A=1, B=2, C=1, D=3

.... --------------------------------

(Imię i nazwisko) (A, B, C, D)

Parametry:
M = 2
N = 12
norma = 1

Raport z Pracowni nr 2

Zadanie 1.

1. Cel zadania

Celem zadania było zbadanie jak zbieżność iteracji Seidla zależy od parametru *n* - wielkości macierzy przy normie 1, korzystając z metody iteruj().

2. Metody

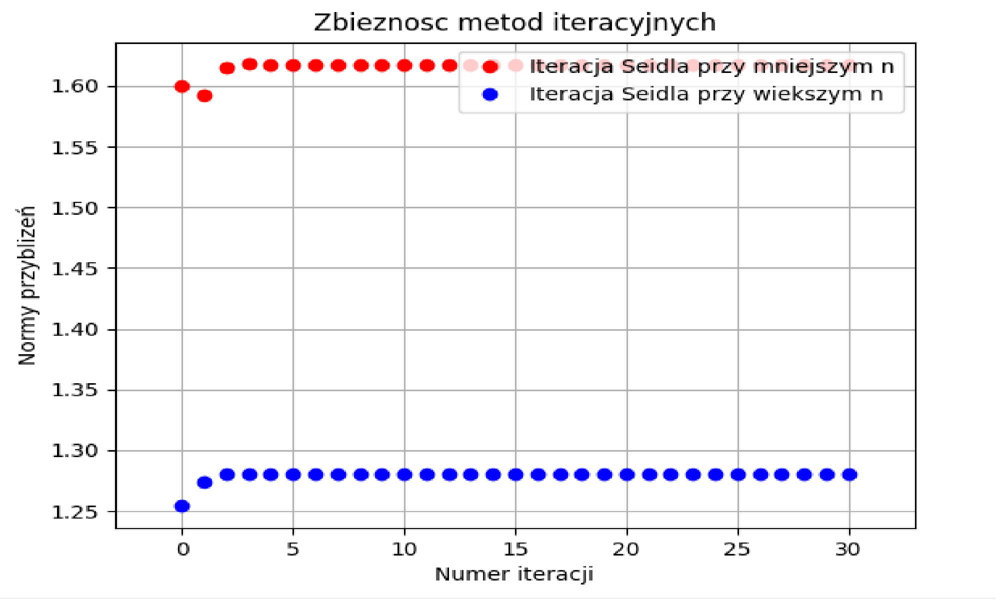
Zadanie rozwiązywano z wykorzystaniem środowiska VSCode zainstalowanym na komputerze stacjonarnym o procesorze Intel Core i5-9600k, za pomocą klas udostępnionych w ramach zajęć z kursu Metody numeryczne, analizę danych wykonano w programie Excel

3. Przyjęte parametry

- *eps* = 1.0E-10 – parametr stopu
- *alfa* = 0.3 – parametr metody losującej układ
- *k* = 200 – liczba iteracji
- *Norma* = 1

4. Przebieg doświadczenia i wyniki

Dla wybranych parametrów przeprowadzono kilka testów kontrolnych. Rozważono dwa różne macierze początkowe o różnych wartościach parametru *n* należącego od 10 do 120



Rys. 1 Normy kolejnych przybliżeń, dla dwóch różnych parametrów *n* (n =10, n = 120)

Uzyskano następujące wyniki:

Dla n = **10**,

Niedokładność rozwiązania wyniosła : 7.288152e-16,
Norma macierzy : 1.067334

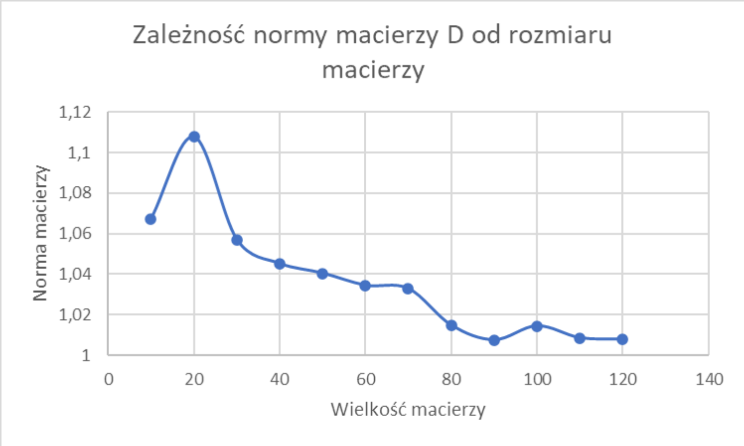
Dla n = **120**
Niedokładność rozwiązania wyniosła : 9.186459e-15
Norma macierzy : 1.007792

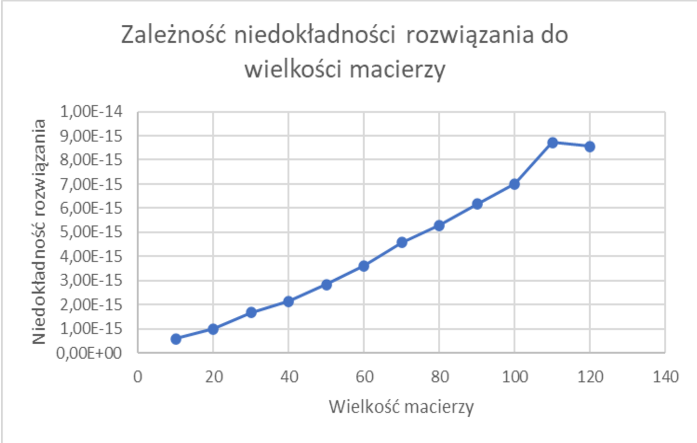
Z otrzymanych wyników sformułowano następującą hipotezę: wraz ze wzrostem wielkości macierzy, maleje norma macierzy

Do eksperymentu wybrano poniższe wartości n : 10,20,30,40,50,60,70,80,90,100,110,120 oraz wykorzystano metodę badaj_zbieznosc() napisaną w języku Python.

Wielkosc macierzy	||D||	Niedokladnosc
10	1.067334	5.940850e-16
20	1.107848	1.001282e-15
30	1.057103	1.666722e-15
40	1.045213	2.142557e-15
50	1.040467	2.839540e-15
60	1.034425	3.608919e-15
70	1.032818	4.569963e-15
80	1.014666	5.293913e-15
90	1.007484	6.163903e-15
100	1.014385	7.006548e-15
110	1.008471	8.720791e-15
120	1.007792	8.555830e-15

- Wraz ze wzrostem wielkości macierzy, nieznacznie maleje norma macierzy ||D||
- Wraz ze wzrostem wielkości macierzy znacząco rośnie niedokładność rozwiązania
- Na samym początku norma macierzy rośnie, aż osiąga maksimum lokalne, a potem maleje





5. <u>Wnioski</u>

Wraz ze wzrostem wielkości macierzy n można zauważyć, że:

1. Dla wielkości macierzy powyżej około 20, norma macierzy D maleje.
2. Wartości błędów rozwiązań znacząco się zwiększają.

Zadanie 2.

1. <u>Cel zadania</u>

Celem zadania było zbadanie wpływu epsilonu na efektywność uzyskiwania rankingu metodą iteracyjną prostej i metodą potęgową. Wykorzystano do tego metodę iteruj_roznice().

2. <u>Metody</u>

Zadanie rozwiązywano z wykorzystaniem środowiska VSCode zainstalowanym na komputerze stacjonarnym o procesorze Intel Core i5-9600k, za pomocą klas udostępnionych w ramach zajęć z kursu Metody numeryczne, analizę danych wykonano w programie Excel

3. <u>Przyjęte parametry</u>

- Norma = 1
- k=5 – liczba pomiarów dla jednej wartości parametru
- n = 30 – rozmiar macierzy
- gamma = 0.4

4. <u>Przebieg doświadczenia i wyniki</u>

Dla wybranych wartości epsilonu przeprowadzono kilka testów kontrolnych. Rozważono macierz, na której zastosowano 2 metody: iteracja prosta i metodę potęgową. Przyjęto dwie różne wartości parametru epsilon : 10^{-13} i 10^{-1}

- Dla 10^{-13} otrzymano:
    - **iteracja prosta**
        - Liczba iteracji: 426
        - Niedokładność: 8.594337e-14
    - **Metoda potęgowa**

        ▪ liczba iteracji: 19.4
        ▪ Niedokładność: 1.652604e-12

- Dla 10^{-1} otrzymano:
    - **iteracja prosta**
        ▪ liczba iteracji 29.4
        ▪ Niedokładność: 8.473893e-02
    - **Metoda potęgowa:**
        ▪ liczba iteracji 2
        ▪ Niedokładność: 4.306270e-02

Z otrzymanych wyników sformułowano następującą hipotezę: Wraz ze wzrostem epsilon, rośnie efektywność iteracji prostej oraz potęgowej.

Do eksperymentu wybrano następujące wartości epsilon: 1e-13,1e-12, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2,1e-1 oraz wykonano metodę badaj_zbieznosc() w języku Python.

Dla każdej wartości epsilon przeprowadzono 5 testów obydwu metod iteracji : prostej oraz potęgowej.
Średnie wartości poszczególnych parametrów pokazano w 2 tabelach poniżej:

```
 def badaj_zbieznosc(self):
        """Badam zbieznosc metody iteracji seidela"""
        # ustalam zbior parametrow
        param = [1e-13, 1e-12, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-
2,1e-1]
        sr_liczba_iteracji1 = []
        sr_liczba_iteracji2 = []
        sr_niedokladnosc1 = []
        sr_niedokladnosc2 = []
        for cur_eps in param:
            niedokladnosc1 = 0.0
            niedokladnosc2 = 0.0
            iteracje1 = 0
            iteracje2 = 0
            iteracje = 0
            while iteracje < self.k:
                rank = pagerank.PageRank(self.n)
                rank.losuj(0.4)
                rank.przygotuj_do_iteracji()
                test1 = iteracjaprosta.IteracjaProsta(rank.v)
                test1.przygotuj()
                iter1 = test1.iteruj_roznica(eps=cur_eps, norma=self.norma)

                niedokl1 = test1.sprawdz_rozwiazanie(norma=self.norma)
                rank.ranking_po_iteracji(test1.X)

                test2 = potegowa.Potegowa(rank.u)
                iter2 = test2.iteruj_roznica(eps=cur_eps)
                niedokl2 = test2.sprawdz_rozwiazanie(norma=self.norma)
```

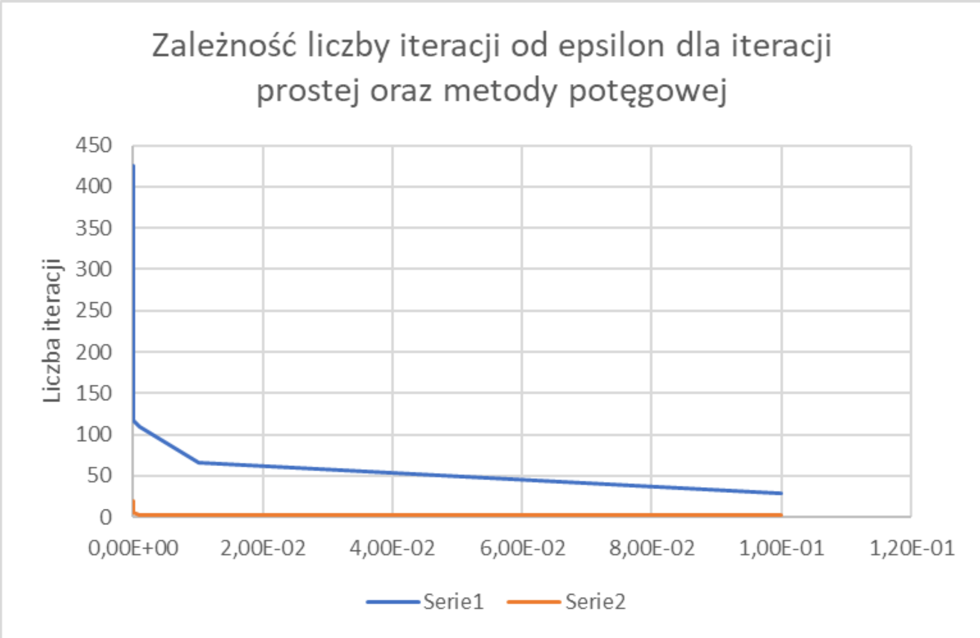
```
            rank.ranking(test2.y)

            niedokladnosc1 += niedokl1
            niedokladnosc2 += niedokl2

            iteracje += 1
            iteracje1 += iter1
            iteracje2 += iter2
        sr_liczba_iteracji1.append(iteracje1/self.k)
        sr_liczba_iteracji2.append(iteracje2/self.k)
        sr_niedokladnosc1.append(niedokladnosc1/self.k)
        sr_niedokladnosc2.append(niedokladnosc2/self.k)
    print("Epsilon  Iteracje  Niedkoladnosc")
    print("------"*9)
    for i in range(len(param)):
        wyniki1 = f"{param[i]} \t"
        wyniki1 += f"{sr_liczba_iteracji1[i]} \t"
        wyniki1 += f"{sr_niedokladnosc1[i]:.6e} \n"
        print(wyniki1)

    print("Epsilon  Iteracje  Niedkoladnosc")
    print("------"*9)
    for i in range(len(param)):
        wyniki2 = f"{param[i]} \t"
        wyniki2 += f"{sr_liczba_iteracji2[i]} \t"
        wyniki2 += f"{sr_niedokladnosc2[i]:.6e} \n"
  print(wyniki2)
```

Epsilon	Iteracje	Niedkoladnosc
1e-13	426.0	8.594337e-14
1e-12	415.6	8.851722e-13
1e-10	322.0	8.758077e-11
1e-09	302.0	8.702812e-10
1e-08	263.0	8.871271e-09
1e-07	224.0	8.682272e-08
1e-06	205.8	8.768303e-07
1e-05	168.4	8.583966e-06
0.0001	116.6	8.571026e-05
0.001	109.8	8.713228e-04
0.01	66.2	8.590151e-03
0.1	29.4	8.473893e-02

Rys.2 Średnie wartości dla metody iteracji prostej

Epsilon	Iteracje	Niedkoladnosc
1e-13	19.4	1.652604e-12
1e-12	19.2	2.026684e-12
1e-10	14.6	2.589579e-09
1e-09	12.6	3.805634e-08
1e-08	12.4	1.515075e-08
1e-07	9.8	7.736999e-07
1e-06	8.2	1.152552e-05
1e-05	7.0	3.210809e-05
0.0001	5.0	5.547564e-04
0.001	2.8	2.208145e-02
0.01	2.0	4.531983e-02
0.1	2.0	4.306270e-02

Rys.3 Średnie wartości dla metody potęgowej



- Metoda potęgowa wymaga znacznie mniejszej liczby iteracji, dla porównywalnych wartości epsilon.
- Dla większej wartości epsilon, niedokładność rosnie.
- Dla porównywalnych wartości epsilon, metoda potęgowa ma większą niedokładność

5. <u>Wnioski</u>

Metoda potęgowa jest znacząco bardziej efektywna w rozwiązywaniu rankingu pagerank w porównaniu do metod iteracji prostej, ale cechuje się większą niedokładnością. Wraz ze wzrostem epsilon, rośnie efektywność iteracji prostej oraz potęgowej.