**Санкт-Петербургский национальный исследовательский университет**

**информационных технологий, механики и оптики**

**Кафедра информатики и прикладной математики**

Вычислительная математика

Лабораторная работа №1

«Решение СЛАУ. Метод Гаусса ».

Выполнил Кудряшов А.А.

Группа 2121

Проверил Шипилов П.А.

2012 г.

Задача : Организация вычислительного алгоритма решения систем линейных алгебраических уравнений методом Гаусса.

1. **Описание метода**

**Метод Гаусса** основан на приведении матрицы системы к треугольному виду. Это достигается последовательным исключением неизвестных из уравнений системы. Сначала с помощью первого уравнения исключается *x1* из всех последующих уравнений системы. Затем с помощью второго уравнения исключается х2 из третьего и всех последующих уравнений. Этот процесс, называемый *прямым ходом метода Гаусса,* продолжается до тех пор, пока в левой части последнего (n-го) уравнения не останется лишь один член с неизвестным *хп,* т. е. матрица системы будет приведена к треугольному виду (к такому виду приводится лишь невырожденная матрица, в противном случае метод Гаусса неприменим).

*Обратный ход метода Гаусса* состоит в последовательном вычислении искомых неизвестных: решая последнее сравнение, находим единственное неизвестное *хп.* Далее, используя это значение, из предыдущего уравнения вычисляем *хп-1* и т. д. Последним найдем *x1* из первого уравнения. Когда матрица системы примет треугольный вид закончится прямой ход метода Гаусса.

1. **Расчетные формулы**

Пусть:

а – матрица коэффициентов СЛАУ

b – матрица (вектор) свободных членов СЛАУ

i – номер строки

j – номер столбца

i,j = 1..n

ki = ai(i-1) / a(i-1)(i-1)

a’ij = aij - a(i-1)j\*k

b’i = bi - bi-1\*k

1. **Программа (алгоритм)**

private void Ok\_Click(object sender, EventArgs e)

 {

 var FirstMatrix = new int [MatrixArray.GetLength(0),MatrixArray.GetLength(1)];

 var FirstAnswer = new int[MatrixArray.GetLength(1)];

 for (int i = 0; i < MatrixArray.GetLength(0); ++i)

 {

 FirstAnswer[i] = int.Parse(Answers[i].Text);

 for (int j = 0; j < MatrixArray.GetLength(1); ++j)

 {

 FirstMatrix[i, j] = int.Parse(MatrixArray[i, j].Text);

 }

 }

 double k = 0;

 //Приведение к треугольной матрице

 int HelpIndex = 1;

 string HelpChance = "0";

 for (int i = 0; i < N; ++i)

 {

 for (int AlgoritmRow = i + 1; AlgoritmRow < N; ++AlgoritmRow)

 {

 if (double.Parse(MatrixArray[AlgoritmRow, i].Text) != 0)

 {

 k = double.Parse(MatrixArray[AlgoritmRow, i].Text) / double.Parse(MatrixArray[i, i].Text);

 for (int i1 = i; i1 < N; ++i1)

 {

 MatrixArray[AlgoritmRow, i1].Text = (double.Parse(MatrixArray[AlgoritmRow, i1].Text) - double.Parse(MatrixArray[i, i1].Text) \* k).ToString();

 }

 Answers[AlgoritmRow].Text = (double.Parse(Answers[AlgoritmRow].Text) - double.Parse(Answers[i].Text) \* k).ToString();

 // Проверка деления на ноль, замена строк

 if (MatrixArray[AlgoritmRow, AlgoritmRow].Text == "0")

 {

 if (AlgoritmRow + HelpIndex >= N-1)

 {

 MessageBox.Show("Определитель матрицы равен нулю");

 return;

 }

 for (int c = 0; c < N; ++c)

 {

 HelpChance = MatrixArray[AlgoritmRow, c].Text;

 MatrixArray[AlgoritmRow, c].Text = MatrixArray[AlgoritmRow + HelpIndex, c].Text;

 MatrixArray[AlgoritmRow + HelpIndex, c].Text = HelpChance;

 }

 HelpIndex++;

 }

 else

 {

 HelpIndex = 1;

 }

 }

 else { continue; }

 }

 }

 //Вычисление определителя

 double Det = double.Parse(MatrixArray[0, 0].Text);

 for (int i = 1; i < N; ++i)

 {

 Det \*= double.Parse(MatrixArray[i, i].Text);

 }

 if (Det == 0)

 {

 MessageBox.Show("Определитель равен нулю");

 }

 //Определение неизвестных

 var Result = new double[N];

 for (int i = N - 1; i >= 0; --i)

 {

 double OpenArgSum = 0;

 for (int j = i+1; j < N; ++j)

 {

 OpenArgSum += double.Parse(MatrixArray[i, j].Text)\*Result[j];

 }

 Result[i] = (double.Parse(Answers[i].Text) - OpenArgSum) / double.Parse(MatrixArray[i, i].Text);

 }

 //Вычисление невязки

 double[] BadNumber = new double[N];

 for (int i = 0; i < N; ++i)

 {

 double RowCheckSumm = 0;

 for (int j = 0; j < N; ++j)

 {

 RowCheckSumm += FirstMatrix[i, j] \* Result[j];

 }

 BadNumber[i] = FirstAnswer[i] - RowCheckSumm;

 }

 string OutString="Det = " + Det +"\n\n\nНеизвестные:\n\n" ;

 for (int i = 0; i < Result.Length; ++i)

 {

 { OutString += "X" + (i + 1) + " = " + Result[i] + " Невязка: " + BadNumber[i] + "\n"; }

 }

 MessageBox.Show(OutString);

 }

 }

}

1. **Сравнение, вывод**

В процессе выполнения лабораторной работы мною сделаны следующие сравнительные выводы: метод Гаусса является одним из наиболее простых методов для переведения его в алгоритмический язык, но при учете алгоритма замены строк и проверки на присутствие строк, подходящих для замены и удовлетворении условия невырожденности матрицы СЛАУ, его сложность приближается и даже в каком то смысле превосходит сложность метода Гаусса с выбором главного элемента, в то же время, являясь более простым, но обладая большим количеством арифметических действий является менее точным чем итерационные методы решения СЛАУ. Но в свою очередь хочу заметить, что даже визуального анализа бывает достаточно для различия значащих значений коэффициентов и их погрешностей.