Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный технический университет имени
Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет: Информатики и систем управления
Кафедра: Теоретической информатики и компьютерных технологий

Лабораторная работа

«Алгоритм Тисменецкого»

по дисциплине «*Численные методы линейной алгебры»*

Студент ИУ9-72: А.Б. Барлука
Преподаватель: А.Ю. Голубков

Москва 2020

1. **Цель:**

Реализовать алгоритм Тисменецкого для построения ILU-предобуславливателей.

2. **Постановка задачи:**

Пусть A – невырожденная матрица размера $n \times n$. Построить неполное LDU-разложение $A = LDU + R$, где L – нижняя треугольная матрица, D – диагональная матрица, U – верхняя треугольная матрица, R – матрица невязок.

3. **Теоретические сведения:**

Пусть

$$A = LDU, \tag{1}$$

где D – диагональная, а L и U – нижняя и верхняя унитреугольные матрицы соответственно. Тисменецкий определяет матрицы L_i и U_i как

$$\mathrm{L}_i = \mathrm{I} + (\mathrm{L} - \mathrm{I})\mathrm{e}_i{\mathrm{e}_i}^T = \text{и} \quad \mathrm{U}_i = \mathrm{I} + \mathrm{e}_i{\mathrm{e}_i}^T(\mathrm{U} - \mathrm{I}),$$

где e_i задает i-ый столбец единичной матрицы. Т.к. L – I является строго нижней треугольной, для j <= I имеем $e_j^T (L - I)\, e_i = 0$, поэтому

$$\mathrm{L}_i^{-1} = \mathrm{I} + (\mathrm{L} - \mathrm{I})\,\mathrm{e}_i\,{\mathrm{e}_i}^T \tag{2}$$

и

$$\mathrm{L}_i\mathrm{L}_j = \mathrm{I} + (\mathrm{L} - \mathrm{I})\,(\mathrm{e}_j\,{\mathrm{e}_j}^T + \mathrm{e}_i\,{\mathrm{e}_i}^T).$$

Простая индукция, базирующаяся на последнем уравнении показывает, что

$$\mathrm{L}_1\mathrm{L}_2\ldots\mathrm{L}_n = \mathrm{I} + (\mathrm{L} - \mathrm{I}) = \mathrm{L} \tag{3}$$

Для U аналогично. Определим теперь матрицы $\overline{\mathrm{L}_i} = \mathrm{L}_1\mathrm{L}_2\ldots\mathrm{L}_i$ и $A_i = \overline{\mathrm{L}_i}^{-1} A\, \overline{\mathrm{U}_i}^{-1}$ так, что если $A_0 = A$,

$$A_i = \mathrm{L}_i^{-1} A_{i-1} U_i^{-1},\ i = 1,2,\ldots,n. \qquad (4)$$

Из уравнений (1) и (3) $A_n = D = diag(d_i)$, а из уравнений (2) и (4) следует, что

$$A_i = (I - (L - I)\, \mathrm{e}_i\mathrm{e}_i^T)\, A_{i-1}(I - \mathrm{e}_i\mathrm{e}_i^T(U - I)). \qquad (5)$$

Теперь i-ый столбец A_i задается $A_i e_i$, поэтому

$$A_i e_i = (I - (L - I)\, \mathrm{e}_i\mathrm{e}_i^T)\, A_{i-1}\mathrm{e}_i\ . \qquad (6)$$

Выберем i-ый столбец так, что $A_i e_i$ формирует i-ый столбец диагональной матрицы D. Замена $e_i d_i$ на $A_i e_i$ в уравнении (6) дает уравнение

$$e_i d_i = A_{i-1}\mathrm{e}_i - (L - I)\mathrm{e}_i\mathrm{e}_i^T A_{i-1}\mathrm{e}_i \qquad (7)$$

которое при умножении слева на e_i^T можно записать как $\mathrm{d}_i = \mathrm{e}_i^T A_{i-1}\mathrm{e}_i$. Обратная замена d_i а уравнении (7) дает

$$L\mathrm{e}_i = A_{i-1}\mathrm{e}_i\, \mathrm{d}_i^{-1} \qquad (8)$$

так что i-ый столбец $L -$ это i-ый столбец матрицы A_i, разделенный на свой i-ый элемент. Общий вид A_i может быть выведен из следующего примера, для которого i = 2 и n = 5:

$$A_2 = \begin{bmatrix} \mathrm{d}_1 & 0 & * & 0 & 0 \\ 0 & \mathrm{d}_2 & * & 0 & 0 \\ 0 & 0 & \mathrm{a}_{33} & \mathrm{a}_{34} & \mathrm{a}_{35} \\ 0 & 0 & \mathrm{a}_{43} & \mathrm{a}_{44} & \mathrm{a}_{45} \\ 0 & 0 & \mathrm{a}_{53} & \mathrm{a}_{54} & \mathrm{a}_{55} \end{bmatrix} \qquad (9)$$

где (*) представляют нули для полного разложения. Если подставить $L\mathrm{e}_i$ из уравнения (8) обратно в уравнение (5), получим

$$A_i = A_{i-1} - \frac{A_{i-1}\mathrm{e}_i\mathrm{e}_i^T A_{i-1}}{d_i} + \mathrm{e}_i d_i \mathrm{e}_i^T. \qquad (10)$$

Чтобы получить неполную версия разложения из вышесказанного, обозначим «$\widehat{\ }$» неполные копии A, L и d_i и положим, что $\widehat{A_{\iota-1}}$ уже известна. Пусть (по аналогии с уравнением (8))

$$\hat{\mathrm{L}}\mathrm{e}_i = (\widehat{A_{\iota-1}}\mathrm{e}_i - \mathrm{f}_i)\widehat{d_\iota}^{-1} \quad и \quad \mathrm{e}_i{}^T\hat{\mathrm{U}} = \widehat{d_\iota}^{-1}(\mathrm{e}_i{}^T\widehat{A_{\iota-1}} - \mathrm{g}_i{}^T)$$

где $\widehat{d_\iota} = \mathrm{e}_i{}^T\widehat{A_{\iota-1}}\mathrm{e}_i$ и векторы f_i и $\mathrm{g}_i{}^T$ состоят из тех элементов $\widehat{A_{\iota-1}}\mathrm{e}_i$ и $\mathrm{e}_i{}^T\widehat{A_{\iota-1}}$, которые должны быть исключены из неполных множителей $\hat{\mathrm{L}}$ и $\hat{\mathrm{U}}$. Пусть r и s – число ненулевых элементов в j-ой строке и j-ом столбце соответственно. Тогда $nrow = alpha * r$ и *ncol* = *alpha* * *s* – число наименьших по модулю элементов, которые будут исключены на j-том шаге алгоритма, а параметр *alpha* задается пользователем. При *alpha* = 1, L и U имеют ту же степень разреженности, что и стандартный алгоритм неполного разложения Холецкого ICCG(0). С увеличением *alpha* улучшается предобуславлитель, но также увеличивается сложность алгоритма.

Определим

$$\mathrm{u}_i = \widehat{A_{\iota-1}}\mathrm{e}_i - \mathrm{f}_i \qquad \text{и} \qquad \mathrm{v}_i{}^T = \mathrm{e}_i{}^T\widehat{A_{\iota-1}} - \mathrm{g}_i{}^T$$

и заметим, что в «шапочных» версиях уравнения (9) некоторые элементы, отмеченные (*), являются нулевыми для полного разложения, но могут быть ненулевыми и должны быть включены в f_i и g_i. Это гарантирует, что u_i и $\mathrm{v}_i{}^T$ имеют нули на корректных позициях и что оставшиеся элементы этих векторов являются соответствующими элементами $\widehat{A_{\iota-1}}\mathrm{e}_i$ и $\mathrm{e}_i{}^T\widehat{A_{\iota-1}}$. Матрица $\widetilde{A_\iota}$ затем задается следующим образом

$$\widetilde{A_\iota} = \widehat{A_{\iota-1}} - (\mathrm{u}_i\mathrm{v}_i{}^T + \mathrm{u}_i\mathrm{g}_i{}^T + \mathrm{f}_i\mathrm{v}_i{}^T)\,\widehat{d_\iota}^{-1} \tag{11}$$

и обновленная матрица $\widehat{A_\iota}$ получается из $\widetilde{A_\iota}$ присвоением её i-му диагональному элементу значения $\widehat{d_\iota}$.

4. Реализация на Python:

Ниже представлен Листинг 1, содержащий реализацию программы *main.py* на языке Python 3.7 с использованием библиотеки NumPy.

На вход подается текстовый файл *in.txt*, в котором задана матрица *A,* а также коэффициент *alpha* – целое число. После завершения чтения входного файла проверяется условие невырожденности матрицы, иначе программа выдает исключение AssertionError.

```
import sys
import numpy as np


def read(file):
    m = []
    rows = file.readlines()
    N = len(rows)
    for row in rows:
        print(row.rstrip().split())
        m.append([float(i) for i in row.rstrip().split()])
        assert len(m[len(m)-1]) == N
    return m, N


def eliminate_n_max(arr, n):
    m = dict()
    for i in range(len(arr)):
        ind = arr[i]
        l = m.get(arr[i], [])
        l.append(i)
        m[ind] = l
    for key in sorted(m.keys(), key=lambda x: abs(x)):
        indexes = m[key]
        for i in indexes:
            if n == 0:
                break
            arr[i] = 0
            n -= 1
    return arr


def tismenetsky_incomplete(A, alpha):
    N = len(A)
    Ai_s = A.copy()

    L = np.zeros((N, N))
```

```python
    U = np.zeros((N, N))

    for i in range(N):
        print("i:", i)

        I = np.eye(N)
        ei = I[:, i]
        ei.shape = (N, 1)
        ei_T = ei.copy()
        ei_T.shape = (1, N)

        di = ei_T @ Ai_s @ ei

        # nrow ncol
        r = np.count_nonzero(Ai_s[i, :])
        nrow = alpha * r
        s = np.count_nonzero(Ai_s[:, i])
        ncol = alpha * s

        fi = eliminate_n_max(Ai_s[i].copy(), nrow)
        fi.shape = (N, 1)
        gi_T = eliminate_n_max(Ai_s[:, i].copy(), ncol)
        gi_T.shape = (1, N)

        ui = Ai_s @ ei - fi
        vi_T = ei_T @ Ai_s - gi_T

        # L
        L[:, i] = ((Ai_s @ ei - fi) / di).flatten()

        # U
        U[i, :] = (ei_T @ Ai_s - gi_T) / di

        # A (An = D)
        Ai_w = Ai_s - (ui @ vi_T + ui @ gi_T + fi @ vi_T + fi @ gi_T) / di
        Ai_w[i][i] = di
        Ai_s = Ai_w.copy()

    return L, Ai_s, U


def main():
    np.set_printoptions(precision=2, suppress=True)
    if len(sys.argv) != 3:
        print("usage python main.py <in.txt> <alpha>")
        sys.exit(-1)
    _, filename, alpha = sys.argv
    alpha = int(alpha)
    with open(filename) as f:
        m, N = read(f)
        A = np.array(m)
        assert np.linalg.matrix_rank(A) == N

        L_hat, D_hat, U_hat = tismenetsky_incomplete(A, alpha)
        print("L\n", L_hat)
        print("D\n", D_hat)
        print("U\n", U_hat)

        ldu = L_hat @ D_hat @ U_hat
```

```
        print("LDU\n", ldu)
        print("R = A - LDU\n", A - ldu)


if __name__ == "__main__":
    main()
```

5. Результат:

На вход была подана матрица, параметр *alpha* = 2

1	2	-1	0	3	5	0	1	1	0
3	0	0	0	0	1	0	0	1	0
5	-7	8	2	0	0	0	2	0	0
7	0	3	4	0	3	0	0	1	0
0	1	0	0	0	1	0	1	0	0
2	0	2	1	0	0	2	0	1	0
1	0	0	0	0	0	1	0	0	0
1	0	0	0	0	1	1	0	1	0
1	0	0	0	1	0	1	0	0	1
1	0	0	1	0	0	1	0	1	0

Построенные матрицы L, D, U:

```
L
 [[ 1.   -0.    0.    0.   -0.   -0.    0.   -0.    0.   -0.  ]
 [ 3.    1.    0.    0.   -0.   -0.    0.   -0.    0.   -0.  ]
 [ 5.    2.83  1.    0.   -0.   -0.    0.   -0.    0.   -0.  ]
 [ 7.    2.33  0.67  1.   -0.   -0.    0.   -0.    0.    0.  ]
 [ 0.   -0.17  0.11 -0.08  1.   -0.    0.   -0.    0.   -0.  ]
 [ 2.    0.67  0.44  0.04  1.35  1.    0.   -0.    0.    0.  ]
 [ 1.    0.33  0.    0.   -0.    0.19  1.   -0.    0.   -0.  ]
 [ 1.    0.33  0.    0.   -0.   -0.37  2.77 -0.    0.   -0.  ]
 [ 1.    0.33  0.    0.   -0.31  0.82 -1.03  0.47  0.   -0.  ]
 [ 1.    0.33  0.    0.37 -0.81 -0.04  1.73 -0.27  1.8  -0.  ]]
```

```
D
 [[ 1.    0.    0.    0.    0.    0.    0.    0.    0.    0.  ]
 [ 0.   -6.    0.    0.    0.    0.    0.    0.    0.    0.  ]
 [ 0.    0.    4.5   0.    0.    0.    0.    0.    0.    0.  ]
 [ 0.    0.    0.    2.67  0.    0.    0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.   -3.25  0.    0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.    0.   -1.79  0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.    0.    0.    0.63  0.    0.    0.  ]
 [ 0.    0.    0.    0.    0.    0.    0.   -1.53  0.    0.  ]
 [ 0.    0.    0.   -0.    0.    0.    0.    0.    0.78  0.  ]
 [ 0.    0.    0.   -0.    0.    0.    0.    0.    0.   -3.09]]
```

```
U
 [[ 1.    2.   -1.    0.    3.    5.    0.    1.    1.    0.  ]
 [-0.    1.   -0.5  -0.    1.5   2.33 -0.    0.5   0.33 -0.  ]
 [ 0.    0.    0.   -0.22  2.33  3.26  0.    1.22  0.15  0.  ]
 [ 0.    0.    0.    0.   -2.62 -3.42  0.   -1.37 -0.67 -0.37]
 [-0.   -0.   -0.   -0.   -0.   -0.2  -0.    0.13  0.17 -0.  ]
 [-0.   -0.   -0.    0.   -0.    1.   -1.11  0.96 -0.48 -0.  ]
 [ 0.    0.    0.    0.    0.    0.    1.   -2.26 -0.78 -1.73]
 [-0.   -0.   -0.    0.   -0.    0.   -0.   -0.   -0.47  0.27]
 [ 0.    0.    0.   -0.    0.   -0.    0.    0.    0.   -1.8 ]
 [-0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.  ]]
```

Произведение LDU:

```
LDU
 [[ 1.    2.   -1.    0.    3.    5.    0.    1.    1.    0.  ]
 [ 3.    0.    0.    0.    0.    1.    0.    0.    1.    0.  ]
 [ 5.   -7.    3.5  -1.    0.   -0.    0.    2.    0.    0.  ]
 [ 7.    0.    0.   -0.67  0.    3.   -0.    0.    1.   -1.  ]
 [ 0.    1.   -0.5  -0.11  3.25  5.38  0.    1.    0.    0.08]
 [ 2.    0.    0.   -0.44  4.38  5.89  2.   -0.    1.   -0.04]
 [ 1.    0.    0.   -0.    0.    0.    1.   -1.74 -0.   -1.09]
 [ 1.    0.    0.    0.    0.    1.    1.   -3.3  -1.35 -3.01]
 [ 1.    0.    0.   -0.    0.   -1.35  1.    0.17  2.06  0.93]
 [ 1.    0.    0.   -0.   -2.62 -3.53  1.   -3.42 -0.97 -4.66]]
```

Матрица невязок:

```
R = A - LDU
 [[ 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.  ]
 [ 0.    0.    4.5   3.   -0.    0.    0.   -0.   -0.    0.  ]
 [ 0.    0.    3.    4.67 -0.    0.    0.   -0.   -0.    1.  ]
 [ 0.    0.    0.5   0.11 -3.25 -4.38 -0.    0.   -0.   -0.08]
 [ 0.    0.    2.    1.44 -4.38 -5.89  0.    0.   -0.    0.04]
 [ 0.    0.    0.    0.    0.   -0.    0.    1.74  0.    1.09]
 [ 0.    0.    0.   -0.    0.   -0.    0.    3.3   2.35  3.01]
 [ 0.    0.    0.    0.    1.    1.35  0.   -0.17 -2.06  0.07]
 [ 0.    0.    0.    1.    2.62  3.53  0.    3.42  1.97  4.66]]
```

6. Вывод:

На практике алгоритм дает один из самых эффективных ILU-предобуславливателей по скорости итераций. Это также приводит к тому, что треугольные множители гарантированно являются вещественными и несингулярными для любой симметричной положительно определенной матрицы A. Это яркий контраст с IC-предобуславливанием, которое может не сработать, если A не является M-матрицей или H-матрицей. Его недостаток — это то, что он требует больше времени и памяти для вычисления множителей в начале.

Список используемой литературы

1) Krylov Solvers for Linear Algebraic Systems – C. G. Broyden, M. T. Vespucci, 2004
2) A New Preconditioning Technique for Solving Large Sparse Linear Systems – M. Tismenetsky, 1991