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 №2

на тему:

«ЧИСЛЕННОЕ РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ (СЛАУ) МЕТОДОМ ПРОСТЫХ ИТЕРАЦИЙ И МЕТОДОМ ЗЕЙДЕЛЯ»

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

Цель работы

- изучить метод простых итераций и метод Зейделя, получить численное решение заданной СЛАУ;
- составить программу решения СЛАУ указанными методами, применимый для организации вычислений на ЭВМ;
- выполнить тестовые примеры и проверить правильность работы программы

Задание:

Методом простых итераций и методом Зейделя 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:

$$C = \begin{bmatrix} 0{,}01 & 0 & -0{,}02 & 0 & 0 \\ 0{,}01 & 0{,}01 & -0{,}02 & 0 & 0 \\ 0 & 0{,}01 & 0{,}01 & 0 & -0{,}02 \\ 0 & 0 & 0{,}01 & 0{,}01 & 0 \\ 0 & 0 & 0 & 0{,}01 & 0{,}01 \end{bmatrix}, \quad D = \begin{bmatrix} 1{,}33 & 0{,}21 & 0{,}17 & 0{,}12 & -0{,}13 \\ -0{,}13 & -1{,}33 & 0{,}11 & 0{,}17 & 0{,}12 \\ 0{,}12 & -0{,}13 & -1{,}33 & 0{,}11 & 0{,}17 \\ 0{,}17 & 0{,}12 & -0{,}13 & -1{,}33 & 0{,}11 \\ 0{,}11 & 0{,}67 & 0{,}12 & -0{,}13 & -1{,}33 \end{bmatrix}.$$

Вектор $b = (1{,}2;\ 2{,}2;\ 4{,}0;\ 0{,}0;\ -1{,}2)^T$.

Вариант 12

Программная реализация

Для проверки решения умножим исходную матрицу на полученный вектор решений и сравним с изначальным вектором свободных членов.

Исходные данные:

Матрица A, полученная в результате вычисления A = 12C + D:

```
Исходные данные:
DenseMatrix 5x5-Double
 1,45   0,21  -0,07   0,12  -0,13
-0,01  -1,21  -0,13   0,17   0,12
 0,12  -0,01  -1,21   0,11  -0,13
 0,17   0,12  -0,01  -1,21   0,11
 0,11   0,67   0,12  -0,01  -1,21

DenseVector 5-Double
 1,2
 2,2
   4
   0
-1,2
```

Код простых итераций:

```
static Vector<double> iterations(Matrix<double> A, Vector<double> b, double err)
{
    var A_modified = Matrix<double>.Build.DenseOfMatrix(A);
    var b_modified = Vector<double>.Build.DenseOfVector(b);

    for (var i = 0; i < A.RowCount; i++)
        for (var j = 0; j < A.ColumnCount; j++)
            A_modified[i, j] /= -A[i, i];

    for (var i = 0; i < b.Count; i++)
        b_modified[i] = b[i] / A[i, i];

    double acc = 1;
    var prev = Vector<double>.Build.Dense(b.Count);
    var curr = Vector<double>.Build.Dense(b.Count);

    int iterations = 0;

    while (acc > err)
    {
        curr = A_modified.Multiply(prev) + b_modified + prev;
        acc = 0;
        for (var i = 0; i < curr.Count; i++)
        {
            if (Math.Abs(curr[i] - prev[i]) > acc)
            {
                acc = Math.Abs(curr[i] - prev[i]);
                break;
            }
        }
        prev = curr;
        iterations++;
    }

    Console.WriteLine($"В итерационном методе кол-во итераций: {iterations}");

    return prev;
}
```

Код метода Зейделя:

```
static Vector<double> zeidel(Matrix<double> A, Vector<double> b, double err)
{
    double acc = 1;
    var ans :Vector<double>? = Vector<double>.Build.Dense(b.Count);
    var problem :Vector<double>? = Vector<double>.Build.Dense(b.Count);

    int iterations = 0;

    while (acc > err)
    {
        for (int j = 0; j < A.RowCount; j++)
        {
            double diff = b[j];
            for (int k = 0; k < A.RowCount; k++)
                diff -= A[j, k] * ans[k];
            diff /= A[j, j];
            diff += ans[j];
            problem[j] = Math.Abs(diff - ans[j]);
            ans[j] = diff;
            acc = Math.Abs(problem.Max());
        }
        iterations++;
    }

    Console.WriteLine($"В методе Зейделя кол-во итераций: {iterations}");

    return ans;
}
```

Код проверок для возможности применения данных методов:

```
static bool checkDominant(Matrix<double> A)
{
    for (int i = 0; i < A.RowCount; i++)
    {
        double rowSum = 0;
        for (int j = 0; j  < A.ColumnCount; j++)
            rowSum += A[i, j];
        rowSum -= A[i, i];
        if (A[i, i] < rowSum)
            return false;
    }

    return true;
}
```

Полученные результаты

```
В методе Зейделя кол-во итераций: 6
DenseVector 5-Double
   0,882889
   -1,48924
   -3,19874
-0,00356536
 -0,0698236
```

```
В итерационном методе кол-во итераций: 7
DenseVector 5-Double
   0,882848
   -1,48915
   -3,19878
-0,00348911
  -0,069648
```

```
Решение системы с помощью математического пакета:
DenseVector 5-Double
   0,882882
   -1,48924
   -3,19874
-0,00356332
 -0,0698225
```

Тестовый пример 1.

С помощью пакета MathNet.Numerics создадим матрицу и вектор свободных членов и заполним их случайными числами:

```
Исходная матрица:
[8.33060289 2.89682981 0.40123842 2.60653485 1.58848179] = 1.99662355
[2.23101953 8.45617941 0.34926343 1.52959562 0.58708367] = 4.69119701
[0.276433   1.19903174 5.54678509 1.27876262 0.39383994] = 4.52123997
[2.21418937 1.10934538 2.37985173 7.72672466 1.94227667] = 4.84621620
[2.66539241 2.92417303 1.31808922 1.65837782 5.28908716] = 4.76331142
Решение методом простых итераций:
Кол-во итераций:  80
Вектор решений:
[-0.13575644  0.48049017  0.61825529  0.29156537  0.45778314]
Получившийся вектор свободных значений:
1.99618849 4.69090388 4.52106078 4.84578185 4.76288830
Отклонение от изначального вектора свободных членов
[-0.00043506 -0.00029314 -0.00017919 -0.00043435 -0.00042313]
Решение методом Зейделя:
Кол-во итераций:  8
Вектор решений:
[-0.13572219  0.48051233  0.61827963  0.29160075  0.45781647]
Получившийся вектор свободных значений:
1.99669297 4.69124992 4.52129021 4.84627828 4.76331143
Отклонение от изначального вектора свободных членов
[ 6.94167244e-05  5.29060881e-05  5.02281960e-05  6.20787742e-05
 -8.88178420e-16]
```

Тестовый пример 2.

В данном примере мы видим матрицу, в которой диагональ не является доминантной.

```
Исходная матрица:
[1. 2. 3.] = 1.0
[2. 1. 3.] = 2.0
[2. 3. 1.] = 3.0
Сумма модулей по строке 0 (5.0) больше модуля диагонального элемента 1.0
Сумма модулей по столбцу 0 (4.0) больше модуля диагонального элемента 1.0
||B|| больше 1
Нельзя решить методом Зейдаля или простых итераций
```

Вывод

В ходе выполнения лабораторной работы я изучил метод простых итераций и метод Зейделя, написал программу их реализации на языке Python для решения СЛАУ, правильность работы программы проверил на тестовых примерах.

На основании тестов можно сделать следующие выводы:

- Программа позволяет получить решения системы с заданной точностью (заданная точность в условиях лабораторной работы 10^-4);
- Метод Зейделя эффективнее по сравнению с методом простых итераций, так как затрачивает меньшее число итераций;
- Имеет ограничение в использовании (главная диагональ должна быть преимущественной), однако существуют матрицы, которые имеют не преимущественную диагональ и решаются с помощью метода простых итераций или метода Зейделя.
- Метод Зейделя более точен, так как использует уже найденные значения вектора решения на данной итерации, в отличие метода простых итераций.