

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ     
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего профессионального образования

**ДАЛЬНЕВОСТОЧНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ**

**Институт математики и компьютерных технологий**

**Департамент математического и компьютерного моделирования**

**ОТЧЕТ**

**по лабораторной работе** **№ 2**

«LU-разложение»

Выполнил: студент гр. Б9120-02.03.01сцт

Пограничный К. О.

Проверил: преподаватель

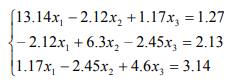
Кузнецов К. С.

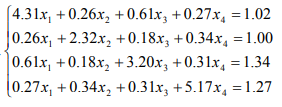
Владивосток 2023

**Постановка задания**

Дана система линейных алгебраических уравнений в виде Ax=b.

Решить следующие две системы с помощью LU-разложения:





**Метод решения**

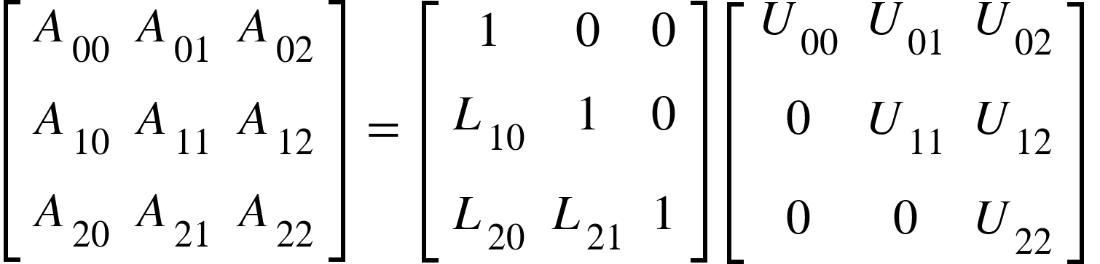
Задача была выполнена на языке программирования Python с использованием средств библиотеки NumPy.

Использовались отдельные функции для создания матриц A1 и A2, элементами которых являются коэффициенты при (x1, …, x3) для первой матрицы, и коэффициенты при (x1, …, x4) для второй матрицы соответственно. Были созданы векторы b1 и b2, элементами которых являются свободные члены соответствующих СЛАУ.

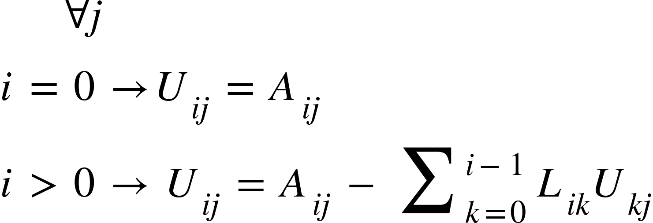
Была реализована функция LU – разложения. Произведены вычисления для уравнений Ly = b и Ux = y, которые и позволили получить решение поставленной задачи, а именно разрешить два тензорных уравнения для каждой из систем линейных алгебраических уравнений.

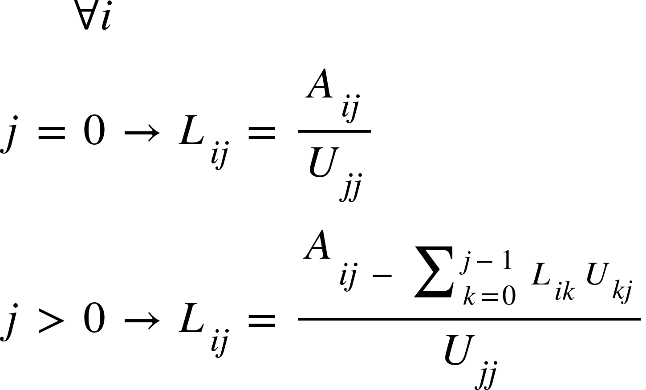
**Алгоритм решения**

Алгоритм заключается в представлении матрицы A, как комбинации матриц LU, то есть необходимо реализовать алгоритм LU-разложения:

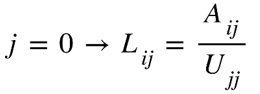


В первую очередь необходимо найти матрицы L и U по следующим формулам:





После вычисления матриц L и U решить СЛАУ в два этапа:

1. Ly=b
2. Ux=y
3. Создание матриц А1 и А2, а также двух векторов b1 и b2 производилось внутри функции «initial».
4. Была реализована функция «LU\_decomposition», которая производила LU-разложение. Были созданы матрицы L и U, где L – нижняя треугольная матрица с единичной диагональю, а U – верхняя треугольная матрица. Для каждого столбца j матрицы [](http://www.codecogs.com/eqnedit.php?latex=L) будем вычислять [](http://www.codecogs.com/eqnedit.php?latex=l_%7bi,j%7d) как 

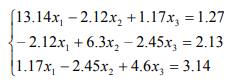
Использовались встроенные функции библиотеки NumPy: shape, zeros, add, identity.

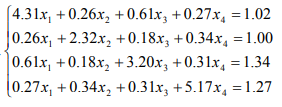
1. Далее была написана функция «solve\_SLAE», в которой разрешались тензорные уравнения Ly = b и Ux = y относительно y и x соответственно. Использовался метод numpy.linalg.tensorsolve().

**Использованные функции**

1. numpy.array() – встроенная функция библиотеки NumPy для создания списков(массивов).
2. numpy.shape() – встроенная функция библиотеки NumPy, которая возвращает размерность массива.
3. numpy.zeros() – встроенная функция библиотеки NumPy, которая создает матрицу из нулей указанной формы и типа данных.
4. numpy.add() – встроенная функция библиотеки NumPy, которая находит сумму двух чисел.
5. numpy.identity() – встроенная функция библиотеки NumPy, которая создает матрицу из нулей с единичной главной диагональю.
6. numpy.linalg.tensorsolve() - используется для разрешения тензорного уравнения Ax=b относительно x.

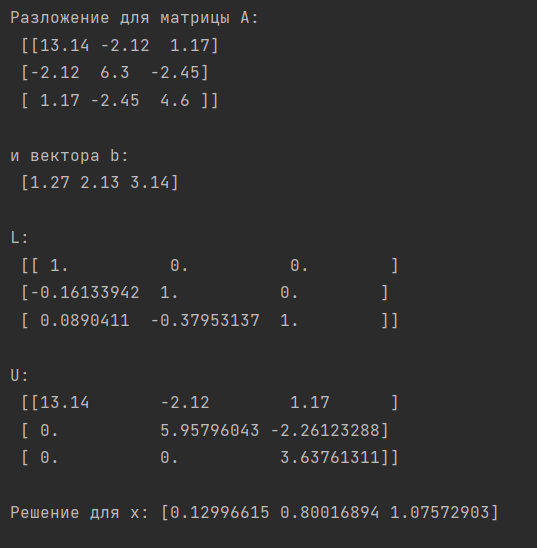
**Описание тестов**

1. Решить заданную систему:
2. Решить заданную систему:

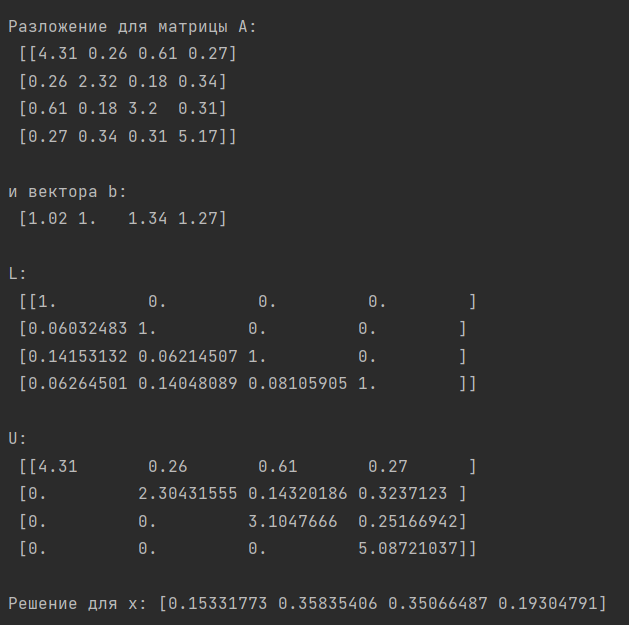


**Результаты тестов**

Отработка для первой СЛАУ:



Отработка для второй СЛАУ:



**Вывод**

В ходе лабораторной работы была реализована программа для решения систем линейных алгебраических уравнений методом LU-разложения.

**Приложение**

Листинг программы:

import numpy as np  
  
  
def initial():  
 A1 = np.array([  
 [13.14, -2.12, 1.17],  
 [-2.12, 6.3, -2.45],  
 [1.17, -2.45, 4.6]  
 ])  
 b1 = np.array([1.27, 2.13, 3.14])  
  
 A2 = np.array([  
 [4.31, 0.26, 0.61, 0.27],  
 [0.26, 2.32, 0.18, 0.34],  
 [0.61, 0.18, 3.20, 0.31],  
 [0.27, 0.34, 0.31, 5.17]  
 ])  
 b2 = np.array([1.02, 1.00, 1.34, 1.27])  
  
 return A1, b1, A2, b2  
  
  
def LU\_decomposition(A):  
 dim = np.shape(A)[0]  
 U = np.zeros(np.shape(A))  
 U += A  
 L = np.add(np.zeros((dim, dim), dtype=float), np.identity(dim, dtype=float))  
  
 for k in range(1, dim):  
 for i in range(k-1, dim):  
 for j in range(i, dim):  
 L[j][i] = U[j][i] / U[i][i]  
  
 for i in range(k, dim):  
 for j in range(k-1, dim):  
 U[i][j] = U[i][j] - L[i][k-1] \* U[k-1][j]  
  
 return L, U  
  
  
def solve\_SLAE(L, U, b):  
 y = np.linalg.tensorsolve(L, b)  
 x = np.linalg.tensorsolve(U, y)  
 return x  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 A1, b1, A2, b2 = initial()  
  
 L, U = LU\_decomposition(A1)  
 print(f"Разложение для матрицы A:\n {A1}")  
 print(f"\nи вектора b:\n {b1}")  
 print("\nL:\n", L)  
 print("\nU:\n", U)  
 x = solve\_SLAE(L, U, b1)  
 print("\nРешение для x:", x)  
  
 L, U = LU\_decomposition(A2)  
 print(f"\n\nРазложение для матрицы A:\n {A2}")  
 print(f"\nи вектора b:\n {b2}")  
 print("\nL:\n", L)  
 print("\nU:\n", U)  
 x = solve\_SLAE(L, U, b2)  
 print("\nРешение для x:", x)