|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ *Робототехники и комплексной автоматизации*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

по дисциплине: «Вычислительная математика»

|  |  |  |
| --- | --- | --- |
| Студент |  | Шашко Олег Владимирович |
| Группа |  | РК6-51Б |
| Тип задания |  | лабораторная работа |
| Тема лабораторной работы |  | LU - разложение |

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шашко О.В.**

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Соколов А.П.**

*подпись, дата фамилия, и.о.*

*Москва, 2022 г.*

Оглавление

[Задание на лабораторную работу 3](#_Toc32399212)

[Цель выполнения лабораторной работы 3](#_Toc32399213)

[Выполненные задачи 3](#_Toc32399214)

[1. @Название раздела в соответствии с задачей 1@ 4](#_Toc32399215)

[2. @Название раздела в соответствии с задачей 2@ 4](#_Toc32399216)

[Заключение 4](#_Toc32399217)

[Список использованных источников 4](#_Toc32399218)

# Задание на лабораторную работу

Дана СЛАУ 𝐴1𝑥 = 𝑏1**:**

*(18)*

Дана СЛАУ 𝐴2𝑥 = 𝑏2:

*(19)*

Требуется (базовая часть):

1. Написать функцию lu(A), которая производит LU-разложение матрицы A и возвращает матрицы L и U.

2. Написать функцию solve(L, U, b), которая возвращает решение СЛАУ 𝐴𝑥 = 𝑏, где матрица 𝐴 представлена в виде LU-разложения.

3. Найти решение СЛАУ (18) с помощью разработанной функции lu(A) и срав- нить с точным решением этой СЛАУ: 𝑥 = [−1, 2, 0, 1]𝑇 .

Требуется (продвинутая часть):

1. Доказать, что для решения СЛАУ (19) необходимо на определенной итерации метода Гаусса произвести перестановку строк.

2. Модифицировать функцию lu(A, permute) так, чтобы она принимала аргумент permute и возвращает матрицы L, U и P. Если permute=True, то LU-разложение должно происходить с частичным выбором главного элемента, а возвращаемая мат-  
рица P при этом должна быть соответствующей матрицей перестановок. Если permute=False, то частичного выбора происходить не должно, а возвращаемая матрица P должна быть единичной.

3. Модифицировать функцию solve(L, U, P, b) так, чтобы она принимала на вход аргумент P и возвращала решение СЛАУ 𝐴𝑥 = 𝑏, где матрица 𝐴 представлена в виде LU-разложения с матрицей перестановок 𝑃 , т.е. 𝑃 𝐴 = 𝐿𝑈 .

4. Найти решение СЛАУ (19) с помощью разработанной функции solve(L, U, P, b), обозначаемое здесь и далее 𝑥ˆ.

5. Доказать, что модифицированная СЛАУ, полученная с помощью добавления к элементам 𝑎11 и 𝑏1 СЛАУ (19) малого числа 10−𝑝, где 𝑝 – произвольное целое число, имеет то же решение, что и исходная СЛАУ. Здесь и далее решение модифицирован- ной СЛАУ будет обозначаться как 𝑥 ̃.

6. С помощью разработанных функций lu(A, permute) и solve(L, U, P, b) найти решения модифицированной СЛАУ для 𝑝 ∈ [0; 12] для двух случаев:

– без частичного выбора главного элемента,  
– с частичным выбором главного элемента.  
7. Для обоих случаев построить log-log график зависимости относительной по-

грешности вычисления 𝐸 = ||𝑥^−𝑥 ̃||∞ от 𝑝. Проанализировав полученные результаты, ||𝑥^ ||∞

сделайте подробный вывод о вычислительной устойчивости/неустойчивости решения СЛАУ с помощью LU-разложения для конкретного рассматриваемого случая и опишите, что нужно в этом контексте иметь в виду потенциальному пользователю ваших функций lu(A, permute) и solve(L, U, P, b).

# Цель выполнения лабораторной работы

**Цель выполнения лабораторной работы** – научится решать СЛАУ методом LU – разложения с частичным выбором главного элемента, а также без такого.

# Выполненные задачи

* **Задача 1 –** написать функцию lu(A)
* **Задача 2 –** написать функцию solve(L, U, b)
* **Задача 3 -** найти решение СЛАУ
* **Задача 4 –** доказать необходимость перестановок строк
* **Задача 5 –** модифицировать функцию lu(A, permute)
* **Задача 6 –** модифицировать функцию solve(L, U, P, b)
* **Задача 7 –** найти решение СЛАУ (19)
* **Задача 8 –** доказать эквивалентность решения СЛАУ
* **Задача 9 –** найти решения модифицированной СЛАУ
* **Задача 10 -** построить log-log график зависимости относительной погрешности

**1. Базовая часть**

1. Задача 1 – написать функцию lu(A).

Для начала, выясним, что из себя представляет LU – разложение. Для этого обратимся к материалу лекций

LU-разложение — это представление матрицы A в виде A=L•U, где L — нижнетреугольная матрица с единичной диагональю, а U — верхнетреугольная матрица. LU-разложение является модификацией метода Гаусса. Основные применения данного алгоритма — решение систем алгебраических уравнений, вычисление определителя, вычисление обратной матрицы и др.

В текущей задаче требуется только разложить исходную матрицу A. Таким образом, задача сводится к нахождению матриц L и U.

Заметим, что k-я итерация прямого хода метода Гаусса эквивалента:

Где индексы в соответствуют строке и столбцу матрицы

Переходим к реализации:

def lu(a1, permute):  
 a = a1  
 n = len(a)  
 u = a  
 l = np.identity(n)  
 for j in range(0, n):  
 m\_j = np.identity(n)  
 for i in range(j + 1, n):  
 m\_j.itemset((i, j), -a[i, j] / a[j, j])  
 l.itemset((i, j), a[i, j] / a[j, j])  
 a = np.matmul(m\_j, a)  
 u = np.matmul(m\_j, u)  
 return [l, u]

*Листинг 1 – реализация LU – разложения*

2. Задача 2 – написать функцию solve(l, u, vec)

В предыдущей задаче мы реализовали функцию lu (реализация представлена на листинге 1), которая возвращает матрицы L и U.

Новая система имеет вид (1):

(1),

Где x – вектор неизвестных.

Для нахождения x необходимо ввести замену и решить относительно y:

*(2.1)*

*(2.2)*

Решив (2.1), выполняем обратную замену и решаем (2.3):

*(2.3)*

Программная реализация представлена на листинге 2:

def solve(l, u, vec):  
 n = len(vec)  
 y = []  
 for k in range(0, n):  
 sum = 0  
 for i in range(0, k):  
 sum += l[k, i] \* y[i]  
 print(sum)  
 y.append(vec[k] - sum)  
 x = [0.0] \* n  
 for k in range(0, n):  
 sum = 0  
 for i in range(k + 1, n):  
 sum += u[k, i] \* x[i]  
 print(sum)  
 x[k] = y[k] / u[k, k] - sum / u[k, k]  
 return x

*Листинг 2 – реализация функции solve*

3. Задача 3 – решить СЛАУ с помощью функции solve и затем сравнить полученный результат с точным решением

Из условия задачи, точное решение этой СЛАУ –

В результате работы программы получен результат

Сравнивая полученное решение с эталонным, можно сделать вывод, что метод LU – разложения является достаточно точным.

**2. Продвинутая часть**

2. Задача 4 – доказать необходимость перестановок строк

Для применения прямого хода метода Гаусса, который лежит в основе LU – разложения, необходимым условием является неравенство нулю угловых миноров. Для примера, возьмем СЛАУ (19) и посчитаем ее угловые миноры:

*(19)*

Мы получили угловой минор, равный нулю. Таким образом, данную СЛАУ нельзя решить без перестановки строк.

5. Задача 5 – модифицировать функцию lu(A, permute)

В задаче 1 базовой части мы реализовали функцию lu, которая возвращала нам матрицы L и U, вычисленные без выбора главного элемента. Таким образом, для случая, когда permute=False, нам полностью подходит реализация, представленная на листинге 1, за исключением return – тут мы будем возвращать помимо матриц L и U единичную матрицу P, которую можно получить с помощью библиотечной функции np.identity().

Для случая, когда permute=True, нам необходимо выбрать условие, при котором будет производится выбор главного элемента. Рационально будет использовать условие, при котором абсолютное значение диагонального элемента не превосходит 10e-10.

Поскольку , то PA

Переходим к реализации:

def lu(a1, permute):  
 a = a1  
 n = len(a)  
  
 p = np.identity(n)  
  
 if not permute:  
  
 n = len(a)  
  
 u = a  
 l = np.identity(n)  
 for j in range(0, n):  
 m\_j = np.identity(n)  
 for i in range(j + 1, n):  
 m\_j.itemset((i, j), -a[i, j] / a[j, j])  
 l.itemset((i, j), a[i, j] / a[j, j])  
 a = np.matmul(m\_j, a)  
 u = np.matmul(m\_j, u)  
 return [l, u, p]  
 else:  
 u = a  
 l = np.identity(n)  
 for j in range(0, n):  
 m\_j = np.identity(n)  
 for i in range(j + 1, n):  
 if abs(a[j, j]) > 10e-10:  
 m\_j.itemset((i, j), -a[i, j] / a[j, j])  
 l.itemset((i, j), a[i, j] / a[j, j])  
  
 else:  
 print('check')  
 max\_ind = index\_of\_max\_element(a[:][j])  
 print(f"max\_ind = ", max\_ind)  
 a = swap(a, j, max\_ind)  
 p = swap(p, j, max\_ind)  
 m\_j.itemset((i, j), -a[i, j] / a[j, j])  
 l.itemset((i, j), a[i, j] / a[j, j])  
 a = np.matmul(m\_j, a)  
 u = np.matmul(m\_j, u)  
 return [np.matmul(np.matmul(p, l), np.linalg.inv(p)), np.matmul(p, u), p]

*Листинг 3 – реализация функции lu с возможностью частичного выбора главного элемента*

6. Задача 6 – модифицировать функцию solve(L, U, P, b)

Для того, чтобы функция корректно вычисляла значения неизвестных, необходимо умножить вектор b, передаваемый в аргументах функции, на матрицу перестановок P. Программная реализация представлена на листинге 4:

def solve(l, u, vec1, p):  
 vec = np.matmul(vec1, p)  
 n = len(vec)  
 y = []  
 for k in range(0, n):  
 sum = 0  
 for i in range(0, k):  
 sum += l[k, i] \* y[i]  
 print(sum)  
 y.append(vec[k] - sum)  
 x = [0.0] \* n  
 for k in range(0, n):  
 sum = 0  
 for i in range(k + 1, n):  
 sum += u[k, i] \* x[i]  
 print(sum)  
 x[k] = y[k] / u[k, k] - sum / u[k, k]  
 return x

*Листинг 4 – реализация функции solve для решения СЛАУ с выбором главного элемента*

7. Задача 7 – найти решение СЛАУ(19) с помощью функции, реализованной в предыдущей задаче.

В результате работы функции были получены значения

Сравним с точным решением этой СЛАУ (точное решение посчитаем с помощью сервиса Wolfram Alpha) - . Таким образом, методом LU – разложения для данной матрицы мы получили точное решение системы.

8. Задача 8 – доказать, что при добавлении к первому элементу матрицы и вектора малого числа, решение этой СЛАУ не поменяется

Добавим к элементу матрицы и вектору число Затем, используя функцию solve, реализованную раннее, решим новое СЛАУ:

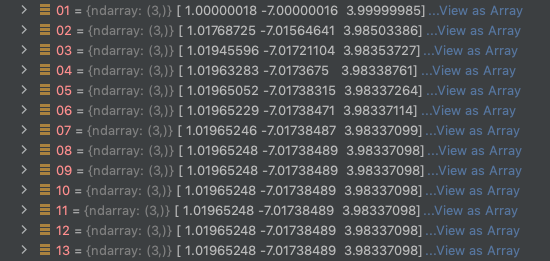
Сохраним решение исходной и модифицированной СЛАУ в переменные и сравним полученные результаты (рис 1):



*Рисунок 1 – результаты решения исходной и модифицированной СЛАУ.*

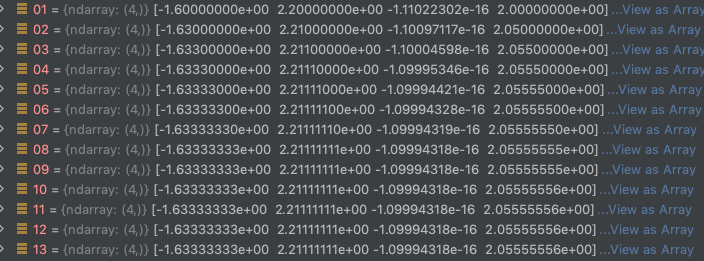
8. Задача 8 – найти решения СЛАУ для

1) Для случая с выбором главного элемента (решаем СЛАУ(19)) полученные решения представлены на рисунке 2:



*Рисунок 2 – результаты решения СЛАУ(19) с выбором главного элемента.*

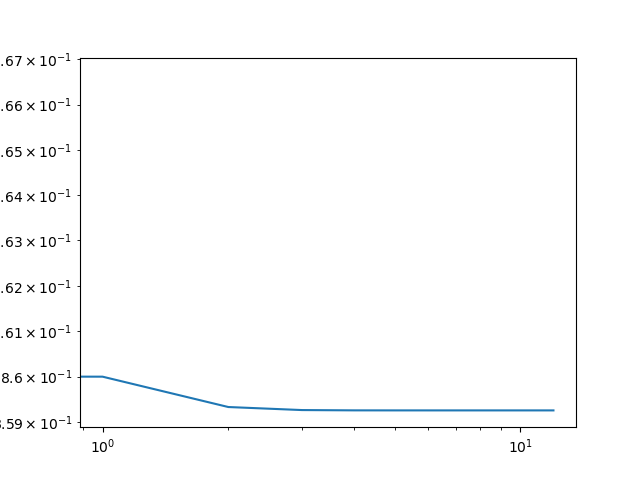
Теперь решим СЛАУ(18) без выбора главного элемента. Полученные значения представлены на рисунке 3:



*Рисунок 3 – результаты решения СЛАУ(18) без выбора главного элемента.*

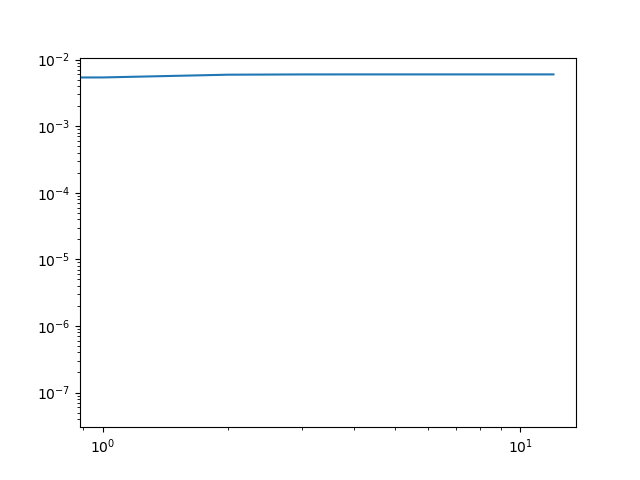
9. Задача 9 – для обоих случаев, рассмотренных в предыдущей задаче, построить log-log графики зависимости погрешности от p.

Для случая без выбора главного элемента график погрешности представлен на рисунке 4:



*Рисунок 4 – график зависимости погрешности от p для случая без выбора главного элемента*

Для случая с выбором главного элемента, график представлен на рисунке 5:



*Рисунок 5 – график зависимости погрешности от p для случая с выбором главного элемента*

# Заключение

Мы изучили метод LU-разложения как с частичным выбором главного элемента, так и без такого, для решения СЛАУ, а также научились реализовать его на языке программирования python.

# Список использованных источников

1. **Першин А.Ю.** *Лекции по курсу “Вычислительная математика”.* //

Москва, 2018-2021. С. 140.

2. *Официальная документация numpy* [Электронный ресурс] // Режим доступа: https://numpy.org/, свободный. Яз. англ.

*Шашко О.В. Отчет о выполнении лабораторной работы по курсу “Вычислительная математика”. [Электронный ресурс] – Москва: 2022 – 13 с. URL: https://sa2systems.ru:88 (система контроля версий кафедры РК6)*