main

April 14, 2023

1 Taylorjev razvoj

1.1 Navodilo naloge

S Taylorjevim razvojem izračunaj funkciji $exp(A)$ in $exp(iA)$, kjer je matrika

$$A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Kako lastne vrednosti približkov n-tega reda konvergirajo k točnemu rezultatu?

1.2 Teorija na kratko

Taylorjev razvoj za eksponentno funkcijo se glasi:

$$e^X = \sum_{n=0}^{\infty} \frac{1}{n!} X^n$$

kjer je X poljubna matrika (poseben primer je matrika velikosti 1x1, kjer dobimo Taylorjev razvoj za skalar, tj. neko kompleksno število).

Poseben primer so diagonalne matrike. Če je matrika A diagonalna, tj. $A_{ij} = 0$ za $i \neq j$, je $(e^A)_{ii} = e^{A_{ii}}$. To lahko izkoristimo tako, da matriko diagonaliziramo in upoštevamo zvezo:

$$e^A = e^{PDP^{-1}} = Pe^D P^{-1}$$

Ker potrebujemo točen rezultat lastnih vrednosti za primerjavo s približki, je najbolje, da to kar takoj izračunamo. Poleg tega izračunamo še celoten eksponent matrike A, da lahko gledamo kako se tudi te približki približujejo.

Za prvi del bomo gledali kako se lastne vrednosti približkov približujejo dejanskim lastnim vredostim e^A, kasneje pa še za matriko e^{iA}.

1.3

$$e^A$$

```python
import numpy as np
from scipy.linalg import expm

A = np.array([[1,1],[1,-1]])
eigenvalues = np.linalg.eigvals(A)

# lastne vrednosti eksponente matrike so kar eksponenti lastnih vrednosti
eigenvalues = np.exp(eigenvalues)
print(f'Eigenvalues of exp(A): {eigenvalues}')

# izracunamo se eksponent matrike
exp_A = expm(A)
print(f'exp(A):')
print(exp_A)
```

```
Eigenvalues of exp(A): [4.11325038 0.24311673]
exp(A):
[[3.54648243 1.36829887]
 [1.36829887 0.80988468]]
```

1.3.1 Pomožne funkcije

Definirajmo pomožne funkcije za izračun približkov.

```python
from diskcache import Cache

cache = Cache('.cache')

@cache.memoize(typed=False, expire=420, tag='factorial')
def factorial(n: int) -> int:
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)


@cache.memoize(typed=False, expire=420, tag='matrix_pow')
def matrix_pow(M: np.ndarray, n: int) -> np.ndarray:
    if n == 0:
        return np.eye(M.shape[0])
    else:
        return M @ matrix_pow(M, n-1)


@cache.memoize(typed=False, expire=420, tag='matrix_exp')
def matrix_exp(M: np.ndarray, n: int) -> np.ndarray:
    if n == 0:
        return matrix_pow(M, 0)
    else:
        return matrix_exp(M, n-1) + matrix_pow(M, n) / factorial(n)
```

1.3.2 Glavna zanka

Vsako iteracijo izračunamo približke kot povedano zgoraj v teoriji. Rezultate si shranimo; spodaj je tabeliran prikaz in izrisan graf

[ ]:
```
# nastavimo zastavice za najvecje stevilo iteracij in za najvecjo toleranco
MAX_ITERATIONS = 100
# osnova zaokrozitvena napaka na 64-bitnem racunalniku je 16 bitov, zato nima␣
↪smisla racunati na vec decimalnih mest kot to
THRESHOLD = 1e-15

# hranimo priblizke za l.v., da jih lahko kasneje izpisemo
first_eigenvalues = []
second_eigenvalues = []

# sproti racunamo tudi njihove napake. vzamemo kar absoultno vrednost
first_eigenvalue_error = []
second_eigenvalue_error = []

# sicer nepotrebno, ampak spremljamo tudi kako hitro se matrike priblizkov␣
↪priblizujejo resitvi
# uporabimo tudi kot zaostavitveni pogoj, tj. ko je frobeniusova norma manjsa␣
↪od THRESHOLD, prekinemo z iteracijami
fro_norm_error = []
inf_norm_error = []
first_norm_error = []

for n in range(MAX_ITERATIONS):
    exp_An = matrix_exp(A, n)

    # izracunamo norme
    fro_norm = np.linalg.norm(exp_An - exp_A, ord='fro')
    inf_norm = np.linalg.norm(exp_An - exp_A, ord=np.inf)
    first_norm = np.linalg.norm(exp_An - exp_A, ord=1)

    if fro_norm < THRESHOLD:
        break

    fro_norm_error.append(fro_norm)
    inf_norm_error.append(inf_norm)
    first_norm_error.append(first_norm)

    eigenvalues_approx = np.linalg.eigvals(exp_An)

    first_eigenvalues.append(eigenvalues_approx[0])
    second_eigenvalues.append(eigenvalues_approx[1])

    # napake priblizkov l.v.
```

```
    e1 = np.abs(eigenvalues[0] - eigenvalues_approx[0])
    e2 = np.abs(eigenvalues[1] - eigenvalues_approx[1])

    first_eigenvalue_error.append(e1)
    second_eigenvalue_error.append(e2)
```

1.3.3 Tabeliran prikaz približkov lastnih vrednosti

[ ]:
```
from tabulate import tabulate

n = len(first_eigenvalues)
eig_results = np.array([np.arange(n).astype(str), eigenvalues[0]*np.ones(n), first_eigenvalues, first_eigenvalue_error, eigenvalues[1]*np.ones(n), second_eigenvalues, second_eigenvalue_error]).T
table = tabulate(eig_results, headers=['Iteration', 'First eigenvalue', 'First eigenvalue approximation', 'First eigenvalue error', 'Second eigenvalue', 'Second eigenvalue approximation', 'Second eigenvalue error'], tablefmt='psql', floatfmt='.16f')
print(table)
```

```
+-------------+--------------------+-----------------------------------+---------
-----------------+---------------------+----------------------------------+----
----------------------+
|   Iteration |   First eigenvalue |   First eigenvalue approximation |   First
eigenvalue error |   Second eigenvalue |   Second eigenvalue approximation |
Second eigenvalue error |
|-------------+--------------------+-----------------------------------+---------
-----------------+---------------------+----------------------------------+----
----------------------|
|           0 | 4.1132503787829267 |                1.0000000000000000 |
3.1132503787829267 |  0.2431167344342142 |                1.0000000000000000 |
0.7568832655657858 |
|           1 | 4.1132503787829267 |                2.4142135623730949 |
1.6990368164098317 |  0.2431167344342142 |               -0.4142135623730951 |
0.6573302968073093 |
|           2 | 4.1132503787829267 |                3.4142135623730949 |
0.6990368164098317 |  0.2431167344342142 |                0.5857864376269049 |
0.3426697031926906 |
|           3 | 4.1132503787829267 |                3.8856180831641272 |
0.2276322956187995 |  0.2431167344342142 |                0.1143819168358734 |
0.1287348175983408 |
|           4 | 4.1132503787829267 |                4.0522847498307932 |
0.0609656289521334 |  0.2431167344342142 |                0.2810485835025400 |
0.0379318490683258 |
|           5 | 4.1132503787829267 |                4.0994252019098969 |
0.0138251768730298 |  0.2431167344342142 |                0.2339081314234369 |
0.0092086030107773 |
```

6	4.1132503787829267	4.1105363130210080	0.0027140657619187	0.2431167344342142	0.2450192425345480	0.0019025081003338
7	4.1132503787829267	4.1127810964533458	0.0004692823295809	0.2431167344342142	0.2427744591022096	0.0003422753320046
8	4.1132503787829267	4.1131779218501716	0.0000724569327550	0.2431167344342142	0.2431712844990350	0.0000545500648208
9	4.1132503787829267	4.1132402769455139	0.0000101018374128	0.2431167344342142	0.2431089294036922	0.0000078050305220
10	4.1132503787829267	4.1132490952876655	0.0000012834952612	0.2431167344342142	0.2431177477458439	0.0000010133116297
11	4.1132503787829267	4.1132502290166721	0.0000001497662545	0.2431167344342142	0.2431166140168376	0.0000001204173766
12	4.1132503787829267	4.1132503626279169	0.0000000161550098	0.2431167344342142	0.2431167476280823	0.0000000131938681
13	4.1132503787829267	4.1132503771629043	0.0000000016200223	0.2431167344342142	0.2431167330930952	0.0000000013411190
14	4.1132503787829267	4.1132503786311592	0.0000000001517675	0.2431167344342142	0.2431167345613506	0.0000000001271364
15	4.1132503787829267	4.1132503787695871	0.0000000000133396	0.2431167344342142	0.2431167344229220	0.0000000000112922
16	4.1132503787829267	4.1132503787818226	0.0000000000011040	0.2431167344342142	0.2431167344351575	0.0000000000009433
17	4.1132503787829267	4.1132503787828405	0.0000000000000862	0.2431167344342142	0.2431167344341397	0.0000000000000745
18	4.1132503787829267	4.1132503787829204	0.0000000000000062	0.2431167344342142	0.2431167344342197	0.0000000000000055
19	4.1132503787829267	4.1132503787829267	0.0000000000000000	0.2431167344342142	0.2431167344342138	0.0000000000000004

1.3.4 Tabeliran prikaz vseh napak, lastnih vrednosti in norm matrike

[ ]:
```
from tabulate import tabulate

n = len(first_eigenvalues)
err_results = np.array([np.arange(n).astype(str), first_eigenvalue_error,␣
 ↪second_eigenvalue_error, fro_norm_error, inf_norm_error, first_norm_error]).T
table = tabulate(err_results, headers=['Iteration', '1st eigenvalue error',␣
 ↪'2nd eigenvalue error', 'Frobenius norm error', 'Inf norm error', 'First␣
 ↪norm error'], tablefmt='psql', floatfmt='.16f')
print(table)
```

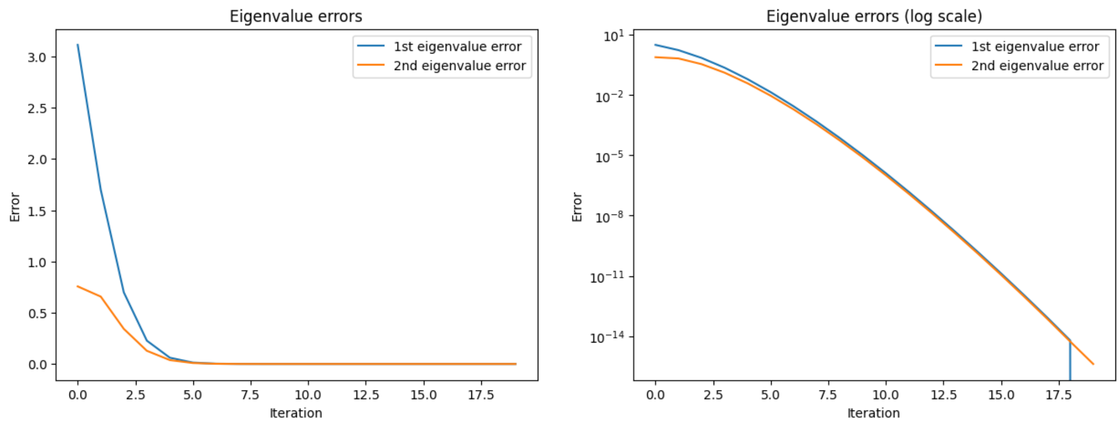
```
+-------------+------------------------+------------------------+---------------
---------+--------------------+--------------------+
|   Iteration |   1st eigenvalue error |   2nd eigenvalue error |   Frobenius
norm error |     Inf norm error |   First norm error |
|-------------+------------------------+------------------------+---------------
---------+--------------------+--------------------|
|           0 |     3.1132503787829267 |     0.7568832655657858 |
3.2039351114973553 | 3.9147813006257524 | 3.9147813006257524 |
|           1 |     1.6990368164098317 |     0.6573302968073093 |
1.8217599245281590 | 1.9147813006257521 | 1.9147813006257521 |
|           2 |     0.6990368164098317 |     0.3426697031926906 |
0.7785081863298294 | 0.9147813006257521 | 0.9147813006257521 |
|           3 |     0.2276322956187995 |     0.1287348175983408 |
0.2615131263832911 | 0.2481146339590854 | 0.2481146339590854 |
|           4 |     0.0609656289521334 |     0.0379318490683258 |
0.0718027373243634 | 0.0814479672924189 | 0.0814479672924189 |
|           5 |     0.0138251768730298 |     0.0092086030107773 |
0.0166112577784071 | 0.0147813006257522 | 0.0147813006257522 |
|           6 |     0.0027140657619187 |     0.0019025081003338 |
0.0033144667794172 | 0.0036701895146412 | 0.0036701895146412 |
|           7 |     0.0004692823295809 |     0.0003422753320046 |
0.0005808427564811 | 0.0004955863400380 | 0.0004955863400380 |
|           8 |     0.0000724569327550 |     0.0000545500648208 |
0.0000906957368148 | 0.0000987609432126 | 0.0000987609432126 |
|           9 |     0.0000101018374128 |     0.0000078050305220 |
0.0000127657988617 | 0.0000105775216956 | 0.0000105775216956 |
|          10 |     0.0000012834952612 |     0.0000010133116297 |
0.0000016352860746 | 0.0000017591795440 | 0.0000017591795440 |
|          11 |     0.0000001497662545 |     0.0000001204173766 |
0.0000001921725159 | 0.0000001558446072 | 0.0000001558446072 |
|          12 |     0.0000000161550098 |     0.0000000131938681 |
0.0000000208581525 | 0.0000000222333625 | 0.0000000222333625 |
|          13 |     0.0000000016200223 |     0.0000000013411190 |
0.0000000021031112 | 0.0000000016777866 | 0.0000000016777866 |
|          14 |     0.0000000001517675 |     0.0000000001271364 |
0.0000000001979829 | 0.0000000002095313 | 0.0000000002095313 |
```

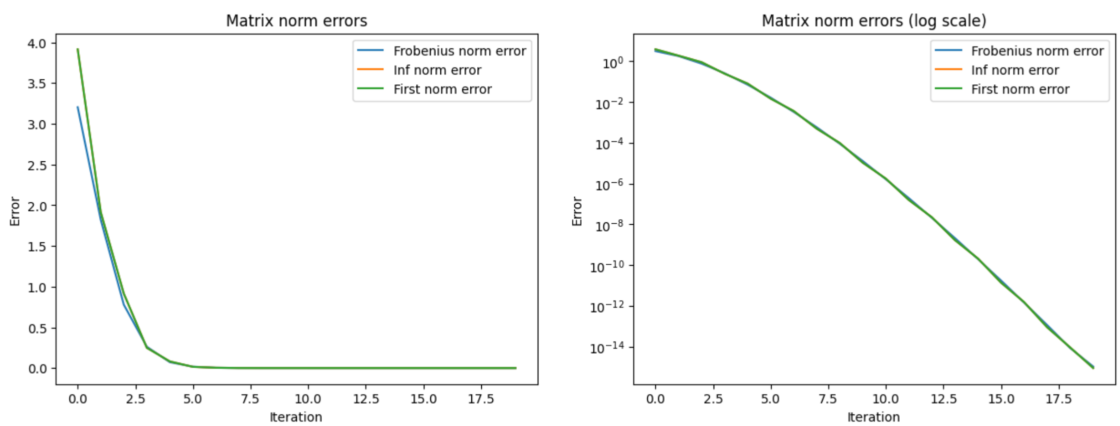
```
|           15 |     0.0000000000133396 |     0.0000000000112922 |
0.0000000000174777 | 0.0000000000137641 | 0.0000000000137641 |
|           16 |     0.0000000000011040 |     0.0000000000009433 |
0.0000000000014524 | 0.0000000000015286 | 0.0000000000015286 |
|           17 |     0.0000000000000862 |     0.0000000000000745 |
0.0000000000001142 | 0.0000000000000890 | 0.0000000000000890 |
|           18 |     0.0000000000000062 |     0.0000000000000055 |
0.0000000000000085 | 0.0000000000000091 | 0.0000000000000091 |
|           19 |     0.0000000000000000 |     0.0000000000000004 |
0.0000000000000010 | 0.0000000000000009 | 0.0000000000000009 |
+-------------+------------------------+------------------------+---------------
---------+--------------------+--------------------+
```

1.3.5 Grafični prikaz napak

[ ]:
```python
import matplotlib.pyplot as plt

fig = plt.figure(figsize=(15, 5))
fig.add_subplot(1, 2, 1) # type: ignore
plt.plot(first_eigenvalue_error, label='1st eigenvalue error')
plt.plot(second_eigenvalue_error, label='2nd eigenvalue error')
plt.title('Eigenvalue errors')
plt.legend()
plt.xlabel('Iteration')
plt.ylabel('Error')

fig.add_subplot(1, 2, 2) # type: ignore
plt.plot(first_eigenvalue_error, label='1st eigenvalue error')
plt.plot(second_eigenvalue_error, label='2nd eigenvalue error')
plt.title('Eigenvalue errors (log scale)')
plt.legend()
plt.xlabel('Iteration')
plt.ylabel('Error')
plt.yscale('log')
plt.show()

fig = plt.figure(figsize=(15, 5))
fig.add_subplot(1, 2, 1) # type: ignore
plt.plot(fro_norm_error, label='Frobenius norm error')
plt.plot(inf_norm_error, label='Inf norm error')
plt.plot(first_norm_error, label='First norm error')
plt.legend()
plt.title('Matrix norm errors')
plt.xlabel('Iteration')
plt.ylabel('Error')

fig.add_subplot(1, 2, 2) # type: ignore
```

```
plt.plot(fro_norm_error, label='Frobenius norm error')
plt.plot(inf_norm_error, label='Inf norm error')
plt.plot(first_norm_error, label='First norm error')
plt.yscale('log')
plt.legend()
plt.title('Matrix norm errors (log scale)')
plt.xlabel('Iteration')
plt.ylabel('Error')
plt.show()
```





1.4

$$e^{iA}$$

[ ]:
```
import numpy as np
from scipy.linalg import expm

A = 1j * np.array([[1,1],[1,-1]])
```

```
eigenvalues = np.linalg.eigvals(A)
eigenvalues = np.exp(eigenvalues)
print(f'Eigenvalues of exp(A): {eigenvalues}')

exp_A = expm(A)
print(f'exp(A):')
print(exp_A)
```

```
Eigenvalues of exp(A): [0.15594369+0.98776595j 0.15594369-0.98776595j]
exp(A):
[[0.15594369+0.698456j 0.        +0.698456j]
 [0.        +0.698456j 0.15594369-0.698456j]]
```

1.4.1 Glavna zanka

[ ]:
```
MAX_ITERATIONS = 100
THRESHOLD = 1e-15

first_eigenvalues = []
second_eigenvalues = []

first_eigenvalue_error = []
second_eigenvalue_error = []

fro_norm_error = []
inf_norm_error = []
first_norm_error = []

for n in range(MAX_ITERATIONS):
    exp_An = matrix_exp(A, n)

    fro_norm = np.linalg.norm(exp_An - exp_A, ord='fro')
    inf_norm = np.linalg.norm(exp_An - exp_A, ord=np.inf)
    first_norm = np.linalg.norm(exp_An - exp_A, ord=1)

    if fro_norm < THRESHOLD:
        break

    fro_norm_error.append(fro_norm)
    inf_norm_error.append(inf_norm)
    first_norm_error.append(first_norm)

    eigenvalues_approx = np.linalg.eigvals(exp_An)

    first_eigenvalues.append(eigenvalues_approx[0])
    second_eigenvalues.append(eigenvalues_approx[1])
```

```
    e1 = np.abs(eigenvalues[0] - eigenvalues_approx[0])
    e2 = np.abs(eigenvalues[1] - eigenvalues_approx[1])

    first_eigenvalue_error.append(e1)
    second_eigenvalue_error.append(e2)
```

1.4.2 Tabeliran prikaz približkov lastnih vrednosti

[ ]:
```
from tabulate import tabulate

n = len(first_eigenvalues)
eig_results = np.array([np.arange(n).astype(str), (eigenvalues[0]*np.ones(n)).
 ↪astype(str), np.array(first_eigenvalues).astype(str),␣
 ↪first_eigenvalue_error, (eigenvalues[1]*np.ones(n)).astype(str), np.
 ↪array(second_eigenvalues).astype(str), second_eigenvalue_error]).T
table = tabulate(eig_results, headers=['Iteration', '1st eigenvalue', '1st␣
 ↪eigenvalue approx', '1st eigenvalue error', '2nd eigenvalue', '2nd␣
 ↪eigenvalue approx', '2nd eigenvalue error'], tablefmt='psql', floatfmt='.
 ↪16f')
print(table)
```

```
+-------------+------------------------------------------+----------------------
---------------------+------------------------+--------------------------------
---------+------------------------------------------------+----------------------
--+
|   Iteration | 1st eigenvalue                           | 1st eigenvalue approx
|   1st eigenvalue error | 2nd eigenvalue                           | 2nd
eigenvalue approx                        |   2nd eigenvalue error |
|-------------+------------------------------------------+----------------------
---------------------+------------------------+--------------------------------
---------+------------------------------------------------+----------------------
--|
|           0 | (0.1559436947653748+0.9877659459927355j) | (1+0j)
|     1.2992738781601245 | (0.1559436947653746-0.9877659459927355j) | (1+0j)
|     1.2992738781601247 |
|           1 | (0.1559436947653748+0.9877659459927355j) |
(0.9999999999999996+1.4142135623730945j)  |     0.9456683435131028 |
(0.1559436947653746-0.9877659459927355j) |
(0.9999999999999998-1.414213562373095j)        |     0.9456683435131034 |
|           2 | (0.1559436947653748+0.9877659459927355j) | 1.4142135623730947j
|     0.4540660804922191 | (0.1559436947653746-0.9877659459927355j) |
(-1.3877787807814457e-17-1.414213562373095j)  |     0.4540660804922192 |
|           3 | (0.1559436947653748+0.9877659459927355j) | 0.9428090415820631j
|     0.1622946677844553 | (0.1559436947653746-0.9877659459927355j) |
(-2.7755575615628914e-17-0.9428090415820635j) |     0.1622946677844550 |
|           4 | (0.1559436947653748+0.9877659459927355j) |
(0.1666666666666663+0.9428090415820632j) |     0.0462180200850947 |
```

```
(0.1559436947653746-0.9877659459927355j) |
(0.1666666666666663-0.9428090415820635j)      |      0.0462180200850945 |
|            5 | (0.1559436947653748+0.9877659459927355j) |
(0.1666666666666663+0.9899494936611662j) |      0.0109430346255599 |
(0.1559436947653746-0.9877659459927355j) |
(0.1666666666666663-0.9899494936611666j)      |      0.0109430346255602 |
|            6 | (0.1559436947653748+0.9877659459927355j) |
(0.1555555555555534+0.9899494936611662j) |      0.0022177764690132 |
(0.1559436947653746-0.9877659459927355j) |
(0.1555555555555553-0.9899494936611666j)      |      0.0022177764690135 |
|            7 | (0.1559436947653748+0.9877659459927355j) |
(0.1555555555555534+0.987704710228828j)  |      0.0003929400272060 |
(0.1559436947653746-0.9877659459927355j) |
(0.1555555555555556-0.9877047102288283j)      |      0.0003929400272055 |
|            8 | (0.1559436947653748+0.9877659459927355j) |
(0.15595238095238095+0.987704710228828j)  |      0.0000618487560589 |
(0.1559436947653746-0.9877659459927355j) |
(0.15595238095238095-0.9877047102288283j)      |      0.0000618487560586 |
|            9 | (0.1559436947653748+0.9877659459927355j) |
(0.15595238095238095+0.9877670653241709j) |      0.0000087580104800 |
(0.1559436947653746-0.9877659459927355j) |
(0.15595238095238098-0.9877670653241711j)      |      0.0000087580104803 |
|           10 | (0.1559436947653748+0.9877659459927355j) |
(0.15594356261022924+0.987767065324171j)  |      0.0000011271059600 |
(0.1559436947653746-0.9877659459927355j) |
(0.15594356261022924-0.987767065324171j)       |      0.0000011271059600 |
|           11 | (0.1559436947653748+0.9877659459927355j) |
(0.15594356261022924+0.9877659315951647j) |      0.0000001329371000 |
(0.1559436947653746-0.9877659459927355j) |
(0.15594356261022935-0.9877659315951649j)      |      0.0000001329370996 |
|           12 | (0.1559436947653748+0.9877659459927355j) |
(0.15594369622147408+0.9877659315951647j) |      0.0000000144710148 |
(0.1559436947653746-0.9877659459927355j) |
(0.15594369622147408-0.9877659315951649j)      |      0.0000000144710146 |
|           13 | (0.1559436947653748+0.9877659459927355j) |
(0.15594369622147397+0.987765946130152j)  |      0.0000000014625690 |
(0.1559436947653746-0.9877659459927355j) |
(0.15594369622147397-0.9877659461301521j)      |      0.0000000014625692 |
|           14 | (0.1559436947653748+0.9877659459927355j) |
(0.15594369475321856+0.987765946130152j)  |      0.0000000001379532 |
(0.1559436947653746-0.9877659459927355j) |
(0.15594369475321856-0.9877659461301521j)      |      0.0000000001379533 |
|           15 | (0.1559436947653748+0.9877659459927355j) |
(0.15594369475321856+0.9877659459917234j) |      0.0000000000121983 |
(0.1559436947653746-0.9877659459927355j) |
(0.15594369475321856-0.9877659459917236j)      |      0.0000000000121981 |
|           16 | (0.1559436947653748+0.9877659459927355j) |
(0.15594369476545422+0.9877659459917235j) |      0.0000000000010151 |
```

```
(0.1559436947653746-0.9877659459927355j) |
(0.155943694765454-0.9877659459917237j)       |     0.0000000000010149 |
|          17 | (0.1559436947653748+0.9877659459927355j) |
(0.155943694765454+0.9877659459927416j)   |     0.0000000000000794 |
(0.1559436947653746-0.9877659459927355j) |
(0.1559436947654541-0.9877659459927417j)      |     0.0000000000000798 |
|          18 | (0.1559436947653748+0.9877659459927355j) |
(0.15594369476537406+0.9877659459927415j) |     0.0000000000000060 |
(0.1559436947653746-0.9877659459927355j) |
(0.15594369476537406-0.9877659459927417j)     |     0.0000000000000062 |
+-------------+------------------------------------------+----------------------
---------------------+-----------------------+-------------------------------
---------+------------------------------------------+----------------------
--+
```

1.4.3 Tabeliran prikaz vseh napak

[ ]:
```python
from tabulate import tabulate

err_results = np.array([np.arange(len(first_eigenvalue_error)),␣
↪first_eigenvalue_error, second_eigenvalue_error, fro_norm_error,␣
↪inf_norm_error, first_norm_error]).T
table = tabulate(err_results, headers=['Iteration', '1st eigenvalue error',␣
↪'2nd eigenvalue error', 'Frobenius norm error', 'Inf norm error', 'First␣
↪norm error'], tablefmt='psql')
print(table)
```

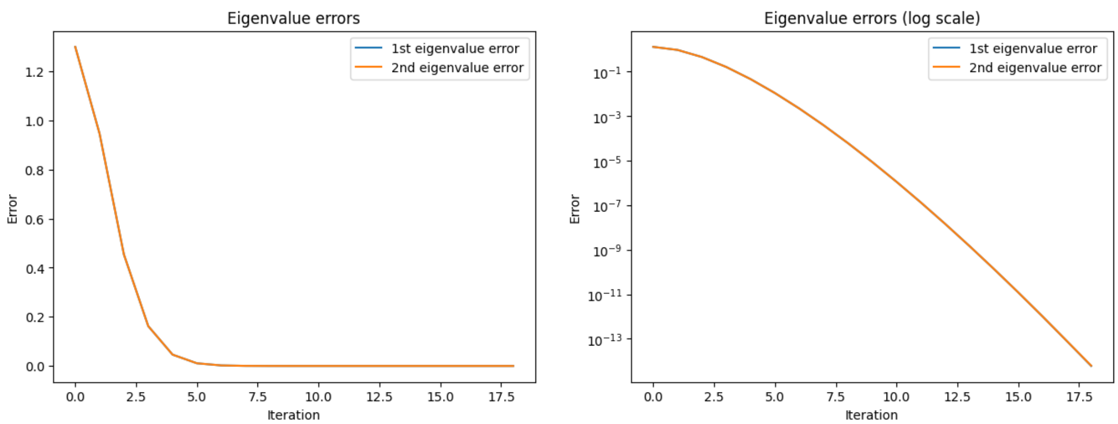
```
+-------------+------------------------+------------------------+---------------
---------+------------------+--------------------+
|   Iteration |   1st eigenvalue error |   2nd eigenvalue error |   Frobenius
norm error |   Inf norm error |   First norm error |
|-------------+------------------------+------------------------+---------------
---------+------------------+--------------------|
|           0 |             1.29927    |             1.29927    |
1.83745     |      1.79403     |        1.79403     |
|           1 |             0.945668   |             0.945668   |
1.33738     |      1.19785     |        1.19785     |
|           2 |             0.454066   |             0.454066   |
0.642146    |      0.641025    |        0.641025    |
|           3 |             0.162295   |             0.162295   |
0.229519    |      0.19094     |        0.19094     |
|           4 |             0.046218   |             0.046218   |
0.0653622   |      0.0653385   |        0.0653385   |
|           5 |             0.010943   |             0.010943   |
0.0154758   |      0.0123776   |        0.0123776   |
|           6 |             0.00221778 |             0.00221778 |
0.00313641  |      0.00313604  |        0.00313604  |
|           7 |             0.00039294 |             0.00039294 |
```

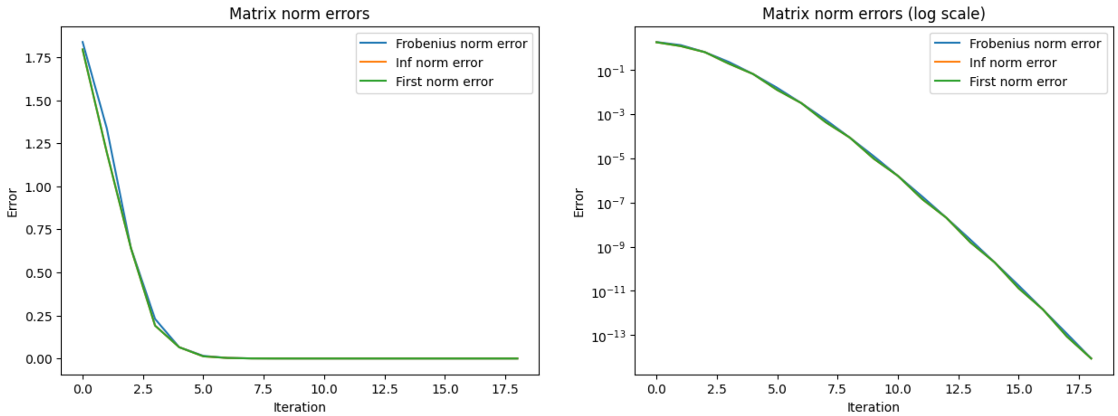
```
0.000555701 |      0.000433847 |        0.000433847 |
|           8 |             6.18488e-05 |            6.18488e-05 |
8.74673e-05 |      8.74631e-05 |        8.74631e-05 |
|           9 |             8.75801e-06 |            8.75801e-06 |
1.23857e-05 |      9.51366e-06 |        9.51366e-06 |
|          10 |             1.12711e-06 |            1.12711e-06 |
1.59397e-06 |      1.59393e-06 |        1.59393e-06 |
|          11 |             1.32937e-07 |            1.32937e-07 |
1.88001e-07 |      1.42727e-07 |        1.42727e-07 |
|          12 |             1.4471e-08  |            1.4471e-08  |
2.04651e-08 |      2.04648e-08 |        2.04648e-08 |
|          13 |             1.46257e-09 |            1.46257e-09 |
2.06839e-09 |      1.55651e-09 |        1.55651e-09 |
|          14 |             1.37953e-10 |            1.37953e-10 |
1.95095e-10 |      1.95094e-10 |        1.95094e-10 |
|          15 |             1.21983e-11 |            1.21981e-11 |
1.72503e-11 |      1.28922e-11 |        1.28922e-11 |
|          16 |             1.01508e-12 |            1.01486e-12 |
1.43529e-12 |      1.43528e-12 |        1.43528e-12 |
|          17 |             7.94217e-14 |            7.97624e-14 |
1.13026e-13 |      8.41337e-14 |        8.41337e-14 |
|          18 |             6.04186e-15 |            6.23957e-15 |
8.66863e-15 |      8.66863e-15 |        8.66863e-15 |
+-------------+------------------------+------------------------+---------------
---------+------------------+--------------------+
```

1.4.4 Grafični prikaz napak

[ ]:
```python
import matplotlib.pyplot as plt

fig = plt.figure(figsize=(15, 5))
fig.add_subplot(1, 2, 1) # type: ignore
plt.plot(first_eigenvalue_error, label='1st eigenvalue error')
plt.plot(second_eigenvalue_error, label='2nd eigenvalue error')
plt.title('Eigenvalue errors')
plt.legend()
plt.xlabel('Iteration')
plt.ylabel('Error')

fig.add_subplot(1, 2, 2) # type: ignore
plt.plot(first_eigenvalue_error, label='1st eigenvalue error')
plt.plot(second_eigenvalue_error, label='2nd eigenvalue error')
plt.title('Eigenvalue errors (log scale)')
plt.legend()
plt.xlabel('Iteration')
plt.ylabel('Error')
plt.yscale('log')
```

```python
plt.show()

fig = plt.figure(figsize=(15, 5))
fig.add_subplot(1, 2, 1) # type: ignore
plt.plot(fro_norm_error, label='Frobenius norm error')
plt.plot(inf_norm_error, label='Inf norm error')
plt.plot(first_norm_error, label='First norm error')
plt.legend()
plt.title('Matrix norm errors')
plt.xlabel('Iteration')
plt.ylabel('Error')

fig.add_subplot(1, 2, 2) # type: ignore
plt.plot(fro_norm_error, label='Frobenius norm error')
plt.plot(inf_norm_error, label='Inf norm error')
plt.plot(first_norm_error, label='First norm error')
plt.yscale('log')
plt.legend()
plt.title('Matrix norm errors (log scale)')
plt.xlabel('Iteration')
plt.ylabel('Error')
plt.show()
```




Matrix norm errors
Frobenius norm error
Inf norm error
First norm error
Error
Iteration
Matrix norm errors (log scale)
Frobenius norm error
Inf norm error
First norm error
Error
Iteration

[ ]: