**Algorytmy macierzowe**

**Laboratorium 2**

**Sprawozdanie**

**Łukasz Stępień, Szymon Urbański**

1. **Temat zadania**

Laboratorium polegało na zaimplementowaniu i przetestowaniu rekurencyjnego algorytmu odwracania, faktoryzacji LU oraz obliczania wyznacznika macierzy. Należało również narysować wykres zależności czasu i ilości obliczeń od rozmiarów macierzy oraz określić złożoność obliczeniową tych algorytmów.

1. **Rozwiązanie**

**2.1. Rekurencyjne odwracanie macierzy**

Zaimplementowano rekurencyjne odwracanie macierzy.

* **Pseudokod algorytmu:**

inverse(A):

if A jest w wymiarach 2x2:

return odwrócone A (według definicji)

Podziel macierz A na macierze A11, A12, A21, A22 (2)

A11\_inv = inverse(A11) (3)

S22 = A22 - A21 \* A11\_inv \* A12

S22\_inv = inverse(S22)

C1 = A11\_inv \* A12

C2 = S22\_inv \* A21 \* A11\_inv

B11 = A11\_inv + C1 \* C2

B12 = - C1 \* S22\_inv

B21 = - C2

B22 = S22\_inv

Połącz B11, B12, B21, B22 (4)

Return B

* **Złożoność obliczeniowa:**

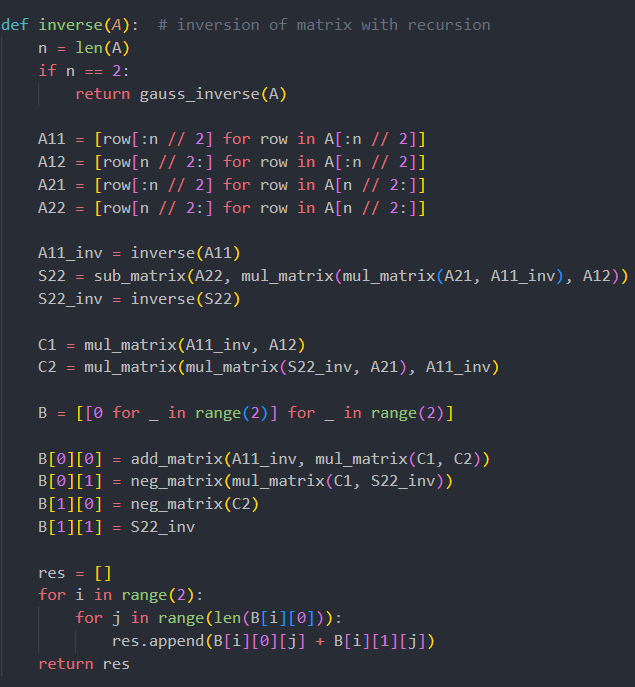
Niech T(n) oznacza czas procedury *inverse* dla dwóch macierzy *n*x*n*.

1. W przypadku bazowym (*n=2*), wykonujemy elementarną operacje odwracania, więc:
2. Zakładamy, że podział wykonuje się w stałym czasie .
3. Wykonujemy łącznie 2 wywołania rekurencyjne procedury *inverse*. Ponieważ w każdym tym wywołaniu argumentem są macierze x , co wnosi do łącznego czasu działania, czas tych dwóch wywołań to . Uwzględniamy 7 mnożeń macierzy zawierających elementów. Każde z siedmiu mnożeń wymaga , a ich liczba jest stała, łączny czas ich wykonywania to . Następnie bierzemy pod uwagę 4 działania addytywne. Każde z nich wymaga , a ich liczba jest stała, więc łączny czas ich wykonywania to .
4. Zakładamy, że łączenie wykonuje się w stałym czasie .

Rekurencja opisująca czas działania algorytmu odwracania macierzy przedstawia się równaniem:

Rozwiązaniem tej rekurencji jest .

* **Najważniejsze fragmenty kodu:**



**2.2. Faktoryzacja LU**

* **Pseudokod algorytmu:**

LU(A):

if A jest w wymiarach 2x2:

return sfaktoryzowane A (według definicji)

Podziel macierz A na macierze A11, A12, A21, A22 (2)

L11, U11 = LU(A11) (3)

U11\_inv = inverse(U11)

L21 = A21 \* U11\_inv

L11\_inv = inverse(L11)

U12 = L11\_inv \* A12

S = A22 - A21 \* U11\_inv \* L11\_inv \* A12

L22, U22 = LU(S)

Połącz L11, L21, L22 (4)

Połącz U11, U12, U22

Return L, U

* **Złożoność obliczeniowa:**

Niech T(n) oznacza czas procedury *LU* dla dwóch macierzy *n*x*n*.

1. W przypadku bazowym (*n=2*), wykonujemy operacje elementarną, więc:
2. Zakładamy, że podział wykonuje się w stałym czasie .
3. Wykonujemy łącznie 2 wywołania rekurencyjnej procedury *LU(A)*. Ponieważ w każdym tym wywołaniu argumentem są macierze x , co wnosi do łącznego czasu działania, czas tych dwóch wywołań to . Uwzględniamy pięć mnożeń oraz 2 odwracania macierzy. zawierających elementów. Łączny czas ich wykonywania to .
4. Zakładamy, że łączenie wykonuje się w stałym czasie .

Rekurencja opisująca czas działania algorytmu faktoryzacji LU przedstawia się równaniem:

Rozwiązaniem tej rekurencji jest

* **Najważniejsze fragmenty kodu:**

**A screenshot of a computer program

Description automatically generated**

**2.3. Obliczanie wyznacznika macierzy**

Zaimplementowano obliczanie wyznacznika macierzy.

* **Pseudokod algorytmu:**

determinant(A):

L, U = LU(A) (1)

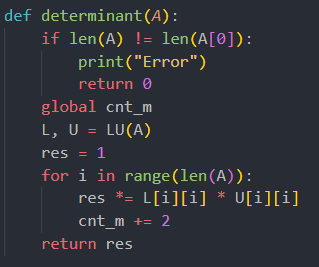
return suma iloczynów liczb występujących na diagonalach macierzy (2)

* **Złożoność obliczeniowa:**

1. Koszt faktoryzacji LU to .
2. Kosz sumowania iloczynów to .

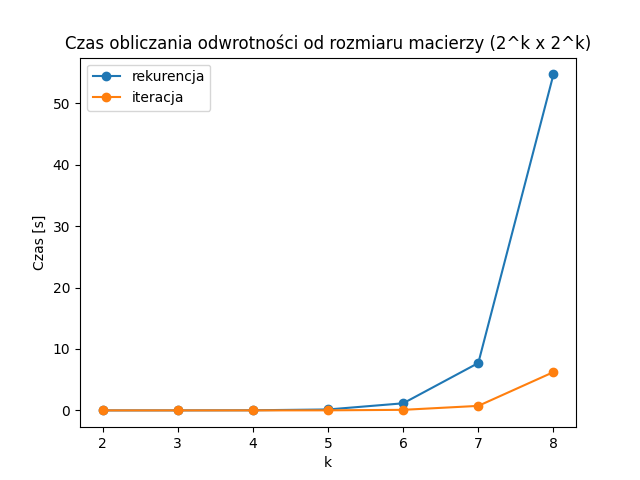
Zatem złożoność obliczeniowa obliczania wyznacznika macierzy to .

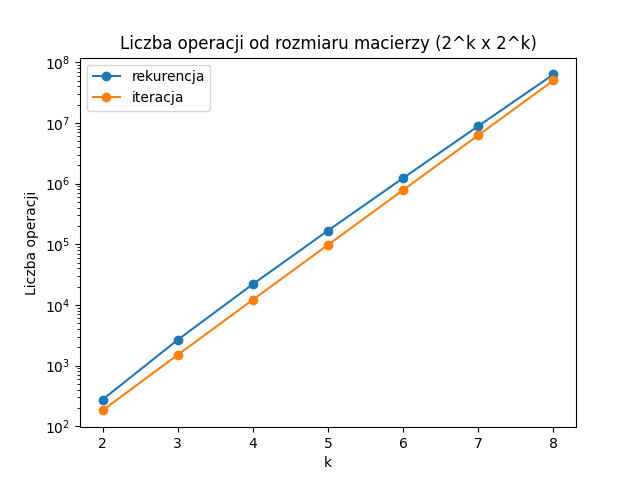
* **Najważniejsze fragmenty kodu:**

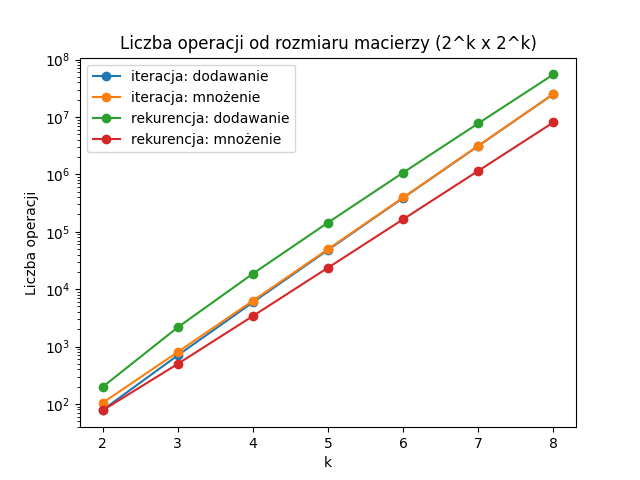


1. **Wyniki**

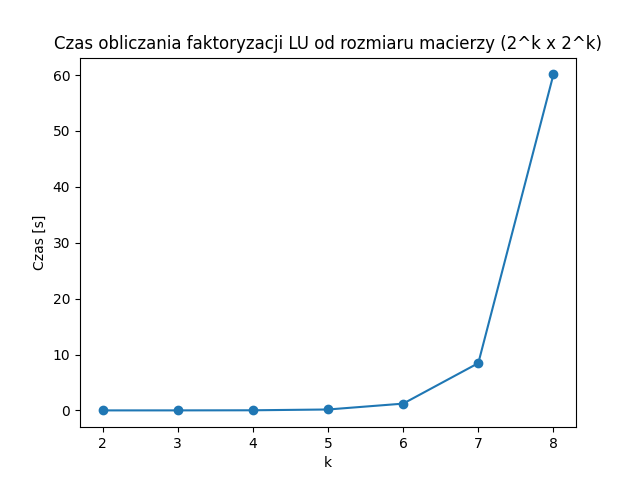
**3.1. Rekurencyjne odwracanie macierzy**

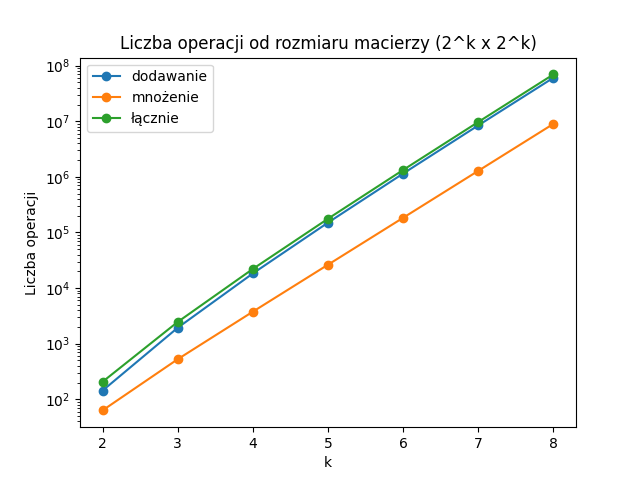
Na poniższym wykresie przedstawiono porównanie czasu działania rekurencyjnego oraz iteracyjnego algorytmu odwracania macierzy w zależności od ilości elementów.

Wygenerowano wykres przedstawiający liczbę wykonanych operacji arytmetycznych w zależności od wielkości macierzy dla obu wariantów algorytmu.

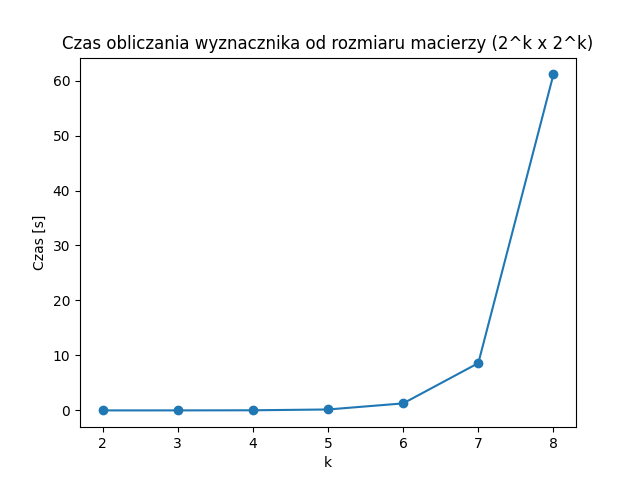
Wygenerowano wykres przedstawiający liczbę wykonanych operacji arytmetycznych z podziałem na dodawanie i odejmowanie w zależności od wielkości macierzy.

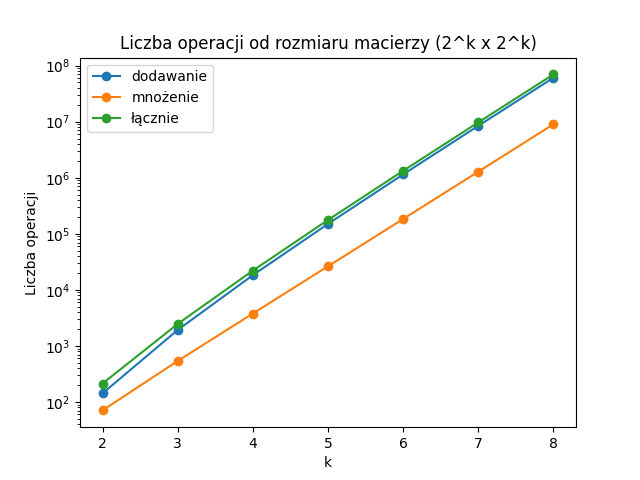
**3.2. Faktoryzacja LU**

Na poniższym wykresie przedstawiono czas działania algorytmu faktoryzacji LU macierzy w zależności od ilości elementów.

Wygenerowano wykres przedstawiający liczbę wykonanych operacji arytmetycznych w zależności od wielkości macierzy.

**3.3. Obliczanie wyznacznika macierzy**

Na poniższym wykresie przedstawiono czas działania algorytmu obliczającego wyznacznik macierzy w zależności od wielkości macierzy.

Wygenerowano wykres przedstawiający liczbę wykonanych operacji arytmetycznych w zależności od wielkości macierzy.

**3.4. Testy w MATLAB**

Przetestowano działanie algorytmów z tymi zaimplementowanymi w programie MATLAB.

Dla wygenerowanej macierzy 4x4:

zaimplementowane algorytmy zwróciły następujące wyniki:

* odwracanie macierzy:

* faktoryzacja LU
* obliczanie wyznacznika

1. **Wnioski**

Wykres przedstawiający porównanie czasu działania rekurencyjnego oraz iteracyjnego algorytmu odwracania macierzy w zależności od ilości elementów pozwala stwierdzić, że o wiele lepiej radzi sobie metoda iteracyjna, pomimo gorszej złożoności obliczeniowej. NA kolejnych wykresach można zaobserwować również większą liczbę działań arytmetycznych w algorytmie rekurencyjnym. Jednakże w algorytmie rekurencyjnym widzimy przeważającą ilość dodawań i odejmowań niż mnożeń i dzieleń, co usprawnia ten algorytm pod względem numerycznym.

Następne algorytmy bazują na rekurencyjnym odwracaniu macierzy, dlatego reszta wykresów jest podobna do tych pierwszych.

Analiza wyników działania zaimplementowanych algorytmów oraz tych zaimplementowanych w MATLAB pozwala wnioskować o poprawności tych pierwszych. Wyniki są praktycznie identyczne z dokładnością do szumu numerycznego.

1. **Bibliografia**

* wykład z przedmiotu „Algorytmy macierzowe” przygotowany przez prof. dr hab. Macieja Paszyńskiego
* <https://en.wikipedia.org/wiki/LU_decomposition>
* Thomas H. Cormen - „Wprowadzenie do algorytmów”