МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Поволжский государственный университет телекоммуникаций и информатики»

Факультет информатики и вычислительной техники

Кафедра ПрИ

# ОТЧЁТ

## ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

**Дисциплина: Численные методы**

Выполнил: студент
ПрИ-21 Морзюков
М.А.
Проверил(а):
Осанов В.А.

Самара 2024

## Вариант №11

**Цель работы:** изучить решение систем линейных уравнений, итерационными методами: методом Якоби, методом Зейделя и методом верхней релаксации (обобщенный метод Зейделя).

| | | | | | |
|---|---|---|---|---|---|
| 11 | 13.4000 | .0581 | .0702 | .0822 | 17.7828 |
| | .0408 | 12.5000 | .0650 | .0770 | 19.0599 |
| | .0356 | .0477 | 11.6000 | .0718 | 19.9744 |
| | .0304 | .0425 | .0546 | 10.7000 | 20.5261 |

**Метод Якоби:** это метод итеративного решения систем линейных уравнений, основанный на разбиении уравнений на отдельные компоненты. Каждый компонент вычисляется независимо, используя текущие значения переменных. Этот метод эффективен для матриц с диагональным преобладанием.

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right)$$

```
public static double[] Yacobi(double[][] A, double[] B, double eps, int maxIterations)
{
    int n = A.Length;
    double[] x = new double[n];
    double[] xNew = new double[n];
    int iterations = 0;

    if (!HasDiagonalDominance(A))
    {
        Console.WriteLine("Матрица не имеет диагонального преобладания.");
        return null;
    }

    while (iterations < maxIterations)
    {
        for (int i = 0; i < n; i++)
        {
            double sum = B[i];
            for (int j = 0; j < n; j++)
            {
                if (j != i)
                {
                    sum -= A[i][j] * x[j];
                }
            }
            xNew[i] = sum / A[i][i];
        }

        bool converged = true;
        for (int i = 0; i < n; i++)
        {
            if (Math.Abs(xNew[i] - x[i]) > eps)
            {
                converged = false;
                break;
            }
        }
        Array.Copy(xNew, x, n);
        iterations++;
        if (converged)
        {
            break;
        }
    }

    Console.WriteLine("Количество итераций: " + iterations);
    return x;
}
```

**Метод Зейделя:** итеративный метод, похожий на метод Якоби, но обновляет значения переменных по мере их вычисления. Это означает, что новые значения используются немедленно для вычисления следующих переменных, что может ускорить сходимость по сравнению с методом Якоби.

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} \right)$$

```
public static double[] Zeidel(double[][] A, double[] B, double eps, int maxIterations)
{
    int n = A.Length;
    double[] x = new double[n];
    double[] prevX = new double[n];
    int iterations = 0;

    if (!HasDiagonalDominance(A))
    {
        Console.WriteLine("Матрица не имеет диагонального преобладания.");
        return null;
    }

    for (int i = 0; i < n; i++)
    {
        x[i] = B[i] / A[i][i];
    }

    while (iterations < maxIterations)
    {
        Array.Copy(x, prevX, n);
        for (int i = 0; i < n; i++)
        {
            double sum = B[i];
            for (int j = 0; j < n; j++)
            {
                if (i != j)
                {
                    sum -= A[i][j] * x[j];
                }
            }
            x[i] = sum / A[i][i];
        }
```

```
        bool converged = true;
        for (int i = 0; i < n; i++)
        {
            if (Math.Abs(x[i] - prevX[i]) > eps)
            {
                converged = false;
                break;
            }
        }

        iterations++;
        if (converged)
        {
            break;
        }
    }

    Console.WriteLine("Количество итераций: " + iterations);
    return x;
}
```

**Метод верхней релаксации (обобщенный метод Зейделя):** модификация метода Зейделя, которая вводит параметр релаксации w для контроля скорости сходимости. Этот метод позволяет "ускорить" процесс,

корректируя новые значения переменных с учетом предыдущих, улучшая тем самым общую производительность итераций.

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right)$$

```
public static double[] ZeidelEx(double[][] A, double[] B, double w, double eps)
{
    int n = A.Length;
    double[] x0 = new double[n];
    double[] x = new double[n];
    double e;
    int step = 0;

    if (!HasDiagonalDominance(A))
    {
        Console.WriteLine("Матрица не имеет диагонального преобладания.");
        return null;
    }

    for (int i = 0; i < n; i++)
    {
        x0[i] = B[i] / A[i][i];
    }

    do
    {
        for (int i = 0; i < n; i++)
        {
            x[i] = w * (B[i] / A[i][i]);
            for (int j = 0; j <= i - 1; j++)
            {
                x[i] -= w * (A[i][j] * x[j] / A[i][i]);
            }
            for (int j = i + 1; j < n; j++)
            {
                x[i] -= w * (A[i][j] * x0[j] / A[i][i]);
            }
            x[i] += (1 - w) * x0[i];
        }

        e = 0;
        for (int i = 0; i < n; i++)
        {
            if (Math.Abs(x[i] - x0[i]) > e)
            {
                e = Math.Abs(x[i] - x0[i]);
            }
            x0[i] = x[i];
        }

        step++;
    } while (e >= eps);

    Console.WriteLine("Количество итераций: " + step);
    return x;
}
```

**Результат выполнения программы:**

```
Решение по методу Якоби: Количество итераций: 4
x0 = 1,300 x1 = 1,500 x2 = 1,700 x3 = 1,900
Решение по методу Зейделя: Количество итераций: 3
x0 = 1,300 x1 = 1,500 x2 = 1,700 x3 = 1,900
Решение по обобщенному методу Зейделя: Количество итераций: 19
x0 = 1,300 x1 = 1,500 x2 = 1,700 x3 = 1,900
Нажмите любую клавишу для выхода...
```

**Проверка:**

| Матрица A | | | | Результат B |
|---|---|---|---|---|
| 13.4000 | 0.0581 | 0.0702 | 0.0822 | 17.7828 |
| 0.0408 | 12.5000 | 0.0650 | 0.0770 | 19.0599 |
| 0.0356 | 0.0477 | 11.6000 | 0.0718 | 19.9744 |
| 0.0304 | 0.0425 | 0.0546 | 10.7000 | 20.5261 |

| N | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $e_1$ | $e_2$ | $e_3$ | $e_4$ |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | | | | |
| 1 | 1.327 | 1.52 | 1.712 | 1.9 | 1.327 | 1.52 | 1.712 | 1.9 |
| 2 | 1.3 | 1.5 | 1.7 | 1.9 | -0.0272 | -0.0205 | -0.0116 | 0.000218 |
| 3 | 1.3 | 1.5 | 1.7 | 1.9 | 0.000148 | 5.8E-5 | -2.0E-6 | -1.0E-6 |

**Сайт для проверки: https://math.semestr.ru/optim/zeidel.php**