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

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.

Исходные данные:

Вектор **b**= (4,2; 4,2; 4,2 ; 4,2; 4,2)^{T},

$$C = \begin{bmatrix} 0,2 & 0 & 0,2 & 0 & 0 \\ 0 & 0,2 & 0 & 0,2 & 0 \\ 0,2 & 0 & 0,2 & 0 & 0,2 \\ 0 & 0,2 & 0 & 0,2 & 0 \\ 0 & 0 & 0,2 & 0 & 0,2 \end{bmatrix}, \quad D = \begin{bmatrix} 2,33 & 0,81 & 0,67 & 0,92 & -0,53 \\ -0,53 & 2,33 & 0,81 & 0,67 & 0,92 \\ 0,92 & -0,53 & 2,33 & 0,81 & 0,67 \\ 0,67 & 0,92 & -0,53 & 2,33 & 0,81 \\ 0,81 & 0,67 & 0,92 & -0,53 & 2,33 \end{bmatrix}.$$

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:

```
Исходная матрица:
[ 2.93  0.81  1.27  0.92 -0.53] = 4.2
[-0.53  2.93  0.81  1.27  0.92] = 4.2
[ 1.52 -0.53  2.93  0.81  1.27] = 4.2
[ 0.67  1.52 -0.53  2.93  0.81] = 4.2
[ 0.81  0.67  1.52 -0.53  2.93] = 4.2
```

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]
```

```python
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

```
Исходная матрица:
[ 2.93  0.81  1.27  0.92 -0.53] = 4.2
[-0.53  2.93  0.81  1.27  0.92] = 4.2
[ 1.52 -0.53  2.93  0.81  1.27] = 4.2
[ 0.67  1.52 -0.53  2.93  0.81] = 4.2
[ 0.81  0.67  1.52 -0.53  2.93] = 4.2
Решение методом Гаусса:
0.85802359 0.87622901 0.61452891 0.67300804 0.79881835
Получившийся вектор свободных членов:
[4.2 4.2 4.2 4.2 4.2]
Отклонение от изначального вектора свободных членов:
[-8.8817842e-16 -8.8817842e-16  0.0000000e+00 -8.8817842e-16
 -8.8817842e-16]
Решение методом Гаусса(частичного выбора):
0.85802359 0.87622901 0.61452891 0.67300804 0.79881835
Получившийся вектор свободных членов:
[4.2 4.2 4.2 4.2 4.2]
Отклонение от изначального вектора свободных членов:
[-8.8817842e-16 -8.8817842e-16  0.0000000e+00 -8.8817842e-16
 -8.8817842e-16]
Решение методом Гаусса(полного выбора):
0.85802359 0.87622901 0.61452891 0.67300804 0.79881835
Получившийся вектор свободных членов:
[4.2 4.2 4.2 4.2 4.2]
Отклонение от изначального вектора свободных членов:
[ 0.0000000e+00  0.0000000e+00  8.8817842e-16 -8.8817842e-16
  0.0000000e+00]
```

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:

```
Исходная матрица:
[1.68487038 2.66603969 1.91516038 4.63748562 2.00836871] = 6.693761213944104
[4.72371577 2.48454414 3.99023282 0.22521519 0.44733542] = 3.933368408846455
[0.98063884 2.15562175 0.5387487  0.18511887 4.17915967] = 0.097097650135528
[3.35075932 3.21289714 2.39734627 0.30899069 3.27583258] = 3.129805485662157
[0.72177109 0.16659261 1.72007686 4.09579363 1.3323955 ] = 5.467324832091167
Решение методом Гаусса(полного выбора):
7.14209624 -2.43400598 -6.13481368 2.66194817 0.27575471
Получившийся вектор свободных членов:
[6.69376121 3.93336841 0.09709765 3.12980549 5.46732483]
Отклонение от изначального вектора свободных членов:
[-8.88178420e-16  8.88178420e-16 -6.66133815e-16 -8.88178420e-16
 -8.88178420e-16]
```

Тестовый пример 2.

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

```
Система с нулевой строкой:
[0. 0. 0.] = 7.0
[1. 2. 3.] = 8.0
[4. 5. 6.] = 9.0
Решение методом Гаусса:
Система несовместна
```

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.

```
Система с нулевой строкой:
[0. 0. 0.] = 0.0
[1. 2. 3.] = 8.0
[4. 5. 6.] = 9.0
Решение методом Гаусса:
Бесконечное кол-во решений
```

Тестовый пример 4.

```
Система с нулевой диагональю:
[0. 1. 2.] = 7.0
[3. 0. 4.] = 8.0
[5. 6. 0.] = 9.0
Решение функцией встроенной в пакет:
[-0.64285714  2.03571429  2.48214286]
Решение методом Гаусса(полного выбора):
[-0.64285714  2.03571429  2.48214286]
Отклонение от встроенной функции:
[ 2.22044605e-16 -4.44089210e-16 -4.44089210e-16]
```

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.