Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ *Робототехника и комплексная автоматизация*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

по дисциплине: «Модели и методы анализа проектных решений»

Студент	Мудриченко Михаил Николаевич
Группа	РК6-72Б
Тема лабораторной работы	Метод конечных элементов
Вариант	35

Студент _____ **Мудриченко М.Н.**
подпись, дата *фамилия, и.о.*

Преподаватель _____ **Трудоношин В.А.**
подпись, дата *фамилия, и.о.*

Оценка _____

Москва, 2022 г.

Оглавление

Задание

Методом конечных элементов решить уравнение:

$$5\frac{d^2u}{dx^2} + 2\frac{du}{dx} - 5 = 0 \tag{1}$$

при следующих граничных условиях:

$$u(0) = 10 \quad u(7) = 1$$

Количество конечных элементов для первого расчета — 20, количество конечных элементов для второго расчета — 40.

Сравнить результат с аналитическим решением, оценить максимальную погрешность.

Аналитическое решение

Решение уравнения (1) будем искать в виде

$$u = u_0 + \tilde{u}, \tag{2}$$

где u_0 — общее решение, $\tilde{u}$ — частное решение.

Общее решение имеет вид:

$$u_0 = C_1 e^{k_1 x} + C_2 e^{k_2 x},$$

где k_1, k_2 — корни следующего характеристического уравнения:

$$5k^2 + 2k = 0.$$

Откуда $k_1 = 0$, $k_2 = -\frac{2}{5}$, следовательно

$$u_0 = C_1 + C_2 e^{-\frac{2}{5}x}. \tag{3}$$

Частное решение $\tilde{u}$ будем искать в виде

$$\tilde{u} = Ax.$$

Подставим $\tilde{u}$ в (1):

$$5\frac{d^2(Ax)}{dx^2} + 2\frac{d(Ax)}{dx} - 5 = 0,$$

$$2A - 5 = 0 \implies A = \frac{5}{2},$$

$$\tilde{u} = \frac{5}{2}x. \tag{4}$$

Подставим (3) и (4) в (2):

$$u = C_1 + C_2 e^{-\frac{2}{5}x} + \frac{5}{2}x.$$

Для получения C_1, C_2 воспользуемся граничными условиями:

$$u(0) = 10 \implies C_1 + C_2 = 10$$

$$u(7) = 1 \implies C_1 + C_2 e^{-\frac{14}{5}} + \frac{35}{2} = 1$$

Откуда $C_1 = \frac{20 + 33e^{\frac{14}{5}}}{2 - 2e^{\frac{14}{5}}}$, $C_2 = \frac{53e^{\frac{14}{5}}}{2e^{\frac{14}{5}} - 2}$.

Итоговое решение:

$$u = \frac{20 + 33e^{\frac{14}{5}}}{2 - 2e^{\frac{14}{5}}} + \frac{53e^{\frac{14}{5}}}{2e^{\frac{14}{5}} - 2} \cdot e^{-\frac{2}{5}x} + \frac{5}{2}x$$

Локальная матрица жесткости и вектор нагрузок для линейной функции формы

Пусть длина конечного элемента (КЭ) равна L, u внутри КЭ изменяется линейно, а значения в узлах равны $u_i(x = 0)$ и $u_j(x = L)$, тогда с помощью линейной интерполяции получим:

$$u = \left(1 - \frac{x}{L}\right) u_i + \frac{x}{L} u_j = \boldsymbol{N_e u},$$

где $\boldsymbol{N_e}$ — вектор функции формы.

Получим локальную матрицу жесткости с помощью метода Галеркина:

$$\int_0^L \boldsymbol{N_e}^T \left(5\frac{d^2u}{dx^2} + 2\frac{du}{dx} - 5\right) dx = \int_0^L \begin{bmatrix} 1 - \frac{x}{L} \\ \frac{x}{L} \end{bmatrix} \left(5\frac{d^2u}{dx^2} + 2\frac{du}{dx} - 5\right) dx$$

$$= 5 \begin{bmatrix} -\frac{du}{dx}\big|_i \\ \frac{du}{dx}\big|_j \end{bmatrix} - 5 \begin{bmatrix} \frac{1}{L} & -\frac{1}{L} \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \end{bmatrix} + 2 \begin{bmatrix} -\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \end{bmatrix} - 5 \begin{bmatrix} \frac{L}{2} \\ \frac{L}{2} \end{bmatrix} = 0$$

$$\begin{bmatrix} -1 - \frac{5}{L} & 1 + \frac{5}{L} \\ -1 + \frac{5}{L} & 1 - \frac{5}{L} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \end{bmatrix} = \begin{bmatrix} \frac{5L}{2} + 5\frac{du}{dx}\big|_i \\ \frac{5L}{2} - 5\frac{du}{dx}\big|_j \end{bmatrix}$$

Локальная матрица жесткости и вектор нагрузок для кубической функции формы

Для получения функции формы воспользуемся интерполяцией Лагранжа, проходящей через узлы $u_i(x=0)$, $u_j(x=\frac{L}{3})$, $u_k(x=\frac{2L}{3})$, $u_l(x=L)$. В результате интерполяции получим следующую функцию формы:

$$u=\left(-\frac{9x^3}{2L^3}+\frac{9x^2}{L^2}-\frac{11x}{2L}+1\right)u_i+\left(\frac{27x^3}{2L^3}-\frac{45x^2}{2L^2}+\frac{9x}{L}\right)u_j+$$

$$+\left(-\frac{27x^3}{2L^3}+\frac{18x^2}{L^2}-\frac{9x}{2L}\right)u_k+\left(\frac{9x^3}{2L^3}-\frac{9x^2}{2L^2}+\frac{x}{L}\right)u_l=\boldsymbol{N_e u}$$

Получим локальную матрицу жесткости с помощью метода Галеркина:

$$\int_0^L \boldsymbol{N_e}^T\left(5\frac{d^2u}{dx^2}+2\frac{du}{dx}-5\right)dx=5\int_0^L \boldsymbol{N_e}^T\frac{d^2u}{dx^2}dx+2\int_0^L \boldsymbol{N_e}^T\frac{du}{dx}dx-5\int_0^L \boldsymbol{N_e}^T dx$$

$$\int_0^L \boldsymbol{N_e}^T\frac{d^2u}{dx^2}dx=\int_0^L\begin{bmatrix}-\frac{9x^3}{2L^3}+\frac{9x^2}{L^2}-\frac{11x}{2L}+1\\ \frac{27x^3}{2L^3}-\frac{45x^2}{2L^2}+\frac{9x}{L}\\ -\frac{27x^3}{2L^3}+\frac{18x^2}{L^2}-\frac{9x}{2L}\\ \frac{9x^3}{2L^3}-\frac{9x^2}{2L^2}+\frac{x}{L}\end{bmatrix}\frac{d^2u}{dx^2}dx=$$

$$=\begin{bmatrix}-\frac{9x^3}{2L^3}+\frac{9x^2}{L^2}-\frac{11x}{2L}+1\\ \frac{27x^3}{2L^3}-\frac{45x^2}{2L^2}+\frac{9x}{L}\\ -\frac{27x^3}{2L^3}+\frac{18x^2}{L^2}-\frac{9x}{2L}\\ \frac{9x^3}{2L^3}-\frac{9x^2}{2L^2}+\frac{x}{L}\end{bmatrix}\frac{du}{dx}\Bigg|_0^L-\int_0^L\frac{d}{dx}\begin{bmatrix}-\frac{9x^3}{2L^3}+\frac{9x^2}{L^2}-\frac{11x}{2L}+1\\ \frac{27x^3}{2L^3}-\frac{45x^2}{2L^2}+\frac{9x}{L}\\ -\frac{27x^3}{2L^3}+\frac{18x^2}{L^2}-\frac{9x}{2L}\\ \frac{9x^3}{2L^3}-\frac{9x^2}{2L^2}+\frac{x}{L}\end{bmatrix}\frac{du}{dx}dx$$

$$=\begin{bmatrix}-\frac{du}{dx}|_i\\ 0\\ 0\\ \frac{du}{dx}|_l\end{bmatrix}-\begin{bmatrix}\frac{37}{10L} & -\frac{189}{40L} & \frac{27}{20L} & -\frac{13}{40L}\\ -\frac{189}{40L} & \frac{54}{5L} & -\frac{297}{40L} & \frac{27}{20L}\\ \frac{27}{20L} & -\frac{297}{40L} & \frac{54}{5L} & -\frac{189}{40L}\\ -\frac{13}{40L} & \frac{27}{20L} & -\frac{189}{40L} & \frac{37}{10L}\end{bmatrix}\begin{bmatrix}u_i\\ u_j\\ u_k\\ u_l\end{bmatrix}$$

$$\int_0^L \boldsymbol{N_e}^T \frac{du}{dx} dx = \int_0^L \begin{bmatrix} -\frac{9x^3}{2L^3} + \frac{9x^2}{L^2} - \frac{11x}{2L} + 1 \\ \frac{27x^3}{2L^3} - \frac{45x^2}{2L^2} + \frac{9x}{L} \\ -\frac{27x^3}{2L^3} + \frac{18x^2}{L^2} - \frac{9x}{2L} \\ \frac{9x^3}{2L^3} - \frac{9x^2}{2L^2} + \frac{x}{L} \end{bmatrix} \frac{du}{dx} dx =$$

$$= \begin{bmatrix} -\frac{1}{2} & \frac{57}{80} & -\frac{3}{10} & \frac{7}{80} \\ -\frac{57}{80} & 0 & \frac{81}{80} & -\frac{3}{10} \\ \frac{3}{10} & -\frac{81}{80} & 0 & \frac{57}{80} \\ -\frac{7}{80} & \frac{3}{10} & -\frac{57}{80} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix}$$

$$\int_0^L \boldsymbol{N_e}^T = \int_0^L \begin{bmatrix} -\frac{9x^3}{2L^3} + \frac{9x^2}{L^2} - \frac{11x}{2L} + 1 \\ \frac{27x^3}{2L^3} - \frac{45x^2}{2L^2} + \frac{9x}{L} \\ -\frac{27x^3}{2L^3} + \frac{18x^2}{L^2} - \frac{9x}{2L} \\ \frac{9x^3}{2L^3} - \frac{9x^2}{2L^2} + \frac{x}{L} \end{bmatrix} dx = \begin{bmatrix} \frac{L}{8} \\ \frac{3L}{8} \\ \frac{3L}{8} \\ \frac{L}{8} \end{bmatrix}$$

Итоговое значение интеграла:

$$\int_0^L \boldsymbol{N_e}^T \left(5\frac{d^2u}{dx^2} + 2\frac{du}{dx} - 5 \right) dx = 5 \begin{bmatrix} -\frac{du}{dx}|_i \\ 0 \\ 0 \\ \frac{du}{dx}|_l \end{bmatrix} - 5 \begin{bmatrix} \frac{37}{10L} & -\frac{189}{40L} & \frac{27}{20L} & -\frac{13}{40L} \\ -\frac{189}{40L} & \frac{54}{5L} & -\frac{297}{40L} & \frac{27}{20L} \\ \frac{27}{20L} & -\frac{297}{40L} & \frac{54}{5L} & -\frac{189}{40L} \\ -\frac{13}{40L} & \frac{27}{20L} & -\frac{189}{40L} & \frac{37}{10L} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix}$$

$$+2 \begin{bmatrix} -\frac{1}{2} & \frac{57}{80} & -\frac{3}{10} & \frac{7}{80} \\ -\frac{57}{80} & 0 & \frac{81}{80} & -\frac{3}{10} \\ \frac{3}{10} & -\frac{81}{80} & 0 & \frac{57}{80} \\ -\frac{7}{80} & \frac{3}{10} & -\frac{57}{80} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} - 5 \begin{bmatrix} \frac{L}{8} \\ \frac{3L}{8} \\ \frac{3L}{8} \\ \frac{L}{8} \end{bmatrix} = 0$$

$$\begin{bmatrix} -\frac{37}{2L}-1 & \frac{189}{8L}+\frac{57}{40} & -\frac{27}{4L}-\frac{3}{5} & \frac{13}{8L}+\frac{7}{40} \\ \frac{189}{8L}-\frac{57}{40} & -\frac{54}{L} & \frac{297}{8L}+\frac{81}{40} & -\frac{27}{4L}-\frac{3}{5} \\ -\frac{27}{4L}+\frac{3}{5} & \frac{297}{8L}-\frac{81}{40} & -\frac{54}{L} & \frac{189}{8L}+\frac{57}{40} \\ \frac{13}{8L}-\frac{7}{40} & -\frac{27}{4L}+\frac{3}{5} & \frac{189}{8L}-\frac{57}{40} & -\frac{37}{2L}+1 \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} = \begin{bmatrix} \frac{5L}{8}+5\frac{du}{dx}|_i \\ \frac{15L}{8} \\ \frac{15L}{8} \\ \frac{5L}{8}-5\frac{du}{dx}|_l \end{bmatrix}$$

Так как нас интересуют только крайние узлы, преобразуем систему к виду

$$\begin{bmatrix} \tilde{a}_{11} & 0 & 0 & \tilde{a}_{14} \\ \tilde{a}_{21} & \tilde{a}_{22} & 0 & \tilde{a}_{24} \\ \tilde{a}_{31} & 0 & \tilde{a}_{33} & \tilde{a}_{34} \\ \tilde{a}_{41} & 0 & 0 & \tilde{a}_{44} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} = \begin{bmatrix} \tilde{b}_1+5\frac{du}{dx}|_i \\ \tilde{b}_2 \\ \tilde{b}_3 \\ \tilde{b}_4-5\frac{du}{dx}|_l \end{bmatrix}$$

После чего можно перейти к системе из двух уравнений:

$$\begin{bmatrix} \tilde{a}_{11} & \tilde{a}_{14} \\ \tilde{a}_{41} & \tilde{a}_{44} \end{bmatrix} \begin{bmatrix} u_i \\ u_l \end{bmatrix} = \begin{bmatrix} \tilde{b}_1+5\frac{du}{dx}|_i \\ \tilde{b}_4-5\frac{du}{dx}|_l \end{bmatrix}$$

Ансамблирование

Пусть локальные матрицы жесткости и вектора нагрузок имеют вид

$$\begin{bmatrix} a_{11} & a_{11} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} u_i \\ u_l \end{bmatrix} = \begin{bmatrix} b_1+5\frac{du}{dx}|_i \\ b_2-5\frac{du}{dx}|_l \end{bmatrix}$$

Тогда после ансамблирования будет получена следующая СЛАУ:

$$\begin{bmatrix} a_{11} & a_{12} & 0 & \dots & 0 \\ a_{21} & a_{22}+a_{11} & a_{12} & \dots & 0 \\ 0 & a_{21} & a_{22}+a_{11} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & a_{12} \\ 0 & 0 & 0 & a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} = \begin{bmatrix} b_1+5\frac{du}{dx}|_1 \\ b_2+b_1 \\ \vdots \\ b_2+b_1 \\ b_2-5\frac{du}{dx}|_n \end{bmatrix}$$

По условию u_1 и u_n известны, что позволяет переписать СЛАУ:

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & a_{22}+a_{11} & a_{12} & \dots & 0 \\ 0 & a_{21} & a_{22}+a_{11} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} = \begin{bmatrix} 10 \\ b_2+b_1 \\ \vdots \\ b_2+b_1 \\ 1 \end{bmatrix}$$

Результат работы программы

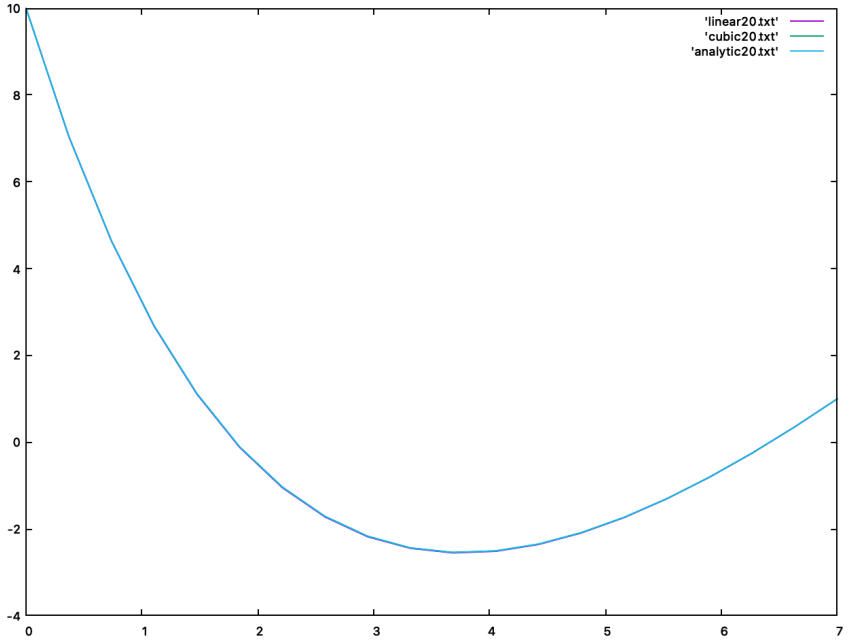

Рисунок 1 — Результат работы программы для 20 КЭ.

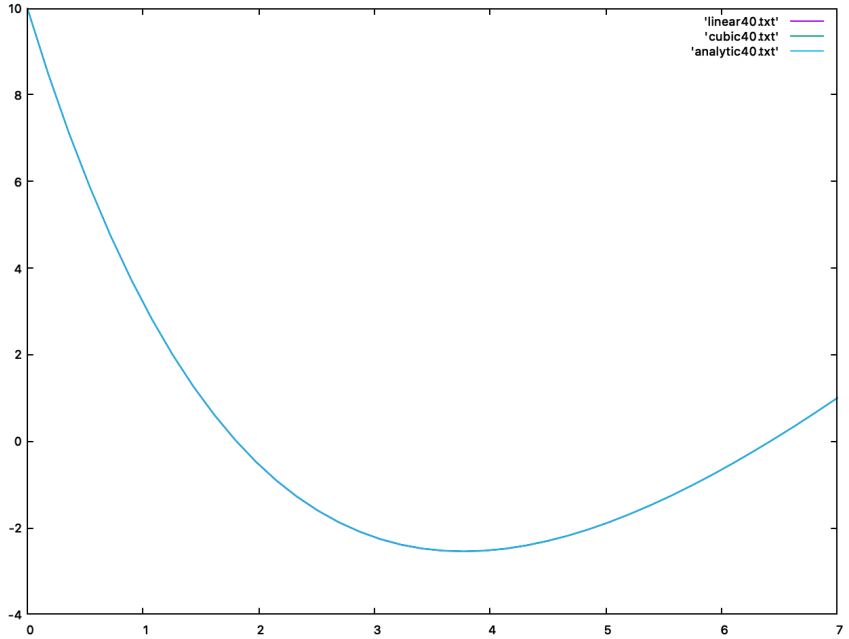

Рисунок 2 — Результат работы программы для 40 КЭ.

Количество КЭ	20	40
Линейная функция формы	0.0132534	0.00314752
Кубическая функция формы	7.42667e-10	1.05391e-11

Таблица 1 — Погрешность вычисления при использовании разного количества КЭ и разных функций формы.

Исходный код

```
#include <math.h>
#include <vector>
#include <string>


typedef std::vector<std::vector<double>> Matrix;
typedef std::vector<double> Vector;

Matrix create_matrix(int rows, int cols) {
    Matrix res;
    res.resize(rows);
    for (int i = 0; i < rows; ++i) {
        res[i].resize(cols, 0);
    }
    return res;
}

void zero(Matrix& m) {
    for (int i = 0; i < m.size(); ++i) {
        std::fill(m[i].begin(), m[i].end(), 0);
    }
}

void print(const Matrix& m) {
    for (int i = 0; i < m.size(); ++i) {
        for (int j = 0; j < m[i].size(); ++j) {
            printf("%-20.8f ", m[i][j]);
        }
        puts("");
    }
}

Vector create_vector(int size) {
    Vector res;
    res.resize(size, 0);
    return res;
}

void zero(Vector& v) {
    std::fill(v.begin(), v.end(), 0);
}

void print(const Vector& v) {
    for (int i = 0; i < v.size(); ++i) {
        printf("%5f ", v[i]);
```

```
    }
    puts("");
}

// it breaks contents of m and b
int gauss(Matrix& m, Vector& b) {
    for (int k = 0; k < m.size(); ++k) {
        if (fabs(m[k][k]) < 1e-17) {
            return 1;
        }
        double diagonal = m[k][k];
        // divide this row by diagonal element
        for (int i = k; i < m.size(); ++i) {
            m[k][i] /= diagonal;
        }
        b[k] /= diagonal;
        for (int i = k+1; i < m.size(); ++i) {
            double elem = m[i][k];
            for (int j = k; j < m.size(); ++j) {
                m[i][j] -= elem * m[k][j];
            }
            b[i] -= elem * b[k];
        }
    }
    for (int i = m.size()-2; i >= 0; --i) {
        for (int j = i + 1; j < m.size(); ++j) {
            b[i] -= m[i][j] * b[j];
        }
    }
    return 0;
}

Vector solve_FEM(const Matrix& local_matrix, const Vector&
local_vector, int n) {
    // assemble matrix
    auto matrix = create_matrix(n+1, n+1);
    for (int i = 0; i < n; ++i) {
        matrix[i][i]     += local_matrix[0][0];
        matrix[i][i+1]   += local_matrix[0][1];
        matrix[i+1][i]   += local_matrix[1][0];
        matrix[i+1][i+1] += local_matrix[1][1];
    }

    // assemble vector
    auto vector = create_vector(n+1);
    for (int i = 0; i < n; ++i) {
        vector[i]     += local_vector[0];
        vector[i+1]   += local_vector[1];
    }

    // assign known variables
    vector[0] = 10;
    matrix[0][0] = 1;
    matrix[0][1] = 0;
    vector[n] = 1;
```

```
    matrix[n][n] = 1;
    matrix[n][n-1] = 0;

    if (gauss(matrix, vector) != 0) {
        puts("Error: degraded matrix");
        exit(-1);
    }

    return vector;
}

void save_to_file(const Vector& vector, const std::string& name) {
    FILE *f = fopen(name.c_str(), "w");
    assert(f);

    double L = 7.0 / (vector.size()-1);
    for (int i = 0; i < vector.size(); ++i) {
        fprintf(f, "%lf %lf\n", i * L, vector[i]);
    }

    fclose(f);
}

void compress(Matrix& matrix, Vector& vector) {
    for (int i = 1; i < 3; ++i) {
        for (int j = 0; j < 4; ++j) {
            if (fabs(matrix[j][i]) < 1e-10 || i == j) {
                continue;
            }
            double val = matrix[j][i]/matrix[i][i];
            vector[j] -= val * vector[i];
            for (int k = 0; k < 4; ++k) {
                matrix[j][k] -= val * matrix[i][k];
            }
        }
    }
}

double max_d(const Vector& a, const Vector& b) {
    double max_d = -10;
    for (int i = 0; i < a.size(); ++i) {
        auto d = fabs(a[i] - b[i]);
        if (d > max_d) {
            max_d = d;
        }
    }
    return max_d;
}

Vector solve_analytical(int node_count) {
    Vector result;
    result.resize(node_count);
    double t = exp(14.0/5.0);
    double L = 7.0 / (node_count-1);
    for (int i = 0; i < node_count; ++i) {
```

```
        double x = L*i;
        double a = (20.0 + 33.0*t)/(2.0 - 2.0*t);
        double b = 53.0*t / (2.0*t-2.0);
        result[i] = a + b * exp(-0.4*x) + 2.5*x;
    }
    return result;
}

Vector solve_linear(int node_count) {
    double L = 7.0 / (node_count-1);

    Matrix local_matrix = {
        {-1 - 5/L, 1 + 5/L},
        {-1 + 5/L, 1 - 5/L}
    };
    Vector local_vector = {
        5*L/2,
        5*L/2
    };
    return solve_FEM(local_matrix, local_vector, node_count-1);
}

Vector solve_cubic(int node_count) {
    double L = 7.0 / (node_count-1);
    Matrix temp_matrix = {
        {-37.0/(2*L)-1.0, 189.0/(8*L)+57.0/40, -27.0/(4*L)-3.0/5, 13.0/
(8*L)+7.0/40},
        {189.0/(8*L)-57.0/40, -54.0/L, 297.0/(8*L)+81.0/40, -27.0/
(4*L)-3.0/5},
        {-27.0/(4*L)+3.0/5, 297.0/(8*L)-81.0/40, -54.0/L, 189/(8*L)
+57.0/40},
        {13.0/(8*L)-7.0/40, -27.0/(4*L)+3.0/5, 189.0/(8*L)-57.0/40,
-37.0/(2*L)+1}
    };
    Vector temp_vector = {5.*L/8, 15.*L/8, 15.*L/8, 5.*L/8};
    compress(temp_matrix, temp_vector);
    Matrix local_matrix = {
        {temp_matrix[0][0], temp_matrix[0][3]},
        {temp_matrix[3][0], temp_matrix[3][3]}
    };
    Vector local_vector = {
        temp_vector[0],
        temp_vector[3]
    };
    return solve_FEM(local_matrix, local_vector, node_count-1);
}

int main() {
    auto analytic20 = solve_analytical(20);
    auto analytic40 = solve_analytical(40);

    auto linear20 = solve_linear(20);
    auto linear40 = solve_linear(40);

    auto cubic20 = solve_cubic(20);
```

```
    auto cubic40 = solve_cubic(40);

    save_to_file(analytic20, "analytic20.txt");
    save_to_file(analytic40, "analytic40.txt");

    save_to_file(linear20, "linear20.txt");
    save_to_file(linear40, "linear40.txt");

    save_to_file(cubic20, "cubic20.txt");
    save_to_file(cubic40, "cubic40.txt");

    printf("Error linear20: %g\n", max_d(analytic20, linear20));
    printf("Error linear40: %g\n", max_d(analytic40, linear40));

    printf("Error cubic20: %g\n", max_d(analytic20, cubic20));
    printf("Error cubic40: %g\n", max_d(analytic40, cubic40));

    return 0;
}
```