## Задание №3

## Решение систем линейных алгебраических уравнений. Метод Гаусса

### Постановка задачи:

Требуется найти решение системы линейных уравнений вида Ax = B, т.е.

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \dots + a_{1,n}x_n = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \dots + a_{2,n}x_n = b_2 \\ \dots \\ a_{n,1}x_1 + a_{n,2}x_2 + \dots + a_{n,n}x_n = b_n \end{cases}$$

где элементы матрицы A и вектора B — вещественные числа.

Методов решения этой задачи существует много, мы начнем с метода Гаусса и его основных модификаций.

### Основная схема метода Гаусса

Введем обозначение  $a_{i,n+1} = b_i$ . Сначала исходная система сводится к эквивалентной системе с верхней треугольной матрицей вида

$$\begin{cases} x_1 + a_{1,2}^{(1)} x_2 + \dots + a_{1,n}^{(1)} x_n = a_{1,n+1}^{(1)} \\ x_2 + \dots + a_{2,n}^{(2)} x_n = a_{2,n+1}^{(2)} \\ \dots \\ x_n = a_{n,n+1}^{(n)} \end{cases}$$

с помощью преобразований

$$a_{k,j}^{(k)} = \frac{a_{k,j}^{(k-1)}}{a_{k,k}^{(k-1)}}, \qquad j = k, \dots, n+1.$$
 (1)

И

$$a_{i,j}^{(k)} = a_{i,j}^{(k-1)} - a_{k,j}^{(k)} a_{i,k}^{(k-1)}, \qquad i = k+1, \dots, n, \qquad j = k, \dots, n+1.$$
 (2)

 $a_{i,j}^{(k)} = a_{i,j}^{(k-1)} - a_{k,j}^{(k)} a_{i,k}^{(k-1)}, \qquad i = k+1, \dots, n, \qquad j = k, \dots, n+1. \tag{2}$  для всех  $k=1,\dots,n$ . В этом случае  $a_{i,j}^{(0)}$  — это элементы исходной матрицы. Следует проводить проверку того, что ведущие элементы (на которые производится деление) не являются близкими к нулю (если это случится, программа должна вывести на экран сообщение о близком к нулю ведущем элементе, а затем продолжить выполнение).

На втором шаге вычисляется решение:

$$x_i = a_{i,n+1}^{(i)} - \sum_{j=i+1}^n a_{i,j}^{(i)} x_j, \qquad i = n, n-1, \dots, 1.$$

Так как при втором шаге элементы матрицы ниже главной диагонали и на главной диагонали не используются (вернее, известно, чему они равны), их можно не вычислять (ради экономии времени). Также можно использовать их для тестирования программы.

# Схема Йордана

Матрица полностью диагонализируется, в результате решение просто совпадает с преобразованным вектором В. Схема диагонализации такая же, как и в случае Гаусса, только вычисление (2) преобразуется в

$$a_{i,j}^{(k)} = a_{i,j}^{(k-1)} - a_{k,j}^{(k)} a_{i,k}^{(k-1)}, \qquad i = 1, 2, \dots, k-1, k+1, \dots, n, \qquad j = k, \dots, n+1.$$
 (3)

Второго шага, естественно, нет.

### Схема с выбором ведущего элемента

Во избежание необходимости деления на малый ведущий элемент, можно реализовать схему Гаусса с добавкой условия: если  $|a_{p,q}|\geqslant |a_{i,j}|$  для всех  $i=k,\ldots,n,\ j=k,\ldots,n,$  то на очередном шаге поменять местами q-й и k-й столбцы и p-ю и k-ю строки. Так как при перестановке столбцов меняется порядок переменных, необходимо после приведения матрицы к верхнему треугольному (или диагональному) виду восстановить исходный порядок.

### Представление данных и результатов:

Требуется написать модуль (на Фортране и других языках, поддерживающих модули) или отдельные файлы (.c/.cpp и .h/.hpp для C/C++ и других языков, не поддерживающих модули) с функцией, получающей матрицу A и вектор B и возвращающей вектор решений. Все три метода должны реализовываться в виде одной функции с дополнительным параметром, задающим тип метода.

Модуль с функцией должен использоваться внешней программой, читающей данные из внешнего файла и записывающей результат в другой внешний файл. Выбор используемого метода должен задаваться ключом командной строки. На экран программа должна выводить модуль вектора невязки (т.е. модуль вектора AX-B, где X— найденное решение).

Файл с исходными данными называется data.dat и содержит:

- В первой строке после символа  $\sharp$  и пробела размер системы n (одно натуральное число).
- $\bullet$  В последующих n строках матрицу A. В каждой строке файла содержатся элементы одной строки матрицы, отделенные друг от друга одним или несколькими пробелами.
- В последующих n строках вектор B, по одному элементу в строке.

Файл для вывода результата называется result.dat и содержит в первой строке символ  $\sharp$ , пробел и размер системы n, в последующих строках — вектор результата X (по одному элементу в строке). Имейте в виду, что n может быть достаточно большим ( $n \sim 10^{(3\div4)}$ ).

#### Примечания:

При использовании Фортрана рекомендуется использовать встроенные средства языка для работы с матрицами, а также конструкцию **forall** 

или ее же в краткой форме

```
forall (<saroловок>) <oneparop>
```

Заголовок имеет вид «триплет, триплет,..., логическая маска» (где триплет — выражение вида «I=1:15:2»). Важно, что это не обычный цикл, а «параллельное выполнение» — сначала независимо вычисляются все правые части присваиваний, находящихся в блоке, а только потом «параллельно» выполняются присваивания. Ничего, кроме присваиваний, внутри блока быть не должно.

```
Возможно, полезной окажется также конструкция выборки
```

```
where (<логическое условие>) <действие>
```

или

```
where (<логическое условие или логический массив>)
<действия>
elsewhere
<другие действия>
end where
```

При реализации схемы с выбором ведущего элемента рекомендуется воспользоваться функциями **maxloc** и **minloc** (поиск максимального и минимального значения в массиве). Аргументы (array,dim,mask) — массив, конкретная размерность (если нужна), логическая маска (также если нужна). Помните, что функции такого типа дают на выходе координаты относительно начала массива-аргумента (считая, что его индексация по каждой из размерностей начинается с 1).

### Исследование:

Попробуйте решить разными модификациями метода Гаусса систему с матрицей Гильберта, для которой  $a_{ij}=\frac{1}{i+j-1}$ . Вектор свободных членов можно взять случайным. Посмотрите, что будет происходить с решением, если вектор свободных членов незначительно изменится (например, отдельные его элементы изменятся на некоторую малую величину). Что получается и как это объяснить?