Московский государственный университет имени М. В.
Ломоносова
Механико-математический факультет
Практикум по ЭВМ

Отчёт 1. Вариант 15

«Решение краевых задач методом конечных элементов»

Работу подготовила:
Кочергина Ксения Михайловна
405 группа кафедры МаТИС

Семинарист:
Безяев Владимир Иванович
доцент кафедры Выч. мат.

Москва, 2025

Содержание

1 Постановка задачи

Необходимо аппроксимировать следующую задачу с помощью метода конечных элементов (кусочно-линейных) и найти решение полученной системы алгебраических уравнений при различных h, f и α:

$$-(ku')' + 4u = f(x), \quad 0 < x < 1,$$

с граничными условиями:

$$u(0) = u(1) = 0,$$

где

$$k(x) = e^{\alpha x}.$$

2 Метод конечных элементов

2.1 Слабая форма

Умножим уравнение на тестовую функцию $v \in H_0^1(0,1)$ и проинтегрируем по частям:

$$\int_0^1 ku'v'dx + 4\int_0^1 uvdx = \int_0^1 fvdx.$$

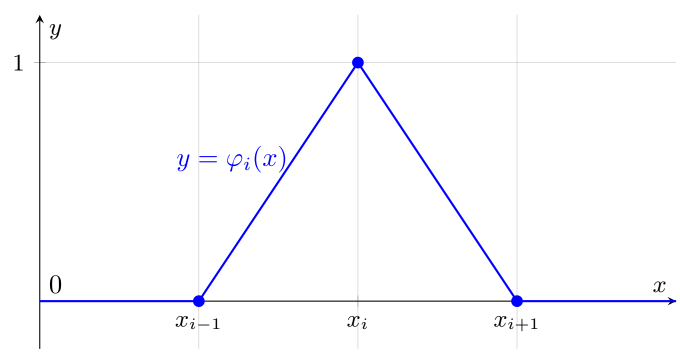
2.2 Конечно-элементная аппроксимация

Введём равномерную сетку:

$$x_i = ih, \quad i = 0, 1, \ldots, N, \quad h = \frac{1}{N}.$$

Используем кусочно-линейные базисные функции $\varphi_i(x)$, такие что:

$$\varphi_i(x_j) = \delta_{ij}, \quad \mathrm{supp}(\varphi_i) = [x_{i-1}, x_{i+1}]$$



Приближённое решение ищем в виде:

$$u_h(x) = \sum_{j=1}^{N-1} c_j \varphi_j(x)$$

2.3 Система уравнений

Подставим u_h в слабую форму и выберем $v = \varphi_i$. Получим СЛАУ:

$$\sum_{j=1}^{N-1} c_j \left[\int_0^1 k\varphi_j'\varphi_i' dx + 4\int_0^1 \varphi_j\varphi_i dx\right] = \int_0^1 f\varphi_i dx, \quad i = 1, \ldots, N-1$$

Обозначим:

$$A_{ij} = \int_0^1 k\varphi_j'\varphi_i' dx + 4\int_0^1 \varphi_j\varphi_i dx, \quad b_i = \int_0^1 f\varphi_i dx$$

Тогда система принимает вид:

$$A\mathbf{c} = \mathbf{b}$$

Матрица A трёхдиагональная, вычислим её ненулевые элементы:

$$A_{ii} = \frac{1}{h^2}\left[\int_{x_{i-1}}^{x_i} k(x)dx + \int_{x_i}^{x_{i+1}} k(x)dx\right] + 4 \cdot \frac{2h}{3},$$
$$A_{i,i+1} = A_{i+1,i} = -\frac{1}{h^2}\int_{x_i}^{x_{i+1}} k(x)dx + 4 \cdot \frac{h}{6}$$

Вычислим правую часть:

$$b_i = \int_{x_{i-1}}^{x_{i+1}} f(x)\varphi_i(x)dx = \int_{x_{i-1}}^{x_i} f(x)\frac{x - x_{i-1}}{h}dx + \int_{x_i}^{x_{i+1}} f(x)\frac{x_{i+1} - x}{h}dx$$

2.3.1 При $\alpha \neq 0$

Для $k(x) = e^{\alpha x}$ при $\alpha \neq 0$ получаем:

$$\int_{x_{i-1}}^{x_i} k(x)dx = \frac{e^{\alpha x_i} - e^{\alpha x_{i-1}}}{\alpha}, \quad \int_{x_i}^{x_{i+1}} k(x)dx = \frac{e^{\alpha x_{i+1}} - e^{\alpha x_i}}{\alpha}$$

и элементы матрицы равны:

$$A_{ii} = \frac{1}{h^2\alpha}(e^{\alpha x_{i+1}} - e^{\alpha x_{i-1}}) + \frac{8h}{3}$$

$$A_{i,i+1} = A_{i+1,i} = -\frac{1}{h^2\alpha}(e^{\alpha x_{i+1}} - e^{\alpha x_i}) + \frac{2h}{3}$$

2.3.2 При $\alpha = 0$

При $\alpha = 0$ $k(x) = 1$ и

$$\int_{x_{i-1}}^{x_i} k(x)dx = x_i - x_{i-1}, \quad \int_{x_i}^{x_{i+1}} k(x)dx = x_{i+1} - x_i$$

Диагональные элементы: $A_{ii} = \frac{2}{h} + \frac{8h}{3}$
Внедиагональные элементы: $A_{i,i+1} = A_{i+1,i} = -\frac{1}{h} + \frac{2h}{3}$

$$A = \frac{1}{h}\begin{pmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & -1 & 2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{pmatrix} + \frac{2h}{3}\begin{pmatrix} 4 & 1 & 0 & \cdots & 0 \\ 1 & 4 & 1 & \ddots & \vdots \\ 0 & 1 & 4 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & 0 & 1 & 4 \end{pmatrix}$$

2.4 Алгоритм решения

1. Задать параметры: h, α, функцию $f(x)$
2. Построить равномерную сетку: $x_i = ih$, $i = 0, 1, \ldots, N$, где $h = 1/N$
3. Вычислить элементы матрицы A по приведённым формулам
4. Вычислить элементы вектора правой части b_i с помощью квадратурных формул
5. Решить трёхдиагональную систему $A\mathbf{c} = \mathbf{b}$ методом прогонки
6. Построить приближённое решение $u_h(x) = \sum\limits_{j=1}^{N-1} c_j \varphi_j(x)$

3 Результаты

3.1 пример 1: $f = 0$

3.1.1 Аналитическое решение для $\alpha = 0$

Для $f = 0$ и $\alpha = 0$ уравнение упрощается до:

$$-u'' + 4u = 0, \quad 0 < x < 1$$

Характеристическое уравнение: $-r^2 + 4 = 0 \Rightarrow r = \pm 2$
Общее решение: $u(x) = C_1 e^{2x} + C_2 e^{-2x}$
С учетом граничных условий:

$$\begin{cases} u(0) = C_1 + C_2 = 0, \\ u(1) = C_1 e^2 + C_2 e^{-2} = 0 \end{cases} \quad \Rightarrow \quad C_1 = C_2 = 0$$

Аналитическое решение: $u(x) = 0$

3.1.2 Случай $\alpha \neq 0$

Уравнение преобразуется к виду:

$$-(e^{\alpha x} u')' + 4u \quad = \quad -e^{\alpha x} u'' - \alpha e^{\alpha x} u' + 4u \quad = \quad 0 \quad | \cdot (-e^{-\alpha x})$$

$$u'' + \alpha u' - 4e^{-\alpha x} u = 0$$

С помощью замены переменной $t = e^{-\alpha x}$ уравнение сводится к:

$$\frac{d^2 u}{dt^2} - \frac{4}{\alpha^2 t} u = 0$$

Далее, замена $s = \ln t$ преобразует уравнение в линейное уравнение с постоянными коэффициентами:

$$u_{ss} - u_s - \frac{4}{\alpha^2} u = 0$$

Характеристическое уравнение:

$$r^2 - r - \frac{4}{\alpha^2} = 0 \quad \Rightarrow \quad r = \frac{\alpha \pm \sqrt{\alpha^2 + 16}}{2\alpha}$$

Общее решение относительно s (где $s = -\alpha x$): $u(s) = Ae^{r_1 s} + Be^{r_2 s}$

$$u(x) = A \exp\left(-\frac{\alpha + \sqrt{\alpha^2 + 16}}{2} x\right) + B \exp\left(-\frac{\alpha - \sqrt{\alpha^2 + 16}}{2} x\right)$$

Применим граничные условия:

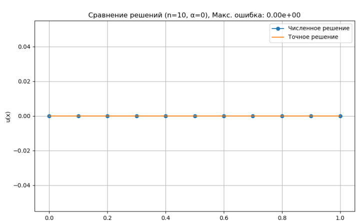
$$u(0) = A + B = 0 \quad \Rightarrow \quad B = -A$$

$$u(1) = A\left[\exp\left(-\frac{\alpha+\sqrt{\alpha^2+16}}{2}\right) - \exp\left(-\frac{\alpha-\sqrt{\alpha^2+16}}{2}\right)\right] =$$

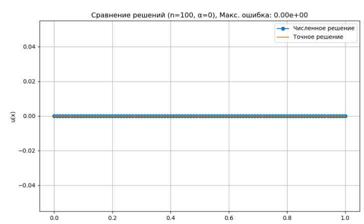
$$= Ae^{-\alpha/2}\left[\exp\left(\frac{\sqrt{\alpha^2+16}}{2}\right) - \exp\left(-\frac{\sqrt{\alpha^2+16}}{2}\right)\right] = 0 \quad \Rightarrow \quad \begin{cases} A = 0, \\ B = 0 \end{cases}$$

Для всех значений α единственным решением краевой задачи является тривиальное решение: $u(x) = 0$.

3.1.3 Графики

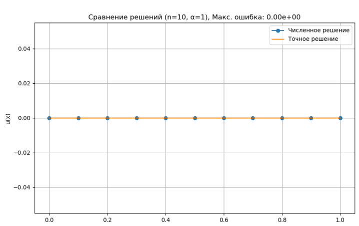


(a) $n = 10$

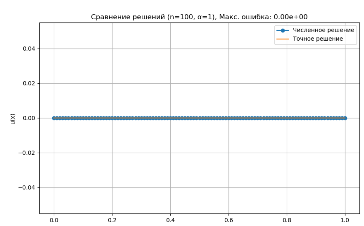


(b) $n = 100$

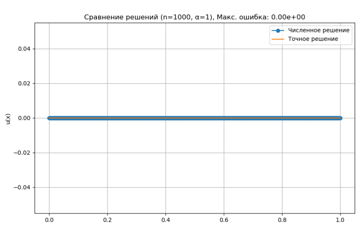
Рис. 1: $f(x) = 0, \alpha = 0$



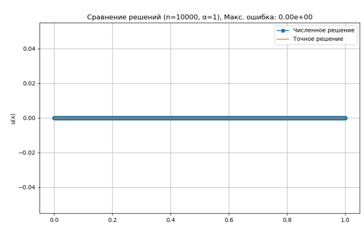
(a) $n = 10$



(b) $n = 100$



(c) $n = 1000$



(d) $n = 10000$

Рис. 2: $f(x) = 0, \alpha = 1$

3.2 пример 2: $f = 1, \alpha = 0$

Для $f = 1$ и $\alpha = 0$ уравнение упрощается до:

$$-u'' + 4u = 1, \quad 0 < x < 1,$$

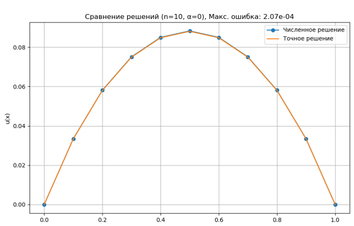
Общее решение неоднородного уравнения:

$$u(x) = C_1 e^{2x} + C_2 e^{-2x} + \frac{1}{4}$$

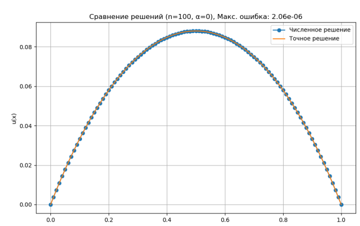
С учетом граничных условий получаем систему уравнений:

$$\begin{cases} u(0) = C_1 + C_2 + \dfrac{1}{4} = 0, \\ u(1) = C_1 e^2 + C_2 e^{-2} + \dfrac{1}{4} = 0 \end{cases} \Rightarrow \begin{cases} C_1 & = \frac{e^{-2}-1}{4(e^2-e^{-2})}, \\ C_2 & = \frac{1-e^2}{4(e^2-e^{-2})} \end{cases}$$

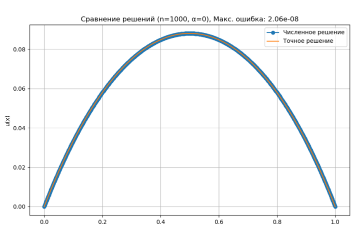
Аналитическое решение: $u(x) = \frac{e^{-2}-1}{4(e^2-e^{-2})} e^{2x} + \frac{1-e^2}{4(e^2-e^{-2})} e^{-2x} + \frac{1}{4}$



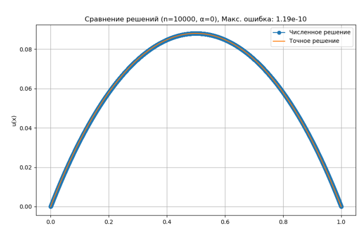
(a) $n = 10$



(b) $n = 100$



(c) $n = 1000$



(d) $n = 10000$

Рис. 3: $f(x) = 1, \alpha = 0$

3.3 пример 3: $f = 1, \alpha \neq 0$

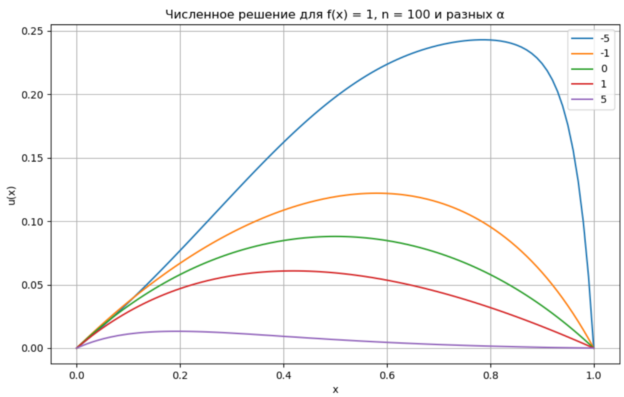


Рис. 4: График зависимости от параметра α для $f(x) = 1, h = 0.01$

3.4 пример 4: $f = \sin(\pi x), \alpha = 0$

Для $f(x) = \sin(\pi x)$ и $\alpha = 0$:

$$-u'' + 4u = \sin(\pi x)$$

Частное решение ищем в виде:

$$u_p = A\sin(\pi x) + B\cos(\pi x)$$

Подставляем в уравнение:

$$\pi^2 A\sin(\pi x) + \pi^2 B\cos(\pi x) + 4A\sin(\pi x) + 4B\cos(\pi x) = \sin(\pi x)$$

Приравниваем коэффициенты:

$$\begin{cases} (\pi^2+4)A = 1 \\ (\pi^2+4)B = 0 \end{cases} \Rightarrow \begin{cases} A = \frac{1}{(\pi^2+4)}, \\ B = 0 \end{cases}$$

Частное решение:

$$u_p = \frac{\sin(\pi x)}{\pi^2+4}$$

Общее решение:

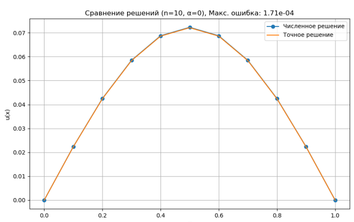
$$u(x) = C_1 e^{2x} + C_2 e^{-2x} + \frac{\sin(\pi x)}{\pi^2+4}$$

С учетом граничных условий:

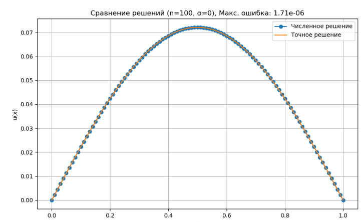
$$\begin{cases} u(0) = C_1 + C_2 = 0 \\ u(1) = C_1 e^2 + C_2 e^{-2} + \frac{\sin(\pi)}{\pi^2+4} = C_1 e^2 + C_2 e^{-2} = 0 \end{cases}$$

Решение системы:

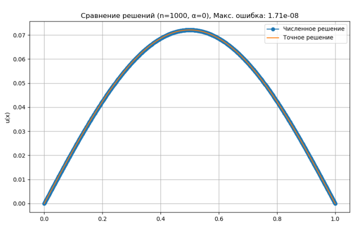
$$C_1 = C_2 = 0 \quad \Rightarrow \quad u(x) = \frac{\sin(\pi x)}{\pi^2 + 4}$$



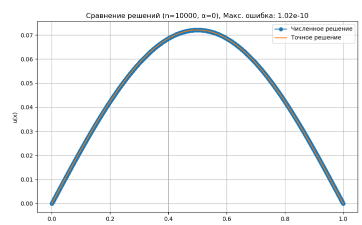
(a) $n = 10$



(b) $n = 100$



(c) $n = 1000$



(d) $n = 10000$

Рис. 5: $f(x) = \sin(\pi x), \alpha = 0$

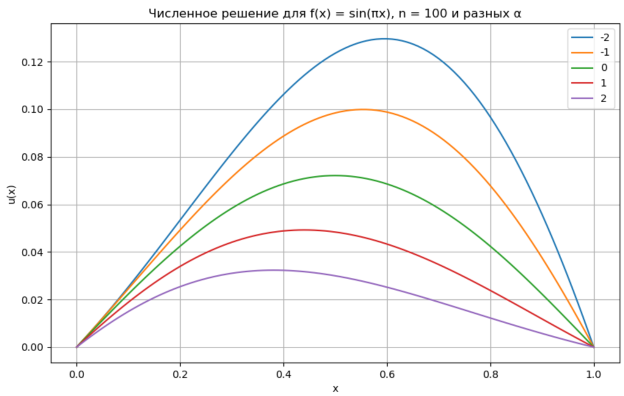


Рис. 6: График зависимости от параметра α для $f(x) = \sin(\pi x), h = 0.01$

3.5 пример 5: $f = \sin(7\pi x), \alpha = 0$

Будем искать частное решение в виде:

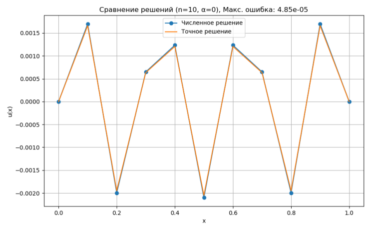
$$u_p(x) = Asin(7x) + Bcos(7x).$$

Вычислим производные и подставим u_p в уравнение:

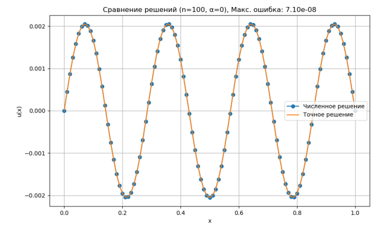
$$-\left(-49^2 Asin(7x) - 49^2 Bcos(7x)\right) + 4\left(Asin(7x) + Bcos(7x)\right) = sin(7x)$$

$$\begin{cases} (49\pi^2 + 4)A = 1, \\ (49\pi^2 + 4)B = 0 \end{cases} \Rightarrow \begin{cases} A = \frac{1}{(49\pi^2+4)}, \\ B = 0 \end{cases}$$

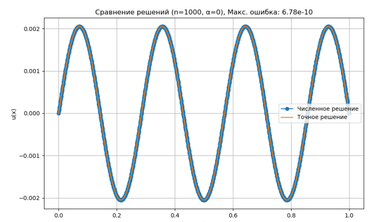
Итак, частное решение: $u_p(x) = \frac{sin(7x)}{(49\pi^2+4)}$



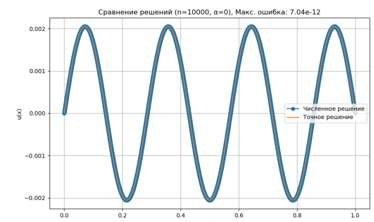
(a) $n = 10$



(b) $n = 100$



(c) $n = 1000$



(d) $n = 10000$

Рис. 7: $f(x) = \sin(7\pi x), \alpha = 0$

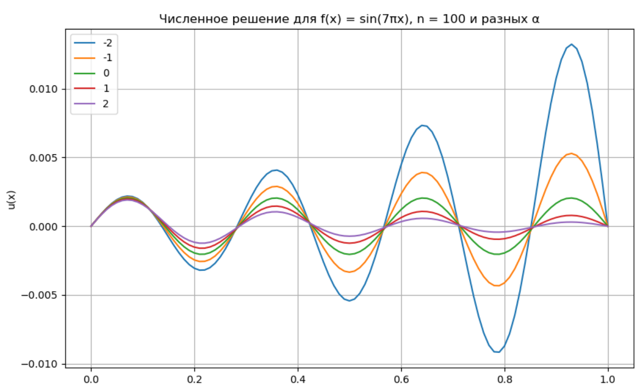


Рис. 8: График зависимости от параметра α для $f(x) = \sin(7\pi x), h = 0.01$

4 Физический смысл

Уравнение может описывать **упругие поперечные колебания** струны переменной плотности:

$u(x)$ - отклонение струны от положения равновесия

$k(x) = exp(\alpha x)$ - коэффициент натяжения струны

Член $4u$ представляет упругое основание (Винклеровский фундамент)

$f(x)$ - внешняя нагрузка

Также уравнение описывает **стационарное распределение температуры** в стержне **с неоднародной теплопроводностью**:

$u(x)$ - температура в точке

$k(x) = exp(\alpha x)$ - коэффициент теплопроводности, который экспоненциально изменяется вдоль стержня: при $\alpha > 0$ теплопроводность увеличивается вдоль стержня, при $\alpha < 0$ - уменьшается

Член $4u$ представляет теплообмен с окружающей средой (конвекцию)

$f(x)$ - плотность источников тепла

5 Выводы

Метод конечных элементов хорошо справляется с решением задачи для различных n, α и $f(x)$. Переменный коэффициент $k(x)$ существенно влияет на форму решения, что соответствует физическому смыслу: в областях с высокой теплопроводностью температура более выровнена, а в областях с низкой теплопроводностью возможны большие перепады температур.

При увеличении числа элементов n численное решение сходится к точному, где оно известно, и становится более гладким. Для случаев, когда аналитическое решение неизвестно, метод конечных элементов является эффективным инструментом для получения приближённого решения.

6 Приложения

6.1 Репозиторий кода

$github.com/ksenia - kochergina/MSU\_term\_4$

6.2 Метод прогонки

Для решения трёхдиагональной системы используется метод прогонки (алгоритм Томаса):

Прямой ход:

$$
\begin{aligned}
C_1 &= \frac{A_{12}}{A_{11}} \\
D_1 &= \frac{b_1}{A_{11}} \\
C_i &= \frac{A_{i,i+1}}{A_{ii} - A_{i,i-1}C_{i-1}} \\
D_i &= \frac{b_i - A_{i,i-1}D_{i-1}}{A_{ii} - A_{i,i-1}C_{i-1}}
\end{aligned}
$$

Обратный ход:

$$
\begin{aligned}
x_{N-1} &= D_{N-1} \\
x_i &= D_i - C_i x_{i+1}
\end{aligned}
$$

6.3 Квадратурный метод Гаусса

Для нахождения правой части системы уравнений, необходимо найти интегралы от $f(x)$, численно это возможно сделать с помощью квадратурного метода Гаусса, который аппроксимирует интеграл функции как взвешенную сумму значений функции в специально выбранных точках x_i (узлах) с весами w_i:

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i)$$

1. Определение узлов и весов

Здесь используется квадратура Гаусса с 2 узлами на стандартном интервале [-1, 1]. Узлы расположены в точках $\pm 1/\sqrt{3}$, а веса равны 1 для каждого узла. Эта квадратура точна для полиномов степени до 3, что достаточно для линейных элементов.

2. Преобразование координат

Поскольку квадратура определена на интервале [-1, 1], а наши элементы находятся на [left_node, right_node], мы выполняем афинное преобразование координат:

$$x(\xi) = \frac{\text{left} + \text{right}}{2} + \frac{h}{2}\xi$$

где ξ - координата в стандартном интервале [-1, 1], x - координата в физическом элементе. Якобиан этого преобразования равен $h/2$, поэтому вес умножается на этот множитель.

3. Вычисление базисных функций

Для линейных элементов на отрезке [a, b] базисные функции имеют вид:

$$\phi_0(x) = \frac{b-x}{b-a}, \quad \phi_1(x) = \frac{x-a}{b-a}$$

Их производные постоянны на каждом элементе:

$$\phi_0'(x) = -\frac{1}{h}, \quad \phi_1'(x) = \frac{1}{h}$$

4. Вычисление вкладов

- Элементы вектора f соответствуют линейной форме:

$$f(\phi_i) = \int f(x)\phi_i dx$$

6.4 Листинг программы

```
import numpy as np

def method_finite_element_func(n, alpha, f):
    h = 1.0 / n
    x_nodes = np.linspace(0, 1, n+1)
    A = np.zeros((n-1, n-1))
    b = np.zeros(n-1)

    for i in range(0, n-1):
        x_i = x_nodes[i+1]
        x_im1 = x_nodes[i]
        x_ip1 = x_nodes[i+2]

        if alpha == 0:
            A[i, i] = 2.0 / h + 8 * h / 3
        else:
            A[i, i] = (np.exp(alpha * x_ip1) - np.exp(alpha * x_im1)) / (h**2 * alpha) + 8 * h / 3

        if i < n-2:
            if alpha == 0:
                A[i, i+1] = -1.0 / h + 2 * h / 3
                A[i+1, i] = -1.0 / h + 2 * h / 3
            else:
                A[i, i+1] = -(np.exp(alpha * x_ip1) - np.exp(alpha * x_i)) / (h**2 * alpha) + 2 * h / 3
                A[i+1, i] = -(np.exp(alpha * x_ip1) - np.exp(alpha * x_i)) / (h**2 * alpha) + 2 * h / 3

    gauss_points = [-1/np.sqrt(3), 1/np.sqrt(3)]
    weights = [1, 1]

    for i in range(n):
        left_node = x_nodes[i]
        right_node = x_nodes[i+1]
        length = right_node - left_node

        f_local = np.zeros(2)

        for point, weight in zip(gauss_points, weights):
            xi = (left_node + right_node)/2 + (length/2)*point
            w = weight * (length/2)

            phi0 = (right_node - xi) / length
            phi1 = (xi - left_node) / length

            f_val = f(xi)

            f_local[0] += w * f_val * phi0
            f_local[1] += w * f_val * phi1

```

```
        if i > 0:
            b[i-1] += f_local[0]
        if i < n-1:
            b[i] += f_local[1]

    return A, b, x_nodes
```

Листинг 1: Реализация метода конечных элементов на Python

```
import numpy as np

def thomas_algorithm(A, b):
    n = len(b)

    main_diag = np.diag(A).astype(float)
    upper_diag = np.diag(A, k=1).astype(float)
    lower_diag = np.diag(A, k=-1).astype(float)

    c_prime = np.zeros(n-1)
    d_prime = np.zeros(n)

    c_prime[0] = upper_diag[0] / main_diag[0]
    d_prime[0] = b[0] / main_diag[0]

    for i in range(1, n-1):
        denominator = main_diag[i] - lower_diag[i-1] * c_prime[i-1]
        c_prime[i] = upper_diag[i] / denominator
        d_prime[i] = (b[i] - lower_diag[i-1] * d_prime[i-1]) / denominator

    denominator = main_diag[n-1] - lower_diag[n-2] * c_prime[n-2]
    d_prime[n-1] = (b[n-1] - lower_diag[n-2] * d_prime[n-2]) / denominator

    x = np.zeros(n)
    x[n-1] = d_prime[n-1]

    for i in range(n-2, -1, -1):
        x[i] = d_prime[i] - c_prime[i] * x[i+1]

    return x
```

Листинг 2: Реализация метода прогонки на Python

Список литературы

[1] Г. И. Марчук, В.И. Агошков. Введение в проекционно-свёрточные методы. Наука, Москва, 1981

[2] Ортега Дж., Пул У. Введение в численные методы решения дифференциальных решений. Наука, Москва, 1986

[3] Богачев К. Ю. Методы решения линейных систем и нахождения собственных значений. Труды механико-математического факультета МГУ, издательство мехмата МГУ, Москва, 1998