**Rozwiązywanie równań liniowych metodami iteracyjnymi**

**Krystian Madej, 14.06.2024**

**1. Treść zadania**

Macierz jest zadana wzorami:

Dany jest układ równań liniowych . Przyjmij wektor jako dowolną n-elementową permutację zbioru i oblicz wektor .

1. Metodą Jacobiego rozwiąż układ równań liniowych przyjmując kryteria stopu:

* – kryterium krokowe
* – kryterium wartościowe

Obliczenia wykonaj dla różnych rozmiarów układu , dla wektorów początkowych: i , oraz różnych wartości w kryteriach stopu. Podaj jak liczono normę. Wyznacz liczbę iteracji oraz sprawdź różnicę w czasie obliczeń dla różnych kryteriów stopu. Sprawdź dokładność obliczeń.

2. Dowolną metodą znajdź promień spektralny macierzy iteracji dla różnych rozmiarów układu. Sprawdź czy spełnione są założenia o zbieżności metody dla zadanego układu. Opisz metodę znajdowania promienia spektralnego.

**2. Środowisko obliczeń**

**­**Obliczenia zostały wykonane przy pomocy języka **C++20** na systemie **Windows 11**, kompilacja 22631.3593, na **12-wątkowym** procesorze **64-bitowym** Intel Core i5-11400H 2.70GHz, kod kompilowany kompilatorem **MSVC** (wersja 19.39). Obliczenia wykonywane na typie double.

**3. Użyte biblioteki i programy pomocnicze**

Wykresy rysowano programem Excel z pakietu Microsoft Office.

Do instalacji bibliotek **C++** użyto programu **conan**, wersja 2.3.2.

Najważniejsze użyte biblioteki:

* <format> - łatwe formatowanie
* <numbers> - stałe matematyczne
* <future> - obiekty std::future oraz std::async
* <thread> - wielowątkowość
* <fstream> - zapisywanie do plików
* <chrono> - zegary i operacje na jednostkach czasu
* Armadillo – algebra liniowa, operacje na macierzach/wektorach

**4. Sposób obliczeń**

**4.1 Obliczanie normy wektora**

Norma z wektora jest normą euklidesową:

**4.2 Obliczanie macierzy iteracji**

Mając układ równań:

gdzie:

* – macierz
* – wektora niewiadomych
* – wektor danych

Następnie sprowadzamy macierz do postacji sumy dwóch macierzy

gdzie:

* – macierz diagonalna, której przekątna jest równa przekątnej macierzy

Przekształcając układ równań otrzymamy:

Zapisując ostatnie równanie w postaci iteracyjnej otrzymamy:

Macierz nazywamy macierzą iteracji.

**4.3 Obliczanie promienia spektralnego**

Promień spektralny macierzy iteracji liczono z definicji jako maksimum wartości bezwzględnych z wartości własnych tej macierzy. Wartości własne macierzy są wyliczane biblioteczną funkcją arma::gen\_eig.

**4.4 Wartości błędów**

Wartości błędów są liczone normą euklidesową z wektora , gdzie jest wektorem zadanym, a wektorem obliczonym.

**5. Metoda postępowania**

Na początku program tworzy macierz według wzorów opisanych w punkcie 1. Następnie tworzy jest wektor i oblicza wektor . Następnie tworzone są dwa wektory początkowe, opisane w punkcie 1. Na koniec rozwiązuje się układ równań metodą Jacobiego, dla wszystkich kombinacji i warunków stopu.

Obliczenia wykonałem dla  
. Ustaliłem limit iteracji na .

**6.1 Wartości promienia spekralnego**

Wykres 1. Promień spektralny

Jak widać na wykresie 1, niezależnie od rozmiaru układu promień spektralny nie zmienia się i jest równy . Różnice w niektórych miejscach są rzędu i wynikają z błędów reprezentacji na typie double. Zatem każdy rozmiar układu spełnia twierdzenie o zbieżności procesu iteracyjnego: „Ciąg z dowolnym wektorem startowym jest zbieżny do jedynego granicznego wtedy i tylko wtedy, gdy ***promień spektralny (spectral radius)*** macierzy iteracji jest mniejszy od 1”.

**6.2 Wyniki obliczeń metody Jacobiego**

**6.2.1 Liczba iteracji**

Wykres 2. Liczba iteracji dla niezerowego wektora początkowego i kryterium krokowego

Wykres 3. Liczba iteracji dla niezerowego wektora początkowego i kryterium wartościowego

Wykres 4. Liczba iteracji dla zerowego wektora początkowego i kryterium krokowego

Wykres 5. Liczba iteracji dla zerowego wektora początkowego i kryterium wartościowego

Jak widać na wykresach 2-5 liczba iteracji pozostaje taka sama lub zwiększa się o 1, mimo zwiększania rozmiaru układu. Liczka iteracji zwiększa się wraz z dokładnością. Można także zauważyć iż liczby iteracji dla poszczególnych kryteriów stopu nie zmieniają się pomimo użycia innego wektora początkowego. Wartym uwagi jest fakt, że dla kryterium wartościowego obliczenia nie zakańczały się po ustalonym limicie iteracji (równym ) już dla niewielkich rozmiarów układu. Jest to widoczne na wykresie jako brak dalszych punków dla . Nie jest to spowodowane niezbieżnością układu, gdyż dla każdy z nich spełnia warunek zbieżności (pokazane w punkcie 6.1), a niedoskonałością artmetyki na typie double.

**6.2.2 Błędy obliczeń**

Tabela 1. Błędy obliczeń dla niezerowego wektora początkowego

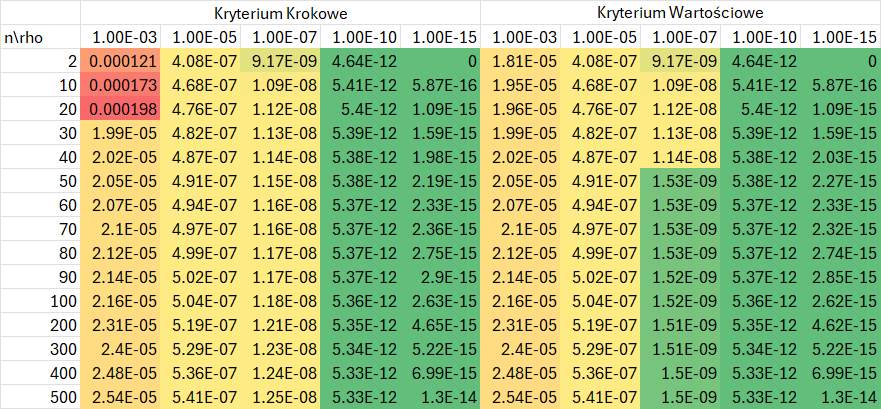
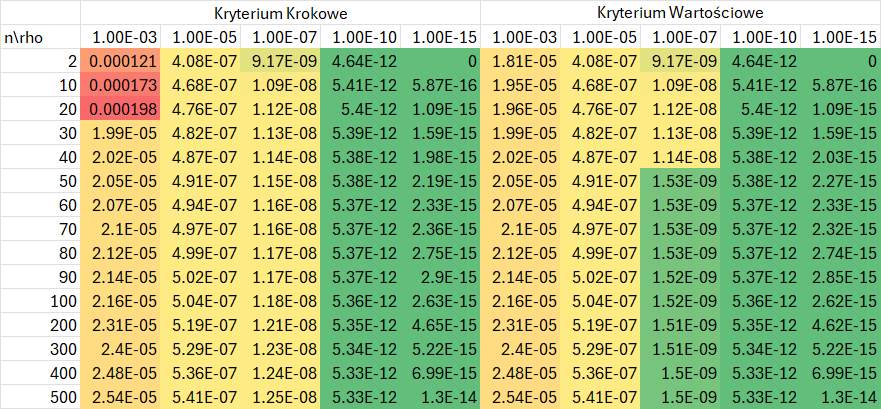


Tabela 2. Błędy obliczeń dla zerowego wektora początkowego

Jak widać na tabelach 1. i 2., wartości błędów dla obu wektorów początkowych są niemal identyczne. Dla kryterium krokowego i błąd zwiększa się dla , po czym gwałtownie spada o jeden rząd wielkości. Podobna sytuację widać dla kryterium wartościowego i . Można też zauważyć, że pomimo wykonania iteracji, więszość wartości dla nie jest nawet bliska spełnieniu kryterium wartościowego. Wartości błędów są niemal identyczne dla większych układów , niezależnie od użytego kryterium.

**6.2.3 Czas obliczeń**

Tabela 3. Czasy obliczeń w milisekundach dla niezerowego wektora początkowego

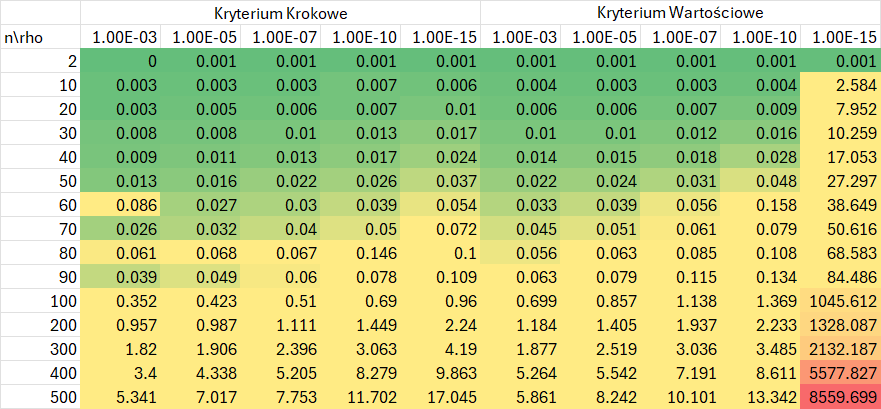
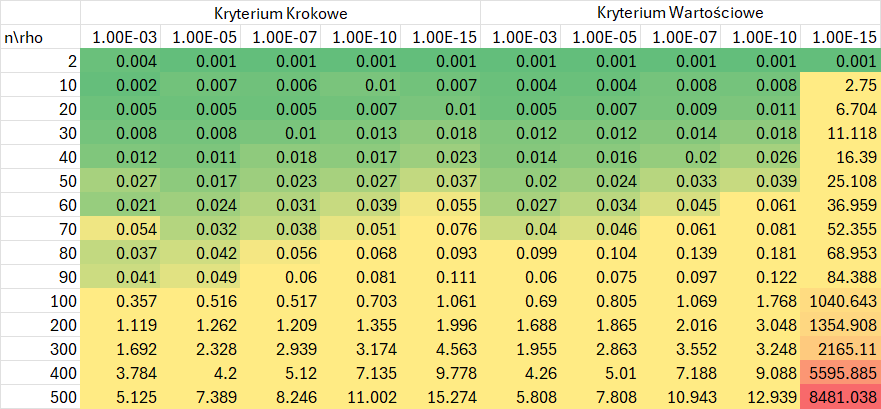
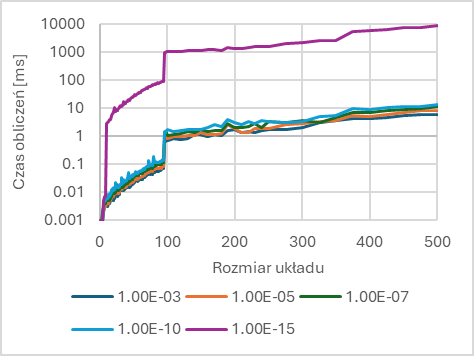
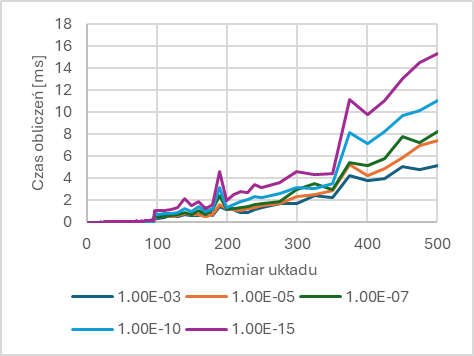


Tabela 4. Czasy obliczeń w milisekundach dla zerowego wektora początkowego

Na wykresach 1. i 2. oraz tabeli 1. widać, że obliczenia dla typu float mają zwykle większy błąd niż dla typu double. Wyraźnie widoczny jest też trend wzrostowy błędu średniego wraz ze wzrostem rozmiaru układu. Błąd dla punktu 2. jest o rzędy wielkości mniejszy od błędu dla punktu 1.



Wykres 6. Czasy obliczeń w milisekundach dla niezerowego wektora początkowego i kryterium krokowego.

Wykres 7. Czasy obliczeń w milisekundach dla niezerowego wektora początkowego i kryterium wartościowego, skala logarytmiczna.

Wykres 7. Czasy obliczeń w milisekundach dla zerowego wektora początkowego i kryterium krokowego.

Wykres 9. Czasy obliczeń w milisekundach dla zerowego wektora początkowego i kryterium wartościowego, skala logarytmiczna.

Jak widać na tabelach 3. i 4. oraz wykresach 6-9, czas obliczeń zwiększa się wraz z rozmiarem układu. Podobnie na czas wpływa dokładność obliczeń. Dobrze widoczny jest nagły skok w czasie obliczeń dla oraz kryterium wartościowego. Następuje on dokładnie tam, gdzie obliczenia zaczęły zakańczać się nie pod wpływem spełnienia kryterium stopu, lecz przekraczania limitu iteracji. Nie widać znaczących różnic w zależności od użycia wektora zerowego lub niezerowego. Na wykresach jest też widoczny nagły skok w czasie obliczeń blisko . Najbardziej prawdopodobnym powodem jest implementacja wewnątrz biblioteki Armadillo.

**7. Wnioski**

Dla zadanego układu równań wartość promienia spektralnego jest praktycznie stała, nie zależy od rozmiaru układu. Dodatkowo dowodzi, że każdy układ jest zbieżny metodą iteracyjną.

Liczba iteracji dla danej dokładności obliczeń jest względnie stała, w niektórych przypadkach rośnie co najwyżej o , po czym dalej nie zmienia się. Zwiększa się natomiast wraz ze zmniejszaniem wartości . Na liczbę iteracji nie wpływa to czy rozpoczniemy obliczenia od wektora zerowego czy niezerowego. Ze względu na ograniczenie artmetyki zmiennoprzecinkowej obliczenia dla wartościowego warunku stopu i nie były zbieżne, pomimo iż udowodniona została zbieżność dla układu.

W większości przypadków wartości błędów pozstają niewielkie. Dla większych rozmiarów układów czas obliczeń jest niemal identyczny dla obu kryteriów. Podobnie jak w przypadku liczby iteracji, wartości wektora początkowego nie wpłynęły na wartości błędów.

Czas obliczeń dla tej samej wartości był większy dla kryterium wartościowego. Zwiększał się wraz z rozmiarem układu. Wybór wektora początkowego nie wpłynął istotnie na czas obliczeń. Dla kryterium wartościowego i obliczenia trwały najdłużej, co nie jest dziwne biorąc pod uwagę liczbę wykonanych iteracji.