OOPython

Задача 8. Метод конечных элементов: решение ОДУ

Мотивация

Метод конечных элементов и метод конечных объемов являются наиболее используемыми методами для решения промышленных задач (в связи со сложной геометрией расчетной 3D-области последних).

Элементы теории

Обозначения

- e – функционал, $e[y]$ – значение функционала на функции y
- y – функция, $y(x)$ – значение функции на числе $x \in \mathbb{R}$.

Постановка задачи:

$$\begin{gathered} -u'' = f,\ x \in I = (0,1) \\ u(0) = u(1) = 0. \end{gathered} \tag{1}$$

В **Задаче 7** мы обсуждали применение метода МКЭ к аппроксимации *аналитически заданной* функции u; аппроксимант мы искали в виде кусочно-линейной функции u_n, которую возможно разложить по базису из функций-«шляпок» $\{\varphi_j\}$:

$$u_n = \sum_{j=0}^{n} \alpha_j \varphi_j \,. \tag{2}$$

С учетом того, что решение на границах отрезка принимает нулевые значения, коэффициенты при функциях-«полушляпках» φ_0, φ_n должны быть равны нулю:

$$u_n = \sum_{j=1}^{n-1} \alpha_j \varphi_j \,. \tag{3}$$

Коэффициенты разложения по базису мы искали исходя из манипуляций со значением функционала-ошибки $e[y] \equiv u - y$ на функции u_n. Рассматривались следующие методы нахождения коэффициентов:

Название метода	Формула
Интерполяция	$e[u_n](x_i; \alpha_1, \dots, \alpha_{n-1}) = 0, \qquad x_i \in$ точки интерполяции

Регрессия	$\sum_{i=0}^{m} e[u_n](x_i;\alpha_1,\dots,\alpha_{n-1})^2 = F_m(\alpha_1,\dots,\alpha_{n-1}) \to \min\limits_{\{\alpha_1,\dots,\alpha_{n-1}\}}$, $x_i \in$ точки регрессии
Наименьших квадратов	$\int_0^1 e[u_n](x;\alpha_1,\dots,\alpha_{n-1})^2 dx = F_\infty(\alpha_1,\dots,\alpha_{n-1}) \to \min\limits_{\{\alpha_1,\dots,\alpha_{n-1}\}}$
L_2-проекция	$(e[u_n], v) = 0, \forall v \in V_n,$ где $V_n = \left\{v: v = \sum_{j=1}^{n-1} \beta_j \varphi_j, v(0) = 0, v(1) = 0\right\}$.

Другими словами, V_n – пространство кусочно-линейных функций, принимающих нулевые значения на границах отрезка.

В случае с ОДУ мы не можем пользоваться функционалом e т.к. нам неизвестно точное решение u. Однако, вместо этого мы можем рассматривать *косвенную* меру ошибки – функционал-невязку r:

$$r[y] \equiv f + y''. \tag{4}$$

Можно заметить, что на точном решении u значение функционала-невязки $r[u] \equiv 0$ (свойство, также свойственное функционалу-ошибке e).

Все методы нахождения коэффициентов в случае решения ОДУ находятся из тех же принципов/манипуляций, что и для аппроксимации функций, только вместо $e[u_n]$ в формулы выше формально подставляют значение функционала-невязки $r[u_n]$:

Название метода	Формула	Название аналога метода
Коллокация	$r[u_n](x_i;\alpha_1,\dots,\alpha_{n-1}) = 0,$ $x_i \in$ точки коллокации	Интерполяция
Наименьших квадратов (дискретный)	$\sum_{i=0}^{m} r[u_n](x_i;\alpha_1,\dots,\alpha_{n-1})^2 = G_m(\alpha_1,\dots,\alpha_{n-1}) \to \min\limits_{\{\alpha_1,\dots,\alpha_{n-1}\}}$, $x_i \in$ точки регрессии	Регрессия
Наименьших квадратов	$\int_0^1 r[u_n](x;\alpha_1,\dots,\alpha_{n-1})^2 dx = G_\infty(\alpha_1,\dots,\alpha_{n-1}) \to \min\limits_{\{\alpha_1,\dots,\alpha_{n-1}\}}$	Наименьших квадратов
Галеркина	$(r[u_n], v) = 0, \quad \forall v \in V_n,$ где $V_n = \left\{v: v = \sum_{j=1}^{n-1} \beta_j \varphi_j, v(0) = 0, v(1) = 0\right\}$	L_2-проекция.

При этом результаты применений метода Галеркина и наименьших квадратов **не совпадают**, в отличие от задачи аппроксимации функций. Далее рассмотрим подробно метод Галеркина.

Метод Галеркина

Получение СЛАУ

Исходная формула метода:

$$\begin{aligned}(r_{u_n}, v) &= 0, \qquad \forall v \in V_n.\\ (f + u_n'', v) &= 0, \qquad \forall v \in V_n,\\ (u_n'', v) &= -(f, v), \qquad \forall v \in V_n.\end{aligned}$$

Преобразуем формулу с помощью интегрирования по частям:

$$(u_n'', v) \equiv \int_0^1 u_n''(x)v(x)dx = \overbrace{\cancel{u_n'(x)v(x)\Big|_0^1}}^{\text{граничные условия для } v} - \int_0^1 u_n'(x)v'(x)dx \equiv -(u_n', v').$$

Примечание: в случае кусочно-линейной функции мы не можем применять формулу интегрирования по частям, но в случае кусочно-полиномиальной степени $p > 1$ это уместно. Поэтому представим, что мы работаем с последней.

Получаем:

$$(u_n', v') = (f, v), \qquad \forall v \in V_n. \tag{5}$$

Введем обозначения:

- билинейная форма: $a[u, v] \equiv \int_0^1 u'(x)v'(x)dx$,
- линейная форма: $l[v] \equiv \int_0^1 f(x)v(x)dx$

и получим финальный вид записи:

$$a[u_n, v] = l[v], \qquad \forall v \in V_n. \tag{6}$$

Т.к. любую кусочно-линейную функцию v можно разложить по базису $v = \sum_{i=1}^{n-1} \beta_i \varphi_i(x)$, то равенство (6) эквивалентно выполнению условий (доказательство см. в теоретическом материале для **Задачи 7**):

$$a[u_n, \varphi_i] = l[\varphi_i], \qquad i = 1, \dots, n-1.$$

Подставим $u_n = \sum_{j=1}^{n-1} \alpha_j \varphi_j$ в предыдущее равенство:

$$a\left[\sum_{j=1}^{n-1} \alpha_j \varphi_j, \varphi_i\right] = l[\varphi_i], \qquad i = 1, \dots, n-1.$$

После раскрытия скобок получаем:

$$\sum_{j=1}^{n-1} \alpha_j \cdot a[\varphi_j, \varphi_i] = l[\varphi_i], \qquad i = 1, \dots, n-1. \tag{7}$$

Эта система равенств представляет собой СЛАУ:

$$\begin{gathered} S\boldsymbol{\alpha} = \boldsymbol{b} \\ S_{ij} = a[\varphi_j, \varphi_i] \\ b_i = l[\varphi_i] \\ i, j = 1, \dots, n-1 \end{gathered} \tag{8}$$

По историческим причинам матрица S именуется **матрицей жесткости,** а вектор правой части $\boldsymbol{b}$ – **вектором нагрузки.** Найдя решение $\boldsymbol{\alpha}$, подставим его в исходное разложение (3) – получим приближенное решение u_n.