

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ
ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря СІКОРСЬКОГО»
Фізико-технічний інститут

КОМП'ЮТЕРНИЙ ПРАКТИКУМ № 3.
РОЗВ'ЯЗАННЯ СЛАБ ІТЕРАЦІЙНИМИ МЕТОДАМИ

Виконав
студент 3 курсу ФТІ
групи ФІ-21
Климентьев Максим Андрійович

Перевірів:

Оцінка:

Зміст

| | |
|--|---|
| 1 Дані | 3 |
| 2 Письмовий етап приведення матриці до діагональної переваги | 3 |
| 3 Результати перших трьох та останньої ітерацій методу | 5 |
| 4 Вектор нев'язки на кожній ітерації | 6 |
| 5 Лістинг програми | 8 |

1 Дані

| Варіант | Матриця системи А | | | | | Вектор правої частини b |
|---------|-------------------|-------|--------|-------|-------|-------------------------|
| 10 | 6.59 | 1.28 | 0.79 | 1.195 | -0.21 | 2.1 |
| | 0.92 | 3.83 | 1.3 | -1.63 | 1.02 | 0.36 |
| | 1.15 | -2.46 | 5.77 | 2.1 | 1.483 | 3.89 |
| | 1.285 | 0.16 | 2.1 | 5.77 | -18 | 11.04 |
| | 0.69 | -1.68 | -1.217 | 9 | -6 | -0.27 |

2 Письмовий етап приведення матриці до діагональної переваги

З ФІ-21 Климентьев Максим

Friday, March 28, 2025 3:53 PM

| Варіант | Матриця системи А | | | | | Вектор правої частини b |
|---------|-------------------|-------|--------|-------|-------|-------------------------|
| 10 | 6.59 | 1.28 | 0.79 | 1.195 | -0.21 | 2.1 |
| | 0.92 | 3.83 | 1.3 | -1.63 | 1.02 | 0.36 |
| | 1.15 | -2.46 | 5.77 | 2.1 | 1.483 | 3.89 |
| | 1.285 | 0.16 | 2.1 | 5.77 | -18 | 11.04 |
| | 0.69 | -1.68 | -1.217 | 9 | -6 | -0.27 |

```

from time import time
A = np.array([
    [6.59, 1.28, 0.79, 1.195, -0.21],
    [0.92, 3.83, 1.3, -1.63, 1.02],
    [1.15, -2.46, 5.77, 2.1, 1.483],
    [1.285, 0.16, 2.1, 5.77, -18],
    [0.69, -1.68, -1.217, 9, -6]
])
unit = [0, 0, 0, 0]
member = [0, 0, 0, 0]
start_time = time()
while not if_diag_pen(A):
    A = np.array([
        [6.59, 1.28, 0.79, 1.195, -0.21],
        [0.92, 3.83, 1.3, -1.63, 1.02],
        [1.15, -2.46, 5.77, 2.1, 1.483],
        [1.285, 0.16, 2.1, 5.77, -18],
        [0.69, -1.68, -1.217, 9, -6]
    ])
    unit = np.random.uniform(-4, 4)
    member = np.random.randint(0, A.shape[0], size=4)
    for member_index in range(member.shape[0]):
        while member_index == member_index-1:
            member_index = np.random.randint(0, A.shape[0])
        # member = [u, v, w, z]
        A[2] += A[member[0]]*unit[0]
        A[2] += A[member[1]]*unit[1]
        A[3] += A[member[2]]*unit[2]
        A[4] += A[member[3]]*unit[3]
        if time() - start_time > 480:
            break
    print(member)
    print(unit)
    print(if_diag_pen(A))

```

Зробив сто випадкових пошук мотивів коефіцієнтів, щоб привести матрицю до діагональної переваги

Отже:

```

A = np.array([
    [6.59, 1.28, 0.79, 1.195, -0.21],
    [0.92, 3.83, 1.3, -1.63, 1.02],
    [1.15, -2.46, 5.77, 2.1, 1.483],
    [1.285, 0.16, 2.1, 5.77, -18],
    [0.69, -1.68, -1.217, 9, -6]
])
b = np.array([
    [2.1],
    [0.36],
    [3.89],
    [11.04],
    [-0.27]
])
A[1] += A[4]*1.6
# f = 0.2880613 0.7071557 -2.2751403 0.41375988
A[2] += A[1]*0.8
# A[2] == A[2]+0.2880613
A[3] += A[4]*-2.28
# A[3] == A[3]-1.18926847
A[4] += A[3]*0.4
# A[4] == A[4]+0.2880613
if_diag_pen(A, print_dev=True)
A

```

```

array([[ 6.59 ,  1.28 ,  0.79 ,  1.195 , -0.21 ],
       [ 1.8025 ,  3.41 ,  0.99575 ,  0.62 , -0.48 ],
       [ 2.824 ,  0.268 ,  0.5666 ,  2.996 ,  1.899 ],
       [-0.2882 ,  3.9984 ,  6.67476 , -34.79 , -6.32 ],
       [ 0.57472 , -0.08384 ,  0.732904 ,  3.1 , -7.728 ]])

```

$$\Rightarrow \begin{pmatrix} 6.59 & 1.28 & 0.79 & 1.195 & -0.21 \\ 0.92 & 3.83 & 1.3 & -1.63 & 1.02 \\ 1.15 & -2.46 & 5.77 & 2.1 & 1.483 \\ 1.285 & 0.16 & 2.1 & 5.77 & -18 \\ 0.69 & -1.68 & -1.217 & 9 & -6 \end{pmatrix} \cdot \frac{1}{9} \Rightarrow$$

$$\Rightarrow \begin{pmatrix} 6.59 & 1.28 & 0.79 & 1.195 & -0.21 \\ 7.0925 & 3.41 & 0.99575 & 0.62 & -0.48 \\ 1.15 & -2.46 & 5.77 & 2.1 & 1.483 \\ 1.285 & 0.16 & 2.1 & 5.77 & -18 \\ 0.69 & -1.68 & -1.217 & 9 & -6 \end{pmatrix} \cdot 0.8 \Rightarrow$$

$$\Rightarrow \left(\begin{array}{ccccc|c} 6.59 & 1.28 & 0.79 & 1.195 & -0.21 & \\ 10945 & 3.41 & 099575 & 0.62 & -0.48 & \\ 2.024 & 0.268 & 6.5666 & 2.596 & 1.099 & \\ 1.285 & 0.16 & 2.1 & 5.77 & -78 & \\ 0.69 & -1.68 & -1.217 & 9 & -6 & \end{array} \right) \xrightarrow{\cdot -2.28 +}$$

$$\Rightarrow \left(\begin{array}{ccccc|c} 6.59 & 1.28 & 0.79 & 1.195 & -0.21 & \\ 10945 & 3.41 & 099575 & 0.62 & -0.48 & \\ 2.024 & 0.268 & 6.5666 & 2.596 & 1.099 & \\ -0.2882 & 3.9902 & 4.87476 & -14.75 & -4.32 & \\ 0.69 & -1.68 & -1.217 & 9 & -6 & \end{array} \right) \xrightarrow{\cdot 0.4 +}$$

$$\Rightarrow \left(\begin{array}{ccccc|c} 6.59 & 1.28 & 0.79 & 1.195 & -0.21 & 6.59 > 3.055 \\ 10945 & 3.41 & 099575 & 0.62 & -0.48 & 3.41 > 3.78825 \\ 2.024 & 0.268 & 6.5666 & 2.596 & 1.099 & 6.5666 > 5.987 \\ -0.2882 & 3.9902 & 4.87476 & -14.75 & -4.32 & 14.75 > 13.47336 \\ 0.5772 & -0.0884 & -0.73204 & 3.1 & -7.728 & 7.728 > 4.491464 \end{array} \right)$$

$$x_1 = \frac{1}{6.59} \left(0 - 1.28x_2 - 0.79x_3 - 1.195x_4 + 0.21x_5 \right) + \frac{2.1}{6.59}$$

$$x_2 = \frac{1}{3.41} \left(-10945x_1 - 0 - 099575x_3 - 0.62x_4 + 0.48x_5 \right) + \frac{0.36}{3.41}$$

$$x_3 = \frac{1}{6.5666} \left(-2.024x_1 - 0.268x_2 - 0 - 2.596x_4 - 1.099x_5 \right) + \frac{3.89}{6.5666}$$

$$x_4 = \frac{1}{-14.75} \left(+0.2882x_1 - 3.9902x_2 - 4.87476x_3 + 0 + 4.32x_5 \right) + \frac{11.04}{-14.75}$$

$$x_5 = \frac{1}{7.728} \left(-0.5772x_1 + 0.0884x_2 - 0.73204x_3 - 3.1x_4 + 0 \right) - \frac{0.27}{7.728}$$

```

C = np.zeros_like(A)
d = np.zeros_like(A)
for row in range(A.shape[0]):
    C[row] = (A[row]) / A[row, row]
    C[row, row] = 0
    d[row] = d[row] / A[row, row]
print(C)
print(d)

```

$$x^{(k+1)} = Cx^{(k)} + d$$

$$C = \begin{pmatrix} 0 & -0.19423369 & -0.1198786 & -0.1803576 & 0.03186646 \\ -0.3203812 & 0 & -0.2520083 & -0.18781878 & 0.14076396 \\ -0.30812648 & -0.0408126 & 0 & -0.3953336 & -0.16736211 \\ -0.01953898 & 0.27053559 & 0.3304022 & 0 & -0.29288136 \\ 0.07436853 & -0.02084886 & 0.09483747 & 0.40113872 & 0 \end{pmatrix}$$

$$d = \begin{pmatrix} 0.31866464 \\ 0.08577713 \\ 0.62802668 \\ -0.79021017 \\ -0.56835404 \end{pmatrix}$$

3 Результати перших трьох та останньої ітерацій методу

$$[0]x : \begin{pmatrix} 0.31866464 \\ 0.08577713 \\ 0.62802668 \\ -0.79021017 \\ -0.56835404 \end{pmatrix}$$

$$[1]x : \begin{pmatrix} 0.35189848 \\ -0.1360347 \\ 0.93382286 \\ -0.39921256 \\ -0.80300943 \end{pmatrix}$$

$$[2]x : \begin{pmatrix} 0.27994407 \\ -0.34009852 \\ 0.81732981 \\ -0.29008047 \\ -0.61228626 \end{pmatrix}$$

$$[15]x : \begin{pmatrix} 0.32955216 \\ -0.26880975 \\ 0.80811961 \\ -0.41730668 \\ -0.63169572 \end{pmatrix}$$

4 Вектор нев'язки на кожній ітерації

$$[0]r : \begin{pmatrix} 0.21901101 \\ 0.75637832 \\ 2.00804118 \\ 5.76721468 \\ 1.81341687 \end{pmatrix}$$

$$[1]r : \begin{pmatrix} 0.47417961 \\ 0.69585762 \\ 0.76496323 \\ 1.60969837 \\ 1.47390867 \end{pmatrix}$$

$$[2]r : \begin{pmatrix} 0.26287021 \\ 0.21849338 \\ 0.29258683 \\ 2.18535873 \\ 0.22868634 \end{pmatrix}$$

$$[3]r : \begin{pmatrix} 0.13645018 \\ 0.10685171 \\ 0.25419385 \\ 0.10085488 \\ 0.47439844 \end{pmatrix}$$

$$[4]r : \begin{pmatrix} 0.07540986 \\ 0.08639293 \\ 0.03490878 \\ 0.57296527 \\ 0.01644702 \end{pmatrix}$$

$$[5]r : \begin{pmatrix} 0.01774368 \\ 0.01585437 \\ 0.07323021 \\ 0.08107878 \\ 0.11986356 \end{pmatrix}$$

$$[6]r : \begin{pmatrix} 0.02458714 \\ 0.0248991 \\ 0.00391974 \\ 0.1391443 \\ 0.02637121 \end{pmatrix}$$

$$[7]r : \begin{pmatrix} 0.00073858 \\ 0.00045966 \\ 0.01873128 \\ 0.0457133 \\ 0.02727432 \end{pmatrix}$$

$$[8]r : \begin{pmatrix} 0.00652564 \\ 0.00657839 \\ 0.00435757 \\ 0.0285816 \\ 0.01177387 \end{pmatrix}$$

$$[9]r : \begin{pmatrix} 0.0009979 \\ 0.00127251 \\ 0.00418348 \\ 0.01722922 \\ 0.00511326 \end{pmatrix}$$

$$[10]r : \begin{pmatrix} 0.00156045 \\ 0.00151075 \\ 0.00189869 \\ 0.00451853 \\ 0.00403224 \end{pmatrix}$$

$$[11]r : \begin{pmatrix} 0.000539 \\ 0.00060713 \\ 0.00077069 \\ 0.0053632 \\ 0.0006388 \end{pmatrix}$$

$$[12]r : \begin{pmatrix} 0.00031669 \\ 0.00029262 \\ 0.00063982 \\ 0.00024233 \\ 0.00118112 \end{pmatrix}$$

$$[13]r : \begin{pmatrix} 1.99277455e-04 \\ 2.12698169e-04 \\ 9.03525690e-05 \\ 1.46380418e-03 \\ 4.09051201e-05 \end{pmatrix}$$

$$[14]r : \begin{pmatrix} 4.85115522e-05 \\ 3.96531294e-05 \\ 1.85525718e-04 \\ 1.95977908e-04 \\ 3.05581645e-04 \end{pmatrix}$$

$$[15]r : \begin{pmatrix} 6.13856585e-05 \\ 6.33931185e-05 \\ 9.05118633e-06 \\ 3.52829061e-04 \\ 6.51510711e-05 \end{pmatrix}$$

5 Лістинг програми

```
1 def Iteration_simple_method(C, d, epsilon=1e-4):
2     xk = np.zeros(b.shape)
3     xk1 = C@xk + d
4
5     count = 0
6
7     r = abs(b_start - A_start@xk1)
8     print(f"[{count}] r: \n {r} \n")
9     print(f"[{count}] x: \n {xk1} \n\n")
10
11     q = np.zeros(b.shape)
12     for row in range(C.shape[0]):
13         q = min(sum(abs(C[row])), sum(abs(C[:, row])))
14
15     while q/(1-q) * np.max(abs(xk1 - xk)) >= epsilon:
16         xk = xk1
17         xk1 = C@xk + d
18
19         count += 1
20
21         r = abs(b_start - A_start@xk1)
22         print(f"[{count}] r: \n {r} \n")
23         print(f"[{count}] x: \n {xk1} \n\n")
24
25     return xk1
26
```

Лістинг 1: Simple Iteration Method