Вычислительная математика

Раевский Григорий, группа 3.2

23.03.2024

Отчет по лабораторной работе 2

Содержание

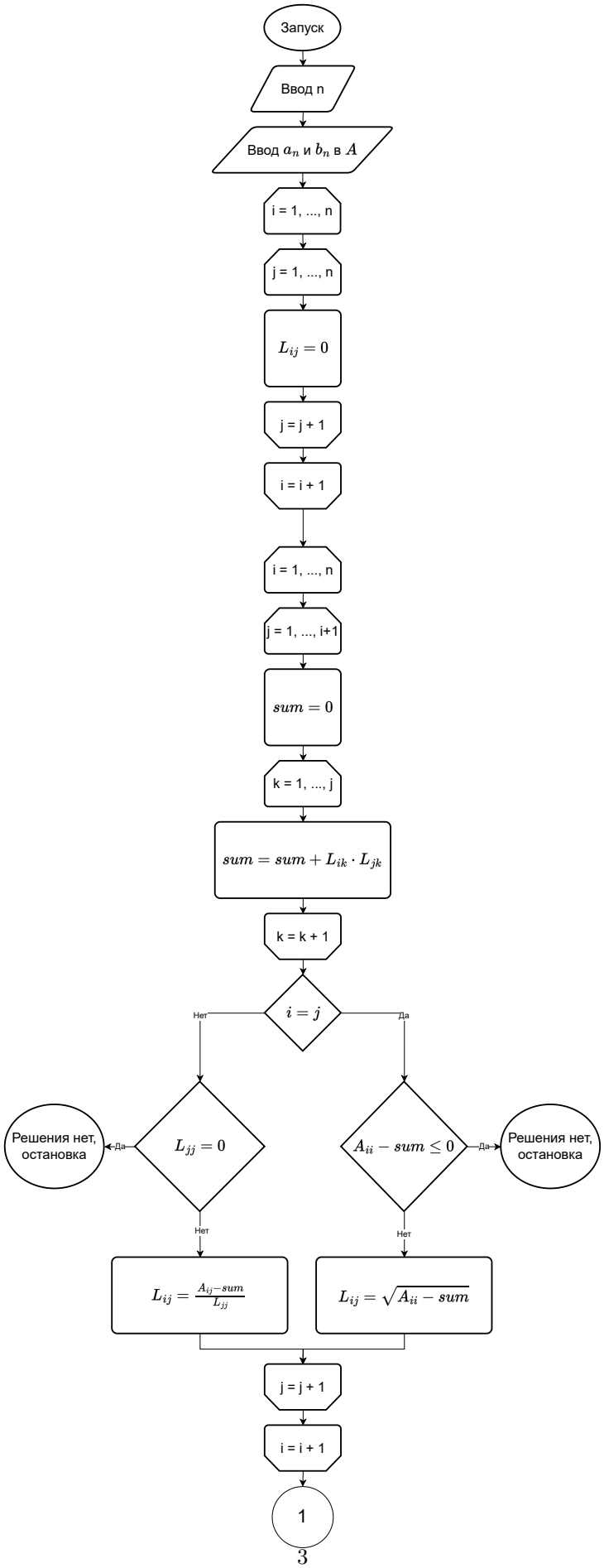
Описание численного метода

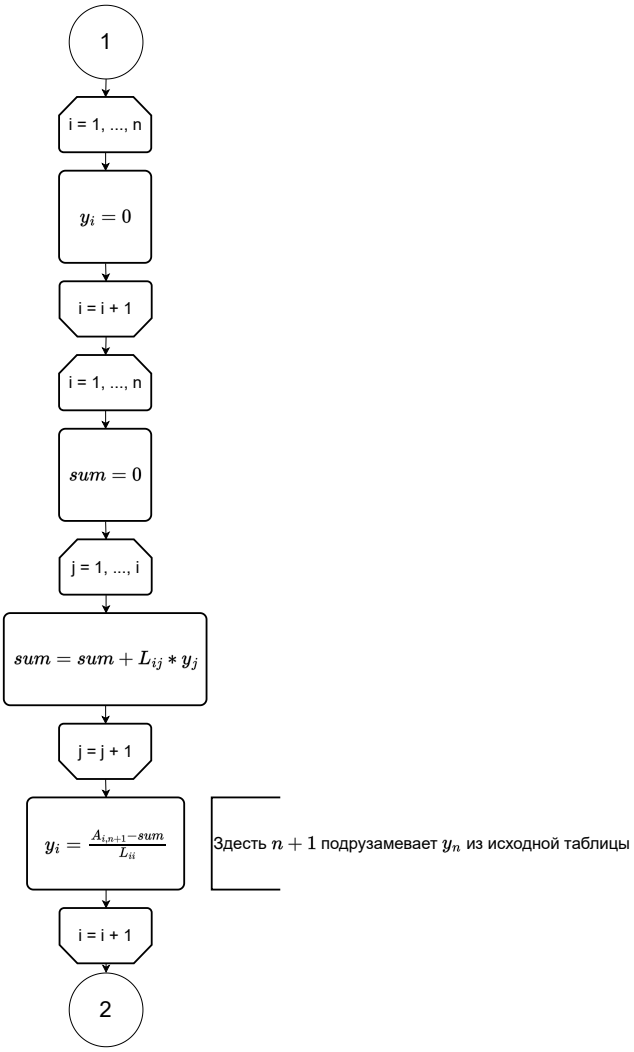
Решение с помощью разложения Холецкого — метод решения системы уравнений с использованием симметричной положительно определенной матрицы. Основан на разложении матрицы A (определяющей коэффициенты в системе СЛАУ) на нижнюю треугольную матрицу L. Таким образом, $A = LL^T$, где L^T — транспонированная треугольная матрица.

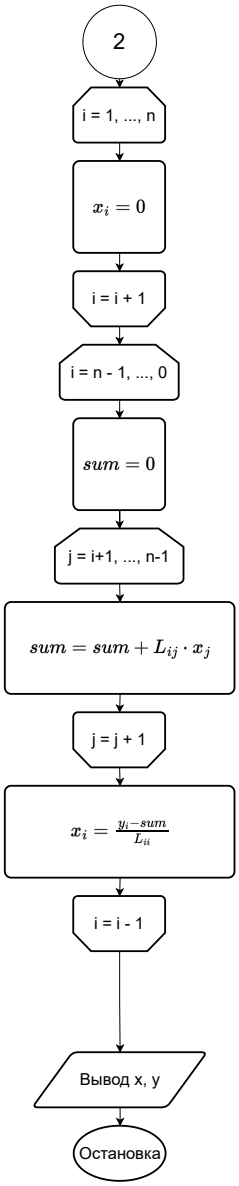
Для каждого элемента l_{ij} суммируется произведения соответствующих пар элементов из предыдущих строк матрицы L, после чего из суммы вычитается a_{ij}. Во время вычисления элементы, стоящие на главной оси матрицы A вычисляются в матрице L, как $l_{ii} = \sqrt{a_{ii} - sum}$, где l_{ii} - i-й элемент диагонали L матрицы, a_{ii} - диагональный элемент матрицы A, а sum - сумма произведения предыдущих пар элементов матрицы L. Так же, если $i = j$, то необходима проверка матрицы на положительную определенность.

Когда матрица L получена, решаются 2 системы: $Ly = b$ и $L^T x = y$ из которых и получаются исходные значения для x и y в уравнении.

Диаграмма


Запуск
Ввод n
Ввод a_n и b_n в A
i = 1, ..., n
j = 1, ..., n
$L_{ij} = 0$
j = j + 1
i = i + 1
i = 1, ..., n
j = 1, ..., i+1
$sum = 0$
k = 1, ..., j
$sum = sum + L_{ik} \cdot L_{jk}$
k = k + 1
$i = j$
Нет
Да
$L_{jj} = 0$
Да
Решения нет, остановка
Нет
$L_{ij} = \frac{A_{ij} - sum}{L_{jj}}$
$A_{ii} - sum \leq 0$
Да
Решения нет, остановка
Нет
$L_{ij} = \sqrt{A_{ii} - sum}$
j = j + 1
i = i + 1
1


1
i = 1, ..., n
$y_i = 0$
i = i + 1
i = 1, ..., n
$sum = 0$
j = 1, ..., i
$sum = sum + L_{ij} * y_j$
j = j + 1
$y_i = \frac{A_{i,n+1} - sum}{L_{ii}}$
Здесь $n + 1$ подрузамевает y_n из исходной таблицы
i = i + 1
2



Листинг кода

```
    def solveByCholeskyDecomposition(n, matrix):
        L = [[0.0] * n for _ in range(n)]

        for i in range(n):
            for j in range(i+1):
                sum = 0
                for k in range(j):
                    sum+= L[i][k]*L[j][k]
                if (i == j):
                    if (matrix[i][i] - sum <= 0):
```

```
                        Solution.isSolutionExists = False
                        return
                    L[i][j] = math.sqrt(matrix[i][i]-sum)
                else:
                    if (L[j][j] == 0):
                        Solution.isSolutionExists = False
                        return
                    L[i][j] = (matrix[i][j]-sum) / L[j][j]
        #print("L: \n")
        #print(L)

        y = [0 for _ in range(n)]
        for i in range (n):
            sum = 0
            for j in range(i):
                sum += L[i][j] * y[j]
            y[i] = (matrix[i][-1]-sum) /L[i][i]

        #print(y)

        x = [0 for _ in range(n)]
        for i in range(n-1,-1,-1):
            sum = 0
            for j in range(i+1,n):
                sum += L[j][i] *x[j]
            x[i] = (y[i]-sum)/L[i][i]
        #print(x)
        result = x+y
        return result
```

Примеры работы программы

Стандартный случай

stdin:

```
3
4 12 -16 1
12 37 -43 2
-16 -43 98 3
```

stdout:

```
28.583333333333332
-7.666666666666666
1.3333333333333333
0.5
-1.0
4.0
```

Матрица положительная и симметрична → метод отработал корректно.

Нулевая строка

stdin:

```
2
1 0 3
0 0 4
```

stdout:

```
The system has no roots of equations or has an infinite set of them.
```

2 строка состоит полностью из нулей (за исключением b_2). Это происходит из-за того, что, матрица вырожденная.

Матрица не положительно определенная

stdin:

```
2
-4 12 2
12 37 1
```

stdout:

```
The system has no roots of equations or has an infinite set of them.
```

Программа обнаружила, что матрица не является положительно определенной, а значит решение методом Холецкого не возможно.

Несимметричная матрица

stdin:

```
2
4 16 2
12 37 1
```

stdout:

```
15.5
-5.0
1.0
-5.0
```

Хоть матрица и не является симметричной, программа отработала корректно и нашла значения x_i и y_i.

Матрица с нулями

stdin:

```
4
4 0 0 0 1
0 1 0 0 2
0 0 3 0 3
0 0 0 4 4
```

stdout:

```
0.25
2.0
1.0000000000000002
1.0
0.5
2.0
1.7320508075688774
2.0
```

Программа посчитала значения x_i и y_i даже в матрице с нулями. Это возможно, так как это не противоречит условиям работы алгоритма.

Выводы

Программа стабильно отрабатывает на данных, если они удовлетворяют условиям возможности решения СЛАУ с использованием метода Холецкого. Однако она не способна выполнить расчет, если входные данные нарушают эти требования (например у матрицы с отрицательными элементами невозможно посчитать код для элемента).

По сравнению с методом Холецкого, метод Гаусса менее требователен к матрице, а так же прост в реализации. Однако он обладает высокой алгоритмической сложностью ($O(n^3)$) и может быть неточным из-за накопления ошибок. Метод Холецкого же в зависимости от реализации может быть более быстрым. Он так же имеет меньше проблем с численной стабильностью, так как имеет меньшее количество вычислений. С другой стороны, он более требователен к матрице, так как работает только с симметричными положительными матрицами, а это ограничивает его применимость.

Моя реализация метода обладает алгоритмической сложностью $O(n^3)$ (самая затратная операция — вычисление матрицы L). Метод Холецкого так же