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

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ №2

Дисциплина: «Методы вычислений»

Тема: «Решение систем линейных алгебраических уравнений (СЛАУ) прямыми 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 —

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

Матрица системы несимметрична, поэтому будем использовать метод Гаусса.

2 РЕЗУЛЬТАТЫ ПО ШАГАМ ПРИВЕДЕНИЯ К ТРЕУГОЛЬНОЙ ФОРМЕ МАТРИЦЫ

Прямой ход — приведение матрицы к треугольной форме. Он состоит в следующем:

1) нахождение максимального по модулю элемента в матрице

$$a_{main} = \max_{i,j} |a_{ij}| \, , \; i,j = 1 \ldots n;$$

2) обнуление элементов матрицы в стоблце с максимальным элементом;
3) переход к пункту 1 с уменьшением размера матрицы (выбрасыванимем стоблца и строки с главным элементом), пока не останется матрица 1×1.

Результат приведения матрицы к диагональному виду по шагам (строки и столбцы с максимальными элементами не удалялись, так как были использованы в дальшейшем, но не учитывались в следующих итерациях)

```
A:
[[  0.00000000e+00  -6.48412000e-01   5.15590000e-02   9.13927000e-01]
 [  0.00000000e+00   7.89442000e-01   4.85454300e+00  -3.74400000e-03]
 [  0.00000000e+00   6.02788300e+00  -7.32079000e-01  -1.25236000e-01]
 [  1.03900000e+01   2.45000000e+00   3.35000000e+00   2.28000000e+00]]
A:
[[  0.         0.        -0.02719    0.900455]
 [  0.        -0.         4.95042    0.012658]
 [  0.         6.027883  -0.732079  -0.125236]
 [ 10.39       0.         3.647549   2.330901]]
A:
[[  0.        -0.         0.         0.900525]
 [  0.        -0.         4.95042    0.012658]
 [  0.         6.027883  -0.        -0.123364]
 [ 10.39       0.         0.         2.321575]]
A:
[[  0.        -0.         0.         0.900525]
 [  0.        -0.         4.95042    0.      ]
 [  0.         6.027883  -0.        -0.      ]
 [ 10.39       0.         0.         0.      ]]
```

3 КОНЕЧНЫЙ РЕЗУЛЬТАТ (РЕШЕНИЕ УРАВНЕНИЯ)

После того, как была получена треугольная матрица, был применен обратный метод Гаусса. Составлена система с треугольной матрицей и найдено решение с шестью значимыми цыфрами

```
Solution:  0.941074 -0.452852  1.100005 -0.383020
```

4 ВЕКТОР НЕВЯЗКИ

Для проверки был найден вектор невязки по формуле $r = b - Ax$, где A — исходная матрица, x — наденное решение, b — столбец правой части

```
Vector of residuals: 5.06611206319e-08 4.83664420514e-07 -1.738530786e-07 1.62868046161e-07
```

5 ЛИСТИНГ ПРОГРАММЫ

Листинг файла main.py

```
from sys import argv

from numpy.random import rand
from numpy import array

from solve import solve_gauss, residuals


if __name__ == '__main__':
    matrix_size = 4
    A = [[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]]
    b = [4.21, 7.47, 2.38, 11.48]
    result = solve_gauss(A, b)
    vector = residuals(b, A, result)
    print 'Solution:', ' '.join(['{:> .06f}'.format(r) for r in result])
    print 'Vector of residuals:', ' '.join(str(v) for v in vector)
```

Листинг файла solve.py

```
from numpy import array, zeros_like, delete, unravel_index, nonzero, round


def solve_gauss(matrix, result):
    """
    >>> solve_gauss([[1]], [1])
    [1.0]
    >>> solve_gauss([[1, 2], [4, 1]], [5, 6])
    [1.0, 2.0]
    >>> solve_gauss([
    ...     [0, 1],
    ...     [1, 1]
    ... ], [2, 3])
    [1.0, 2.0]
    >>> solve_gauss([
    ...     [1, 0, 0],
    ...     [0, 1, 0],
    ...     [0, 0, 1],
    ... ], [1, 2, 3])
    [1.0, 2.0, 3.0]
    >>> solve_gauss([
    ...     [1, 0, 0, 0],
    ...     [0, 1, 0, 0],
```

```
...     [0, 0, 1, 0],
...     [0, 0, 0, 1],
... ], [1, 2, 3, 4])
[1.0, 2.0, 3.0, 4.0]
>>> solve_gauss([
...     [1, 0, 0, 0, 0],
...     [0, 1, 0, 0, 0],
...     [0, 0, 1, 0, 0],
...     [0, 0, 0, 1, 0],
...     [0, 0, 0, 0, 1],
... ], [1, 2, 3, 4, 5])
[1.0, 2.0, 3.0, 4.0, 5.0]
>>> solve_gauss([
...     [1, 1, 1],
...     [0, 1, 1],
...     [0, 0, 1],
... ], [6, 5, 3])
[1.0, 2.0, 3.0]
"""
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]

n = result.size

path = []

while n > 0:
    i, j = argmax(abs(matrix), path)
    m = matrix[i, j]
    for k in range(matrix.shape[0]):
        if i != k and i not in path:
            el = matrix[k][j] / m
            matrix[k] -= el * matrix[i]
            result[k] -= el * result[i]
    path.append(i)
    n -= 1
    print('A:')
    print(round(matrix, 6))

assert len(path) == result.size

solutions = zeros_like(result)
solved = set()

matrix = round(matrix, 6)

for i in path[::-1]:
```

```
        k = list(set(int(t) for t in nonzero(matrix[i])[0]) - solved)
        assert len(k) == 1, 'Found {} non-zero coefficients in {}:{}'.format(len(k), matrix[i], list(solved))
        k = k[0]
        solved.add(k)
        result[i] -= matrix[i].dot(solutions)
        result[i] /= matrix[i, k]
        solutions[k] = result[i]

    return solutions.tolist()

def argmax(matrix, ignored=()):
    result = ((0, 0), float('-inf'))
    for i in range(matrix.shape[0]):
        if i in ignored:
            continue
        for j in range(matrix.shape[1]):
            if matrix[i, j] > result[1]:
                result = ((i, j), matrix[i, j])
    return result[0]

def residuals(result, matrix, solution):
    return (result - array(matrix).dot(solution)).tolist()
```

ВЫВОДЫ

Система линейных алгебраических уравенений была решена с помощью метода Гаусса. Была использована схема с выбором главного элемента.

Получен вектор невязки, который показывает погрешность найденного решения. Полученная точность — до 10^{-8}. Это больше, чем 10^{-16}, потому решение было найдено не очень точно. Исходная система была решена с помощью сайта `http://ru.onlinemschool.com/math/assistance/equation/gaus/` для решения СЛАР методом Гаусса. Полученные решения совпадают.