# Практическая задача, пункт b

Натальченко Александр

11 ноября 2020 г.

## Содержание



## 1 Постановка задачи

12. ПРАКТИЧЕСКАЯ ЗАДАЧА 3 курс 5 семестр

**Решить систему**

$$\begin{cases} c_1x_1 + d_1x_2 + e_1x_3 & = & f_1 \\ b_2x_1 + c_2x_2 + d_2x_3 + e_2x_4 & = & f_2 \\ a_3x_1 + b_3x_2 + c_3x_3 + d_3x_4 + e_3x_5 & = & f_3 \\ a_4x_2 + b_4x_3 + c_4x_4 + d_4x_5 + e_4x_6 & = & f_4 \\ \cdots & \cdots & \cdots \\ a_mx_{m-2} + b_mx_{m-1} + c_mx_m + d_mx_{m+1} + e_mx_{m+2} & = & f_m \\ \cdots & \cdots & \cdots \\ a_{n-1}x_{n-3} + b_{n-1}x_{n-2} + c_{n-1}x_{n-1} + d_{n-1}x_n & = & f_{n-1} \\ a_nx_{n-2} + b_nx_{n-1} + c_nx_n & = & f_n \end{cases}, \quad n = 20,\ c_i = 10,\ f_i = i,\ i = 1,\ldots,n;$$

$b_{i+1} = d_i = 1$, $i = 1,\ldots,n-1$; $a_{i+2} = e_i = 0$, $i = 1,\ldots,n-2$.

**a.** **прямым методом (методом Гаусса):**
вывести на печать решение и невязки;

**b.** **итерационным методом (методом Зейделя) с точностью** $\varepsilon = 10^{-4}$:
вывести на печать решение;

**c.** **определить число обусловленности матрицы системы** $\mu = \|A\| \cdot \|A^{-1}\|$

(УКАЗАНИЕ: найти $\lambda_{\max}$ степенным методом, затем $\lambda_{\min}$, используя сдвиг спектра матрицы, для определения числа обусловленности воспользоваться евклидовой нормой).

# 2 Краткое описание алгоритма

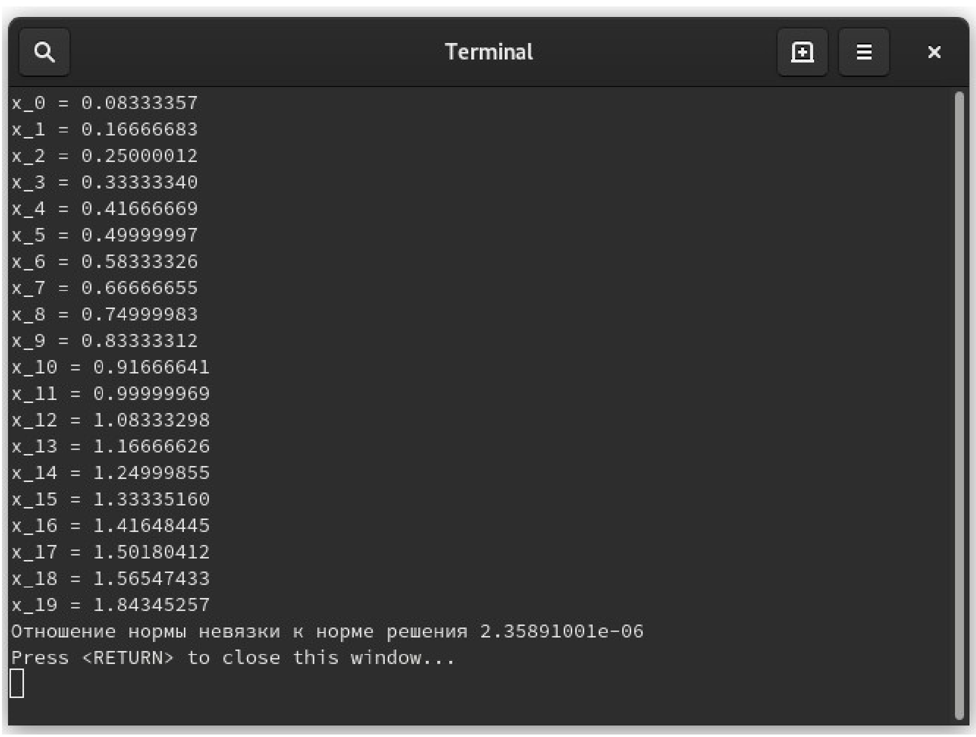
Будем реализовывать итерационный процесс в следующем виде:

$$\begin{cases} x_1{}^{(k+1)} & = & c_{12}x_2^{(k)} & + & c_{13}x_3^{(k)} & + & \dots & + & c_{1n}x_n{}^{(k)} & + & d_1 \\ x_2{}^{(k+1)} & = & c_{21}x_1^{(k+1)} & + & c_{23}x_3^{(k)} & + & \dots & + & c_{2n}x_n{}^{(k)} & + & d_2 \\ \dots \\ x_n{}^{(k+1)} & = & c_{n1}x_1^{(k+1)} & + & c_{n2}x_2^{(k+1)} & + & \dots & + & c_{n(n-1)}x_{n-1}{}^{(k+1)} & + & d_n \end{cases},$$

Задаём значения элементов матрицы (с. 35-43). Для реализации итерационного процесса нам понадобиться хранить предыдущий итерационный вектор (вектор h, с. 45). Мы реализуем итерационный процесс в цикле (с. 48-63), и продолжаем его до тех пор, пока норма разности двух последовательных векторов не достигнет заданной точности $\varepsilon = 10^{-4}$.

Функция error(vector<vector<double> > A, vector<double> X) по заданной расширенной матрице системы $A$ и вектору решения $X$ вычисляет отношение нормы невязки к норме решения.

# 3 Результаты




Мы получили решение и невязки, и вычислили отношение нормы невязки к норме решения:

$$\frac{\|r\|}{\|x^*\|} = 2.35891 \cdot 10^{-6}$$

Если сравнить это решение с полученным прямым методом Гаусса, то видно, что уже седьмые знаки после запятой могут быть различны. Отношение нормы невязки к норме решения стало на 9 порядков больше по сравнению с методом Гаусса.

# 4 Код программы (C++)

```cpp
#include <iostream>
#include <vector>
#include <cmath>
#include <iomanip>

using namespace std;

const double c_i = 10; // Значения элементов матрицы на главной диагонали
const double d_i = 1; // Значения элементов матрицы на соседних диагоналях
const size_t N = 20; // Размер матрицы
const double epsilon = 1e-4;

//Считает отношение нормы невязки к норме решения
double error(vector<vector<double> > A, vector<double> X) {
    vector<double> Err(N);
    for (size_t i = 0; i < N; i++) {
        for (size_t j = 0; j < N; j++) {
            Err[i] += A[i][j]*X[j];
        }
        Err[i] -= A[i][N];
    }
    double NX = 0;
    double err = 0;
    for (size_t i = 0; i < N; i++) {
        err += Err[i]*Err[i];
        NX += X[i]*X[i];
    }
    return sqrt(err/NX);
}

int main() {
    vector<vector<double> > A(N);

    // Задаём значения элементов матрицы
    for (size_t i = 0; i < N; i++) {
        A[i].resize(N + 1);
        A[i][i] = c_i;
        if (i < N - 1)
            A[i][i + 1] = d_i;
        if (i > 0)
            A[i][i - 1] = d_i;
        A[i][N] = i + 1;
    }

    vector<double> X(N, 0), h(N, 1);

    double d;
    do {
        d = 0;
        for (size_t i = 0; i < N; i++) {
            X[i] = A[i][N]/A[i][i];
            for (size_t j = 0; j < i; j++) {
                X[i] -= X[j]*A[i][j]/A[i][i];
            }
```

```
            for (size_t j = i + 1; j < N; j++) {
                X[i] -= h[j]*A[i][j]/A[i][i];
            }
        }
        for (size_t i = 0; i < N; i++) {
            d += (X[i] - h[i])*(X[i] - h[i]);
        }
        h = X;
    } while (sqrt(d) > epsilon);

    for (size_t j = 0; j < N; j++) {
        cout << fixed << setprecision(8)
             << "x_" << j << " = " << X[j] << endl;
    }
    cout << scientific << "Отношение нормы невязки к норме решения " << error(A, X) << endl;

    return 0;
}
```