Jan Doniec A=5, B=1, C=2, D=5

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

(Imię i nazwisko) (A, B, C, D)

Parametry:
M = 10
N = 14
norma = 1

Raport z Pracowni nr 2

Zadanie 1.

1. Cel zadania

Celem zadania było zbadanie jak zbieżność Iteracji Prostej jest zależna od parametru n (Parametr n - wielkość macierzy), przy normie macierzowej równej 1, wykorzystując metodę iteruj_roznica().

2. Metody

Do przeprowadzenia doświadczenia wykorzystano komputer MacBook Air z procesorem Apple M1 ze zainstalowanym środowiskiem Visual Studio Code, wykorzystując pliki zawierające klasy napisane w języku Python, udostępnione studentom podczas kursu Metody Numeryczne. Część analizy danych odbyła się również w programie Microsoft Excel.

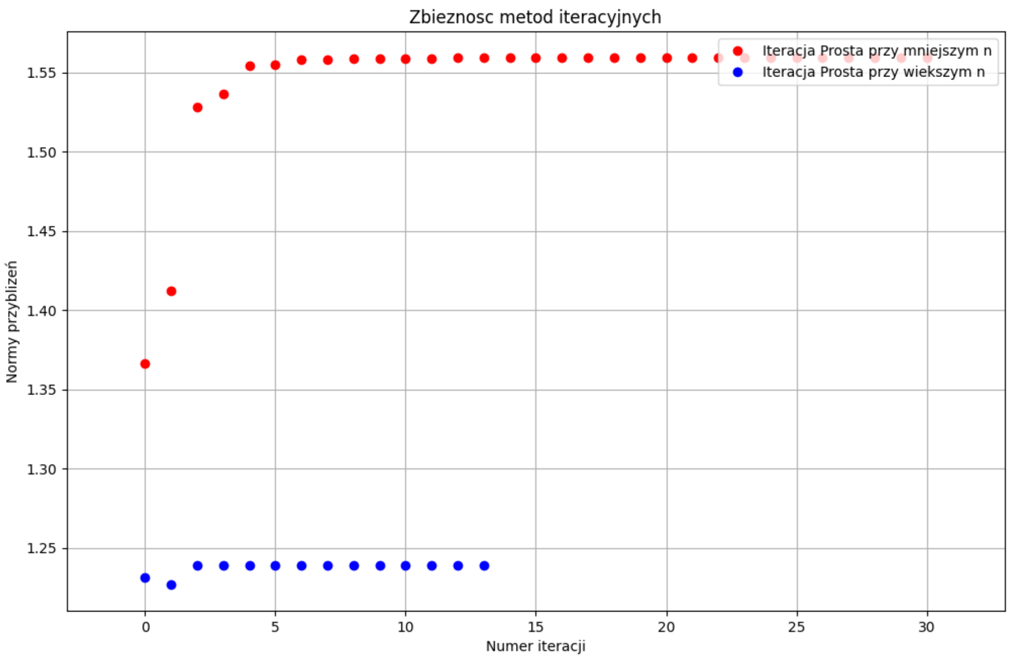
3. Przyjęte parametry

Do wykonania zadania przyjęto poniższe parametry:

- *eps*=1.0E-10 – parametr stopu
- *alfa*=0.3 – parametr zmiennej losującej układ
- *k*=5 – liczba pomiarow dla jednej wartosci parametru
- *norma*=1

4. Przebieg doświadczenia i wyniki

W zgodzie z wybranymi parametrami, przeprowadzone zostało kilka testów kontrolnych. Podczas sesji testowej rozważono dwie różne macierze o rożnych wartościach parametru n należących od 10 do 140.



Rys.1 – Przedstawia normy kolejnych przybliżeń dla dwóch różnych parametrów n (n=10, n=140).

Uzyskano następujące wyniki:

Dla n=10:

Norma macierzy: 1.017589
Niedokładność rozwiązania: 6.619938e-11

Dla n=140:
Norma macierzy: 1.012952
Niedokładność rozwiązania: 1.355744e-10

Na podstawie powyższych wyników można sformułować następującą hipotezę: Niezależnie od wzrostu rozmiaru macierzy norma oscyluje w zbliżonych wartościach.

W eksperymencie wykorzystano następujące wartości n=[10,20,30,40,50,60,70,80,90,100,110,120,130,140]

Wykorzystano również funkcje badaj_zbieznosc():

```
def badaj_zbieznosc(self):
        param = [10,20,30,40,50,60,70,80,90,100,110,120,130,140]
        sr_norma_macierzy = []

        sr_niedokladnosc = []
        sr_liczba_iteracji = []
        for n in param :
            u1 = uklad.Uklad(wymiar=self.n)
            norma_macierzy = 0.0
            liczba_iteracji=0.0
            niedokladnosc = 0.0
            iteracje = 0

            while iteracje < self.k:
                u1.losuj_uklad_symetryczny_dodatnio_okreslony()
                test1 = iteracjaprosta.IteracjaProsta(ukl=u1)
                test1.przygotuj()
                norma_D = u1.norma_macierzy(
                    typ=self.norma,
                    macierz=test1.D
                )
                iter=test1.iteruj_roznica(
                    norma = self.norma,
                    eps= self.eps
                )
                niedokl = test1.sprawdz_rozwiazanie(norma=self.norma)
                if iter == 0:
                    continue
                else:
                    norma_macierzy += norma_D
                    niedokladnosc += niedokl
                    liczba_iteracji += iter
                    iteracje += 1

            sr_liczba_iteracji.append(liczba_iteracji/self.k)
            sr_norma_macierzy.append(norma_macierzy / self.k)
            sr_niedokladnosc.append(niedokl / self.k)
        print("Wielkosc \nmacierzy \t \t ||D|| \t Iteracje \t   Niedokladnosc")
        print("-------" * 9)
        for i in range(len(param)):
```

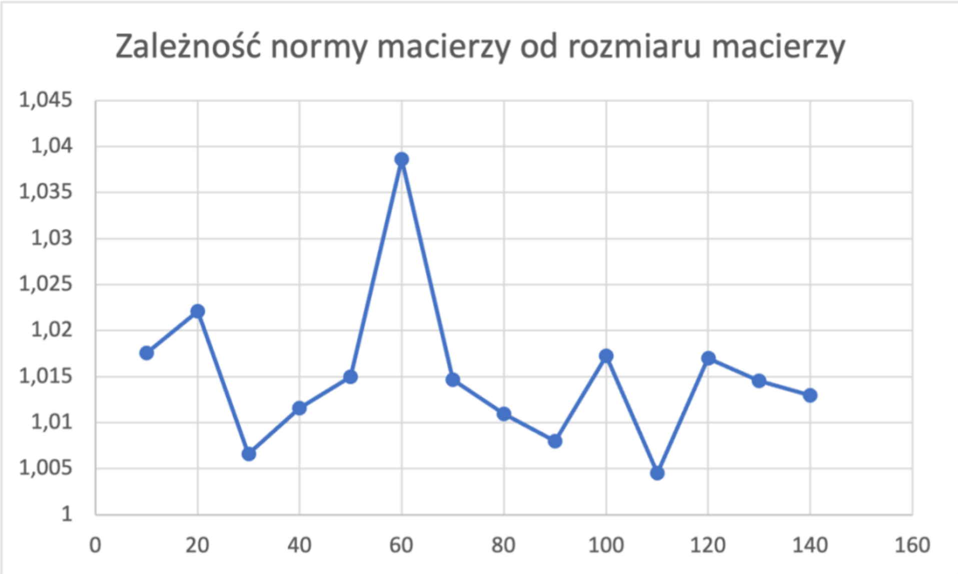
```
        wyniki = f"{param[i]} \t\t\t"
        wyniki += f"{sr_norma_macierzy[i]:.6f} \t\t"
        wyniki += f"{sr_liczba_iteracji[i]:.2f} \t"
        wyniki += f"{sr_niedokladnosc[i]:.6e} \n"
        print(wyniki)
```

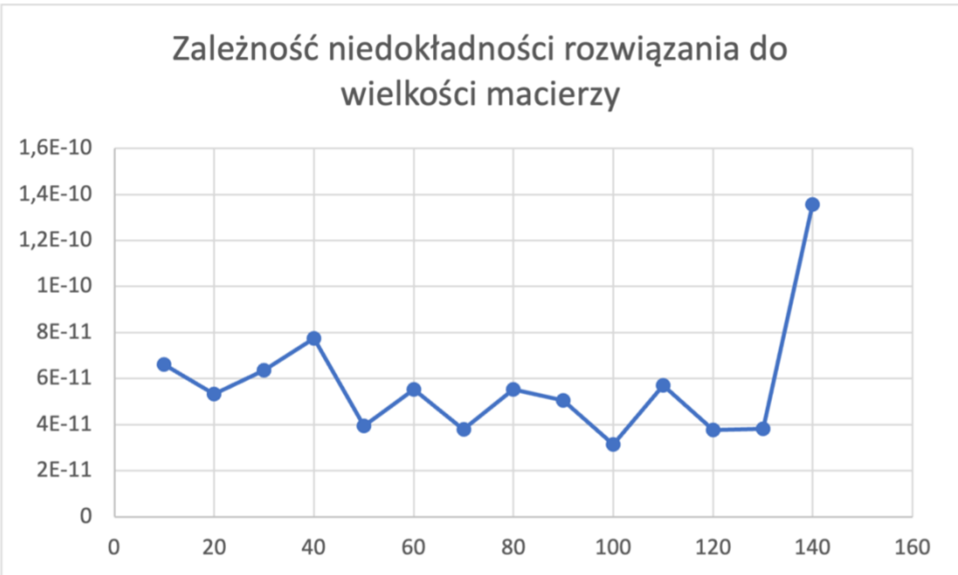
Wynikiem jej działania dla wcześniejszych parametrów jest zwrócenie w konsoli następującej sekwencji danych:

Wielkosc macierzy	||D||	Iteracje	Niedokladnosc
10	1.017589	14.00	6.619938e-11
20	1.022122	13.80	5.337707e-11
30	1.006595	13.60	6.349509e-11
40	1.011566	13.80	7.751698e-11
50	1.014971	14.00	3.949318e-11
60	1.038630	13.80	5.534426e-11
70	1.014687	14.00	3.800554e-11
80	1.010966	14.00	5.530306e-11
90	1.007955	13.80	5.043926e-11
100	1.017232	14.00	3.133300e-11
110	1.004543	13.80	5.700721e-11
120	1.017006	14.00	3.759572e-11
130	1.014551	13.80	3.825631e-11
140	1.012952	13.80	1.355744e-10

Zauważmy, że:

-Wartości norm niewiele się zmieniają wraz ze wzrostem rozmiaru macierzy
-Im większy rozmiar macierzy tym mniejsza niedokładność





5. <u>Wnioski</u>

Przy wzroście rozmiaru macierzy n:

- Nie wydaje się, aby rozmiar macierzy miał wpływ na normę macierzy
- Wydaje się, że średnio im większa macierzy tym mniejsza niedokładność

Zadanie 2.

1. <u>Cel zadania</u>

Celem zadania było zbadanie wpływu wartości gamma na efektywność uzyskiwania rankingu PageRank metodą Iteracji Seidla oraz Metodą Potęgową przy normie równej 0. Wykorzystano do tego metodę iteruj_roznica().

2. Metody

Do przeprowadzenia doświadczenia wykorzystano komputer MacBook Air z procesorem Apple M1 ze zainstalowanym środowiskiem Visual Studio Code, wykorzystując pliki zawierające klasy napisane w języku Python, udostępnione studentom podczas kursu Metody Numeryczne. Część analizy danych odbyła się również w programie Microsoft Excel.

3. Przyjęte parametry

Do wykonania zadania przyjęto poniższe parametry:

- eps=1.0E-10 – parametr stopu
- k=5 – liczba pomiarow dla jednej wartosci parametru
- norma=0
- n=30 – rozmiar macierzy

4. Przebieg doświadczenia i wyniki

Dla wybranych wartości parametru gamma przeprowadzono kilka testów kontrolnych. Rozważono macierz na której zastosowano dwie metody – Iteracji Seidela oraz Metodę Potęgową. Przyjęto następujące wartości parametru gamma: 0.99 oraz 0.01.

- Dla 0.99:

Średnia liczba linkow: 29.733333333333334

Iteracja Seidela

-Liczba iteracji: 147
-Niedokładność: 7.547418939823913e-11

Metoda Potęgowa

-Liczba iteracji: 7
-Niedokładność: 0.035522854386918365

- Dla 0.01:

Średnia liczba linkow: 0.3333333333333333

Iteracja Seidela

-Liczba iteracji: 136
-Niedokładność: 6.886611420009459e-11

Metoda Potęgowa

-Liczba iteracji: 18
-Niedokładność: 1.6999999998748558

Teraz można sformułować hipotezę: Wraz ze wzrostem wartości parametru gamma rośnie efektywność Metody Potęgowej.

Do badań wybrano następujące wartości gamma:
[0.01,0.1,0.2,0.25,0.3,0.4,0.5,0.55,0.6,0.7,0.8,0.9,0.95,0.99]

Dla każdej z wartości zostało przeprowadzonych pięć testów w obydwu iteracjach – Iteracji Seidela oraz Metodzie Potęgowej.

Oto ciało metody badaj_zbieznosc(), która odpowiada za przygotowanie i przeprowadzenie testów.

```
def badaj_zbieznosc(self):
    gammy=[0.01,0.1,0.2,0.25,0.3,0.4,0.5,0.55,0.6,0.7,0.8,0.9,0.95,0.99]

    u1 = uklad.Uklad(wymiar = self.n)

    sr_gamma1 = []
    sr_niedokladnosc1 = []
    sr_liczba_iteracji1 = []
    sr_norma_macierzy1 = []
    sr_liczba_iteracji2 = []
    sr_norma_macierzy2 = []
    sr_gamma2 = []
    sr_niedokladnosc2 = []

    for gamma in gammy:
        norma_macierzy1 = 0.0
        niedokladnosc1 = 0.0
        iteracje1 = 0
        liczba_iteracji1 = 0.0
        gamma1 = 0.0
        norma_macierzy2 = 0.0
        liczba_iteracji2 = 0.0
        gamma2 = 0.0
        niedokladnosc2 = 0.0

        while iteracje1 < self.k:
```

```
pgr = pagerank.PageRank(nn=30)
pgr.losuj(gamma)
pgr.srednia_liczba_linkow()
print("Metoda potegowa")

test1 = potegowa.Potegowa(ukl=pgr.u)

iter1 = test1.iteruj_roznica(
    eps = self.eps
    )
test1.wypisz_rozwiazanie(iter1)
pgr.ranking(test1.y)
niedokl1 = test1.sprawdz_rozwiazanie(self.norma)
if iter1 == 0:
        continue
else:
    norma_macierzy1 += uklad.Uklad.norma_macierzy(pgr.u,typ=self.norma)
    gamma1 += gamma
    niedokladnosc1 += niedokl1
    liczba_iteracji1 += iter1
    iteracje1 += 1
print("Metoda Seidela")

pgr.przygotuj_do_iteracji()
test2 = iteracjaseidela.IteracjaSeidela(ukl=pgr.v)
test2.przygotuj()
iter2 = test2.iteruj_roznica(
    eps = self.eps,
    norma = self.norma
    )
test2.wypisz_rozwiazanie(iter2)
pgr.ranking_po_iteracji(test2.X)
niedokl2 = test2.sprawdz_rozwiazanie(self.norma)
if iter2 == 0:
    continue
else:
    norma_macierzy2 +=          uklad.Uklad.norma_macierzy(pgr.u,typ=self.norma)
    gamma2 += gamma
    niedokladnosc2 += niedokl2
    liczba_iteracji2 += iter2
```

```
        sr_norma_macierzy1.append(norma_macierzy1/self.k)
        sr_gamma1.append(gamma1/self.k)
        sr_liczba_iteracji1.append(liczba_iteracji1/self.k)
        sr_niedokladnosc1.append(niedokladnosc1/self.k)
        sr_norma_macierzy2.append(norma_macierzy2/self.k)
        sr_gamma2.append(gamma2/self.k)
        sr_liczba_iteracji2.append(liczba_iteracji2/self.k)
        sr_niedokladnosc2.append(niedokladnosc2/self.k)

    print("Metoda Potegowa")
    print("Gamma \t \t ||D|| \t     Iteracje   Niedokladnosc")
    print("------"*9)
    for i in range(len(gammy)):
        wyniki = f"{sr_gamma1[i]} \t"
        wyniki += f"{sr_norma_macierzy1[i]:.6f} \t"
        wyniki += f"{sr_liczba_iteracji1[i]:.2f} \t"
        wyniki += f"{sr_niedokladnosc1[i]:.6e} \n"
        print(wyniki)

    print("Metoda Seidela")
    print("Gamma \t \t ||D|| \t     Iteracje   Niedokladnosc")
    print("------"*9)
    for i in range(len(gammy)):
        wyniki = f"{sr_gamma2[i]} \t"
        wyniki += f"{sr_norma_macierzy2[i]:.6f} \t"
        wyniki += f"{sr_liczba_iteracji2[i]:.2f} \t"
        wyniki += f"{sr_niedokladnosc2[i]:.6e} \n"
        print(wyniki)
```

Poniżej znajdują się wyniki testów dla obu metod:

Metoda Potęgowa

```
Gamma           ||D||       Iteracje   Niedokladnosc
------------------------------------------------------
0.01     2.446667       17.00    9.016667e-01

0.1      2.290000       186.60   8.848159e-01

0.2      1.877063       24.20    5.309395e-01

0.25     1.738268       20.00    5.137208e-01
```

0.3	1.730088	18.80	4.508544e-01
0.4	1.534355	15.20	4.599518e-01
0.5	1.432270	13.40	3.781356e-01
0.55	1.339407	12.00	3.170948e-01
0.6	1.271511	11.40	3.206489e-01
0.7	1.287942	10.00	3.168111e-01
0.8	1.167588	9.20	2.406856e-01
0.9	1.102763	7.80	1.503479e-01
0.95	1.052362	7.00	9.786271e-02
0.99	1.010640	5.80	3.897587e-02

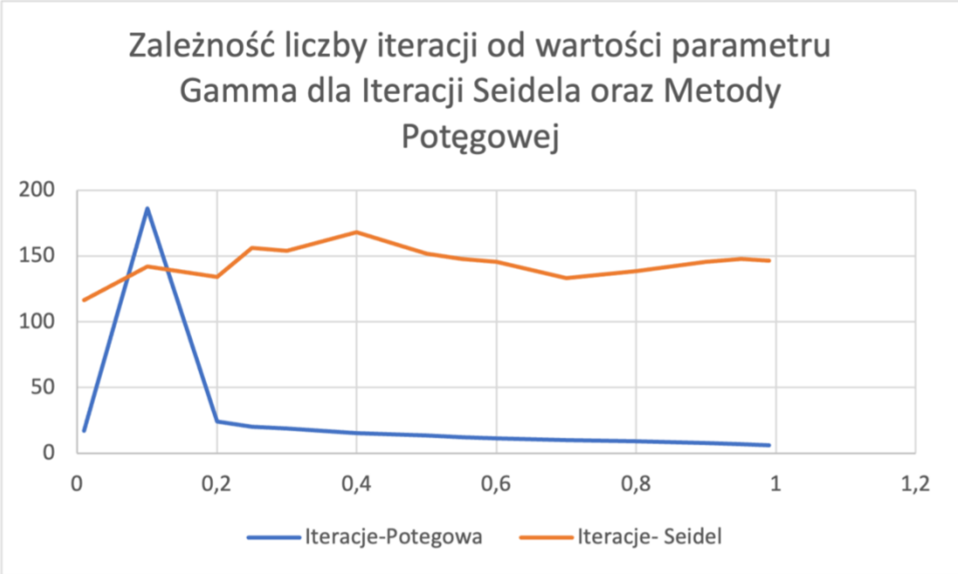
Iteracja Seidela

Gamma	||D||	Iteracje	Niedokladnosc
0.01	2.446667	116.60	5.080476e-11
0.1	2.290000	142.00	1.000000e-01
0.2	1.877063	134.40	6.756213e-11
0.25	1.738268	156.20	6.486115e-11
0.3	1.730088	154.00	6.589118e-11
0.4	1.534355	168.40	7.127794e-11
0.5	1.432270	152.00	7.037886e-11
0.55	1.339407	147.80	7.071342e-11
0.6	1.271511	145.60	7.345058e-11
0.7	1.287942	133.40	7.709446e-11

0.8	1.167588	138.80	7.803019e-11
0.9	1.102763	145.80	7.697077e-11
0.95	1.052362	147.80	7.539689e-11
0.99	1.010640	146.60	8.053303e-11

Średnia liczba iteracji dla Metody Potęgowej: 25,6
Średnia liczba iteracji dla Iteracji Seidela: 144,9571429

Średnia Niedokładność Rozwiązania dla Metody Potęgowej: 4,00E-01
Średnia Niedokładność Rozwiązania dla Iteracji Seidela: 0,00714286



Z powyższych danych możemy wywnioskować:
-Generalny trend mówi, że wraz ze wzrostem parametru Gamma liczba iteracji dla metody potęgowej maleje, a dla Iteracji Seidela liczba iteracji utrzymuje się na podobnym poziomie.
-Iteracja Potęgowa potrzebuje znacznie mniej iteracji
-Średnio niedokładność jest mniejsza dla Iteracji Seidela

5. Wnioski

Na podstawie przeprowadzonego eksperymentu można stwierdzić, że Metoda Potęgowa jest o wiele bardziej efektywna od Iteracji Seidela w określaniu rankingu Google PageRank. Jednak Iteracji Seidela cechuje się większą dokładnością. Wraz ze wzrostem parametru gamma efektywność Metody Potęgowej rośnie, natomiast efektywność Metody Iteracyjnej wydaje się utrzymywać na podobnym poziomie.