МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования

«Дальневосточный федеральный университет» (ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

Департамент математического и компьютерного моделирования

ОТЧЁТ по лабораторной работе № 2

«LU-разложение»

Вариант № 8

Выполнила: студент гр. Б9122-02.03.01сцт
Ф.И.О.
Ильяхова Алиса Алексеевна

Проверил: преподаватель
Ф.И.О.
Павленко Елизавета Робертовна

Владивосток

2024

Цели работы:

Решить систему линейных алгебраических уравнений в виде Ax = b методом LU разложения.

Ход работы:

1. Находим матрицы L и U по формулам.
2. После нахождения значений матриц L и U решить СЛАУ в два этапа:
   - Из системы Ly = b находим вектор значений y. Вычислив массив “y” решаем СЛАУ вида Ux = y.
   - Полученный массив x будет являться решением исходной системы Ax = b.
3. Сравнить полученные результаты с точным решением x^* тестовых СЛАУ, по-другому отладить свой алгоритм на тестах.
4. После отладки программы решить два СЛАУ, указанных в задании.
5. Вывести полученные решения СЛАУ.

Основное:

1) Функции:

Функция **calc_LU** принимает на вход матрицу original и возвращает две матрицы: matrixL (нижняя треугольная матрица) и matrixU (верхняя треугольная матрица), которые составляют LU-разложение.

- Заполняется первый столбец матрицы matrixL, деля элементы первого столбца исходной матрицы на первый элемент первого столбца.
- Внешний цикл проходит по строкам, а внутренний - по столбцам.
- Если строка меньше или равна столбцу, вычисляется элемент матрицы U. Если строка больше столбца, вычисляется элемент матрицы L.

Функция **solver** решает систему линейных уравнений Ax = b, где A - это матрица (первый аргумент), а b - вектор правых частей (второй аргумент).

- Сначала выполняется LU-разложение с помощью функции **calc_LU**.
- Затем решается система уравнений ML * y = b (где ML - это L) и получается промежуточный вектор y.
- После этого решается система MU * x = y (где MU - это U) и возвращается вектор решения x.

2) Результаты:

```
Результаты для системы уравнений:
rest1: [[ 1.34022645]
 [-4.75800672]
 [ 2.57775289]]
```

```
Разница с точным решением t1: [[ 2.35473435e-05]
 [ 2.67189250e-05]
 [-6.52892870e-04]]
rest2: [[0.72965517]
 [1.2137931 ]
 [0.15310345]]
Разница с точным решением t2: [[ 4.48275862e-05]
 [ 6.89655172e-06]
 [-3.44827586e-06]]
rest3: [[ 2.]
 [ 1.]
 [-3.]]
Разница с точным решением t3: [[-4.44089210e-16]
 [-2.22044605e-16]
 [ 4.44089210e-16]]
```

```
rest4: [[ 3.]
 [-1.]
 [ 4.]
 [ 2.]]
Разница с точным решением t4: [[ 0.0000000e+00]
 [-4.4408921e-16]
 [ 8.8817842e-16]
 [-4.4408921e-16]]
rest5: [[-1.00000000e+00]
 [-5.18104078e-16]
 [ 1.00000000e+00]]
Разница с точным решением t5: [[-4.44089210e-16]
 [ 5.18104078e-16]
 [ 2.22044605e-16]]
res1: [[0.12996615]
 [0.80016894]
 [1.07572903]]
```

```
Произведение matA1 и res1: [[1.27]
 [2.13]
 [3.14]]
res2: [[0.15331773]
 [0.35835406]
 [0.35066487]
 [0.19304791]]
Произведение matA2 и res2: [[1.02]
 [1.  ]
 [1.34]
 [1.27]]
```

3) Вывод:

В ходе выполнения данной лабораторной работы были приобретены практические навыки применения метода LU-разложения для решения систем линейных алгебраических уравнений. В процессе работы были освоены следующие навыки:

- Проведение LU-разложения матриц, что включает в себя разложение исходной матрицы на нижнюю (L) и верхнюю (U) треугольные матрицы.
- Решение систем линейных уравнений с использованием методов прямой и обратной подстановки, что упрощает вычисления.
- Анализ и оценка точности полученных решений для тестовых систем, а также интерпретация результатов. Метод LU-разложения зарекомендовал себя как надежный и эффективный способ решения систем линейных уравнений, позволяющий разбивать задачу на подзадачи. Этот метод особенно полезен для больших систем, где важна оптимизация вычислений.
- Кроме того, LU-разложение дает возможность использовать одни и те же матрицы L и U для решения нескольких систем с различными правыми частями, что делает его ценным инструментом в линейной алгебре и численных методах.

4) Листинг программы:

```
import numpy as np
import numpy.linalg as LA

def calc_LU(original: np.matrix) -> (np.matrix, np.matrix):
    matrixU = np.matrix(np.zeros(original.shape))
    matrixU[0] = original[0]
    matrixL = np.diag(np.ones((original.shape[0])))

    for i in range(original.shape[1]):
        matrixL[i, 0] = original[i, 0] / original[0, 0]

    for row in range(1, original.shape[0]):
        for column in range(1, original.shape[1]):
            if row <= column:
                elem_sum = sum(matrixL[row, i] * matrixU[i, column]
for i in range(0, row - 1 + 1))
                matrixU[row, column] = original[row, column] -
elem_sum
            else:
                elem_sum = sum(matrixL[row, i] * matrixU[i, column]
for i in range(0, column - 1 + 1))
                matrixL[row, column] = (original[row, column] -
elem_sum) / matrixU[column, column]

    return matrixL, matrixU

def solver(matrix, b):
```

```
    ML, MU = calc_LU(matrix)
    y = LA.solve(ML, b)
    x = LA.solve(MU, y)
    return x


mAt1 = np.matrix('2.1 -4.5 -2.0; 3.0 2.5 4.3; -6.0 3.5 2.5')
vecbt1 = np.matrix('19.07; 3.21; -18.25')
precise_res_t1 = np.matrix('1.34025; -4.75798; 2.5771')
rest1 = solver(mAt1, vecbt1)

mAt2 = np.matrix('5 -1 5; -3 6 2; 10 -7 0')
vecbt2 = np.matrix('3.2; 5.4; -1.2')
precise_res_t2 = np.matrix('0.7297; 1.2138; 0.1531')
rest2 = solver(mAt2, vecbt2)

mAt3 = np.matrix('5 2 3; 1 6 1; 3 -4 -2')
vecbt3 = np.matrix('3; 5; 8')
precise_res_t3 = np.matrix('2; 1; -3')
rest3 = solver(mAt3, vecbt3)

matrixAt4 = np.matrix([[1, 2, 1, 4], [2, 0, 4, 3], [4, 2, 2, 1], [-
3, 1, 3, 2]])
vecbt4 = np.matrix('13; 28; 20; 6')
precise_res_t4 = np.matrix('3; -1; 4; 2')
rest4 = solver(matrixAt4, vecbt4)

mAt5 = np.matrix('2 1 3; 11 7 5; 9 8 4')
vecbt5 = np.matrix('1; -6; -5')
precise_res_t5 = np.matrix('-1; 0; 1')
rest5 = solver(mAt5, vecbt5)

matA1 = np.matrix('13.14 -2.12 1.17; -2.12 6.3 -2.45; 1.17 -2.45
4.6')
b1 = np.matrix('1.27; 2.13; 3.14')
res1 = solver(matA1, b1)

matA2 = np.matrix('4.31 0.26 0.61 0.27; 0.26 2.32 0.18 0.34; 0.61
0.18 3.2 0.31; 0.27 0.34 0.31 5.17')
b2 = np.matrix('1.02; 1; 1.34; 1.27')
res2 = solver(matA2, b2)

#вывод результатов
print("Результаты для системы уравнений:")
print("rest1:", rest1)
```

```
print("-----------------------------------------------------")
print("Разница с точным решением t1:", precise_res_t1 - rest1)
print("rest2:", rest2)
print("Разница с точным решением t2:", precise_res_t2 - rest2)
print("rest3:", rest3)
print("Разница с точным решением t3:", precise_res_t3 - rest3)
print("rest4:", rest4)
print("Разница с точным решением t4:", precise_res_t4 - rest4)
print("rest5:", rest5)
print("Разница с точным решением t5:", precise_res_t5 - rest5)
print("res1:", res1)
print("-----------------------------------------------------")
print("Произведение matA1 и res1:", matA1 @ res1)
print("res2:", res2)
print("Произведение matA2 и res2:", matA2 @ res2)
```