

Jak widać metodzie LU udało się obliczyć normę residuum w przeciwieństwie do naszych dwóch poprzednich metod. Jest to metoda kosztowna obliczeniowo (O(n^3), dlatego jak widać zajęła też sporo czasu, bo prawie 43 sekundy, ale dała nam bardzo dobry i dokładny wynik w postaci residuum na poziomie 1.025688835872897e-13.

Analiza zadania E

Zadaniem było stworzenie wykresu zależności czasu wyznaczenia rozwiązania dla trzech badanych metod w zależności od liczby niewiadomych N, ja przyjąłem za N = {200, 400, 600, 800, 1000, 1200, 1400, 1600}. Tak wygląda wygenerowany wykres:



Na wykresie można zauważyć, jak bardzo czasochłonna staje się metoda faktoryzacji LU wraz ze wzrostem rozmiaru macierzy w porównaniu do naszych dwóch metod iteracyjnych. Natomiast metody Jacobiego oraz Gaussa–Seidla mają bardzo zbliżony do siebie, wręcz niezauważalny czas wykonania z minimalną korzyścią na stronę tej drugiej.

Podsumowanie - zadanie F

Każda z analizowanych metod ma swoje plusy i minusy. Za metodą bezpośrednią LU przemawia jej dokładność i pewność uzyskania wyniku, lecz jej zauważalną wadą jest złożoność obliczeniowa, która przekłada się na długi czas wykonywania, szczególnie dla dużych rozmiarowo macierzy. Metody Jacobiego oraz Gaussa–Seidla charakteryzują się o wiele lepszym czasem wykonywania, ale dają też mniej zadowalające i dokładne wyniki, a nawet może się zdarzyć, że wystąpi

przypadek kiedy będą się one rozbiegać. Natomiast jeśli chodzi o porównanie tych dwóch metod iteracyjnych to metoda Gussa-Seidla wykonuje się minimalnie szybciej i w mniejszej ilości iteracji, lecz skutkuje to także w drugą stronę szybszym wzrostem normy residuum w przypadku, kiedy metody nie zbiegają się. Podsumowując wybór metody jest zależy w głównej mierze od tego czego oczekujemy i też jakim sprzętem i mocą obliczeniową dysponujemy, czy postawimy na dokładność, czy na czas wykonania.