Практическая задача, пункт c

Натальченко Александр

1 ноября 2020 г.

Содержание

1 Постановка задачи

12. ПРАКТИЧЕСКАЯ ЗАДАЧА 3 курс 5 семестр

Решить систему

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

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

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

В этой задаче нам понадобится вычислять наибольшее по модулю собственное значение матрицы. В программе это функция get_lambda_abs_max(vector<vector<double> > A), которая принимает на вход матрицу A, для которой нужно вычислить собственное значение, и возвращает его величину.

Мы строим последовательности

$$u^{(k+1)} = Au^{(k)},\ u^{(0)} = (1, 1, \ldots, 1)^T; \quad \lambda^{(k)} = \frac{(u^{(k+1)}, u^{(k)})}{(u^{(k)}, u^{(k)})}$$

и останавливаемся, когда $\lambda^{(i)}$ и $\lambda^{(i+1)}$ отличаются меньше чем на заданный ε. В памяти хранится по два элемента последовательности (с. 19-25, с. 27-34).

Далее мы задаём матрицу A из условия (с. 43-52) и находим его наибольшее по модулю собственное значение $\lambda_{max} = \lambda_1$ (с. 55). Затем, сделав сдвиг, находим так же собственное значение λ матрицы $A - \lambda_{max}E$ (с. 57-61), тогда наименьшее по модулю собственное значение матрицы A находим как $\lambda_{min} = \lambda_2 = \lambda + \lambda_1$. Тогда, пользуясь тем, что в данной задаче $A = A^T$

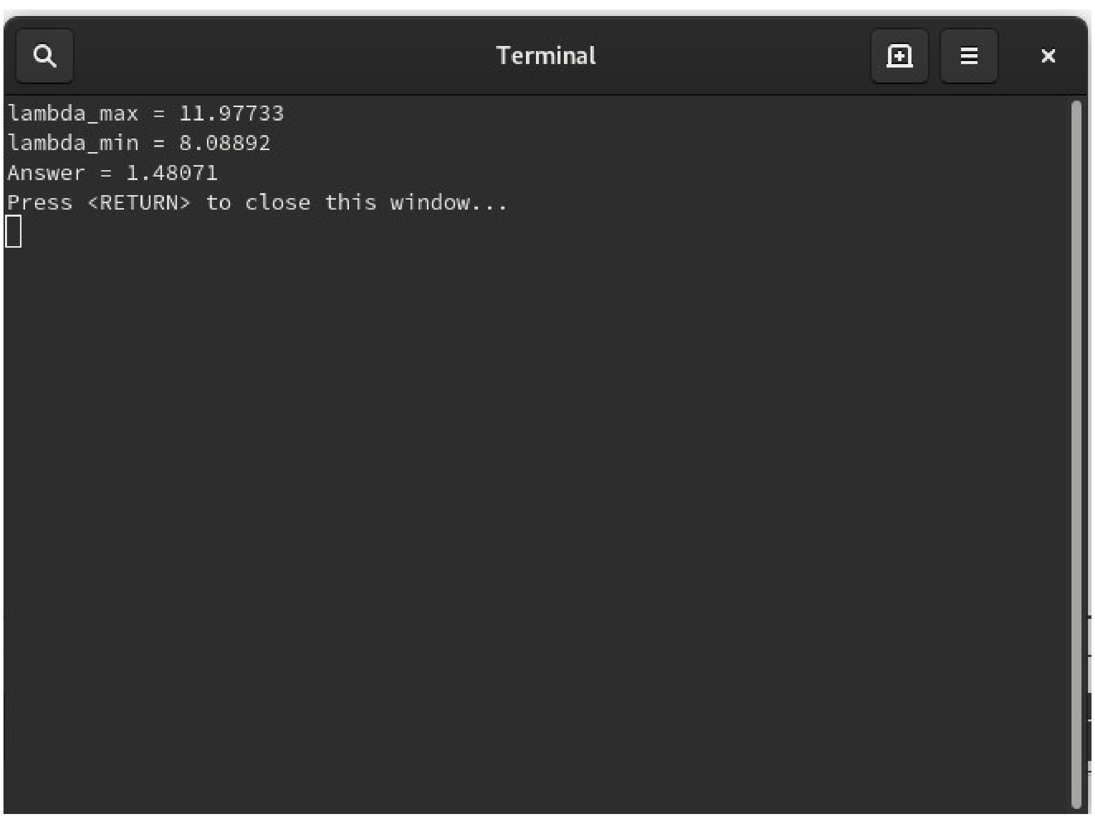
$$\|A\| = \max_i \sqrt{\lambda_i(A^T A)} = \max_i \sqrt{\lambda_i(A^2)} = \max_i \sqrt{\lambda_i^2(A)} = \max_i |\lambda_i(A)|,$$

$$\|A^{-1}\| = \max_i \sqrt{\lambda_i(A^{-2})} = \min_i \sqrt{\frac{1}{\lambda_i(A^2)}} = \min_i \sqrt{\frac{1}{\lambda_i^2(A)}} = \min_i \frac{1}{|\lambda_i(A)|}.$$

Тогда число обусловловленности

$$\mu(A) = \|A\|\|A^{-1}\| = \max_i |\lambda_i(A)| \min_j |\frac{1}{\lambda_j(A)}| = \frac{|\lambda_1|}{|\lambda_2|}.$$

3 Результаты



Для матрицы A наибольшее и наименьшее по модулю собственные значения получились соответственно $\lambda_{max} = 11.97733$ и $\lambda_{min} = 8.08892$, откуда число обусловленности:

$$\mu = 1.48071,$$

что означает, что матрица очень хорошо обусловлена.

4 Код программы (C++)

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

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-5;

using namespace std;

double get_lambda_abs_max(vector<vector<double> > A) {
    vector<double> X_1(N, 0), X_2(N, 1);
    double lambda1 = 0, lambda2 = 1;

    do {
        X_1 = X_2;
        for (size_t i = 0; i < N; i++) {
            X_2[i] = 0;
            for (size_t j = 0; j < N; j++) {
                X_2[i] += A[i][j]*X_1[j];
            }
        }

        double X_1_X_2 = 0, X_1_X_1 = 0;
        for (size_t i = 0; i < N; i++) {
            X_1_X_2 += X_1[i]*X_2[i];
            X_1_X_1 += X_1[i]*X_1[i];
        }

        lambda1 = lambda2;
        lambda2 = X_1_X_2/X_1_X_1;

    } while (fabs(lambda1 - lambda2) > epsilon);
    return lambda2;
}

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;
    }

    // Наибольшее по модулю СЗ матрицы A
    double lambda_max = get_lambda_abs_max(A);

    for (size_t i = 0; i < N; i++)
        A[i][i] -= lambda_max;

    // Наименьшее по модулю СЗ матрицы A
    double lambda_min = get_lambda_abs_max(A) + lambda_max;

    cout << fixed << setprecision(5)
         << "lambda_max = " << lambda_max << endl
         << "lambda_min = " << lambda_min << endl
         << "Answer = " << fabs(lambda_max/lambda_min) << endl;

    return 0;
}
```