Вопрос 13. Точные и приближенные методы решения систем линейных уравнений. Метод Гаусса решения систем линейных уравнений. Погрешности вычисления.

Точными являются такие методы, которые позволяют получить решение системы после выполнения конечного числа арифметических операций над коэффициентами системы и их свободными членами. Причем решение получится точным только тогда, когда коэффициенты и правые части системы (3) известны точно и все арифметические действия над ними выполняются без округлений. Из точных методов рассмотрим метод Гаусса и правило Крамера. Однако на практике даже этими методами не всегда удается получить точное решение, ибо в ЭВМ точные коэффициенты представляются приближенно с некоторой погрешностью, а в процессе вычислений необходимо проводить округление чисел.

Алгоритм метода Гаусса

Пусть дана система n линейных уравнений с n переменными:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n. \end{cases}$$

Коэффициенты a_{ij} при переменных будем рассматривать как элементы двумерного массива A (N, N), а свободные члены b_i— как элементы одномерного массива B (N).Решение x_i($i = \overline{1,n}$) разместим в одномерном массиве B (N). Коэффициенты a_{ij} и свободные члены b_i будем рассматривать как элементы расширенной матрицы

$$\begin{pmatrix} a_{11}a_{12}\ldots a_{1n}b_1 \\ a_{21}a_{22}\ldots a_{2n}b_2 \\ \ldots\ldots\ldots\ldots \\ a_{n1}a_{n2}\ldots a_{nn}b_n \end{pmatrix}.$$

Предписываемые методом Гаусса преобразования будем выполнять над элементами расширенной матрицы. Опишем формально алгоритм решения линейной системы методом Гаусса без выбора главного элемента.

1. Элементы первой строки расширенной матрицы (A | B)делим на a_{11}. Полученную после такого деления первую строку умножаем последовательно на a_{k1}($k = \overline{2,n}$) и вычитаем ее затем из k-ой строки ($k = \overline{2,n}$). После этого преобразования в первом столбце массива A (кроме $a_{11}^1 = 1$) все элементы будут равны нулю, то есть получим матрицу:

$$\left\|\begin{pmatrix} 1 & a_{12}^{(1)} & a_{13}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} & b_2^{(1)} \\ \dots & & \dots & & & \dots \\ 0 & a_{n2}^{(1)} & a_{n3}^{(1)} & \dots & a_{nn}^{(1)} & b_n^{(1)} \end{pmatrix}\right.$$

2. Элементы второй строки расширенной матрицы делим на $a_{22}^{(1)}$, затем умножаем ее последовательно на $a_{k2}^{(1)}$ и вычитаем из оставшихся строк при $k = \overline{3,n}.$

3. Продолжаем этот процесс исключения переменных (получения нулей) до тех пор, пока подобная процедура не будет проделана с (n — 1)-й строкой матрицы. После этого получим матрицу:

$$\left\|\begin{pmatrix} 1 & a_{12}^{(1)} & a_{13}^{(1)} & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & 1 & a_{23}^{(2)} & a_{2n}^{(2)} & b_2^{(2)} \\ 0 & 0 & 0 & 1\, a_{n-1,n}^{(n-1)} & b_{n-1}^{(n-1)} \\ 0 & 0 & 0 & 0\, a_{nn}^{(n-1)} & b_n^{(n-1)} \end{pmatrix}\right.$$

4. Элементы n-й строки делим на $a_{nn}^{(n-1)}$ и в результате получаем:

$$\left\|\begin{pmatrix} 1 & a_{12}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & 1 & \dots & a_{2n}^{(2)} & b_2^{(2)} \\ .. & .. & .. & .. & .. \\ 0 & 0 & \dots & 1 & b_n^{(n)} \end{pmatrix}\right. \Rightarrow x_n = \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}} = b_n^{(n)}.$$

На этом закончился прямой ход метода Гаусса.

5. Выполняем обратный ход метода Гаусса: в *(n*—1)-ю строку последней матрицы подставляем значение x_n и находим значение x_{n-1}, затем последовательно находим x_{n-2}, x_{n-3}, ... , x_2, x_1 по формулам:

$$x = b_k^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)} x_j.$$

Этот алгоритм является экономичным в смысле использования памяти, так как все промежуточные и окончательные значения элементов в процессе преобразования матриц последовательно хранятся в тех же ячейках памяти, что и массивы *A* и *B*.Очередные значения диагональных элементов $a_{ii}^{(k)}$ перед началом преобразования строк будем присваивать простой переменной *D*, что позволит хранить их до окончания преобразования очередной строки матрицы.

Значения переменных x_n, x_{n-1}, ...,x_1 присваиваются элементам массива свободных членов *B*.

Метод Гаусса с выбором главного элементазаключается в том,что при прямом ходе производится выбор наибольшего по модулю (главного) элемента и перестановка строк или столбцов. Последнее исключает деление на 0, если матрица коэффициентов содержит нулевые элементы, и повышает точность вычислений при наличии ошибок округления. Обычно для программ, ведущих вычисления с числами с плавающей точкой, достаточен выбор A_{ii} ¹ 0.

Метод вращения является разновидностью метода Гаусса. Он обладает повышенной устойчивостью к “провалам” промежуточных вычислений. Этот метод обеспечивает приведение исходной системы к системе с верхней треугольной матрицей (см. литературу).

Погрешность метода.

Точность результатов будет определяться только точностью выполнения арифметических операций при преобразовании элементов матрицы, т.е. ошибкой округления. Контроль правильности полученного решения осуществляется подстановкой полученных значений $x_1 \ldots x_n$ в исходную систему уравнений и вычислением *невязок*, т.е. разностей между правыми и левыми частями уравнений:

$$r_k = a_{k,n+1} - \sum_{j=1}^{n} a_{kj} x_j$$, где $k = 1 \ldots n$. (6)

Специально отметим, что подставлять найденные значения $\vec{x}$ следует в *исходную* (не преобразованную к верхнетреугольному виду) систему.

Преимущества и недостатки метода.

Преимущество метода в том, что он позволяет достичь результата за заранее известное и фиксированное число действий. Точность результатов будет определяться правильным выбором порядка коэффициентов в матрице $\hat{A}$ и ее размерностью. Недостатком метода является резкое увеличение времени и погрешности вычислений с ростом n.

```
import java.util.Scanner;
public class labasel3111 {

public static void main(String[] args) {
	Scanner scan=new Scanner(System.in);
	int i=0;
	int j=0;
	int k=0;

	double c;
	int n=scan.nextInt();
	//double [][] a = {{2,1,2,3},{5,3,10,8},{4,2,9,9},{1,1,7,2},{20,11,40,37}};

	double [][] a=new double[n][n-1];
	//ввод массива
	for(i=0;i<n-1;i++) {
		for(j=0;j<n;j++) {
			System.out.print("a["+j+"]["+i+"]");
			a[j][i]=scan.nextInt();
		}
	}
	scan.close();
```

```
	//конец ввода

	for(k=0;k<n-1;k++) {
		//for(t=0;t<n;t++) {
	for(i=k;i<n-1;i++) {
		if(a[k][i]!=0) {break;}
		}
		if(i!=k) {
			for(j=k;j<n;j++) {
				c=a[j][k];
				a[j][k]=a[j][i];
				a[j][i]=c;
			}
		}
		c=a[k][k];
		i=k;
	for(j=k;j<n;j++) {
		a[j][i]=a[j][i]/c;
	}
	for(i=k+1;i<n-1;i++) {
		c=a[k][i];
		for(j=0;j<n;j++) {
			a[j][i]=a[j][i]-a[j][k]*c;

		}
	}
	}
	double [] x=new double[n-1];

	x[n-2]=a[n-1][n-2];
	for(i=n-2;i>0;i--) {
		double s=0;
		for( j=i+1;j<n;j++) {
			s=a[j-1][i-1]*x[j-1];
			x[i-1]=a[n-1][i-1]-s;
			a[n-1][i-1]=x[i-1];
		}
	}

	/*вывод массива
	for(i=0;i<n-1;i++) {
		for(j=0;j<n;j++) {
			System.out.print(a[j][i]+"\t");}
		System.out.println("");}
	*///конец вывода
	System.out.print("Корни уравнения"+"\t");
	for(i=0;i<n-1;i++) {
		System.out.print(x[i]+" ");
	}
}
}
```