Задание

Наименование задачи: решение системы линейных уравнений с помощью метода итераций.

Вид решения: программа и отчёт.

Реализация решения: язык C или C++.

Разработать алгоритм и написать программу реализующую: решение системы линейных уравнений, на множестве вещественных чисел, методом итераций. Предусмотреть возможность решения уравнений различного порядка без перекомпиляции программы. Исходные данные (матрицу коэффициентов уравнения, вектор правых частей), вводить в программу из консоли или из файла. Оценить вычислительную сложность решения задачи.

Task: solution of a system of linear equations using the iteration method.

Элементы теории

Метод итерации является приближенным методом решения совместных определенных систем линейных алгебраических уравнений.

Пусть дана линейная система

$$\left.\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n. \end{array}\right\} \qquad (1)$$

Введя в рассмотрение матрицы

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix}, \qquad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}, \qquad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{bmatrix},$$

систему (1) можно записать в виде матричного уравнения

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}. \qquad (1')$$

Предполагая, что диагональные коэффициенты

$$a_{ii} \neq 0 \qquad (i = 1,\ 2,\ \ldots,\ n),$$

разрешим первое уравнение системы (1) относительно x_1, второе – относительно x_2 и т. д. Тогда получим эквивалентную систему

$$\left.\begin{aligned} x_1 &= \beta_1 + \alpha_{12}x_2 + \alpha_{13}x_3 + \ldots + \alpha_{1n}x_n, \\ x_2 &= \beta_2 + \alpha_{21}x_1 + \alpha_{23}x_3 + \ldots + \alpha_{2n}x_n, \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n &= \beta_n + \alpha_{n1}x_1 + \alpha_{n2}x_2 + \ldots + \alpha_{n,\,n-1}x_{n-1}, \end{aligned}\right\} \qquad (2)$$

где

$$\beta_i = \frac{b_i}{a_{ii}}; \qquad \alpha_{ij} = -\frac{a_{ij}}{a_{ii}} \qquad \text{при } i \neq j$$

и $\alpha_{ij} = 0$ при $i = j$ $(i, j = 1, 2, \ldots, n)$.

Введя матрицы

$$\alpha = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \ldots & \alpha_{1n} \\ \alpha_{21} & \alpha_{22} & \ldots & \alpha_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ \alpha_{n1} & \alpha_{n2} & \ldots & \alpha_{nn} \end{bmatrix} \qquad \text{и} \qquad \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \cdot \\ \cdot \\ \cdot \\ \beta_n \end{bmatrix},$$

систему (2) можем записать в матричной форме

$$x = \beta + \alpha x. \qquad (2')$$

Систему (2) будем решать методом последовательных приближений. За нулевое приближение принимаем, например, столбец свободных членов $x^{(0)} = \beta$.

Далее, последовательно строим матрицы-столбцы

$x^{(1)} = \beta + \alpha x^{(0)}$ (первое приближение), $x^{(2)} = \beta + \alpha x^{(1)}$ (второе приближение) и т. д.

Вообще говоря, любое (k+1)-е приближение вычисляют по формуле

Если последовательность приближений $x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots$ имеет предел

$$x = \lim_{k \to \infty} x^{(k)},$$

то этот предел является решением системы (2). В самом деле, переходя к пределу в равенстве (3), будем иметь:

$$\lim_{k\to\infty} x^{(k+1)} = \beta + \alpha \lim_{k\to\infty} x^{(k)}$$

или

$$x = \beta + \alpha x,$$

т. е. предельный вектор x является решением системы (2'), а следовательно, и системы (1).

Напишем формулы приближений в развернутом виде:

$$\left.\begin{array}{l} x_i^{(0)} = \beta_i, \\ x_i^{(k+1)} = \beta_i + \sum\limits_{j=1}^{n} \alpha_{ij} x_j^{(k)} \\ (\alpha_{ii} = 0;\ i = 1, \ldots, n;\ k = 0, 1, 2, \ldots). \end{array}\right\} \qquad (3')$$

Оценим количество итераций и впоследствии – вычислительную сложность.

Пусть $x^{(k-1)}$ и $x^{(k)}$ (k >= 1) – два последовательных приближения решения линейной системы x = αx+β. При p >= 1 имеем:

$$\begin{aligned} \| x^{(k+p)} - x^{(k)} \| \leqslant \| x^{(k+1)} - x^{(k)} \| &+ \\ + \| x^{(k+2)} - x^{(k+1)} \| + \ldots &+ \| x^{(k+p)} - x^{(k+p-1)} \|. \end{aligned} \qquad (1)$$

Так как

$$x^{(m+1)} = \alpha x^{(m)} + \beta$$

и

$$x^{(m)} = \alpha x^{(m-1)} + \beta,$$

то

$$x^{(m+1)} - x^{(m)} = \alpha \left(x^{(m)} - x^{(m-1)}\right)$$

и, следовательно,

$$\begin{aligned} \| x^{(m+1)} - x^{(m)} \| &\leqslant \| \alpha \| \, \| x^{(m)} - x^{(m-1)} \| \leqslant \\ &\leqslant \| \alpha \|^{m-k} \| x^{(k+1)} - x^{(k)} \| \quad \text{при} \quad m > k \geqslant 1. \end{aligned}$$

Поэтому из формулы (1) получаем:

$$\| x^{(p+k)} - x^{(k)} \| \leqslant \| x^{(k+1)} - x^{(k)} \| + \\ + \|\alpha\| \, \| x^{(k+1)} - x^{(k)} \| + \ldots + \|\alpha\|^{p-1} \| x^{(k+1)} - x^{(k)} \| \leqslant \\ \leqslant \frac{1}{1-\|\alpha\|} \| x^{(k+1)} - x^{(k)} \|.$$

Переходя в последнем неравенстве к пределу при $p \to \infty$, получим окончательно:

$$\| x - x^{(k)} \| \leqslant \frac{\| x^{(k+1)} - x^{(k)} \|}{1-\|\alpha\|} \qquad (2)$$

при k>=1, или

$$\| x - x^{(k)} \| \leqslant \frac{\|\alpha\|}{1-\|\alpha\|} \| x^{(k)} - x^{(k-1)} \|.$$

Из формулы (2) будем иметь:

$$\| x - x^{(k)} \| \leqslant \frac{\|\alpha\|^k}{1-\|\alpha\|} \| x^{(1)} - x^{(0)} \|.$$

В частности, если выбрать

$$x^{(0)} = \beta,$$

то

$$x^{(1)} = \alpha\beta + \beta$$

$$\| x^{(1)} - x^{(0)} \| = \| \alpha\beta \| \leqslant \|\alpha\| \, \|\beta\|.$$

$$\| x - x^{(k)} \| \leqslant \frac{\|\alpha\|^{k+1}}{1-\|\alpha\|} \|\beta\|.$$

Для поиска количества итераций (k) примем, что правая часть этого неравенства меньше заданной точности. Тогда получим оценку k:

$$k > \frac{log(\frac{\varepsilon \, * \, (1 - ||\alpha||)}{||\beta||})}{log(||\alpha||)}$$

Таким образом, $k = \Omega(\frac{log(\frac{\varepsilon * (1-||\alpha||)}{||\beta||})}{log(||\alpha||)})$.

Листинг программы

```
#include <iostream>

#include <cmath>

using namespace std;

bool eps_check(double*, double*, int, double);

int main() {

	setlocale(LC_ALL, "Russian");

	int n;

	cout << "Введите размерность матрицы коэффициентов A" << endl;

	cin >> n;

	double** A = new double* [n];

	cout << "Введите элементы матрицы коэффициентов A" << endl;

	for (int i = 0; i < n; i++) {

		A[i] = new double[n];

		for (int j = 0; j < n; j++) {

			cin >> A[i][j];

		}

		if (A[i][i] == 0) {

			cout << "Диагональные элементы должны быть отличны от нуля!" <<
endl;

			return 0;

		}

	}

	cout << "Введите элементы столбца B" << endl;

	double* B = new double[n];

	for (int i = 0; i < n; i++) {

		cin >> B[i];

	}

	double** Alpha = new double* [n];
```

```
double* Beta = new double[n];
for (int i = 0; i < n; i++) {
	Alpha[i] = new double[n];
}
for (int i = 0; i < n; i++) {
	for (int j = 0; j < n; j++) {
		Alpha[i][j] = -A[i][j] / A[i][i];
	}
	Beta[i] = B[i] / A[i][i];
	Alpha[i][i] = 0;
}
double* x = new double[n];
double* x_prev = new double[n];
for (int i = 0; i < n; i++) {
	x[i] = Beta[i];
}
double eps;
cout << "Задайте точность вычислений" << endl;
cin >> eps;
double mul;
int iterations = 0;
do {
	for (int i = 0; i < n; i++) {
		x_prev[i] = x[i];
		mul = 0;
		for (int k = 0; k < n; k++) {
			mul += Alpha[i][k] * x[k];
		}
		x[i] = Beta[i] + mul;
```

```
		}
		iterations++;
	} while (!(eps_check(x_prev, x, n, eps)));
	cout << "Решение:" << endl;
	for (int i = 0; i < n; i++) {
		cout << "x" << i << " = " << x[i] << endl;
	}
	cout << "Количество итераций: " << iterations << endl;
	cout << "Проверка решения подстановкой:" << endl;
	double resi;
	for (int i = 0; i < n; i++) {
		resi = 0;
		for (int j = 0; j < n; j++) {
			resi += x[j] * A[i][j];
		}
		cout << "Уравнение " << i << ": " << resi << " " << "B" << i << " = " << B[i] << 
endl;;
	}
	for (int i = 0; i < n; i++) {
		delete[] A[i];
		delete[] Alpha[i];
	}
	delete[] B;
	delete[] Beta;
	return 0;
}
bool eps_check(double* x_prev, double* x, int dimension, double eps) {
	double norma_x;
	norma_x = 0; // k-норма
```

```
        for (int i = 0; i < dimension; i++) {
                norma_x += pow(abs(x_prev[i] - x[i]), 2);
        }
        norma_x = sqrt(norma_x);
        if (norma_x < eps) return 1;
        else return 0;
}
```

Вывод программы

Введите размерность матрицы коэффициентов A

4

Введите элементы матрицы коэффициентов A

3 2 -1 0

9.8 29 3 2

-2.337 2 5 4

0.01 0.89 -2 10.33

Введите элементы столбца B

2 8 3 3

Задайте точность вычислений

0.0001

Решение:

x0 = 1.07163

x1 = -0.202125

x2 = 0.81073

x3 = 0.463759

Количество итераций: 12

Проверка решения подстановкой:

Уравнение 0: 1.99992 B0 = 2

Уравнение 1: 8.00009 B1 = 8

Уравнение 2: 3.00003 B2 = 3

Уравнение 3: 3 B3 = 3

Введите размерность матрицы коэффициентов A

4

Введите элементы матрицы коэффициентов A

10 -1 2 -3

1 10 -1 2

2 3 20 -1

3 2 1 20

Введите элементы столбца B

0 5 -10 15

Задайте точность вычислений

0.0001

Решение:

x0 = 0.344695

x1 = 0.271871

x2 = -0.540344

x3 = 0.698126

Количество итераций: 5

Проверка решения подстановкой:

Уравнение 0: 1.34362e-05 B0 = 0

Уравнение 1: 5 B1 = 5

Уравнение 2: -10 B2 = -10

Уравнение 3: 15 B3 = 15

Введите размерность матрицы коэффициентов A

3

Введите элементы матрицы коэффициентов A

4 0.24 -0.08

0.09 3 -0.15

0.04 -0.08 4

Введите элементы столбца B

8 9 20

Задайте точность вычислений

0.0001

Решение:

x0 = 1.9092

x1 = 3.19496

x2 = 5.04481

Количество итераций: 4

Проверка решения подстановкой:

Уравнение 0: 8 B0 = 8

Уравнение 1: 9 B1 = 9

Уравнение 2: 20 B2 = 20

Выводы

По результатам работы программы можно сделать вывод, что процесс итерации хорошо сходится, если элементы матрицы α малы по абсолютной величине. То есть для успешного применения метода итерации модули диагональных коэффициентов исходной системы должны быть велики по сравнению с модулями недиагональных коэффициентов, причем свободные члены при этом роли не играют. Также была проведена теоретическая оценка количества итераций. И как выяснилось, практически она оказывается завышенной.

Заключение

В ходе выполнения домашнего задания были изучены теоретические основы метода итерации для приближенного решения систем линейных алгебраических уравнений. Также была разработана программа на языке C++ и оценена вычислительная сложность.

Список литературы

1. Демидович Б.П., Марон И.А. Основы вычислительной математики // М.: Наука, 1970. – 664 с.