НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ
“КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ ИМЕНИ ИГОРЯ СИКОРСКОГО”
ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ №3

Дисциплина: «Методы вычислений»

Тема: «Решение систем линейных алгебраических уравнений (СЛАУ) итерационными методами»

Выполнила

студентка 3 курса группы ФИ-41

Лавягина Ольга Алексеевна

Проверила

Стёпочкина Ирина Валерьевна

Киев — 2017 года

ОГЛАВЛЕНИЕ

1 ИСХОДНАЯ СИСТЕМА

Рассматирваем систему вида $Ax = b$, где $A\,(n \times n)$ — матрица системы, b — вектор правой части, x — вектор решения.

Для варианта 8 матрица системы имеет вид

$$A = \begin{bmatrix} 3.81 & 0.25 & 1.28 & 1.75 \\ 2.25 & 1.32 & 5.58 & 0.49 \\ 5.31 & 7.28 & 0.98 & 1.04 \\ 10.39 & 2.45 & 3.35 & 2.28 \end{bmatrix},$$

а вектор правой части —

$$b = \begin{bmatrix} 4.21 \\ 7.47 \\ 2.38 \\ 11.48 \end{bmatrix}.$$

2 ПРИВЕДЕНИЕ МАТРИЦЫ К ДИАГОНАЛЬНОМУ ПРЕОБЛАДАНИЮ

Нужно привести матрицу к такому виду, чтобы на диагоналях стояли максимальные элементы, причём сумма не диагональных была меньше самих диагональных элементов.

Для этого поменяем местами первую и четвёртую строки, а также вторую и третью.

$$A = \begin{bmatrix} 10.39 & 2.45 & 3.35 & 2.28 \\ 5.31 & 7.28 & 0.98 & 1.04 \\ 2.25 & 1.32 & 5.58 & 0.49 \\ 3.81 & 0.25 & 1.28 & 1.75 \end{bmatrix},$$

а вектор правой части —

$$b = \begin{bmatrix} 11.48 \\ 2.38 \\ 7.47 \\ 4.21 \end{bmatrix}.$$

Все строки матрицы имеют нужный вид, кроме второй четвёртой. От второй строки отнимем четвёртую. Четвёртую умножим на -21 и прибавим к ней первую строку полученной матрицы, умноженную на 8

$$A = \begin{bmatrix} 10.39 & 2.45 & 3.35 & 2.28 \\ 1.5 & 7.03 & -0.3 & -0.71 \\ 2.25 & 1.32 & 5.58 & 0.49 \\ -3.11 & -14.35 & 0.08 & 18.51 \end{bmatrix},$$

а вектор правой части —

$$b = \begin{bmatrix} 11.48 \\ -1.83 \\ 7.47 \\ -3.43 \end{bmatrix}.$$

От системы $Ax = b$ перейдём к $x = Bx + d$.

Оставляем слева диагональные элементы, вправо переносим все остальные

$$\begin{cases} 10.39x_1 = -2.45x_2 - 3.35x_3 - 2.28x_4 + 11.48, \\ 7.03x_2 = -1.5x_1 + 0.3x_3 + 0.71x_4 - 1.83, \\ 5.58x_3 = -2.25x_1 - 1.32x_3 - 0.49x_4 + 7.47, \\ 18.51x_4 = 3.11x_1 + 14.35x_2 - 0.08x_3 - 3.43. \end{cases}$$

Поделим на коэффициенты при диагональных элементах

$$\begin{cases} x_1 = -2.45/10.39x_2 - 3.35/10.39x_3 - 2.28/10.39x_4 + 11.48/10.39, \\ x_2 = -1.5/7.03x_1 + 0.3/7.03x_3 + 0.71/7.03x_4 - 1.83/7.03, \\ x_3 = -2.25/5.58x_1 - 1.32/5.58x_3 - 0.49/5.58x_4 + 7.47/5.58, \\ x_4 = 3.11/18.51x_1 + 14.35/18.51x_2 - 0.08/18.51x_3 - 3.43/18.51. \end{cases}$$

Получаем новый вектор

$$b = \begin{bmatrix} 11.48/10.39 \\ -1.83/7.03 \\ 7.47/5.58 \\ -3.43/18.51 \end{bmatrix}.$$

Формируем матрицу B, где на диагонали стоят нули, а остальные элементы сформированы из правой части новой системы

$$B = \begin{bmatrix} 0 & -2.45/10.39 & -3.35/10.39 & -2.28/10.39 \\ -1.5/7.03 & 0 & 0.3/7.03 & 0.71/7.03 \\ -2.25/5.58 & -1.32/5.58 & 0 & -0.49/5.58 \\ 3.11/18.51 & 14.35/18.51 & -0.08/18.51 & 0 \end{bmatrix}.$$

3 РЕЗУЛЬТАТЫ ПЕРВЫХ ТРЁХ И ПОСЛЕДНЕЙ ИТЕРАЦИИ МЕТОДА

```
Iteration: 1
Solution:  1.1049085659287776 -0.2603129445234708  1.3387096774193548 -0.1853052404105889
Vector of residuals: -3.42441475714 -1.38731666636 -2.05163161877 0.406321888067
Error: 25.5458929165
Iteration: 2
Solution:  0.7753210050879507 -0.4576552868220310  0.9710337600774946 -0.2072567200468318
Vector of residuals: 1.7652524353 0.368493015517 1.01282012875 3.82746585281
Error: 7.01616621649
Iteration: 3
Solution:  0.9452201807662315 -0.4052380726659926  1.1525427437240392 -0.4140349941047443
Vector of residuals: -0.265022805046 -0.347208643005 -0.350142513674 -1.26605274081
Error: 3.94584108537

Iteration: 15
Solution:  0.9410714922862460 -0.4528510227249862  1.1000060601470580 -0.3830176876980250
Vector of residuals: 1.22272809726e-05 -5.28889419638e-06 -2.65629562346e-06 2.2920614648e-05
Error: 5.19530018732e-05
```

4 ЛИСТИНГ ПРОГРАММЫ

Листинг файла __main__.py

```
from solve import iteration


if __name__ == '__main__':
    matrix_size = 4
    B = [[0, -2.45/10.39, -3.35/10.39, -2.28/10.39],
         [-1.5/7.03, 0, 0.3/7.03, 0.71/7.03],
         [-2.25/5.58, -1.32/5.58, 0, -0.49/5.58],
         [3.11/18.51, 14.35/18.51, -0.08/18.51, 0]]
    d = [11.48/10.39, -1.83/7.03, 7.47/5.58, -3.43/18.51]
    e = 10**(-4)
    result = iteration(B, d, e)
```

Листинг файла solve.py

```
from numpy import array, zeros_like

def iteration(matrix, result, accuracy):
    matrix = array(matrix, dtype='f8', copy=True)
    result = array(result, dtype='f8', copy=True)
    assert matrix.ndim == 2 and matrix.shape[0] == matrix.shape[1]
    assert result.ndim == 1 and result.size == matrix.shape[0]

    solution = zeros_like(result)

    iteration = 1
    while True:
        solution_prev, solution = solution, matrix.dot(solution) + result
        print 'Iteration: {}'.format(iteration)
        print 'Solution:', ' '.join(['{:> .016f}'.format(r) for r in solution])
        vector = residuals(solution)
        print 'Vector of residuals:', ' '.join(str(v) for v in vector)
        print 'Error: {}'.format(max(abs(solution - solution_prev))/(1 - max(abs(matrix).sum(axis=1))))
        if max(abs(solution - solution_prev))/(1 - max(abs(matrix).sum(axis=1))) < accuracy:
            break
        iteration += 1

    return solution.tolist()

def residuals(solution):
    A = [[10.39, 2.45, 3.35, 2.28],
         [1.5, 7.03, -0.3, -0.71],
         [2.25, 1.32, 5.58, 0.49],
         [3.11, 14.35, -0.08, -18.51]]
    b = [11.48, -1.83, 7.47, 3.43]
```

```
    return (b - array(A).dot(solution)).tolist()
```

ВЫВОДЫ

Система линейных алгебраических уравенений была решена с помощью метода простой итерации. Для этого исходная система $Ax = b$ была преобразована так, что матрица A приобрела диагональное преимущество.

Получен вектор невязки, который показывает погрешность найденного решения. Полученная точность — до 10^{-6}. Её можно повысить, если задать ошибку ε для критерия завершения процесса меньше (была использована $\varepsilon = 10^{-4}$). Исходная система была решена с помощью метода Гаусса. Полученные решения совпадают.