

Parametry:*liczba\_prob* = 4*liczba\_rozmiarow* = 22**Raport z Pracowni nr 1****Zadanie 1.**1. Cel zadania

Celem zadania było zbadanie złożoności obliczeniowej algorytmów rozwiązywania układów Cramerowskich za pomocą metody iteracji prostej.

2. Metody

W doświadczeniu wykorzystano kilka klas stworzonych w języku Python. Odpowiedni projekt stworzono w środowisku Visual Studio Code i kompilowano w zintegrowanym ze środowiskiem terminalu PowerShell na komputerze personalnym o procesorze AMD Ryzen 7 3700X.

3. Przebieg doświadczenia i wyniki

Doświadczenie rozpoczęto od ustalenia maksymalnego rozmiaru macierzy.

Przyjęto  $n = 1111$ , gdzie dla macierzy o tym rozmiarze czas, w jakim algorytm określił przybliżenie o różnicy dwóch ostatnich norm mniejszej od  $\epsilon = 1e-10$  wyniósł około 0.92s.

Opracowano metodę *mierz\_czas*, która pozwala obliczyć czas określania przybliżenia rozwiązań układu metodą iteracji prostej. Metoda wywoływana była w instancji klasy Zadanie o parametrach konstruktora  $n = 1111$ ,  $M = 4$ ,  $N = 22$ . Poniżej zamieszczono kod (zawierający jednocześnie algorytmy służące do pomiaru czasu pracy zarówno algorytmu opartego na metodzie iteracji prostej, jak i iteracji Seidela) tej metody:

```
def mierz_czas(self, metoda, k):
```

```
    """Metoda mierzaca czas rozwiazywania problemu wybrana metoda
```

```
        k - rozmiar macierzy"""
```

```
        czas = 0.0
```

```
        macierzA = ukklad.Uklad(wymiar = k)
```

```

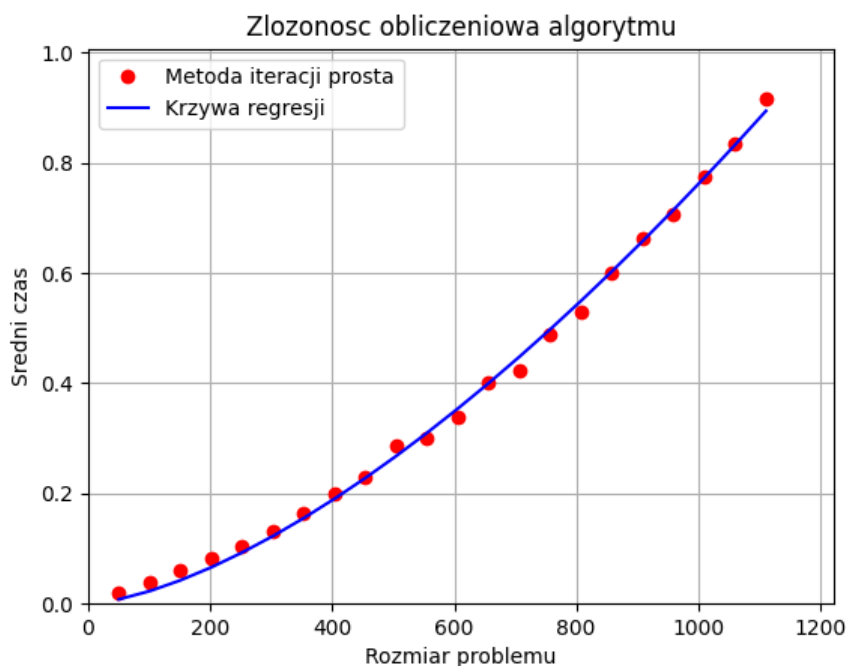
if metoda == 1:
    pomiary = 0
    while pomiary < self.M:
        macierzA.losuj_uklad()
        macierzAprosta = iteracjaprosta.IteracjaProsta(macierzA)
        stoper = time.time()
        if(macierzAprosta.przygotuj() == 1):
            macierzAprosta.iteruj_roznica(eps = 1e-10, norma = 0,
wyswietlaj = 0, X0 = None)
            czas += time.time() - stoper
            pomiary += 1

if metoda == 2:
    pomiary = 0
    while pomiary < self.M:
        macierzA.losuj_uklad()
        macierzASeidela = iteracjaseidela.IteracjaSeidela(macierzA)
        stoper = time.time()
        if(macierzASeidela.przygotuj() == 1):
            macierzASeidela.iteruj_roznica(eps = 1e-10, norma = 0,
wyswietlaj = 0, X0 = None)
            czas += time.time() - stoper
            pomiary += 1

return czas/self.M

```

W wyniku wykonania kodu zawartego w metodzie otrzymano wyniki pomiarów czasu działania algorytmu dla macierzy o coraz większym rozmiarze, aż po  $n = 1111$ , zilustrowane na wykresie załączonym poniżej:



Wykres 1. Zależność czasu obliczania przybliżenia metodą iteracji prostej od rozmiaru macierzy

Empiryczna złożoność obliczeniowa wyniosła:  $n^{1,528}$ , co jest ponadprzeciętnie oddalone od teoretycznej wartości  $n^2$ .

#### 4. Wnioski

W wyniku przeprowadzonego eksperymentu udało się słabo oszacować złożoność obliczeniową algorytmu iteracji prostej. Otrzymana eksperymentalna złożoność jest w otoczeniu teoretycznej wartości  $O(n^2)$ , lecz jednocześnie w zauważalnym stopniu od niej oddalona.

### Zadanie 2.

#### 1. Cel zadania

Celem zadania było porównanie dwóch metod obliczania wektorów rozwiązań macierzy - metodą iteracji prostej oraz metodą iteracji Seidela.

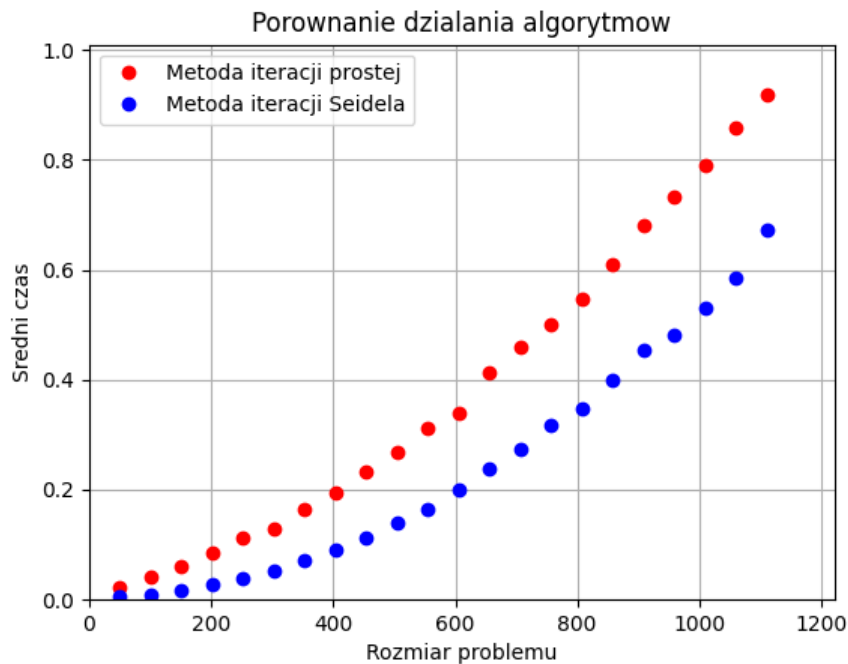
#### 2. Metody

W doświadczeniu wykorzystano kilka klas stworzonych w języku Python. Odpowiedni projekt stworzono w środowisku Visual Studio Code i skompilowano w zintegrowanym ze środowiskiem terminalu PowerShell na komputerze personalnym o procesorze AMD Ryzen 7 3700X.

#### 3. Przebieg doświadczenia i wyniki

Wykorzystano maksymalny rozmiar macierzy ustalony w zadaniu 1. Ze względu na

to, że w zadaniu 1 opracowano metodę *mierz\_czas* w ten sposób, by umożliwiła zastosowanie obu metod iteracyjnych można ją było wykorzystać w tym zadaniu. Po wywołaniu metody *porownaj\_metody* dla obiektu klasy *Zadanie* powołanego do istnienia z parametrami  $n = 1111$ ,  $M = 4$ ,  $N = 22$  udało się uzyskać wykres zamieszczony poniżej.



Wykres 2. Porównanie metod iteracyjnych  
(czerwone punkty - metoda iteracji prostej,  
niebieskie - metoda iteracji Seidela)

#### 4. Wnioski

W wyniku przeprowadzonego doświadczenia okazało się, że metoda iteracji Seidela jest szybsza niż metoda iteracji prostej - różnice czasów dla małych macierzy są niewielkie, ale rosną wraz ze wzrostem rozmiaru macierzy.