Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Методы численного анализа»

**ОТЧЕТ**

к лабораторной работе №1

на тему:

**«РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ (СЛАУ) МЕТОДОМ ГАУССА И С ПОМОЩЬЮ ЕГО Модификаций»**

БГУИР 1-40 04 01

|  |
| --- |
| Выполнил студент группы 253505  БЕКАРЕВ Станислав Сергеевич |
|  |
| (дата, подпись студента) |
| Проверил доцент кафедры информатики  АНИСИМОВ Владимир Яковлевич |
|  |
| (дата, подпись преподавателя) |

Минск 2023

**Содержание**

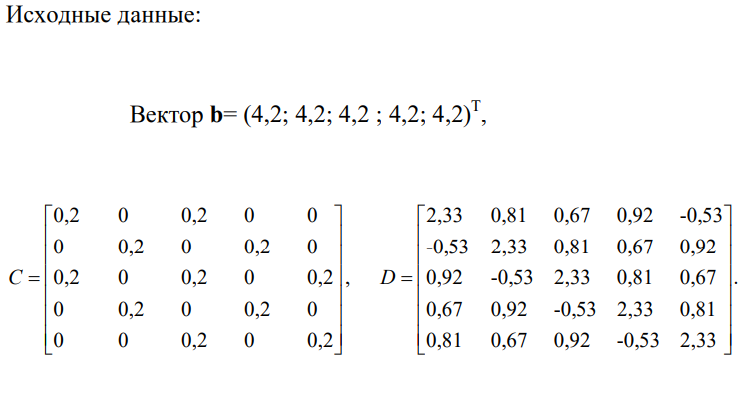
1. Цель работы
2. Задание
3. Программная реализация
4. Полученные результаты
5. Оценка полученных результатов
6. Вывод

**Цель работы**

* изучить метод Гаусса и его модификации, составить программу его реализации, получить численное решение данной СЛАУ;
* составить программу решения СЛАУ указанными методами, применимый для организации вычислений на ЭВМ;
* выполнить тестовые примеры и проверить правильность работы программы.

**Задание**

Методом Гаусса и методом выбора главного элемента найти с точностью 0,0001 численное решение системы **Ax = b**, где **A = kC + D**, **A** - исходная матрица для расчёта, **k** - номер варианта (0–15), матрицы **C, D** и вектор свободных членов **b** задаются ниже.



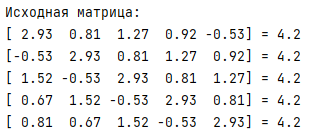
Вариант 3

**Программная реализация**

Умножим исходную матрицу на полученный вектор решений и сравним с изначальным вектором свободных членов.

*Исходные данные:*

Матрица A, полученная в результате вычисления A = 3C + D:



Код прямого обхода:

def gauss(A, b):  
 run\_straight(A, b)  
 run\_reverse(A, b)

def run\_straight(matrix, vector):  
 amount\_row = len(matrix)for i in range(0, amount\_row):  
 curr\_row = matrix[i]  
 devider = curr\_row[i]if(devider == 0):  
 print("Система несовместна")  
 exit()  
 curr\_row /= devider  
 vector[i] /= devider  
 for j in range(i+1, amount\_row):  
 diag\_el = matrix[j][i]  
 matrix[j] -= diag\_el \* curr\_row  
 b[j] -= diag\_el \* vector[i]

def run\_reverse(matrix, vector):amount\_row = len(matrix)  
 for i in reversed(range(0, amount\_row)):  
 for j in range(i, amount\_row):  
 vector[i] -= X[j] \* matrix[i][j]  
 X[i] = vector[i]

Были реализованы модификации метода Гаусса: метод частичного выбора по столбцу и по всей матрице.

Метод частичного выбора:

def run\_straight\_column(matrix, vector):  
 amount\_row = len(matrix)for i in range(0, amount\_row):  
 swap\_rows(matrix, max\_index\_in\_column(matrix, i), i)  
 curr\_row = matrix[i]  
 devider = curr\_row[i]if (devider == 0):  
 print("Система несовместна")  
 exit()  
 curr\_row /= devider  
 vector[i] /= devider  
 for j in range(i + 1, amount\_row):  
 diag\_el = matrix[j][i]  
 matrix[j] -= diag\_el \* curr\_row  
 b[j] -= diag\_el \* vector[i]  
def max\_index\_in\_column(matrix, i\_column):  
 max = matrix[i\_column][i\_column]save\_i = i\_column  
 for i in range(i\_column, len(matrix)):  
 if matrix[i\_column][i] > max:  
 max = matrix[i\_column][i]  
 save\_i = i  
 return save\_i

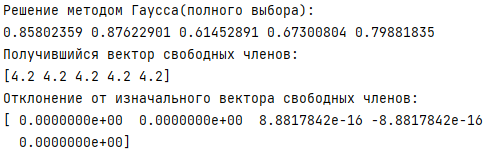
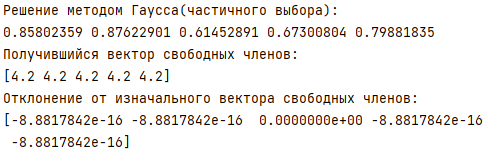
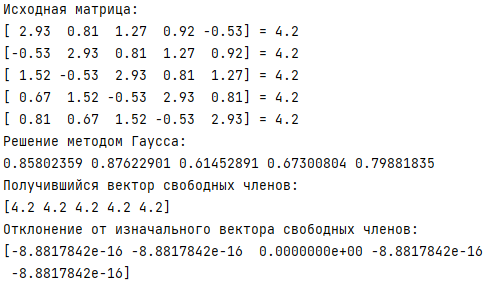
Метод выбора по всей матрице:

def run\_straight\_max(matrix, vector):  
 amount\_row = len(matrix)for i in range(0, amount\_row):  
 max\_indexes = max\_element(matrix, i)  
 swap\_rows(matrix, max\_indexes[0], i)  
 swap\_columns(matrix, max\_indexes[1], i)  
 curr\_row = matrix[i]  
 devider = curr\_row[i]

if (devider == 0):  
 print("Система несовместна")  
 exit()  
 curr\_row /= devider  
 vector[i] /= devider  
 for j in range(i + 1, amount\_row):  
 diag\_el = matrix[j][i]  
 matrix[j] -= diag\_el \* curr\_row  
 b[j] -= diag\_el \* vector[i]

def max\_element(A, k):  
 max = A[k][k]  
 max\_indexes = [k, k]  
 for i in range(k, len(A)):  
 for j in range(k, len(A)):  
 if max < A[i][j]:  
 max = A[i][j]  
 max\_indexes = [i, j]  
 return max\_indexes

**Полученные результаты**

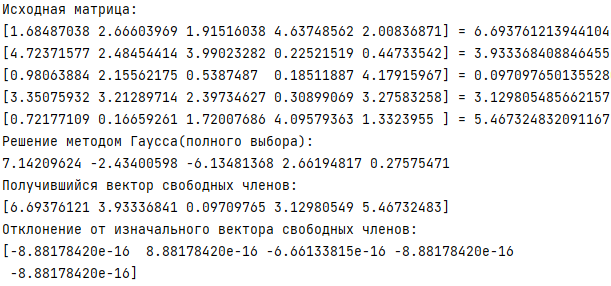


*Результаты вычислений:*

|  |  |  |
| --- | --- | --- |
| *Метод Гаусса* | *Метод Гаусса с выбором главного элемента по столбцу* | *Метод Гаусса с выбором главного элемента по всей матрице* |
| 0.85802359 | 0.85802359 | 0.85802359 |
| 0.87622901 | 0.87622901 | 0.87622901 |
| 0.61452891 | 0.61452891 | 0.61452891 |
| 0.67300804 | 0.67300804 | 0.67300804 |
| 0.79881835 | 0.79881835 | 0.79881835 |

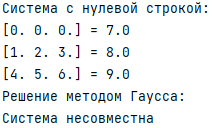
*Тестовый пример 1.*

С помощью пакета numpy создадим матрицу и вектор свободных членов и заполним их случайными числами:

**

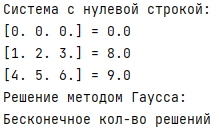
*Тестовый пример 2.*

*В данном примере мы видим матрицу без решений, так как ранг матрицы коэффициентов меньше трех.*

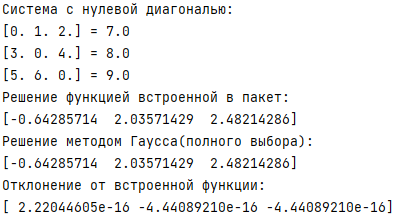


*Тестовый пример 3.*

*В данном примере мы видим матрицу с бесконечным количеством решений, так как ранг матрицы коэффициентов равен двум, как и ранг матрицы ответов.*



*Тестовый пример 4.*



**Вывод**

В ходе выполнения лабораторной работы я изучил метод Гаусса и его 2 модификации: метод Гаусса с выбором главного элемента по столбцу (схема частичного выбора) и метод Гаусса с выбором главного элемента по всей матрице(схема полного выбора), написал программу их реализации на языке Python для решения поставленной задачи, правильность работы программы проверил на тестовых примерах.

На основании тестов можно сделать следующие выводы:

* имеет ограничение в использовании (на главной диагонали не должно быть нулевых элементов), однако его можно обойти, используя метод Гаусса полного выбора или же поменяв строки и диагонали в исходной

матрице.