УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Вычислительная математика

Лабораторная работа №1

Метод Гаусса

Преподаватель: Малышева Татьяна Алексеевна

Выполнил: Цыпандин Николай Петрович

Группа: P3210

Вариант: 24

Санкт-Петербург
2022

Цель работы

Реализовать и протестировать программу для решения системы линейных уравнений методом Гаусса, размерность до 20 включительно неизвестных.

Описание метода

Суть метода заключается в преобразования расширенной СЛАУ к треугольному виду и последующему нахождению всех неизвестных. Если матрица квадратная и она имеет определитель, не равный нулю, то мы имеем единственное решение. Далее мы находим все неизвестные начиная с последней строки. Каждая неизвестная выражается через предыдущие, а последняя известна сразу.

$$\left(\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & b_1 \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} & b_n \end{array}\right) \sim \left(\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & b_1 \\ 0 & a_{2,2} - \frac{a_{2,1} * a_{1,2}}{a_{1,1}} & \cdots & a_{2,n} - \frac{a_{2,1} * a_{1,n}}{a_{1,1}} & b_2 - \frac{b_1 * a_{2,1}}{a_{1,1}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n,2} - \frac{a_{n,1} * a_{1,2}}{a_{1,1}} & \cdots & a_{n,n} - \frac{a_{n,1} * a_{1,n}}{a_{1,1}} & b_n - \frac{b_1 * a_{n,1}}{a_{1,1}} \end{array}\right) \sim$$

$$\sim \left(\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & b_1 \\ 0 & a_{2,2} & \cdots & a_{2,n} & b_2' \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & a_{n,n} & b_n \end{array}\right)$$

Прямой ход - к треугольному виду

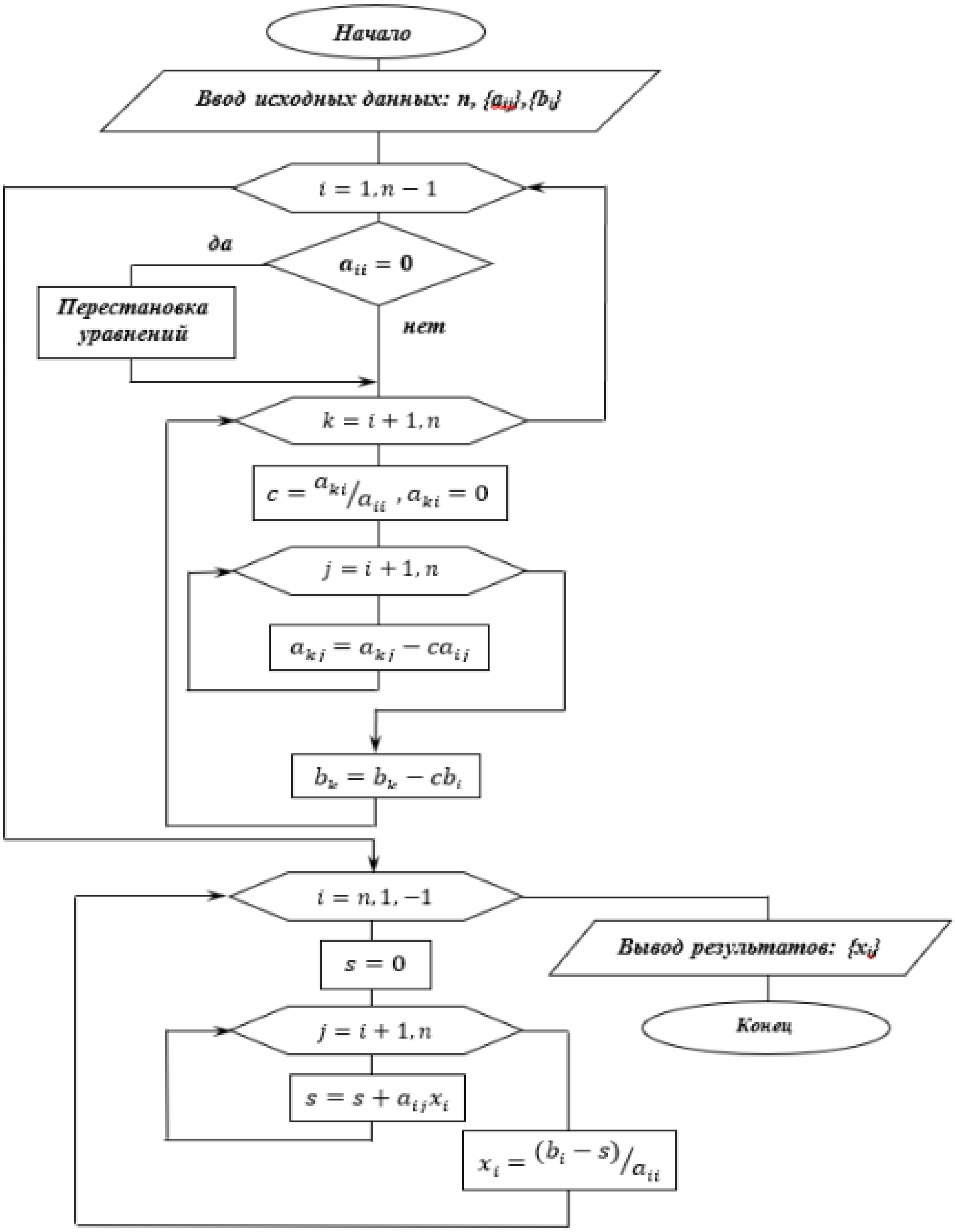
$$x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{i,j} * x_j}{a_{i,i}}$$

Обратный ход - вычисление вектора неизвестных

$$detA = (-1)^k \prod_{i=1}^{n} a_{ii}$$

Определитель - произведение элементов главной диагонали, с учётом перестановок

Блок-схема метода Гаусса:



Листинг программы (код численного метода)

```
# Решение системы
def solve_gauss(slae):
    to_triangular(slae)
    return reverse_course(slae)


# Прямой ход
def to_triangular(slae):
    n, m = slae.coefficients.row, slae.coefficients.col
```

```python
    for i in range(n - 1):
        slae.swap_equations(i, find_not_zero(slae.coefficients, i, i))
        a_i_i = slae.coefficients.get(i, i)
        for j in range(i + 1, m):
            c_i_i = -slae.coefficients.get(j, i) / a_i_i
            slae.add_eq_to_eq(j, i, c_i_i)
    return


# Обратный ход
def reverse_course(slae):
    n, m = slae.coefficients.row, slae.coefficients.col
    if is_zero(det_triangular(slae.coefficients)):
        return None

    slae_roots = Matrix(m, 1)
    for i in range(n - 1, -1, -1):
        linear_comb = 0
        for j in range(i + 1, m):
            linear_comb += slae.coefficients.get(i, j) * slae_roots.get(j, 0)
        b_i = slae.free_members.get(i, 0)
        a_i_i = slae.coefficients.get(i, i)
        slae_roots.set_row(i, [(b_i - linear_comb) / a_i_i])
    return slae_roots


# Вычисление определителя
def det_triangular(matr):
    if matr.row != matr.col:
        return None
    res = 1
    for i in range(matr.col):
        res *= matr.get(i, i)
    return res
```

Полный код можно найти по ссылке:

https://github.com/kkkooolllyyyaaa/computational_math/tree/master/comp-math-1

Пример 1

[file] - получить матрицу с файла

[cl] - получить матрицу с клавиатуры

>>> file

Введенная система линейных алгебраических уравнений:

13.0 x1 + 15.0 x2 + 13.0 x3 + 18.0 x4 + 13.0 x5 = 16.0

12.0 x1 + 12.0 x2 + 16.0 x3 + 17.0 x4 + 13.0 x5 = 15.0

14.0 x1 + 18.0 x2 + 16.0 x3 + 19.0 x4 + 17.0 x5 = 14.0

18.0 x1 + 16.0 x2 + 18.0 x3 + 20.0 x4 + 18.0 x5 = 16.0

19.0 x1 + 15.0 x2 + 18.0 x3 + 13.0 x4 + 15.0 x5 = 18.0

СЛАУ, приведенная к ступенчатому (треугольному) виду:

13.0 x1 + 15.0 x2 + 13.0 x3 + 18.0 x4 + 13.0 x5 = 16.0

0.0 x1 + -1.85 x2 + 4.0 x3 + 0.38 x4 + 1.0 x5 = 0.23

0.0 x1 + 0.0 x2 + 6.0 x3 + 0.0 x4 + 4.0 x5 = -3.0

0.0 x1 + 0.0 x2 + 0.0 x3 + -5.92 x4 + 4.31 x5 = -11.92

0.0 x1 + 0.0 x2 + 0.0 x3 + 0.0 x4 + -7.82 x5 = 15.46

Определитель матрицы:

Элементарные преобразования (диагональ): -6660.000000000001

Разложение по первой строке: -6660.0

Через встроенную функцию numpy: -6659.999999999996

Корни уравнения:

x1 = 0.78978978978979 x2 = 0.696696696696696 x3 = 0.818318318318318 x4 = 0.575075075075075 x5 = -1.977477477477477

Вектор невязок:

r1 = 0.0 r2 = 0.0 r3 = 0.0 r4 = 7e-15 r5 = -4e-15

Пример 2

[file] - получить матрицу с файла

[cl] - получить матрицу с клавиатуры

>>> cl

5

13 15 13 18 13 16

12 12 16 17 13 15

14 18 16 19 17 14

18 16 18 20 18 16

19 15 18 13 15 18

Введенная система линейных алгебраических уравнений:

13.0 x1 + 15.0 x2 + 13.0 x3 + 18.0 x4 + 13.0 x5 = 16.0

$12.0\ x1 + 12.0\ x2 + 16.0\ x3 + 17.0\ x4 + 13.0\ x5 = 15.0$

$14.0\ x1 + 18.0\ x2 + 16.0\ x3 + 19.0\ x4 + 17.0\ x5 = 14.0$

$18.0\ x1 + 16.0\ x2 + 18.0\ x3 + 20.0\ x4 + 18.0\ x5 = 16.0$

$19.0\ x1 + 15.0\ x2 + 18.0\ x3 + 13.0\ x4 + 15.0\ x5 = 18.0$

СЛАУ, приведенная к ступенчатому (треугольному) виду:

$13.0\ x1 + 15.0\ x2 + 13.0\ x3 + 18.0\ x4 + 13.0\ x5 = 16.0$

$0.0\ x1 + -1.85\ x2 + 4.0\ x3 + 0.38\ x4 + 1.0\ x5 = 0.23$

$0.0\ x1 + 0.0\ x2 + 6.0\ x3 + 0.0\ x4 + 4.0\ x5 = -3.0$

$0.0\ x1 + 0.0\ x2 + 0.0\ x3 + -5.92\ x4 + 4.31\ x5 = -11.92$

$0.0\ x1 + 0.0\ x2 + 0.0\ x3 + 0.0\ x4 + -7.82\ x5 = 15.46$

Определитель матрицы:

Элементарные преобразования (диагональ): -6660.000000000001

Разложение по первой строке: -6660.0

Через встроенную функцию numpy: -6659.999999999996

Корни уравнения:

$x1 = 0.789789789789 79\ x2 = 0.696696696696696\ x3 = 0.818318318318318\ x4 = 0.575075075075075\ x5 = -1.977477477477477$

Вектор невязок:

$r1 = 0.0\ r2 = 0.0\ r3 = 0.0\ r4 = 7e\text{-}15\ r5 = -4e\text{-}15$

Вывод

Метод Гаусса подходит для простого вычисления СЛАУ, так как он более универсален и прост в реализации, а также работает за конечное число арифметических операций. Но и у него есть недостатки так как весь массив приходится хранить в оперативной памяти компьютера. Происходит накопление погрешности в процессе решения, поскольку на любом этапе используют результаты предыдущих операций. Можно улучшить этот метод выбирая главный элемент.