Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение
высшего профессионального 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:	Крылов Александр Сергеевич
Группа:	РК6-75б
Тип задания:	Лабораторная работа
Название:	Метод конечных элементов
Вариант:	10

Студент ______ (подпись, дата) Крылов А. С. (Фамилия, И.О.)

Преподаватель ______ (подпись, дата) Трудоношин В. А. (Фамилия, И.О.)

Оценка: ______

Москва, 2023

Содержание

Метод конечных элементов

1 Цель выполнения лабораторной работы

Цель выполнения лабораторной работы – решение дифференциального уравнения методом конечных элементов (МКЭ), используя линейную и кубическую функции формы, и анализ точности относительной аналитического способа решения

2 Задание

Решить с помощью МКЭ уравнение 1

$$4\frac{d^2u}{dx^2} - 11u + 7 = 0, \tag{1}$$

при следующих граничных условиях (г. у.):

$$u(x = 1) = -10, \tag{2}$$

$$u'(x = 32) = 5. \tag{3}$$

Количество конечных элементов

- для первого расчета – 20,
- для второго – 40.

Также необходимо:

1. Сравнить результаты с аналитическим решением. Оценить максимальную погрешность.
2. Определить количество линейных КЭ, обеспечивающих такую же точность как и кубические.

3 Аналитическое решение

На рисунке 1 представлено аналитическое решение поставленной задачи.

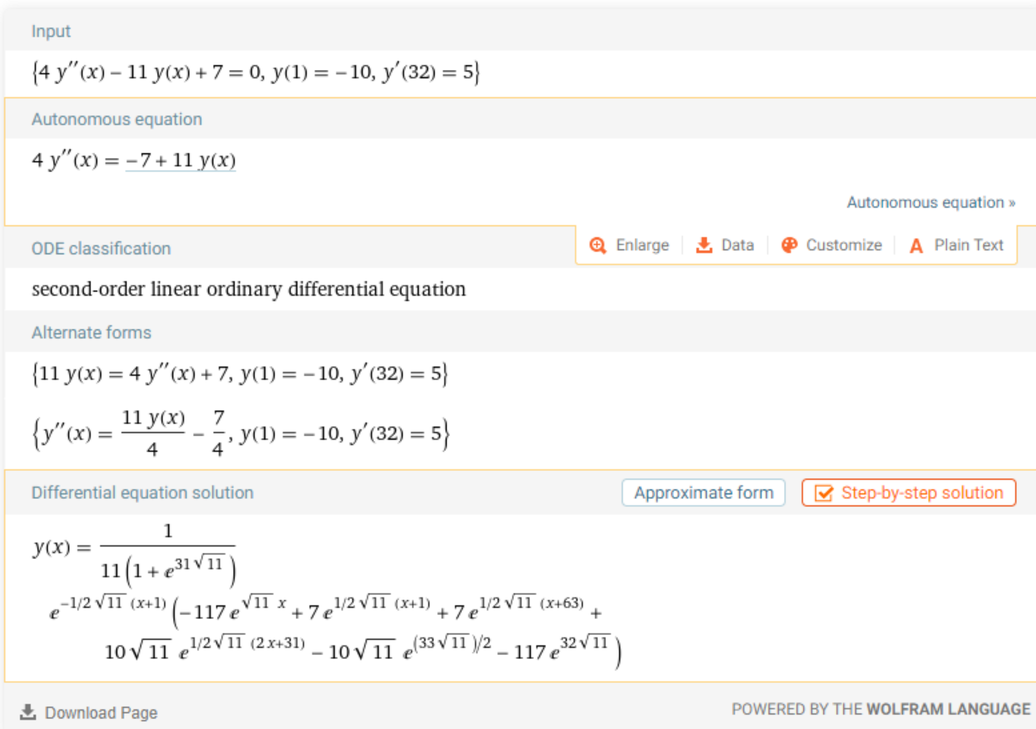

Рис. 1. Аналитическое решение

Таким образом, получаем:

$$u(x) = \begin{aligned} &\frac{1}{11\left(1+e^{31\sqrt{11}}\right)} e^{-1/2\sqrt{11}(x+1)} \cdot \\ &\cdot \left(-117e^{\sqrt{11}x} + 7e^{1/2\sqrt{11}(x+1)} + 7e^{1/2\sqrt{11}(x+63)} + \right. \\ &\left. +10\sqrt{11}e^{1/2\sqrt{11}(2x+31)} - 10\sqrt{11}e^{(33\sqrt{11})/2} - 117e^{32\sqrt{11}}\right) \end{aligned}.$$

4 Получение локальных матрицы жесткости и вектора нагрузок

Составим локальные матрицу жесткости и вектор нагрузок для уравнения 1.

Линейная функция-формы КЭ

$$\mathbf{u} = \left[\left(1-\frac{x}{L}\right); \quad \frac{x}{L}\right]\begin{bmatrix} u_i \\ u_j \end{bmatrix} = \mathbf{N_e U},$$

где $\mathbf{N_e}$ – вектор функции формы конечного элемента (в данном случае линейной), его составляющие элементы - глобальные базисные функции, отличные от нуля в пределах этого элемента, L – длина КЭ.

В соответствии с методом Галеркина для уравнения 1:

$$\int_0^L \mathbf{W_e}\left(4\frac{d^2\mathbf{u}}{dx^2} - 11\mathbf{u} + 7\right)dx = 0, \tag{4}$$

где $\mathbf{W_e} = \mathbf{N_e}^T$.

$$\int_0^L \mathbf{W_e}\left(4\frac{d^2\mathbf{u}}{dx^2} - 11\mathbf{u} + 7\right)dx = 4\int_0^L \mathbf{W_e}\frac{d^2\mathbf{u}}{dx^2}dx - 11\int_0^L \mathbf{W_e}\mathbf{u}dx + 7\int_0^L \mathbf{W_e}dx = 0$$

Распишем каждое слагаемое отдельно:

$$4\int_0^L \mathbf{W_e}\frac{d^2\mathbf{u}}{dx^2}dx = 4\int_0^L \begin{bmatrix}(1-\frac{x}{L})\\ \frac{x}{L}\end{bmatrix}\frac{d^2\mathbf{u}}{dx^2}dx = 4\begin{bmatrix}(1-\frac{x}{L})\\ \frac{x}{L}\end{bmatrix}\frac{d\mathbf{u}}{dx}\Big|_0^L -$$

$$-4\int_0^L \frac{d}{dx}\begin{bmatrix}(1-\frac{x}{L})\\ \frac{x}{L}\end{bmatrix}\frac{d}{dx}\begin{bmatrix}(1-\frac{x}{L}); & \frac{x}{L}\end{bmatrix}\begin{bmatrix}u_i\\ u_j\end{bmatrix} = \begin{bmatrix}-4\frac{d\mathbf{u}}{dx}|_i\\ 4\frac{d\mathbf{u}}{dx}|_j\end{bmatrix} - 4\begin{bmatrix}\frac{1}{L}, & -\frac{1}{L}\\ -\frac{1}{L}, & \frac{1}{L}\end{bmatrix}\begin{bmatrix}u_i\\ u_j\end{bmatrix}$$

$$-11\int_0^L \mathbf{W_e}\mathbf{u}dx = -11\int_0^L \begin{bmatrix}(1-\frac{x}{L})\\ \frac{x}{L}\end{bmatrix}\begin{bmatrix}(1-\frac{x}{L}); & \frac{x}{L}\end{bmatrix}\begin{bmatrix}u_i\\ u_j\end{bmatrix}dx = -11\begin{bmatrix}\frac{L}{3}, & \frac{L}{6}\\ \frac{L}{6}, & \frac{L}{3}\end{bmatrix}\begin{bmatrix}u_i\\ u_j\end{bmatrix}$$

$$7\int_0^L \mathbf{W_e}dx = 7\begin{bmatrix}\frac{L}{2}\\ \frac{L}{2}\end{bmatrix}$$

Таким образом, для уравнения 4, при использовании линейной функции-формы, получаем (матмодель линейного КЭ):

$$\begin{bmatrix}4\frac{1}{L} + 11\frac{L}{3}, & -4\frac{1}{L} + 11\frac{L}{3}\\ -4\frac{1}{L} + 11\frac{L}{3}, & 4\frac{1}{L} + 11\frac{L}{3}\end{bmatrix}\begin{bmatrix}u_i\\ u_j\end{bmatrix} = \begin{bmatrix}-4\frac{du}{dx}|_i + 7\frac{L}{2}\\ 4\frac{du}{dx}|_j + 7\frac{L}{2}\end{bmatrix}$$

Кубическая функция-формы КЭ

$$\mathbf{u} = \left[-\frac{9x^3}{2L^3} + \frac{18x^2}{2L^2} - \frac{11x}{2L} + 1; \frac{27x^3}{2L^3} - \frac{45x^2}{2L^2} + \frac{9x}{L}; -\frac{27x^3}{2L^3} + \frac{36x^2}{2L^2} - \frac{9x}{2L}; \frac{9x^3}{2L^3} - \frac{9x^2}{2L^2} - \frac{x}{L};\right]\begin{bmatrix}u_i\\ u_j\\ u_k\\ u_l\end{bmatrix} = \mathbf{N_e}\mathbf{U},$$

Как и для линейной функции-формы применим метод Галеркина (см. уравнение 4) и рассмотрим каждое слагаемое отдельно.

$$4\int_0^L \mathbf{W_e}\frac{d^2\mathbf{u}}{dx^2}dx = 4\int_0^L \begin{bmatrix} -\frac{9x^3}{2L^3} & + & \frac{18x^2}{2L^2} & - & \frac{11x}{2L} & + & 1 \\ \frac{27x^3}{2L^3} & - & \frac{45x^2}{2L^2} & + & \frac{9x}{L} & & \\ -\frac{27x^3}{2L^3} & + & \frac{36x^2}{2L^2} & - & \frac{9x}{2L} & & \\ \frac{9x^3}{2L^3} & - & \frac{9x^2}{2L^2} & - & \frac{x}{L} & & \end{bmatrix}\frac{d^2\mathbf{u}}{dx^2}dx =$$

$$= 4\begin{bmatrix} -\frac{9x^3}{2L^3} & + & \frac{18x^2}{2L^2} & - & \frac{11x}{2L} & + & 1 \\ \frac{27x^3}{2L^3} & - & \frac{45x^2}{2L^2} & + & \frac{9x}{L} & & \\ -\frac{27x^3}{2L^3} & + & \frac{36x^2}{2L^2} & - & \frac{9x}{2L} & & \\ \frac{9x^3}{2L^3} & - & \frac{9x^2}{2L^2} & - & \frac{x}{L} & & \end{bmatrix}\frac{d\mathbf{u}}{dx}\Big|_0^L - 4\int_0^L \frac{d}{dx}\begin{bmatrix} -\frac{9x^3}{2L^3} & + & \frac{18x^2}{2L^2} & - & \frac{11x}{2L} & + & 1 \\ \frac{27x^3}{2L^3} & - & \frac{45x^2}{2L^2} & + & \frac{9x}{L} & & \\ -\frac{27x^3}{2L^3} & + & \frac{36x^2}{2L^2} & - & \frac{9x}{2L} & & \\ \frac{9x^3}{2L^3} & - & \frac{9x^2}{2L^2} & - & \frac{x}{L} & & \end{bmatrix}\frac{d}{dx}\mathbf{u} =$$

$$= \begin{bmatrix} -4\frac{d\mathbf{u}}{dx}\big|_i \\ 0 \\ 0 \\ 4\frac{d\mathbf{u}}{dx}\big|_l \end{bmatrix} - 4\begin{bmatrix} \frac{37}{10L} & -\frac{189}{40} & \frac{27}{20L} & -\frac{13}{40L} \\ -\frac{189}{40} & \frac{54}{5L} & -\frac{297}{40L} & \frac{27}{20L} \\ \frac{27}{20L} & -\frac{297}{40L} & \frac{54}{5L} & -\frac{189}{40} \\ -\frac{13}{40L} & \frac{27}{20L} & -\frac{189}{40} & \frac{37}{10L} \end{bmatrix}\begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix}$$

$$-11\int_0^L \mathbf{W_e}\mathbf{u}dx = -11\int_0^L \begin{bmatrix} -\frac{9x^3}{2L^3} + \frac{18x^2}{2L^2} - \frac{11x}{2L} + 1 \\ \frac{27x^3}{2L^3} - \frac{45x^2}{2L^2} + \frac{9x}{L} \\ -\frac{27x^3}{2L^3} + \frac{36x^2}{2L^2} - \frac{9x}{2L} \\ \frac{9x^3}{2L^3} - \frac{9x^2}{2L^2} - \frac{x}{L}; \end{bmatrix}\mathbf{u}dx = -11\begin{bmatrix} \frac{8L}{105} & \frac{33L}{560} & -\frac{3L}{140} & \frac{19L}{1680} \\ \frac{33L}{560} & \frac{27L}{70} & -\frac{27L}{560} & -\frac{3L}{140} \\ -\frac{3L}{140} & -\frac{27L}{560} & \frac{27L}{70} & \frac{33L}{560} \\ \frac{19L}{1680} & -\frac{3L}{140} & \frac{33L}{560} & \frac{8L}{105} \end{bmatrix}\begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix}$$

$$7\int_0^L \mathbf{W_e}dx = 7\begin{bmatrix} \frac{L}{8} \\ \frac{3L}{8} \\ \frac{3L}{8} \\ \frac{L}{8} \end{bmatrix}$$

Таким образом, для уравнения 4, при использовании кубической функции-формы, получаем:

$$\begin{bmatrix} 4\frac{37}{10L}+11\frac{8L}{105} & -4\frac{189}{40L}+11\frac{33L}{560} & 4\frac{27}{20L}-11\frac{3L}{140} & -4\frac{13}{40L}+11\frac{19L}{1680} \\ -4\frac{189}{40L}+11\frac{33L}{560} & 4\frac{54}{5L}+11\frac{27L}{70} & -4\frac{297}{40L}-11\frac{27L}{560} & 4\frac{27}{20L}-11\frac{3L}{140} \\ 4\frac{27}{20L}-11\frac{3L}{140} & -4\frac{297}{40L}-11\frac{27L}{560} & 4\frac{54}{5L}+11\frac{27L}{70} & -4\frac{189}{40L}+11\frac{33L}{560} \\ -4\frac{13}{40L}+11\frac{19L}{1680} & 4\frac{27}{20L}-11\frac{3L}{140} & -4\frac{189}{40L}+11\frac{33L}{560} & 4\frac{37}{10L}+11\frac{8L}{105} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} = \begin{bmatrix} 7\frac{L}{8}-4\frac{du}{dx}\big|_i \\ 7\frac{3L}{8} \\ 7\frac{3L}{8} \\ 7\frac{L}{8}+4\frac{du}{dx}\big|_l \end{bmatrix} \tag{5}$$

Локальные матрица жесткости и вектор нагрузок могут быть представлены в виде:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} = \begin{bmatrix} b_1 - 2\frac{du}{dx}\big|_i \\ b_2 \\ b_3 \\ b_4 + 2\frac{du}{dx}\big|_l \end{bmatrix}$$

Выполним матричные преобразования.

$$\begin{bmatrix} a_{11}-\frac{a_{12}}{a_{22}}a_{21} & a_{12}-\frac{a_{12}}{a_{22}}a_{22} & a_{13}-\frac{a_{12}}{a_{22}}a_{23} & a_{14}-\frac{a_{12}}{a_{22}}a_{24} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31}-\frac{a_{32}}{a_{22}}a_{21} & a_{32}-\frac{a_{32}}{a_{22}}a_{22} & a_{33}-\frac{a_{32}}{a_{22}}a_{23} & a_{34}-\frac{a_{32}}{a_{22}}a_{24} \\ a_{41}-\frac{a_{42}}{a_{22}}a_{21} & a_{42}-\frac{a_{42}}{a_{22}}a_{22} & a_{43}-\frac{a_{42}}{a_{22}}a_{23} & a_{44}-\frac{a_{42}}{a_{22}}a_{24} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} = \begin{bmatrix} b_1-\frac{a_{12}}{a_{22}}b_2-4\frac{du}{dx}\big|_i \\ b_2 \\ b_3-\frac{a_{32}}{a_{22}}b_2 \\ b_4-\frac{a_{42}}{a_{22}}b_2+4\frac{du}{dx}\big|_l \end{bmatrix}$$

$$\begin{bmatrix} a'_{11}-\frac{a'_{13}}{a'_{33}}a'_{31} & 0-\frac{a'_{13}}{a'_{33}}0 & a'_{13}--\frac{a'_{13}}{a'_{33}}a'_{33} & a'_{14}-\frac{a'_{13}}{a'_{33}}a'_{34} \\ a'_{21}-\frac{a'_{23}}{a'_{33}}a'_{31} & a'_{22}-\frac{a'_{23}}{a'_{33}}0 & a'_{23}-\frac{a'_{23}}{a'_{33}}a'_{33} & a'_{24}-\frac{a'_{23}}{a'_{33}}a'_{34} \\ a'_{31} & 0 & a'_{33} & a'_{34} \\ a'_{41}-\frac{a'_{43}}{a'_{33}}a'_{31} & 0-\frac{a'_{43}}{a'_{33}}0 & a'_{43}-\frac{a'_{43}}{a'_{33}}a'_{33} & a'_{44}-\frac{a'_{43}}{a'_{33}}a'_{34} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} = \begin{bmatrix} b'_1-\frac{a'_{13}}{a'_{33}}b'_3-4\frac{du}{dx}\big|_i \\ b'_2-\frac{a'_{23}}{a'_{33}}b'_3 \\ b'_3 \\ b'_4-\frac{a'_{43}}{a'_{33}}b'_3+4\frac{du}{dx}\big|_l \end{bmatrix}$$

Итого получаем:

$$\begin{bmatrix} a''_{11} & 0 & 0 & a''_{14} \\ a''_{21} & a''_{22} & 0 & a''_{24} \\ a''_{31} & 0 & a''_{33} & a''_{34} \\ a''_{41} & 0 & 0 & a''_{44} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix} = \begin{bmatrix} b''_1-4\frac{du}{dx}\big|_i \\ b''_2 \\ b''_3 \\ b''_4+4\frac{du}{dx}\big|_l \end{bmatrix}$$

Для упрощения расчетов преобразуем систему выше, исключив внутренние узлы. Таким образом, при использовании кубической функции-формы, получаем:

$$\begin{bmatrix} a''_{11} & a''_{14} \\ a''_{41} & a''_{44} \end{bmatrix} \begin{bmatrix} u_i \\ u_l \end{bmatrix} = \begin{bmatrix} b''_1-4\frac{du}{dx}\big|_i \\ b''_4+4\frac{du}{dx}\big|_l \end{bmatrix}$$

5 Получение глобальных матрицы жесткости и вектора нагрузок

Проведем процедуры ансамблирования и учет граничных условий для формирования итоговой математической модели.

Ансамблирование

Пусть локальные матрица жесткости и вектор неизвестных заданы следующим образом

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} u_i \\ u_j \end{bmatrix} = \begin{bmatrix} b_1 - 4\frac{du}{dx}|_i \\ b_2 + 4\frac{du}{dx}|_l \end{bmatrix},$$

тогда, при разбитие области на n КЭ, глобальная матрица жесткости будет иметь размерность $(n+1)\cdot(n+1)$:

$$\begin{bmatrix} a_{11}^1 & a_{12}^1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ a_{21}^1 & a_{22}^1 + a_{11}^2 & a_{12}^2 & 0 & \cdots & 0 & 0 & 0 \\ 0 & a_{21}^2 & a_{22}^2 + a_{11}^3 & a_{12}^3 & 0 & \cdots & 0 & 0 \\ 0 & 0 & a_{21}^3 & a_{22}^3 + \cdots & & & & 0 \\ \vdots & \vdots & \vdots & \vdots & & & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots + a_{11}^n & a_{12}^n \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{21}^n & a_{22}^n \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} = \begin{bmatrix} b_1^1 - 4\frac{du}{dx}|_0 \\ b_2^1 + b_1^2 \\ b_2^2 + b_1^3 \\ b_2^3 + b_1^4 \\ \vdots \\ b_2^{n-1} + b_1^n \\ b_2^n + 4\frac{du}{dx}|_L \end{bmatrix}$$

Учет граничных условий

Применим граничные условия первого (см. 2) и второго рода (см. 3) к выведенной выше системе.

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ a_{21}^1 & a_{22}^1 + a_{11}^2 & a_{12}^2 & 0 & \cdots & 0 & 0 & 0 \\ 0 & a_{21}^2 & a_{22}^2 + a_{11}^3 & a_{12}^3 & 0 & \cdots & 0 & 0 \\ 0 & 0 & a_{21}^3 & a_{22}^3 + \cdots & & & & 0 \\ \vdots & \vdots & \vdots & \vdots & & & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots + a_{11}^n & a_{12}^n \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{21}^n & a_{22}^n \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} = \begin{bmatrix} -10 \\ b_2^1 + b_1^2 \\ b_2^2 + b_1^3 \\ b_2^3 + b_1^4 \\ \vdots \\ b_2^{n-1} + b_1^n \\ b_2^n + 4\cdot 5 \end{bmatrix}$$

6 Анализ результатов

Проведем сравнение результатов согласно заданию.

Линейная функция-формы

На рисунках 2, 3 представлены графики полученные с помощью МКЭ (линейная функция-формы).

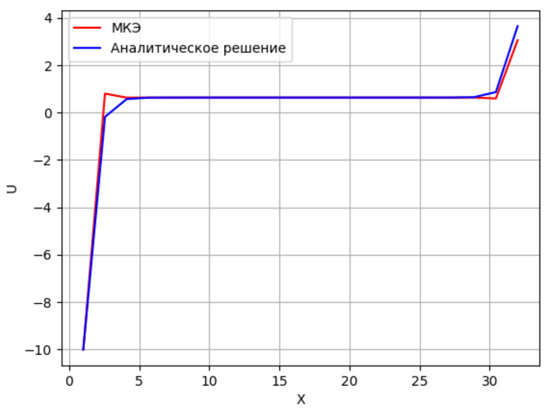

Рис. 2. Результат работы программы для 20 КЭ

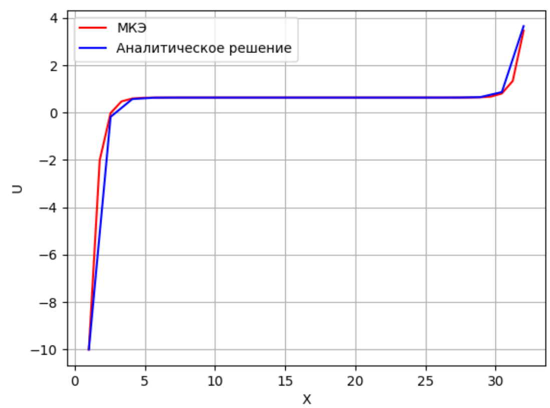

Рис. 3. Результат работы программы для 40 КЭ

X	Аналитическое решение	МКЭ-решение	Абсолютная погрешность
1.000000	-10.000000	-10.000000	5.151435e-14
2.550000	-0.177384	0.804383	9.817663e-01
4.100000	0.574107	0.633709	5.960252e-02
5.650000	0.631601	0.636406	4.804944e-03
7.200000	0.635999	0.636363	3.637378e-04
8.750000	0.636336	0.636364	2.788931e-05
10.300000	0.636362	0.636364	2.132738e-06
11.850000	0.636363	0.636364	1.631828e-07
13.400000	0.636364	0.636364	1.248412e-08
14.950000	0.636364	0.636364	9.535381e-10
16.500000	0.636364	0.636364	5.236078e-11
18.050000	0.636364	0.636364	2.651520e-10
19.600000	0.636364	0.636364	3.538516e-09
21.150000	0.636364	0.636364	4.625762e-08
22.700000	0.636364	0.636364	6.045811e-07
24.250000	0.636372	0.636364	7.905259e-06
25.800000	0.636467	0.636364	1.031465e-04
27.350000	0.637714	0.636354	1.359727e-03
28.900000	0.654012	0.636968	1.704382e-02
30.450000	0.867038	0.598114	2.689240e-01
32.000000	3.651477	3.057714	5.937635e-01

Таблица 1. 20 линейных КЭ

X	Аналитическое решение	МКЭ-решение	Абсолютная погрешность
1.000000	-10.000000	-10.000000	5.151435e-14
1.775000	-2.305627	-2.002239	3.033885e-01
2.550000	-0.177384	-0.018204	1.591792e-01
3.325000	0.411283	0.473983	6.269927e-02
4.100000	0.574107	0.596081	2.197417e-02
4.875000	0.619144	0.626371	7.227005e-03
5.650000	0.631601	0.633885	2.284009e-03
6.425000	0.635046	0.635749	7.024616e-04
7.200000	0.635999	0.636211	2.118405e-04
7.975000	0.636263	0.636326	6.294610e-05
8.750000	0.636336	0.636354	1.849024e-05
9.525000	0.636356	0.636361	5.382151e-06
10.300000	0.636362	0.636363	1.555123e-06
11.075000	0.636363	0.636363	4.466234e-07
11.850000	0.636363	0.636364	1.276233e-07
12.625000	0.636364	0.636364	3.631445e-08
13.400000	0.636364	0.636364	1.029597e-08
14.175000	0.636364	0.636364	2.909886e-09
14.950000	0.636364	0.636364	8.190102e-10
15.725000	0.636364	0.636364	2.255953e-10
16.500000	0.636364	0.636364	4.627598e-11
17.275000	0.636364	0.636364	4.786616e-11
18.050000	0.636364	0.636364	2.298726e-10
18.825000	0.636364	0.636364	8.331842e-10
19.600000	0.636364	0.636364	2.956999e-09
20.375000	0.636364	0.636364	1.045017e-08
21.150000	0.636364	0.636364	3.680703e-08
21.925000	0.636364	0.636364	1.291425e-07
22.700000	0.636364	0.636364	4.510616e-07
23.475000	0.636366	0.636364	1.566918e-06
24.250000	0.636372	0.636366	5.407660e-06
25.025000	0.636392	0.636374	1.851339e-05
25.800000	0.636467	0.636404	6.275140e-05
26.575000	0.636737	0.636527	2.100142e-04
27.350000	0.637714	0.637022	6.913348e-04
28.125000	0.641245	0.639019	2.225545e-03
28.900000	0.654012	0.647070	6.942118e-03
29.675000	0.700168	0.679520	2.064786e-02
30.450000	0.867038	0.810329	5.670958e-02
31.225000	1.470336	1.337627	1.327089e-01
32.000000	3.651477	3.463200	1.882774e-01

Таблица 2. 40 линейных КЭ

Максимальная абсолютная погрешность 9.817663e-01 и 3.033885e-01 соответственно.

Кубическая функция-формы

На рисунках 4, 5 представлены графики полученные с помощью МКЭ (кубическая функция-формы).

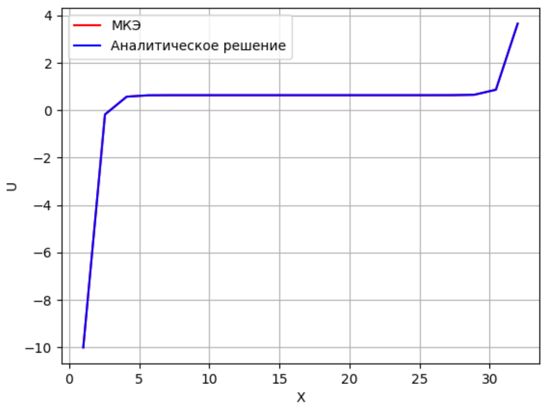

Рис. 4. Результат работы программы для 20 КЭ

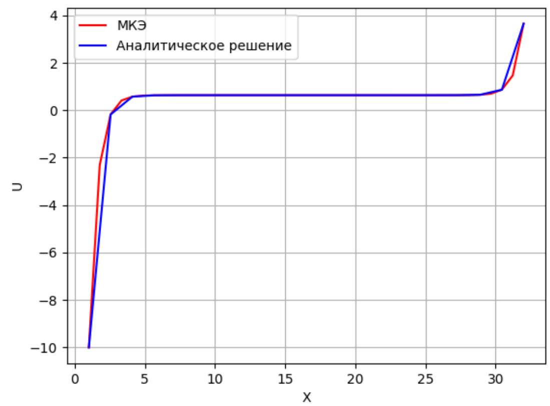

Рис. 5. Результат работы программы для 40 КЭ

X	Аналитическое решение	МКЭ-решение	Абсолютная погрешность
1.000000	-10.000000	-10.000000	5.151435e-14
2.550000	-0.177384	-0.173641	3.742709e-03
4.100000	0.574107	0.574678	5.713634e-04
5.650000	0.631601	0.631666	6.541855e-05
7.200000	0.635999	0.636006	6.657905e-06
8.750000	0.636336	0.636336	6.352525e-07
10.300000	0.636362	0.636362	5.818715e-08
11.850000	0.636363	0.636363	5.181725e-09
13.400000	0.636364	0.636364	4.520149e-10
14.950000	0.636364	0.636364	3.873557e-11
16.500000	0.636364	0.636364	2.346789e-12
18.050000	0.636364	0.636364	1.091083e-11
19.600000	0.636364	0.636364	1.306549e-10
21.150000	0.636364	0.636364	1.502185e-09
22.700000	0.636364	0.636364	1.693172e-08
24.250000	0.636372	0.636371	1.858183e-07
25.800000	0.636467	0.636465	1.962729e-06
27.350000	0.637714	0.637694	1.953440e-05
28.900000	0.654012	0.653837	1.749664e-04
30.450000	0.867038	0.865807	1.231670e-03
32.000000	3.651477	3.649235	2.241716e-03

Таблица 3. 20 кубических КЭ

X	Аналитическое решение	МКЭ-решение	Абсолютная погрешность
1.000000	-10.000000	-10.000000	5.151435e-14
1.775000	-2.305627	-2.305538	8.948718e-05
2.550000	-0.177384	-0.177334	4.950310e-05
3.325000	0.411283	0.411304	2.053833e-05
4.100000	0.574107	0.574115	7.574352e-06
4.875000	0.619144	0.619146	2.618768e-06
5.650000	0.631601	0.631601	8.692003e-07
6.425000	0.635046	0.635046	2.804841e-07
7.200000	0.635999	0.635999	8.866285e-08
7.975000	0.636263	0.636263	2.758900e-08
8.750000	0.636336	0.636336	8.478808e-09
9.525000	0.636356	0.636356	2.579692e-09
10.300000	0.636362	0.636362	7.783906e-10
11.075000	0.636363	0.636363	2.332303e-10
11.850000	0.636363	0.636363	6.947676e-11
12.625000	0.636364	0.636364	2.058398e-11
13.400000	0.636364	0.636364	6.064815e-12
14.175000	0.636364	0.636364	1.786349e-12
14.950000	0.636364	0.636364	5.188072e-13
15.725000	0.636364	0.636364	1.482148e-13
16.500000	0.636364	0.636364	3.308465e-14
17.275000	0.636364	0.636364	4.352074e-14
18.050000	0.636364	0.636364	1.409983e-13
18.825000	0.636364	0.636364	5.325740e-13
19.600000	0.636364	0.636364	1.791900e-12
20.375000	0.636364	0.636364	6.072920e-12
21.150000	0.636364	0.636364	2.056177e-11
21.925000	0.636364	0.636364	6.919876e-11
22.700000	0.636364	0.636364	2.318180e-10
23.475000	0.636366	0.636366	7.715921e-10
24.250000	0.636372	0.636372	2.549332e-09
25.025000	0.636392	0.636392	8.347920e-09
25.800000	0.636467	0.636467	2.703956e-08
26.575000	0.636737	0.636737	8.640092e-08
27.350000	0.637714	0.637714	2.713100e-07
28.125000	0.641245	0.641244	8.324309e-07
28.900000	0.654012	0.654009	2.472810e-06
29.675000	0.700168	0.700161	6.999577e-06
30.450000	0.867038	0.867020	1.829008e-05
31.225000	1.470336	1.470296	4.075939e-05
32.000000	3.651477	3.651421	5.565028e-05

Таблица 4. 40 кубических КЭ

Максимальная абсолютная погрешность 3.742709e-03 и 8.948718e-05 соответственно.

Нахождение количества линейных КЭ, обеспечивающих ту же точность, что и 20 кубических

Так как очевидно, что при увлечении числа КЭ точность растет, найдем искомое следуя алгоритму, представленному на рисунке 6.

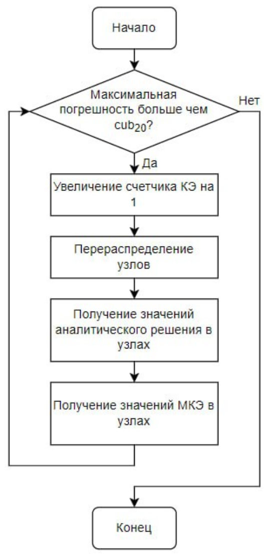

Рис. 6. Алгоритм нахождения количества КЭ, заданную точность ($cub_{20} = 3.742709e - 03$)

Реализовав данный алгоритм с начальным количеством КЭ=20 и увеличивая счетчик всегда на 1 получаем необходимое количество КЭ, равное 341 .

7 Код

Листинг 1. *Реализация МКЭ*

```

#include <iostream>
#include <vector>
#include <cmath>

//define OSN
//define TABLE
//define CUBE

constexpr double EPS = 1e−16,CUB= 0.003742709;
double EPS = 1e−16;
```

```cpp
double X_BEGIN = 1.0;
double X_END = 32.0;
size_t ELEMS_NUM = ;
double L = (X_END - X_BEGIN) / ELEMS_NUM;

constexpr double a = 4.0, B = 0.0, C = -11.0, D = 7.0, usl_left = -10.0, usl_right = 5.0; // au''+Bu'+Cu+D=0

std::vector<double> solve_with_gauss(std::vector<std::vector<double>>& A, std::vector<double>& b){
    size_t row_size = A.size();
    size_t col_size = A.back().size();
    // Прямой ход Гаусса
    double pivot = 0.0;
    for (size_t i = 0; i < row_size; i++) {
        for (size_t j = i + 1; j < col_size; j++) {
            if (std::abs(A.at(j).at(i)) < EPS) {
                continue;
            }
            pivot = A.at(j).at(i) / A.at(i).at(i);
            b.at(j) -= pivot * b.at(i);
            for (size_t k = 0; k < row_size; k++) {
                A.at(j).at(k) -= pivot * A.at(i).at(k) ;
            }
        }
    }
    // Обратный ход Гаусса
    std::vector<double> x(row_size);
    for (int i = row_size - 1.0; i >= 0; i--) {
        x.at(i) = b.at(i);
        for (size_t j = i + 1; j < row_size; j++) {
            x.at(i) -= x.at(j) * A.at(i).at(j);
        }
        x.at(i) /= A.at(i).at(i);
    }
    return x;
}

double analytical_solution(double x) {
    double rez = (exp(-1. / 2. * sqrt(11.) * (x + 1.)) * (-117. * exp(sqrt(11.) * x) + 7. *
        exp(1. / 2. * sqrt(11.) * (x + 1.)) + 7. * exp(1. / 2. * sqrt(11.) * (x + 63.)) + 10.
        * sqrt(11.) * exp(1. / 2. * sqrt(11.) * (2. * x + 31.)) - 10. * sqrt(11.) * exp((33. *
        sqrt(11.))/2.) - 117. * exp(32. * sqrt(11.))))/(11. * (1. + exp(31. * sqrt(11.))));
    return rez;
}
```

```
std::vector<double> build_analytical_solution(std::vector<double>& x_vec) {
    size_t x_vec_size = x_vec.size();
    std::vector<double> y_vec = std::vector<double>(x_vec_size);
    for (size_t i = 0; i < x_vec_size; i++) {
        y_vec.at(i) = analytical_solution(x_vec.at(i));
    }
    return y_vec;
}

std::vector<double> build_linear_solution(size_t elems_num) {
    double L = (X_END - X_BEGIN) / elems_num;
    size_t size = elems_num + 1;
    std::vector< std::vector<double> > A(size, std::vector<double>(size));
    std::vector<double> b(size);

    // Локальная матрица жесткости для линейного КЭ
    std::vector< std::vector<double> > local_matrix = {
        { a/L - C * L/3.0 + B*1.0/2.0, -a/L - C * L/6.0 - B*1.0/2.0},
        { -a/L - C * L/6.0 + B*1.0/2.0, a/L - C*L/3.0 - B*1.0/2.0},
    };

    // Ансамблирование и получение глобальной матрицы жесткости для линейного КЭ
    for (size_t i = 0; i < elems_num; i++) {
        for (size_t j = 0; j < 2; j++) {
            for (size_t k = 0; k < 2; k++) {
                A.at(i + j).at(i + k) += local_matrix.at(j).at(k);
            }
        }
    }

        for (size_t i = 0; i < size ; i++) {
        b.at(i) = D * L;
    }

    // Учет ГУ
    b.at(0) = usl_left;
    A.at(0).at(0) = 1;
    A.at(0).at(1) = 0;

    b.at(size - 1) = D * L /2. + a*usl_right;

    // Решение полученной СЛАУ методом Гаусса
    std::vector<double> res = solve_with_gauss(A, b);
    return res;
}
```

```
std::vector<double> build_cube_solution(size_t elems_num) {
    double L = (X_END - X_BEGIN) / elems_num;
    size_t size = elems_num + 1;
    std::vector< std::vector<double> > A(size,std::vector<double>(size));
    std::vector<double> b(size);

    // Локальная матрица жесткости для кубического КЭ
        std::vector< std::vector<double> > local_matrix = {

          { a*37.0/(10.0*L) - C*8*L/105.0 +B*1.0/2.0, -a*189.0/(40.0*L) - C*33*L/560.0
              - B*57/80.0, a*27.0/(20.0*L) + C*3*L/140.0 + B*3.0/10.0,
              -a*13.0/(40.0*L) - C*19.0*L/1680.0 - B*7/80.0},
          { -a*189.0/(40.0*L) - C*33*L/560.0 + B*57/80.0, a*54.0/(5.0*L)-C*27*L/70.0,
              -a*297.0/(40*L) + C*27*L/560.0 - B*81.0/80.0, a*27.0/(20.0*L) +
              C*3*L/140.0 + B*3.0/10.0},
          { a*27.0/(20.0*L) + C*3*L/140.0 - B*3.0/10.0, -a*297.0/(40.0*L) +
              C*27*L/560.0 + B*81.0/80.0, a*54.0/(5.0*L) - C*27*L/70.0,
              -a*189.0/(40.0*L) - C*33*L/560.0 - B*57/80.0},
          { -a*13.0/(40.0*L) - C*19.0*L/1680.0 + B*7/80.0, a*27.0/(20.0*L) +
              C*3*L/140.0 - B*3.0/10.0 , -a*189.0/(40.0*L) - C*33*L/560.0 +
              B*57/80.0, a*37.0/(10.0*L) - C*8*L/105.0 - B*1.0/2.0}

   };

    // Локальный вектор нагрузок (дополнительные слагаемые для первого и последнего
        элементов учитываются далее)
    std::vector<double> local_b = { D * L / 8.0,
                                    D*3.0 * L / 8.0,
                                    D*3.0 * L / 8.0,
                                    D * L / 8.0 };



    // Производим матричные преобразования для обнуления элементов локальной
        матрицы жесткости, относящихся к внутренним узлам
    for (size_t i = 1; i < 3; i++) {
        for (size_t j = 0; j < 4; j++) {
            if (std::fabs(local_matrix.at(j).at(i)) > EPS && i != j) {
                double val = local_matrix.at(j).at(i) /local_matrix.at(i).at(i);
                local_b.at(j) -= val * local_b.at(i);
                for (size_t k = 0; k < 4; k++) {
                    local_matrix.at(j).at(k) -= val *local_matrix.at(i).at(k);
                }
            }
            continue;
        }
```

```cpp
    }


    // Исключаем внутренние узлы из рассмотрения
    std::vector< std::vector<double> > local_matrix_mod = { { local_matrix.at(0).at(0), local_matrix.at(0).at(3) },
                                                            { local_matrix.at(3).at(0), local_matrix.at(3).at(3) } };
    std::vector<double> local_b_mod = { local_b.at(0),
                                        local_b.at(3)};

    // Ансамблирование и получение глобальной матрицы жесткости для кубического КЭ
    for (size_t i = 0; i < elems_num; i++) {
        for (size_t j = 0; j < 2; j++) {
            for (size_t k = 0; k < 2; k++) {
                A.at(i + j).at(i + k) += local_matrix_mod.at(j).at(k);
            }
        }
    }
        for (size_t i = 0; i < elems_num; i++) {
        b.at(i) += local_b_mod.at(0);
        b.at(i+1) += local_b_mod.at(1);
    }
    // Учет ГУ
    b.at(0) = usl_left;
    A.at(0).at(0) = 1;
    A.at(0).at(1) = 0;

    b.at(size - 1) = local_b_mod.at(1) + a * usl_right;

    // Решение полученной СЛАУ методом Гаусса
    std::vector<double> res = solve_with_gauss(A, b);
    return res;
}

double calc_abs_error(const std::vector<double>& y_real, const std::vector<double>& y) {
    double max_err = 0.0;
    for (size_t i = 0; i < y_real.size(); i++) {
        double err = std::fabs(y_real.at(i) - y.at(i));
        if (err > max_err) {
            max_err = err;
        }
    }
    return max_err;
```

```
}

int main() {
#ifdef OSN

    std::vector<double> x(ELEMS_NUM + 1);
    for (size_t i = 0; i < x.size(); i++) {
        x.at(i) = X_BEGIN + i * L;
    }
    size_t x_size = x.size();


#ifdef CUBE
    std::vector<double> y = build_cube_solution(ELEMS_NUM);
#else
    std::vector<double> y = build_linear_solution(ELEMS_NUM);
#endif
    std::vector<double> y_real = build_analytical_solution(x);

#ifdef TABLE
    for (size_t i = 0; i < x.size(); i++) {
        std::cout<<x.at(i)<<" & "<<y_real.at(i)<<" &"<<y.at(i)<<"
            &"<<std::fabs(y_real.at(i) - y.at(i))<<"\\\\"<<std::endl;
    }
#endif

    FILE* gp = popen("gnuplot -persist", "w");
    fprintf(gp, "$predict << EOD\n");
    for (size_t i = 0; i < x_size; i++) {
        fprintf(gp, "%lf %lf\n", x.at(i), y.at(i));
    }
    fprintf(gp, "EOD\n");
    fprintf(gp, "$real << EOD\n");
    for (size_t i = 0; i < x_size; i++) {
        fprintf(gp, "%lf %lf\n", x.at(i), y_real.at(i));
    }
    fprintf(gp, "EOD\n");
    fprintf(gp, "set grid\n");
     fprintf(gp, "plot '$predict' using 1:2 with lp lc '#ba55d3' lw 1.5 pt 7 ps 0.5 title
         МКЭрешение'- (%zu КЭ)'," "'$real' using 1:2 with lines lc rgb '#afdafc' lt 1 lw 2
         title аналитическое' решение(%zu КЭ)',\n", ELEMS_NUM, ELEMS_NUM);
     printf("Абсолютная погрешность: %e\n", calc_abs_error(y_real, y));


    //нахождение количества линейных КЭ
#else
```

```cpp
    int N=20,n=10000;
    double err=10;
    FILE* gp = popen("gnuplot -persist", "w");
    fprintf(gp, "$predict << EOD\n");
    while (err>CUB && n<=19000){

        double L = (X_END - X_BEGIN) / N;
        std::vector<double> x(N + 1);
        for (size_t i = 0; i < x.size(); i++) {
            x.at(i) = X_BEGIN + i * L;
        }

        std::vector<double> y_r(N + 1);
        std::vector<double> y_s(N + 1);

        y_s = build_linear_solution(N);
        y_r = build_analytical_solution(x);

        err=calc_abs_error(y_r, y_s);

        fprintf(gp, "%d %e\n", N, err);
        printf("Абсолютная погрешность: %e количествоКЭ: %d\n", calc_abs_error(y_r, y_s), N);
        N+=1;
        n+=1;
    }

    fprintf(gp, "EOD\n");
    fprintf(gp, "set grid\n");
    fprintf(gp, "set logscale y 2\n");
    fprintf(gp, "plot '$predict' using 1:2 with lp lc '#cd853f' lw 1.5 pt 7 ps 0.5 title 'Абсолютная погрешность',\n" );
    std::cout<<"Количество линейныхКЭ"<<N-1<<std::endl;
    printf("Абсолютная погрешность: %e\n", err);

#endif
    return 0;
}
```

8 Вывод

В ходе выполнения лабораторной работы был реализован МКЭ для различных функций форм, а также найдено количество линейных КЭ обеспечивающих точность 20ти кубических КЭ.

Постановка:	© *доцент кафедры РК-6, кандидат технических наук, доцент, Трудоношин В.А.*
Решение и вёрстка:	© *студент группы РК6-75б, Крылов А. С.*

2023, осенний семестр