Rozwiązywanie układów równań liniowych metodami bezpośrednimi

Metody Obliczeniowe w Nauce i Technice

Laboratorium 9 i 10

Aleksandra Smela

SPIS TREŚCI

1 UŻYTE NARZĘDZIA I ŚRODOWISKO

- Komputer z systemem Windows 10
- Procesor: AMD Ryzen 7 3700X 3,6GHz
- Pamięć RAM: 32 GB
- Język programowania: Python 3
- Biblioteki: pandas, numpy, time

2 OPIS ZADANIA

Dany jest układ równań liniowych $Ax = b$.

W zadaniach I, II i II zadane są macierze A o wymiarze $n \times n$. Przyjmuję, że wektor x jest dowolną n–elementową permutacją ze zbioru { 1, -1 } i na tej podstawie obliczam wektor b.

Następnie zadaną w każdym zadaniu metodą rozwiązuję układ równań liniowych $Ax = b$ (przyjmując jako niewiadomą wektor x).

Przeprowadzam odpowiednie eksperymenty opisane w każdym z zadań.

2.1 Zadanie I

Elementy macierzy A są określone wzorem:

$$\begin{cases} a_{1j} = 1 \\ a_{ij} = \dfrac{1}{i+j-1} \quad dla \;\; i \neq 1 \end{cases} \qquad i, j = 1, \ldots, n$$

Do rozwiązania układu zastosuję metodę eliminacji Gaussa.

Sprawdzę, jak błędy zaokrągleń zaburzają rozwiązanie dla różnych rozmiarów układu (porównam zgodnie z wybraną normą – wektory: x obliczony z x zadany).

Przeprowadzę eksperymenty dla różnych rozmiarów układu oraz różnej precyzji dla znanych wartości macierzy A i wektora b.

2.2 Zadanie II

Macierz A jest zadana wzorem:

$$\begin{cases} a_{ij} = \dfrac{2i}{j} \quad dla \;\; j \geq i \\ a_{ij} = a_{ji} \quad dla \;\; j < i \end{cases} \qquad i, j = 1, \ldots, n$$

Do rozwiązania układu ponownie zastosuję metodę eliminacji Gaussa.

Porównam wyniki z tym, co otrzymałam w przypadku układu z zadania I. Spróbuję uzasadnić, skąd biorą się różnice w wynikach. Sprawdzę uwarunkowanie obu układów.

2.3 Zadanie III

Elementy macierzy A są określone wzorem:

$$\begin{cases} a_{i,i} = 4 \\ a_{i,i+1} = \frac{1}{i+5} \text{ dla i} < \text{n} \\ a_{i,i-1} = \frac{4}{i+6} \quad dla \; i > 1 \\ a_{i,j} = 0 \quad dla \; j < i-1 \; oraz \; j > i+1 \end{cases} \qquad \text{dla } i, j \; = \; 1, \ldots, n$$

Układ rozwiążę metodą przeznaczoną do rozwiązywania układów z macierzą trójdiagonalną.

Porównam wyniki otrzymane dwoma metodami (czas, dokładność obliczeń i zajętość pamięci) dla różnych rozmiarów układu. Przy porównywaniu czasów pominę czas tworzenia układu.

Opiszę, jak w metodzie dla układów z macierzą trójdiagonalną przechowywałam i wykorzystywałam macierz A.

3 SPOSÓB REALIZACJI ZADANIA

3.1 Wstęp teoretyczny

3.1.1 Metoda eliminacji Gaussa

Metoda eliminacji Gaussa jest bezpośrednią metodą rozwiązywania układów równań liniowych $Ax = b$ i składa się z dwóch etapów:

1) Redukcja macierzy A do macierzy górnotrójkątnej

Aby doprowadzić macierz A do postaci macierzy górnotrójkątnej stosujemy elementarne operacje takie jak: pomnożenie wiersza macierzy przez liczbę różną od 0, dodanie jednego wiersza macierzy do drugiego, zamienienie kolejności dwóch wierszy.

Ten krok algorytmu można zrealizować w złożoności czasowej $O(n^3)$ zerując elementy pod diagonalą w kolejnych wierszach macierzy. Co istotne, należy pamiętać o odpowiednich operacjach również na wektorze b, gdy realizujemy operacje na macierzy A.

2) Obliczanie wartości metodą *backward substitution*

Mając macierz A w postaci górntrójkątnej można obliczyć wartości kolejnych elementów wektora x zaczynając od x_n i wykorzystując wcześniej wyliczone wartości w liczeniu kolejnych x_i zgodnie z algebrą na macierzach. Krok ten jest realizowany w złożoności czasowej $O(n^2)$.

3.1.2 Metoda Thomasa dla układów trójdiagonalnych

Układ trójdiagonalny to układ postaci:

$$\begin{bmatrix} d_1 & c_1 & 0 & & & 0 & 0 & 0 \\ a_2 & d_2 & c_2 & \cdots & & 0 & 0 & 0 \\ 0 & a_3 & d_3 & & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & & \\ 0 & 0 & 0 & & a_{n-1} & d_{n-1} & c_{n-1} & \\ 0 & 0 & 0 & \cdots & 0 & a_n & d_n & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}$$

wzór I

gdzie poza elementami a_i, d_i, c_i *i=1, ..., n* w macierzy współczynników występują jedynie zera.

Aby rozwiązać ten układ można wykorzystać metodę Thomasa. W tym celu układ najpierw należy sprowadzić do postaci:

$$\begin{bmatrix} 1 & \gamma_1 & 0 & & & 0 & 0 & 0 \\ 0 & 1 & \gamma_2 & \cdots & & 0 & 0 & 0 \\ 0 & 0 & 1 & & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & & \\ 0 & 0 & 0 & & 0 & 1 & \gamma_{n-1} & \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \\ \vdots \\ \rho_{n-1} \\ \rho_n \end{bmatrix}$$

wzór II

A następnie zastosować metodę *backward substitution*.

Wartości γ_i oraz ρ_i liczę zgodnie ze wzorami:

$$\gamma_i = \begin{cases} \dfrac{c_i}{d_i} \; dla \; i = 1 \\ \dfrac{c_i}{d_i - a_i\gamma_{i-1}} \; dla \; i = 2, \ldots, n-1 \end{cases}$$

$$\rho_i = \begin{cases} \dfrac{b_i}{d_i} \; dla \; i = 1 \\ \dfrac{b_i - a_i\rho_{i-1}}{d_i - a_i\gamma_{i-1}} \; dla \; i = 2, \ldots, n \end{cases}$$

Algorytm Thomasa to uproszczenie metody eliminacji Gaussa, które wykorzystuje informację o tym, że macierz współczynników jest trójdiagonalna. Dzięki temu oba etapy algorytmu działają w czasie liniowym:

1) wyliczenie współczynników z wzoru II według wzorów III i IV,
2) zastosowanie metody *backward substitution* – dzięki temu, że macierz ze wzoru II ma w każdym wierszu zera poza maksymalnie dwiema kolumnami sprawia, że wyliczanie kolejnych wartości z wektora x zachodzi w czasie liniowym.

Ponadto macierz trójdiagonalna jest rzadka i ma bardzo specyficzną strukturę. Dzięki temu nie trzeba jej przechowywać w dwuwymiarowej strukturze (np. tablicy) rozmiaru $n \times n$, tak jak macierze gęste. Wystarczy zapamiętać wartości na trzech przekątnych w liniowych strukturach rozmiaru *n*. W rezultacie złożoność pamięciowa również jest liniowa.

3.1.3 Zastosowane normy do oceny jakości rozwiązań

Mając dwa wektory x: zadany (x^{zad}) oraz obliczony (x^{ob}) można policzyć wskaźnik, który określi jak bardzo się różnią. Zastosowałam w tym celu normę euklidesową wyliczoną zgodnie z wzorem:

$$\sqrt{\sum_{i=0}^{n} (x_i^{zad} - x_i^{ob})^2}$$

gdzie n to rozmiar wektorów.

Poza tym policzyłam również błąd maksymalny, czyli różnicę między poszczególnymi liczbami z wektorów o największej wartości bezwzględnej.

$$max_{i=0,\ldots,n} |x_i^{zad} - x_i^{ob}|$$

3.1.4 Wskaźnik uwarunkowania macierzy

Wskaźnik uwarunkowania macierzy definiuje się jako maksymalny stosunek błędu względnego wektora rozwiązania *x* do błędu względnego *b*. Wartość tego wskaźnika informuje o tym w jakim stopniu błąd reprezentacji numerycznej wpływa na błąd wyniku. Problemy o wysokim wskaźniku uwarunkowania są źle uwarunkowane, natomiast te o niskim – dobrze uwarunkowane.

Wskaźnik uwarunkowania macierzy został obliczony ze wzoru:

$$cond(A) = \|A^{-1}\| \cdot \|A\|$$

Gdzie norma $\|\cdot\|$ jest zdefiniowana jako: $\|A\| = max_{1<i<n} \sum_{j=1}^{n} |a_{ij}|$

3.2 Szczegóły techniczne

3.2.1 Sposób przechowywania macierzy

Wektory x i b były przechowywane w postaci jednowymiarowej tablicy o rozmiarze n. Macierz A wykorzystywana do metody eliminacji Gaussa była przechowywana w postaci dwuwymiarowej tablicy o rozmiarze $n \times n$. Do metody Thomasa macierz A była przechowywana w trzech tablicach jednowymiarowych o rozmiarze n lub $n-1$.

n to liczba równań w układzie równań $Ax=b$.

3.2.2 Precyzja zmiennych

Wykorzystano funkcje z biblioteki numpy aby określić precyzję liczb w macierzy A i b w zadaniu I i II:

- np.float16()
- np.float32()
- np.float64()

Funkcje te odpowiednio zapewniają szesnasto-, trzydziestodwu- i sześćdziesięciocztero- bitową precyzję.

4 WYNIKI I ICH ANALIZA

Uwaga: w poniższych tabelach MAX oznacza błąd maksymalny, a EUK normę euklidesową zgodnie z tym co zostało opisane w punkcie 3.1.3.

4.1 Zadanie I

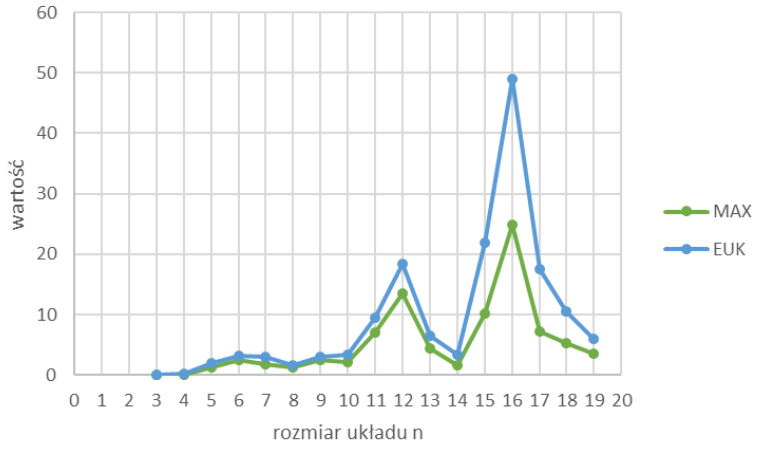
4.1.1 Wyniki

Dla układu z zadania I przeprowadzono testy dla 23 różnych rozmiarów układu (n) oraz każdej z trzech precyzji opisanych w punkcie 3.2.2. Wyniki przedstawiono w tabeli I.

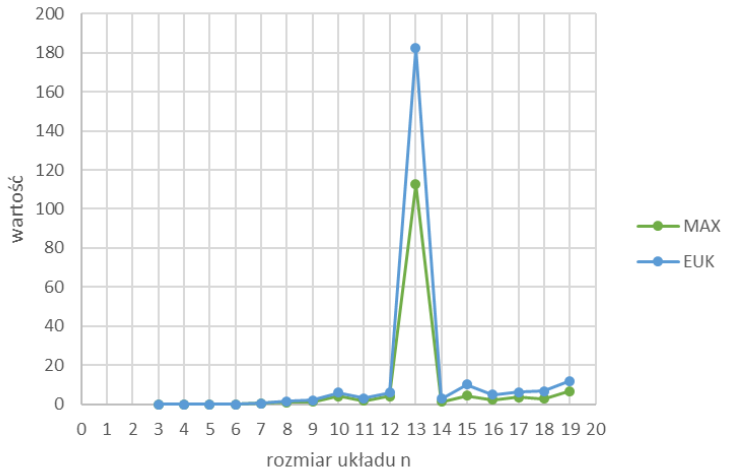
n	np.float16		np.float32		np.float64	
	MAX	EUK	MAX	EUK	MAX	EUK
3	1.05E-05	1.314E-05	8.94E-07	1.115E-06	0	0
4	0.11226	0.14341	1.04E-05	1.334E-05	3.00E-13	3.831E-13
5	1.29538	1.89505	8.89E-05	1.217E-04	4.39E-12	6.272E-12
6	2.40440	3.24048	2.49E-03	3.569E-03	3.39E-10	4.951E-10
7	1.81189	2.93623	0.44238	0.63786	3.96E-09	5.713E-09
8	1.34930	1.62892	1.06734	1.66639	3.53E-08	5.298E-08
9	2.54397	3.04259	1.20176	2.08302	5.05E-06	8.215E-06
10	2.15053	3.42959	4.14841	5.84776	7.90E-05	1.245E-04
11	7.04635	9.53820	1.76427	3.12421	1.82E-05	2.876E-05
12	13.56237	18.41698	4.08978	6.10431	0.34111	0.57558
13	4.39157	6.54020	112.66356	182.39976	5.55624	9.40796
14	1.67000	3.40372	1.40678	2.75689	0.37366	0.63070
15	10.20406	21.86908	4.37566	10.10397	4.15633	6.70252
16	24.92401	49.03512	2.31774	4.84486	1.61105	3.16696
17	7.21866	17.50773	3.69892	6.34561	1.94144	3.86983
18	5.20789	10.44980	2.85368	6.73426	22.80902	43.64588
19	3.54820	5.97064	6.85723	11.85719	37.28203	82.07795
50	8.79037	22.29948	1.89E+04	6.393E+04	52.76327	130.71137
100	4.10689	17.05057	65.12101	180.57236	143.77647	537.51052
150	19.81233	64.63227	22.53539	93.63995	3.61E+02	1.465E+03
200	4.49173	21.53608	43.30639	144.37917	5.01E+02	2.611E+03
300	89.05831	265.22651	23.41565	97.06817	4.55E+02	1.910E+03
500	52.72751	214.80696	1.06E+02	4.632E+02	3.52E+02	1.911E+03

Tabela I: Wyniki obliczeń dla zadania I

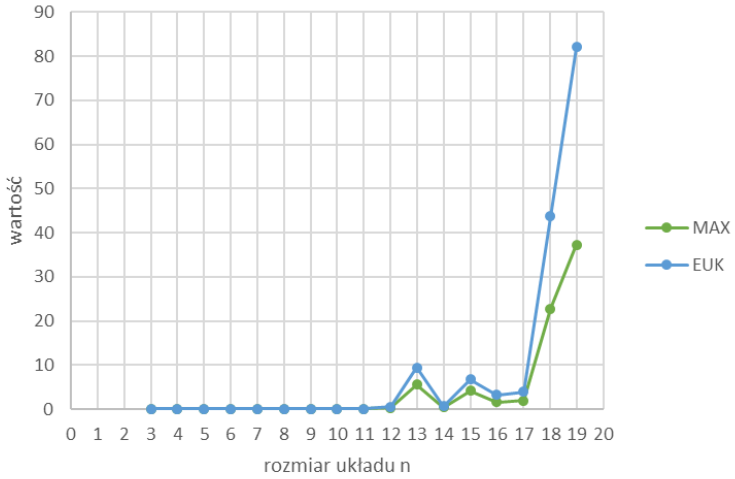
Stworzono również wykresy dla wyników eksperymentów, gdzie *n=3,..., 19.*



*Wykres I: Wyniki obliczeń dla zadania I i precyzji **np.float16***



Wykres II: Wyniki obliczeń dla zadania I i precyzji ***np.float32***



Wykres III: Wyniki obliczeń dla zadania I i precyzji ***np.float64***

4.1.2 Analiza i wnioski

Można zauważyć, że jakość rozwiązania spada wraz z rosnącym rozmiarem układu. Co można łatwo zobaczyć na wykresach I-III tendencja ta nie jest liniowa.

Rozważmy najpierw wyniki obliczeń dla n=3,…, 19.

Co istotne, błędy są różne w zależności od precyzji. Nie tylko ich wartości się różnią, ale również kształt wykresu, czyli sposób w jaki się zmieniają wraz z zmianą rozmiaru układu.

Dla precyzji np.float16 błędy bardzo szybko zaczynają przekraczać 1 – już dla n=5. Maksima są osiągane dla n=12 oraz n=16. W przypadku normy euklidesowej są to rzędu kilkudziesięciu, ale w granicach 50.

Dla precyzji np.flat32 błędy przekraczają wartość 1 dla n=8, co jest lepszym wynikiem niż w przypadku np.float16. W przypadku tej precyzji maksimum jest osiągnięte dla n=13 i osiąga bardzo duże wartości błędów. Odległość euklidesowa wynosi ponad 180, a błąd maksymalny około 112.

Z precyzją np.64 osiągnięto najlepsze wyniki. Wartość błędów 1 jest przekraczana dopiero dla n=13, a maksimum jest osiągnięte w ostatnim analizowanym przypadku – czyli dla n=19.

Obliczenia dla n równego 50 i więcej dały wyniki o bardzo dużych błędach rzędu kilku dziesiątek, setek, a dla niektórych przypadków również tysięcy.

Analiza wyników obliczeń pozwoliła ocenić jakość rozwiązań dla układu z zadania I jako bardzo słabą.

4.2 Zadanie II

4.2.1 Wyniki

Dla układu z zadania II przeprowadzono analogiczne testy jak w zadaniu I. Wyniki zostały przedstawione w tabeli II.

n	np.float16		np.float32		np.float64	
	MAX	EUK	MAX	EUK	MAX	EUK
3	2.93E-04	3.613E-04	9.54E-08	1.271E-07	2.22E-16	3.14E-16
4	5.86E-04	7.90E-04	4.77E-08	6.722E-08	8.88E-16	1.071E-15
5	1.29E-03	1.65E-03	9.04E-08	1.370E-07	4.44E-16	6.474E-16
6	1.38E-03	2.22E-03	3.12E-07	4.095E-07	2.00E-15	2.427E-15
7	1.71E-03	3.06E-03	1.75E-06	2.176E-06	2.22E-15	3.128E-15
8	3.52E-03	5.19E-03	5.07E-07	7.903E-07	4.44E-16	9.805E-16
9	6.11E-03	1.02E-02	1.47E-06	2.062E-06	1.33E-15	2.011E-15
10	8.88E-03	1.62E-02	1.71E-06	2.960E-06	2.00E-15	3.706E-15
11	1.58E-02	2.25E-02	2.63E-06	3.703E-06	4.88E-15	7.323E-15
12	1.60E-02	3.00E-02	2.39E-06	4.177E-06	1.13E-14	1.771E-14
13	2.37E-02	3.42E-02	4.56E-06	6.017E-06	9.10E-15	1.377E-14
14	3.46E-02	5.65E-02	5.58E-06	7.907E-06	6.99E-15	1.441E-14
15	1.87E-02	3.87E-02	3.62E-06	5.618E-06	9.77E-15	1.893E-14
16	3.12E-02	6.10E-02	4.57E-06	8.309E-06	8.66E-15	1.529E-14
17	4.77E-02	7.65E-02	6.64E-06	1.093E-05	4.88E-15	9.490E-15
18	4.36E-02	8.59E-02	6.72E-06	1.105E-05	1.69E-14	2.456E-14
19	0.06923	0.13541	4.56E-06	8.067E-06	1.75E-14	4.428E-14
50	0.45825	0.72508	6.30E-05	1.015E-04	1.02E-13	2.501E-13
100	1.55717	5.25045	2.23E-04	6.383E-04	6.37E-13	1.677E-12
150	19.11297	59.02199	4.96E-04	1.786E-03	1.42E-12	3.975E-12
200	28.53031	104.00276	8.04E-04	3.118E-03	5.63E-12	1.547E-11
300	135.80316	722.31413	1.91E-03	9.285E-03	4.70E-12	2.211E-11
500	2.80E+01	1.727E+02	8.19E-03	3.623E-02	1.56E-11	8.159E-11

Tabela II: Wyniki obliczeń dla zadania II

4.2.2 Analiza i wnioski

Analizując wyniki zestawione w tabeli II można zauważyć, że wykonane obliczenia dały stosunkowo dobre rezultaty. Ponownie wraz ze wzrostem rozmiaru układu, spada jakość rozwiązań. Tendencja ta jest jednak silnie monotoniczna i nie obserwujemy „skoków" jak w przypadku tych z zadania I.

Błędy znacznie różnią się w zależności od precyzji obliczeń, ale istnieje zależność – im większa precyzja tym mniejsze błędy. Co więcej wyliczone błędy dla n od 3 do 19 są podobnego rzędu co precyzja wartości w macierzac układu.

Dla n równego 50 i więcej jakość rozwiązań bardzo zależy od precyzji obliczeń. Wyniki te są znacząco gorsze od tych dla mniejszych rozmiarów układu, ale dla precyzji np.float64 wciąż są bardzo dobre – błędy rzędu 10^{-11}. Różnież dla precyzji np.float32 wyniki są akceptowalne – błędy rzędu 10^{-3}. Jednak dla precyzji np.float16 wyniki dla dużych rozmiarów układu mają bardzo duże błędy rzędu kilkudziesięciu lub nawet kilkuset. Różnice te pokazują jak bardzo precyzja zmiennych wpływa na jakość obliczeń.

4.3 Porównanie wyników z zadania I i II

n	Zadanie I	Zadanie II
3	186.00	9.53
4	4800.14	17.61
5	118084.92	28.54
6	2.87E+06	42.50
7	5.59E+07	59.66
8	1.77E+08	80.14
9	3.36E+08	104.07
10	3.83E+08	131.54
11	6.34E+08	162.67
12	1.71E+09	197.52
13	1.47E+10	236.19
14	1.50E+10	278.75
15	1.93E+10	325.27
16	1.93E+10	375.82
17	1.94E+10	430.44
18	1.96E+10	489.21
19	1.97E+10	552.18
50	1.10E+13	4798.04
100	1.13E+13	22293.56
150	1.18E+13	5.44E+04
200	1.68E+13	1.02E+05
300	1.77E+13	2.47E+05
500	3.11E+13	7.49E+05

Tabela III: Wskaźniki uwarunkowania układów

Różnice w jakości wyników otrzymanych dla układów z zadań I i II pozwalają podejrzewać, że istnieje pewien problem z układem I. W celu jego zidentyfikowania można policzyć wskaźniki uwarunkowania zgodnie z wstępem teoretycznym z punktu 3.1.4. Obliczenia zostały przeprowadzone na precyzji np.float32 i przedstawione w tabeli III.

Można zauważyć, że wskaźnik uwarunkowania dla układu z zadania I jest wielokrotnie większy od wskaźnika dla układu II. Co więcej wskaźnik ten jest tym większy im większa jest macierz.

Jak widać układ z zadania I jest o wiele gorzej uwarunkowany niż ten z zadania II. Wyjaśnia to duże błędy w rozwiązaniach zadania I oraz ich skokową tendencję.

Jedną z przyczyn prawdopodobnie jest sposób wyboru elementu wiodącego, przez który dzielimy poszczególne wiersze w metodzie Gaussa. Jeżeli moduł tego elementu jest bardzo mały (<1), to każdy wiersz jest mnożony przez dużą wartość (bo odwrotność tego elementu jest duża). Takie operacje i nakładające się na siebie błędy zaokrągleń skutkują bardzo niską jakością wyników. Natomiast macierz A z zadania II jest lepiej wyważona, w związku z czym błędy nie są tak duże.

4.4 Zadanie III

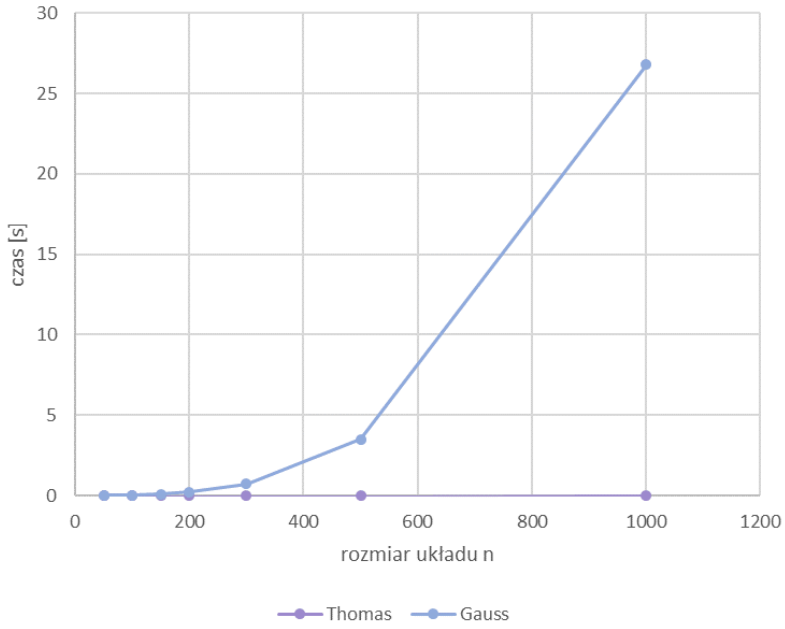
4.4.1 Wyniki

Dla układu z zadania II przeprowadzono testy dla 24 różnych rozmiarów układu. Oprócz zmierzenia jakości wyników poprzez zastosowanie norm opisanych w punkcie 3.1.3, zmierzono również czas działania algorytmu. Wyniki przedstawiono w tabeli IV.

n	EUK		MAX		TIME	
	Thomas	Gauss	Thomas	Gauss	Thomas	Gauss
3	0	0	0	0	0	0
4	2.48E-16	2.48E-16	2.22E-16	2.22E-16	0	0
5	2.22E-16	3.33E-16	2.22E-16	2.22E-16	0	0
6	2.72E-16	2.72E-16	2.22E-16	2.22E-16	0	0
7	2.72E-16	4.30E-16	2.22E-16	2.22E-16	0	0
8	2.48E-16	3.33E-16	2.22E-16	2.22E-16	0	0
9	3.14E-16	3.85E-16	2.22E-16	2.22E-16	0	0
10	2.22E-16	4.00E-16	2.22E-16	2.22E-16	0	0
11	1.11E-16	3.14E-16	1.11E-16	2.22E-16	0	0
12	4.15E-16	4.44E-16	2.22E-16	2.22E-16	0	0
13	2.94E-16	4.84E-16	2.22E-16	2.22E-16	0	0
14	3.51E-16	5.44E-16	2.22E-16	2.22E-16	0	0.0010
15	3.85E-16	4.97E-16	2.22E-16	2.22E-16	0	0
16	3.68E-16	5.44E-16	2.22E-16	2.22E-16	0	0
17	4.97E-16	4.58E-16	2.22E-16	2.22E-16	0	0
18	5.32E-16	5.98E-16	2.22E-16	2.22E-16	0	0.0010
19	4.84E-16	5.09E-16	2.22E-16	2.22E-16	0	0
50	6.84E-16	8.46E-16	2.22E-16	2.22E-16	0	0.0040
100	1.29E-15	1.26E-15	2.22E-16	2.22E-16	0	0.0270
150	1.31E-15	1.62E-15	2.22E-16	2.22E-16	0	0.0860
200	1.65E-15	1.80E-15	3.33E-16	3.33E-16	0	0.2240
300	1.94E-15	2.24E-15	3.33E-16	2.22E-16	0	0.7142
500	2.54E-15	2.81E-15	4.44E-16	4.44E-16	0	3.5038
1000	3.40E-15	3.91E-15	3.33E-16	4.44E-16	0.0010	26.8080

Tabela IV: Wyniki obliczeń dla zadania III

Czas realizacji obliczeń dla *n* od 50 do 1000 przedstawiono również na wykresie IV.



Wykres IV: Czas działania algorytmu Thomasa i eliminacji Gaussa

4.4.2 Analiza i wnioski

Można zauważyć, że niezależnie od obranej metody błędy są bardzo podobne, szczególnie dla mniejszych układów. W szczególności błędy są tego samego rzędu.

To co znacząco odróżnia te dwa algorytmy to czas działania – szczególnie dla dużych rozmiarów układów. Dla przeprowadzonych testów algorytm Thomasa nie przekroczył 0.001 sekundy, gdy algorytm eliminacji Gaussa potrzebował ponad 26 sekund. Co istotne czas działania algorytmu eliminacji Gaussa wzrasta tym szybciej im większy jest układ. Zależność tę można zauważyć na wykresie IV.

Takie wyniki czasowe są spodziewane w związku z różną złożonością czasową algorytmów. Dla algorytmu eliminacji jest to $O(n^3)$, a dla algorytmu Thomasa $O(\mathrm{n})$, co zostało szerzej omówione w wstępie teoretycznym.

Warto również wspomnieć o różnicy w zajmowanej pamięci – zgodnie z opisem w punkcie 3.2.1. algorytm Thomasa ma liniową złożoność pamięciową, algorytm eliminacji Gaussa – kwadratową.

5 PODSUMOWANIE

Wyniki zadań I i II pokazują w jaki sposób uwarunkowanie zadania i precyzja zmiennych wpływa na wyniki. Przeprowadzone testy dowodzą, że błędy związane z ograniczoną dokładnością reprezentacji liczb w komputerze i precyzją operacji arytmetycznych nie mogą być pomijane.

Zadanie III pozwala wysunąć wniosek, że specyficzny typ danych może znacząco wpłynąć na możliwość poprawy złożoności sposobu wyznaczania rozwiązania. Chociaż metoda Gaussa jest skuteczna i prosta, to niektóre układa można rozwiązać szybciej i z wykorzystaniem mniejszej ilości pamięci bez utraty dokładności rozwiązania.

6 BIBLIOGRAFIA

[1] Wykłady dr Katarzyny Rycerz;
[2] www.industrial-maths.com/ - strona z materiałami prof. Williama Lee z University of Huddersfield i University of Limerick