**Rozwiązywanie równań liniowych metodami bezpośrednimi**

**Krystian Madej, 05.06.2024**

**1. Treść zadania**

Przyjmij wektor jako dowolną n-elementową permutację ze zbioru .

1. Elementy macierzy o wymiarze są określone wzorami:

Oblicz wektor zadany wzorem . Następnie metodą eliminacji Gaussa rozwiąż układ równań . Przyjmij różną precyzję obliczeń dla wartości macierzy i wektora . Sprawdź w jaki sposób błędy zaokrągleń wpływają na rozwiązania różnych rozmiarów układu, porównując zgodnie z wybraną normą wektor zadany i wektor obliczony.

1. Powtórz obliczenia dla macierzy zadanej wzorem:  
   Porównaj z wynikami punktu **1.** Uzasadnij, skąd biorą się różnice w wynikach oraz sprawdź uwarunkowanie obu układów.
2. Powtórz eksperyment dla macierzy zadanej wzorem:  
   Następnie ponownie rozwiąż układ równań, tym razem algorytmem Thomasa, dla macierzy trójdiagonalnych. Porównaj wyniki metod Gaussa i Thomasa (dokładność obliczeń, czas) dla różnych rozmiarów układu. Opisz sposób przechowywania macierzy trójdiagonalnej.

**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, z najwyższym możliwym priorytetem, kod kompilowany kompilatorem **MSVC** (wersja 19.39). Obliczenia wykonane na typach wbudowanych: float (32-bitowy) i double (64-bitowy).

**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
* <random> - algorytm Mersenne Twister (std::mt19937)

**4. Sposób obliczeń**

**4.1 Norma porównywania błędów**

Za normę do porównywania błędów przyjęto błąd średni:

– wektor zadany  
 – wektor obliczony  
 – rozmiar obu wektorów

**4.2 Uwarunkowanie układu**

Uwarunkowanie układów jest obliczane wzorem:

gdzie:

– macierz współczynników układu

– norma macierzy (maksymalna suma wartości bezwzględnych wierszy)  
 – jest obliczana metodą Gaussa, sprowadzając macierz blokową do postaci , gdzie to macierz odwrotna

**4.3 Metoda Gaussa**

Do obliczeń użyto metody Gaussa z Partial Pivotingiem.

**5. Implementacja obliczeń**

Na początku zaimplementowano funkcje operacji na  
macierzach i wektorach:

* add\_rows – funkcja dodająca jeden wiersz macierzy do innego, złożoność czasowa:
* subtract\_rows – funkcja odejmująca jeden wiersz od innego, złożoność czasowa: lub , gdzie k to długość zadanego przedziału
* multiply\_rows – funkcja mnożąca wiersz przez skalar,  
  złożoność czasowa:
* divide\_rows – funkcja dzieląca wiersz przez skalar,  
  złożoność czasowa:
* swap\_rows – funkcja zamieniająca kolejnośc wierszy w macierzy, złożoność czasowa:

Funkcje te przyjmują macierz, na której operacja ma zostać wykonana, drugą macierz lub wektor na których operacje także zostanie wynonana. Funkcje add\_rows i subtract\_rows przyjmują też numery wiersza do którego dodać/od którego odjąć inny wiersz oraz numer wiersza dodawanego/odejmowanego oraz skalar, przez który wiersz dodawany/odejmowany ma być przemnożony przed operacją. Dodatkowo subtract\_rows przyjmuje opcjonalne parametry określające przedział odejmowania. Funkcje multiply\_rows i divide\_rows przyjmują dodatkowo numer wiersza mnożonego/dzielonego oraz skalar. Funkcja swap\_rows przymuje dodatkowo numery zamienianych wierszy.

Przy użyciu tych funkcji zaimplementowano funkcje solve, oraz solve\_tridiag, rozwiązujące zadane układy równań odpowiednio metodą Gaussa i Thomasa. Zakładając postąć układu , obie funkcje przyjmują macierz oraz wektor , a zwracają trójkę: obliczony wektor , macierz po przekształceniach oraz wektor po przekształceniach. Implementacja algorytmu Thomasa przyjmuje macierz tylko w postaci kwadratowej .

Zaimplementowano też funkcję gauss\_inv, obliczająca macierz odwrotną zadanej macierzy metodą Gaussa opisaną w punkcie 4.2. Przy jej pomocy napisano funkcję condition obliczającą wskaźnik uwarunkowana zadanej macierzy, wedługo metody w punkcie 4.2.

Zaimplementowana też funkcję avg\_error, obliczającą błąd średni wg. wzoru w punkcie 4.1.

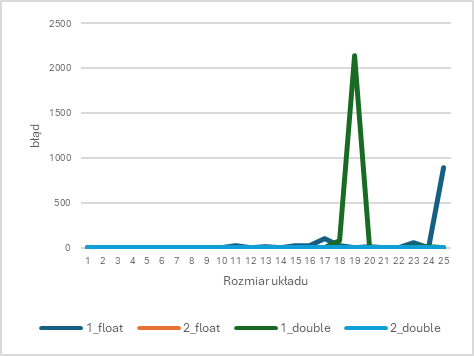
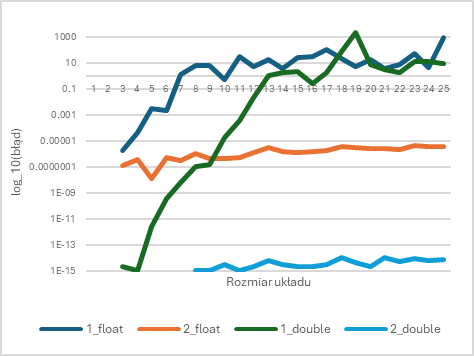
Opisane wyżej funkcje są szablonowe, podczas kompilacji dostosują się do zadanego im typu zmiennoprzecinkowego.

Początkowy wektor jest wyznaczany osobno dla typu float i double, przy pomocy osobnych instancji algorytmu pseudolosowego Mersenne Twister. Obie instancje mają identyczne ustawienia początkowe, co gwarantuje identyczność obu wariantów wektora .

Pliki z wynikami obliczeń zostaną zapisane w folderze z plikiem wykonywalnym.

**6. Porównanie wyników**

**6.1 Błędy obliczeń w podpunktach 1. i 2.**

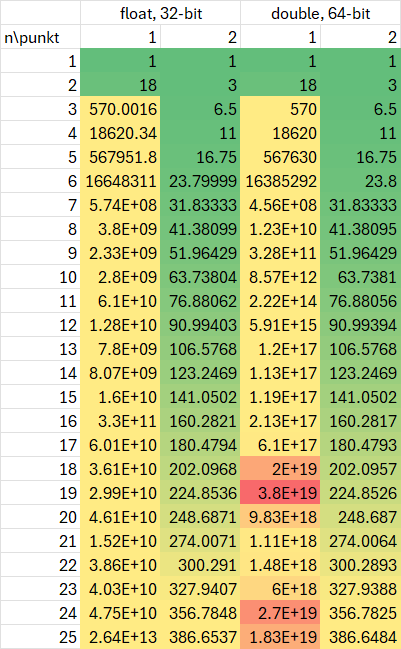


Wykres 1. Średnie błędy obliczeń w podpunktach 1. i 2., skala logarytmiczna

Tabela 1. Średnie błędy obliczeń w podpunktach 1. i 2.

Wykres 2. Średnie błędy obliczeń w podpunktach 1. i 2.

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.

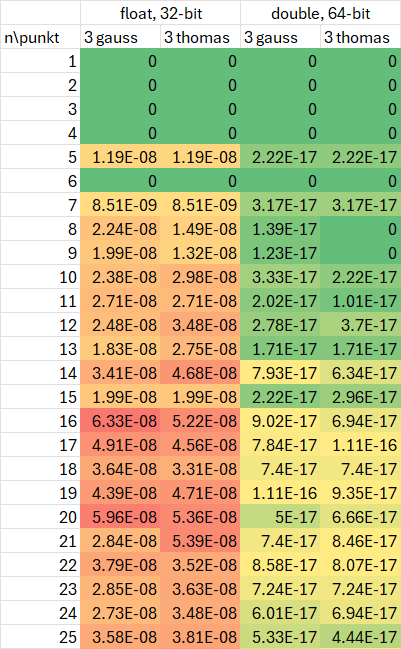
** 6.2 Uwarunkowanie w podpunktach 1. i 2.**

Jak widać na tabeli 2. i wykresach 3. i 4. współczynnik uwarunkowania rośnie wraz z rozmiarem układu. Uwarunkowanie jest o rzędu wielości większe w podpunkcie 2. w porównaniu do podpunktu 1. To dlatego wartości błędów były mniejsze w podpunkcie 1. w porównaniu do podpunktu 2. Miejsca nagłych przyrostów uwarunkowania pokrywają się z miejscami nagłych przyrostów błędów, np. n=25 i n=19 dla punktu pierwszego, odpowiednio dla typów float i double. Łatwo także zauważyć, iż uwarunkowanie jest zbliżone w podpunkcie 2., niezależnie od typu zmiennoprzecinkowego.

Wykres 3. Uwarunkowanie układów  
w podpunktach 1. i 2., skala logarytmiczna

Tabela 2. Uwarunkowanie układów  
w podpunktach 1. i 2.

Wykres 4. Uwarunkowanie układów  
w podpunktach 1. i 2.

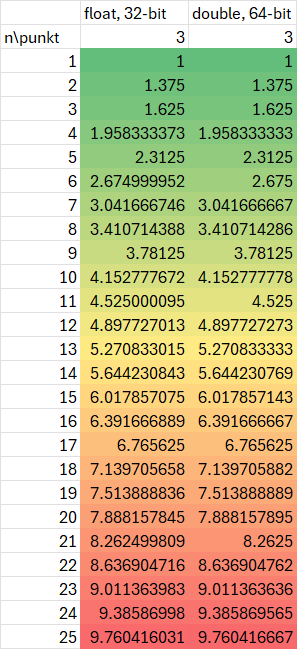
**6.3 Błędy obliczeń w podpunkcie 3.**

Wykres 5. Średnie błędy obliczeń w podpunkcie 3.

Na wykresach 5. i 5. oraz tabeli 3. widać, że dla podpunktu 3. obliczenia są wyraźnie dokładniejsze od tych w podpunkcie 1. Występują różnice pomiędzy metodą Gaussa a algorytmem Thomasa. Metoda Gaussa jest nieco dokładniejsza od algorytmu Thomasa. Po przekroczeniu pewnego rozmiaru układu, błąd utrzymuje się na względnie stałym poziomie.

Tabela 3. Średnie błędy obliczeń w podpunkcie 3.

Wykres 6. Średnie błędy obliczeń w podpunkcie 3. dla typu double

** 6.4 Uwarunkowanie w podpunkcie 3.**

Jak widać na tabeli 4. i wykresie 7. współczynnik uwarunkowania rośnie liniowo wraz z rozmiarem układu. Uwarunkowanie jest o rzędy wielkości mniejsze niż w podpunkcie 2. i 1. To tłumaczy niewielkie wartości błędów. Różnica pomiędzy float a double jest prawie żadna.

Wykres 7. Uwarunkowanie układów w podpunkcie 3.

Tabela 4. Uwarunkowanie układów w podpunkcie 3.

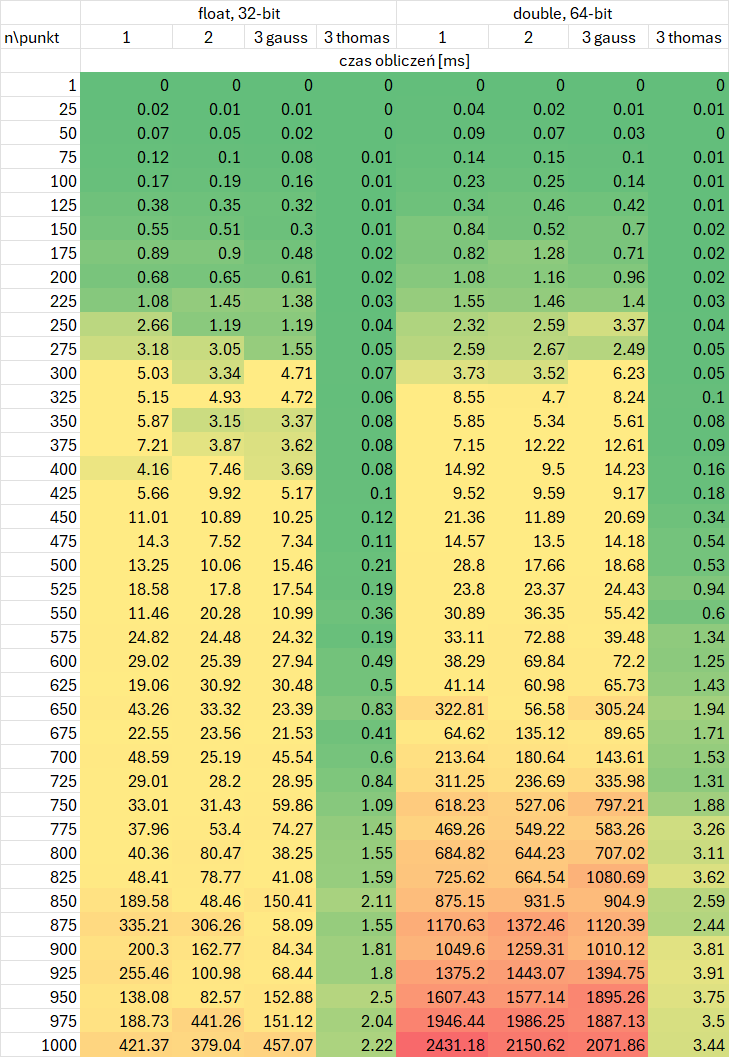
**6.5 Porównanie czasów obliczeń**

Tabela 5. Czasy obliczeń

Wykres 8. Czasy obliczeń

Wykres 9. Czasy obliczeń algorytmu Thomasa

Jak widać na tabeli 5. oraz wykresach 8 i 9. czas obliczeń rośnie wraz z rozmiarem układu. Poza nielicznymi odstępstwami, obliczenia metodą Gaussa zajmują podobną ilość czasu. Obliczenia na typie double zajmują więcej czasu niż na typie float, a różnica pomiędzy nimi zwiększa się wraz z rozmiarem układu.

**7. Wnioski**

Eksperymenty pokazały, że wyniki obliczeń na układach o dużej wartości wskaźnika uwarunkowania miały duże wartości średniego błędu, natomiast te dobrze uwarunkowane miały mniejsze, a czasami znikome błędy.

Innym istotnym czynnikiem wpływającym na dokładność obliczeń jest precyzja typu zmiennoprzecinkowego.

Porównanie czasów obliczeń metody Gaussa i algorytmu Thomasa pokazuje, że dobór uproszczonego algorytmu znacznie zmniejsza czas obliczeń, przy praktycznie żadnej stracie na dokładności.