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
Форинов Егор Вячеславович

____
(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 алгоритм метода и программу его реализации, получить численное решение данной СЛАУ;
- составить алгоритм решения СЛАУ указанными методами, применимый для организации вычислений на ЭВМ;
- составить программу решения СЛАУ по разработанному алгоритму;
- 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 12

Программная реализация

Полученные результаты сверим с решением, полученным с использованием пакета *Math.Net.Numerics*.

```
Console.WriteLine("Solution of packet Math.Net.Numerics");
Console.WriteLine(A.Solve(b));
```

Исходные данные:

Матрица А, полученная в результате вычисления А = 12С + D:

```
Source matrx A:
DenseMatrix 5x5-Double
 4,73   0,81   3,07   0,92  -0,53
-0,53   4,73   0,81   3,07   0,92
 3,32  -0,53   4,73   0,81   3,07
 0,67   3,32  -0,53   4,73   0,81
 0,81   0,67   3,32  -0,53   4,73
```

Код прямого обхода:

```
static void gauss(ref Matrix<double> A, ref Vector<double> b)
{
    var aCopy = Matrix<double>.Build.Dense(A.RowCount, A.ColumnCount);
    A.CopyTo(aCopy);
    var bCopy = Vector<double>.Build.Dense(b.Count);
    b.CopyTo(bCopy);
    var check = checkDiagonal(ref aCopy);
    if (!check)
    {
        Console.WriteLine("Бесконечное количество решений или нет решений!");
        return;
    }
    straightRun(ref aCopy, ref bCopy);
    var ans = reverseRun(ref aCopy, ref bCopy);
    Console.WriteLine(ans);
    Console.WriteLine($"Error: {error(residual(ref A, ref b, ref ans))}");
}
```

```
static void run(ref Matrix<double> A, ref Vector<double> b, int i)
{
    for (var j = i; j < A.RowCount; j++)
    {
        var factor = A[j, i - 1] / A[i - 1, i - 1];
        var rowToSubstract = A.Row(i - 1) * factor;
        b[j] -= factor * b[i - 1];
        A.SetRow(j, A.Row(j) - rowToSubstract);
    }
}

// ReSharper disable once InconsistentNaming
static void straightRun(ref Matrix<double> A, ref Vector<double> b)
{
    for (var i = 1; i < A.RowCount; i++)
        run(ref A, ref b, i);
}
```

```
static Vector<double> reverseRun(ref Matrix<double> A, ref Vector<double> b)
{
    var ans = Vector<double>.Build.Dense(A.RowCount);
    for (var i = A.RowCount - 1; i >= 0; i--)
    {
        double diff = 0;
        for (var j = A.ColumnCount - 1; j > i; j--)
            diff += ans[j] * A[i, j];
        ans[i] = (b[i] - diff) / A[i, i];
    }
    return ans;
}
```

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

```
static void gaussColumn(ref Matrix<double> A, ref Vector<double> b)
{
    var aCopy = Matrix<double>.Build.Dense(A.RowCount, A.ColumnCount);
    A.CopyTo(aCopy);
    var bCopy = Vector<double>.Build.Dense(b.Count);
    b.CopyTo(bCopy);
    var check = checkDiagonal(ref aCopy);
    if (!check)
    {
        Console.WriteLine("Бесконечное количество решений или нет решений!");
        return;
    }
    straightRunColumn(ref aCopy, ref bCopy);
    var ans = reverseRun(ref aCopy, ref bCopy);
    Console.WriteLine(ans);
    Console.WriteLine($"Error: {error(residual(ref A, ref b, ref ans))}");
}
```

```
static void gaussMax(ref Matrix<double> A, ref Vector<double> b)
{
    var aCopy = Matrix<double>.Build.Dense(A.RowCount, A.ColumnCount);
    A.CopyTo(aCopy);
    var bCopy = Vector<double>.Build.Dense(b.Count);
    b.CopyTo(bCopy);
    var check = checkDiagonal(ref aCopy);
    if (!check)
    {
        Console.WriteLine("Бесконечное количество решений или нет решений!");
        return;
    }
    var positions = straightRunMax(ref aCopy, ref bCopy);
    var ans = reverseRun(ref aCopy, ref bCopy);
    var ansRebuild = Vector<double>.Build.DenseOfVector(ans);
    for (var i = 0; i < ans.Count; i++)
    {
        ansRebuild[positions[i]] = ans[i];
    }
    Console.WriteLine(ansRebuild);
    Console.WriteLine($"Error: {error(residual(ref A, ref b, ref ansRebuild))}");
}
```

Также была реализована вспомогательная функция для проверки матрицы на нули на главной диагонали и, в случае наличия нулей, перестановки ее столбцов.

```
static bool checkDiagonal(ref Matrix<double> A)
{
    for (var i = 0; i < A.RowCount; i++)
    {
        if (A[i, i] == 0)
        {
            var check = true;
            for (var j = 0; j < A.ColumnCount; j++)
            {
                if (A[i, j] != 0 && A[j, i] != 0)
                {
                    swapColumns(ref A, i, j);
                    check = false;
                    break;
                }
            }
            if (check)
                return false;
        }
    }
    return true;
}
```

Полученные результаты

При получении результатов также была вычислена невязка для оценки погрешности метода.

Решение с помощью встроенного пакета.

```
Solution of packet Math.Net.Numerics
DenseVector 5-Double
  0,864869
  0,827955
-0,0816668
 0,0576127
  0,686341
```

Решение методом Гаусса.

```
Solution of gauss method:
DenseVector 5-Double
  0,864869
  0,827955
-0,0816668
 0,0576127
  0,686341

Error: 1,2560739669470201E-15
```

Решение методом Гаусса с модификацией частичного выбора по столбцу.

```
Solution of gaussColumn method:
DenseVector 5-Double
  0,864869
  0,827955
-0,0816668
 0,0576127
  0,686341

Error: 1,2560739669470201E-15
```

Решение методом Гаусса с модификацией выбора по всей матрице.

```
Solution of gaussMax method:
DenseVector 5-Double
  0,864869
  0,827955
-0,0816668
 0,0576127
  0,686341

Error: 1,2560739669470201E-15
```

Также для моего варианта я изменил исходную матрицу коэффициентов и посчитал число обусловленности исходной матрицы :

```
Condition number:
12,050681902445357

Source matrix b:
DenseVector 5-Double
4,3
4,1
4,3
4,1
4,3
```

Ответы получились следующими :

```
Solution of gaussMax method:
DenseVector 5-Double
  0,900777
  0,829634
-0,0921409
 0,0258561
   0,70489

Error: 8,881784197001252E-16
```

Максимальное расхождение с ответами для предыдущего вектора свободных коэффициентов получилось 0.2. Это связано с числом обусловленности исходной матрицы, которое равно 12, что считается, хоть и не идеальным результатом, но приемлимым.

Тестовый пример 1.

С помощью пакета MathNet.Numerics 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]
```

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), составил алгоритм решения СЛАУ данными методами и создал программу их реализации на языке C# для решения поставленной задачи, правильность работы программы проверил на тестовых примерах.

На основании тестов можно сделать следующие выводы:

- данный метод является универсальным методом решения СЛАУ, т.к нет необходимости предварительно исследовать систему или проверять её на совместность;
- погрешность метода обусловлена точностью представления в памяти компьютера дробных чисел.
- имеет ограничение в использовании (на главной диагонали не должно быть нулевых элементов), однако его можно обойти, взаимно поменяв местами уравнения системы и на главной диагонали будут не нули.