ЖУРНАЛ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И МАТЕМАТИЧЕСКОЙ ФИЗИКИ, 2020, том 60, № 2, с. 234–252

УДК 519.63

БЫСТРЫЕ ФУРЬЕ-СОЛВЕРЫ ДЛЯ МКЭ ВЫСОКОГО ПОРЯДКА С ТЕНЗОРНЫМИ ПРОИЗВЕДЕНИЯМИ ДЛЯ УРАВНЕНИЯ ТИПА ПУАССОНА[1)]

© 2020 г. А. А. Злотник[1],*, И. А. Злотник[2],**

[1] *109028 Москва, Покровский б-р, 11, НИУ Высшая школа экономики, Россия;*

[2] *115419 Москва, 2-й Верхний Михайловский пр., 9, стр. 2, ЗАО РДК, Россия*

**e-mail: azlotnik@hse.ru*

***e-mail: ilya.zlotnik@gmail.com*

Поступила в редакцию 22.08.2019 г.
Переработанный вариант 22.08.2019 г.
Принята к публикации 17.10.2019 г.

Представлены прямые логарифмически оптимальные в теории и быстрые на практике алгоритмы реализации метода конечных элементов (МКЭ) на основе тензорных произведений 1D пространств МКЭ высокого порядка на многомерных прямоугольных параллелепипедах для решения уравнения типа Пуассона. Они основаны на хорошо известных фурье-подходах. Ключевыми новыми элементами являются детальное описание собственных пар 1D задач на собственные значения для МКЭ высокого порядка и быстрые прямой и обратный алгоритмы разложения по соответствующим собственным векторам, использующие одновременно несколько версий быстрого дискретного преобразования Фурье. Представлены результаты численных экспериментов в 2D и 3D случаях. Алгоритмы могут быть использованы для многочисленных приложений, в частности, для реализации методов конечных элементов высокого порядка с тензорными произведениями для различных эволюционных уравнений в частных производных, включая многомерные уравнение теплопроводности, волновое уравнение и уравнение Шрёдингера. Библ. 17. Фиг. 8. Табл. 6.

Ключевые слова: быстрый прямой алгоритм, метод конечных элементов высокого порядка, БДПФ, уравнение Пуассона.

DOI: 10.31857/S0044466920020143

ВВЕДЕНИЕ

В работе представлены быстрые прямые алгоритмы реализации метода конечных элементов (МКЭ) на основе тензорных произведений 1D пространств МКЭ n-го порядка ($n \geq 2$) на прямоугольных параллелепипедах [1] для решения N-мерного уравнения типа Пуассона $-\Delta u + \alpha u = f$ ($N \geq 2$) с краевыми условиями Дирихле. Алгоритмы основаны на хорошо известных Фурье-подходах (см., например, [2]–[5] и ссылки в них).

Ключевыми новыми элементами являются детальное описание собственных пар 1D задач на собственные значения для МКЭ высокого порядка и быстрые прямой и обратный алгоритмы разложения по соответствующим собственным векторам, использующие одновременно несколько версий БДПФ (быстрого дискретного преобразования Фурье), связанных не только с узлами сетки, но и с центрами элементов [6]. Это решает давно известную задачу (см., например, [5, с. 271]), и делает полные алгоритмы логарифмически оптимальными по отношению к числу элементов так же, как в случае полилинейных элементов ($n = 1$) или стандартных разностных схем. В краткой форме и в случае $N = 2$ алгоритмы представлены (без доказательств) в [7]. См. также методы другого типа для той же цели в [8], [9].

[1)]Статья подготовлена в ходе выполнения проекта № 19-01-021 в рамках Программы “Научный фонд НИУ ВШЭ” в 2019–2020 гг. и в рамках государственной поддержки ведущих университетов Российской Федерации “5-100”, а также проекта РФФИ № 19-01-00262.

Отметим, что постоянная α может быть вещественной любого знака или даже комплексной, так что оператор $-\Delta + \alpha$ может быть незнакоопределенным (в противоположность некоторым стандартным используемым итерационным методам).

Алгоритмы являются быстрыми с возрастанием количества элементов K и на практике (причем более быстрыми, чем при стандартном теоретическом анализе, вследствие векторной обработки данных в обычных современных процессорах) и демонстрируют только слабое снижение эффективности с ростом n, начиная со стандартного случая $n = 1$. Для примера, в случае элементов 9-го порядка 2D МКЭ-система уравнений для $K = 2^{20}$ элементов, содержащая почти 85×10^6 неизвестных, и 3D МКЭ-система для $K = 2^{18}$ элементов, содержащая более чем 190×10^6 неизвестных, решаются за менее чем 2 и 15 мин соответственно на ординарном ноутбуке с использованием кода на Matlab R2016a, детали см. ниже.

Алгоритмы могут быть использованы для многочисленных других приложений, включая общие эллиптические уравнения 2-го порядка (для предобуславливания), для N-мерных уравнения теплопроводности, волнового уравнения, нестационарного уравнения Шрёдингера(α может быть комплексным) и т.д., поскольку для их неявных дискретизаций по времени обычно получаются задачи рассматриваемого типа на верхнем слое по времени. В комбинации с другими методами они могут быть использованы для некоторых непрямоугольных областей и неравномерных сеток, в частности, с применением сеток, топологически эквивалентных равномерным прямоугольным [10]. Другие стандартные краевые условия также могут быть охвачены, см. краткое описание в [7]. Алгоритмы обладают высокой степенью параллелизуемости, так что они представляются полезными в научных вычислениях.

Подчеркнем также, что Фурье-структура алгоритмов является особенно ценной для решения некоторых задач волновой физики, в частности, использующих нелокальные граничные условия (см., например, [5], [11], [12]), откуда и возник наш собственный интерес.

Работа построена следующим образом. В разд. 1 даны формулировки стандартной 1D МКЭ-задачи на собственные значения и вспомогательных МКЭ-задач на собственные значения на эталонном элементе и внутри него. Основной разд. 2 посвящен описанию собственных значений и собственных векторов 1D МКЭ-задачи на собственные значения и быстрым прямому и обратному алгоритмам разложения по этим собственным векторам. Приложения к решению уравнения типа Пуассона в N-мерном прямоугольном параллелепипеде с краевым условием Дирихле описаны в разд. 3. Результаты численных экспериментов для $N = 2$ и 3 детально представлены в разд. 4; все они включают для сравнения известный случай $n = 1$.

1. ФОРМУЛИРОВКА 1D МКЭ ЗАДАЧИ НА СОБСТВЕННЫЕ ЗНАЧЕНИЯ

Сначала подробно рассмотрим МКЭ для простейшей 1D задачи на собственные значения для обыкновенного дифференциального уравнения (ОДУ)

$$-u''(x) = \lambda u(x) \quad \text{на} \quad [0, X], \quad u(0) = u(X) = 0, \quad u(x) \not\equiv 0. \tag{1.1}$$

Возьмем равномерную сетку $\overline{\omega}_h$ с узлами $x_j = jh$, $j = \overline{0, K}$ (т.е. $0 \le j \le K$) и шагом $h = X/K$. Пусть $H_h^{(n)}[0, X]$ – МКЭ-пространство кусочно-полиномиальных функций $\varphi \in C[0, X]$ таких, что $\varphi(x) \in \mathcal{P}_n$ при $x \in [x_{j-1}, x_j]$, $j = \overline{1, K}$, с $\varphi(0) = \varphi(X) = 0$; здесь $\mathcal{P}_n$ – пространство многочленов степени не выше n, $n \ge 2$.

Пусть $S_K^{(n)}$ – пространство вектор-функций w таких, что $w_j \in R$ при $j = \overline{0, K}$ с $w_0 = w_K = 0$ и $w_{j-1/2} \in \mathbb{R}^{n-1}$, $j = \overline{1, K}$. Ясно, что $\dim S_K^{(n)} = nK - 1$. Функция $\varphi \in H_h^{(n)}[0, X]$ однозначно определяется своими значениями в узлах сетки $\varphi_j = \varphi(x_j)$, $j = \overline{0, K}$, с $\varphi_0 = \varphi_K = 0$, и внутри элементов $\varphi_{j-1/2} = \{\varphi(x_{j-1} + (l/n)h)\}_{l=1}^{n-1}$, $j = \overline{1, K}$, которые формируют элемент из $S_K^{(n)}$.

Воспользуемся следующей масштабированной операторной формой стандартной МКЭ-дискретизации задачи (1.1)

$$\mathcal{A}v = \lambda \mathcal{C}v, \quad v \in S_K^{(n)}, \quad v \ne 0. \tag{1.2}$$

Здесь $\mathscr{A} = \mathscr{A}^{\mathrm{T}} > 0$ и $\mathscr{C} = \mathscr{C}^{\mathrm{T}} > 0$ – глобальные (масштабированные) операторы (матрицы) жесткости и масс, действующие в $S_K^{(n)}$ и вместе с λ *не зависящие от* h; истинные приближенные собственные значения – это $\lambda_h = 4h^{-2}\lambda$.

Пусть $A = \{A_{kl}\}_{k,l=0}^{n}$ и $C = \{C_{kl}\}_{k,l=0}^{n}$ – локальные матрицы жесткости и масс, связанные с эталонным элементом $\sigma_0 = [-1,1]$, со следующими элементами

$$A_{kl} = \int\limits_{\sigma_0} e_k'(x)e_l'(x)dx, \quad C_{kl} = \int\limits_{\sigma_0} e_k(x)e_l(x)dx,$$

где $\{e_l\}_{l=0}^{n}$ – лагранжев базис в $\mathscr{P}_n$ такой, что $e_l(-1+(2k)/n) = \delta_{kl}$ при $k,l = \overline{0,n}$, а δ_{kl} – символ Кронекера. Матрицы A, C и соответствующий матричный пучок $G(\lambda) := A - \lambda C$ имеют следующую 3×3-блочную форму

$$A = \begin{pmatrix} a_0 & a^{\mathrm{T}} & a_n \\ a & \tilde{A} & \check{a} \\ a_n & \check{a}^{\mathrm{T}} & a_0 \end{pmatrix}, \quad C = \begin{pmatrix} c_0 & c^{\mathrm{T}} & c_n \\ c & \tilde{C} & \check{c} \\ c_n & \check{c}^{T} & c_0 \end{pmatrix}, \quad G(\lambda) = \begin{pmatrix} g_0(\lambda) & g^{\mathrm{T}}(\lambda) & g_n(\lambda) \\ g(\lambda) & \tilde{G}(\lambda) & \check{g}(\lambda) \\ g_n(\lambda) & \check{g}(\lambda) & g_0(\lambda) \end{pmatrix}. \tag{1.3}$$

Здесь $\tilde{A}$, $\tilde{C}$ и $\tilde{G}(\lambda) = \tilde{A} - \lambda\tilde{C}$ – квадратные матрицы порядка $n-1$ с векторами-столбцами $a, c, g(\lambda) = a - \lambda c \in \mathbb{R}^{n-1}$, а $\check{p}_l \equiv (Pp)_l = p_{n-l}$, $l = \overline{1,n-1}$ для $p \in \mathbb{R}^{n-1}$. Матрицы A, C и $G(\lambda)$ являются *бисимметричными* (т.е. симметричными по отношению к их основной и побочной диагоналям).

Отметим, что $P_{ij} = \delta_{i(n-j)}$ и $P^{\mathrm{T}} = P^{-1} = P$. Пусть $\mathbb{R}_e^{n-1}$ и $\mathbb{R}_o^{n-1}$ – подпространства четных и нечетных векторов в $\mathbb{R}^{n-1}$, т.е. таких, что соответственно $Pp = p$ и $Pp = -p$. Разложение $\mathbb{R}^{n-1} = \mathbb{R}_e^{n-1} \oplus \mathbb{R}_o^{n-1}$ (при $n \geq 3$) реализуется формулами

$$p = p_e + p_o, \quad p_e := 0.5(p + \check{p}), \quad p_o := 0.5(p - \check{p}). \tag{1.4}$$

Отметим, что $\dim \mathbb{R}_e^{n-1} = [n/2]$ и $\dim \mathbb{R}_o^{n-1} = [(n-1)/2]$, с $\mathbb{R}_o^{n-1} = \{0\}$ при $n = 2$. Ясно, что

$$\check{p}\cdot q = p \cdot \check{q}, \quad \check{p} \cdot \check{q} = p \cdot q \quad \text{для всех} \quad p,q \in \mathbb{R}^{n-1}. \tag{1.5}$$

Здесь и ниже символ $\cdot$ означает скалярное произведение векторов в $\mathbb{R}^{n-1}$.

Теперь задачу (1.2) можно переписать в следующей явной форме

$$g_n(\lambda)v_{j-1} + \check{g}(\lambda)\cdot v_{j-1/2} + 2g_0(\lambda)v_j + g(\lambda)\cdot v_{j+1/2} + g_n(\lambda)v_{j+1} = 0, \quad j = \overline{1,K-1}, \tag{1.6}$$

$$g(\lambda)v_{j-1} + \tilde{G}(\lambda)v_{j-1/2} + \check{g}(\lambda)v_j = 0, \quad j = \overline{1,K}, \tag{1.7}$$

с $v_0 = v_K = 0$, $v \not\equiv 0$.

Рассмотрим также вспомогательные задачи на собственные значения на эталонном элементе σ_0 и внутри него

$$Ae = \lambda Ce, \quad e \in \mathbb{R}^{n+1}, \quad e \neq 0, \tag{1.8}$$

$$\tilde{A}e = \lambda\tilde{C}e, \quad e \in \mathbb{R}^{n-1}, \quad e \neq 0, \tag{1.9}$$

где ясно, что $A \geq 0$, $C > 0$ и $\tilde{A} = \tilde{A}^{\mathrm{T}} > 0$, $\tilde{C} = \tilde{C}^{\mathrm{T}} > 0$; см. некоторые их свойства в [11] (где была изучена задача, аналогичная (1.6), (1.7) на равномерной сетке на $[0,\infty)$ при $\lambda \in \mathbb{C}$). Обозначим через S_n и $\tilde{S}_n$ их спектры. Пусть $\{\lambda_0^{(l)}, e^{(l)}\}_{l=1}^{n-1}$ – собственные пары задачи (1.9).

Лемма 1.1. 1. *Подпространства* $\mathbb{R}_e^{n-1}$ *и* $\mathbb{R}_e^{n-1}$ *инвариантны по отношению к* $\tilde{A}$ *и* $\tilde{C}$. *Поэтому каждый собственный вектор в* $\{e^{(l)}\}_{l=1}^{n-1}$ *может быть выбран четным или нечетным. Также* $\lambda_0^{(l)} > 0$, $l = \overline{1,n-1}$.

2. *Аналогичные свойства верны для задачи* (1.8) *за исключением наличия одного простого нулевого собственного значения.*

Доказательство. Любая бисимметричная матрица B порядка $n-1$ коммутирует с P, т.е.

$$BP = PB, \tag{1.10}$$

откуда следует основной результат п. 1. Свойство $\lambda_0^{(l)} > 0$, $l = \overline{1, n-1}$, хорошо известно.

Для п. 2 рассуждение аналогично с учетом того, что $A(1\ldots1)^{\mathrm{T}} = 0$ (относительно простоты $\lambda = 0$ см. [11, утверждение 5]).

Прямым вычислением можно проверить, что все собственные значения в S_n и $\tilde{S}_n$ являются простыми и $S_n \cap \tilde{S}_n = \emptyset$ по крайней мере при $1 \le n \le 9$, см. [11].

Для малых n можно найти S_n и $\tilde{S}_n$ аналитически (с помощью Mathematica), в частности, $\tilde{S}_2 = \{2.5\}$, $\tilde{S}_3 = \{2.5, 10.5\}$, $\tilde{S}_4 = \{14 \pm \sqrt{133}, 10.5\}$ и $\tilde{S}_5 = \{14 \pm \sqrt{133}, 30 \pm 9\sqrt{5}\}$ (повторяемость собственных значений не случайна, см. [11]).

Выберем $\{e^{(l)}\}_{l=1}^{n-1}$ как указано в лемме 1.1, с использованием масштабирования $\tilde{C}e^{(l)} \cdot e^{(l)} = 1$.

Лемма 1.2. *Пусть* $\tilde{G}(\lambda)p = -g(\lambda)$, *где* $\lambda \notin \tilde{S}_n$. *Пусть векторы* a *и* c *разложены как*

$$a = \sum_{l=1}^{n-1} a^{(l)} \tilde{C} e^{(l)}, \quad c = \sum_{l=1}^{n-1} c^{(l)} \tilde{C} e^{(l)}, \quad \text{с} \quad a^{(l)} = a \cdot e^{(l)}, \quad c^{(l)} = c \cdot e^{(l)}. \tag{1.11}$$

См. $\tilde{G}(\lambda)$, $g(\lambda)$, a *и* c *в* (1.3). *Тогда верны следующие формулы*:

$$p = \sum_{l=1}^{n-1} \frac{a^{(l)} - \lambda c^{(l)}}{\lambda - \lambda_0^{(l)}} e^{(l)} = \sum_{l=1}^{n-1} \frac{a^{(l)} - \lambda_0^{(l)} c^{(l)}}{\lambda - \lambda_0^{(l)}} e^{(l)} - \tilde{C}^{-1} c.$$

Доказательство. Для разложений $p = \sum_{l=1}^{n-1} p_l e^{(l)}$ и (1.11) имеем

$$\tilde{G}(\lambda)p = \sum_{l=1}^{n-1} (\lambda_0^{(l)} - \lambda) p_l \tilde{C} e^{(l)}, \quad g(\lambda) = \sum_{l=1}^{n-1} (a^{(l)} - \lambda c^{(l)}) \tilde{C} e^{(l)},$$

откуда легко следует результат.

2. РЕШЕНИЕ 1D МКЭ-ЗАДАЧ НА СОБСТВЕННЫЕ ЗНАЧЕНИЯ И СООТВЕТСТВУЮЩИЕ АЛГОРИТМЫ, ОСНОВАННЫЕ НА БДПФ

Сделаем следующее

Предположение.

$$(A) \quad \text{собственные значения в } S_n \quad \text{и} \quad \tilde{S}_n \text{ простые} \quad \text{и} \quad S_n \cap \tilde{S}_n = \emptyset.$$

Напомним, что это верно по крайней мере при $2 \le n \le 9$ (ниже в разд. 4 это проверяется вплоть до $n = 21$).

Введем вспомогательное уравнение

$$\hat{\gamma}(\lambda) \equiv -(g_0 - g \cdot \tilde{G}^{-1} g)(\lambda)/(g_n - \breve{g} \cdot \tilde{G}^{-1} g)(\lambda) = \theta, \tag{2.1}$$

где $\lambda \notin \tilde{S}_n$, с параметром θ. Его решение эквивалентно нахождению корней многочлена степени не выше n, см. [11]. В частности, в силу леммы 1.2 это уравнение может быть переписано в виде

$$a_0 - \lambda c_0 + \sum_{l=1}^{n-1} \frac{(a^{(l)} - \lambda c^{(l)})^2}{\lambda - \lambda_0^{(l)}} = -\theta \left(a_n - \lambda c_n + \sum_{l=1}^{n-1} \frac{(\breve{a}^{(l)} - \lambda \breve{c}^{(l)})(a^{(l)} - \lambda c^{(l)})}{\lambda - \lambda_0^{(l)}} \right). \tag{2.2}$$

Здесь $\check{a}^{(l)} = \check{a} \cdot e^{(l)}$ и $\check{c}^{(l)} = \check{c} \cdot e^{(l)}$. Более того, при $2 \le n \le 9$ вычисления помогают подтвердить, что векторы $e^{(l)}$ являются четными/нечетными соответственно для нечетных/четных l при условии, что $\lambda_0^{(1)} < \ldots < \lambda_0^{(n-1)}$; следовательно, тогда $\check{a}^{(l)} = (-1)^l a^{(l)}$ и $\check{c}^{(l)} = (-1)^l c^{(l)}$, $l = \overline{1, n-1}$.

Введем простейшее скалярное произведение в $S_K^{(n)}$ и соответствующий квадрат $\mathscr{C}$-нормы

$$(y, v)_{S_K^{(n)}} := \sum_{j=1}^{K-1} y_j v_j + \sum_{j=1}^{K} y_{j-1/2} \cdot v_{j-1/2}, \qquad \|v\|_C^2 := (\mathscr{C}v, v)_{S_K^{(n)}}.$$

В следующей теореме описываются собственные значения и собственные векторы задачи (1.2).

Теорема 2.1. 1. *Спектр задачи* (1.2) *состоит из* $\tilde{S}_n$ *и чисел* $\{\lambda_k^{(l)}\}_{l=1}^n$, *которые являются всеми* n (*и всеми положительными вещественными*) *решениями уравнения* (2.2) *с* $\theta = \theta_k := \cos\dfrac{\pi k}{K}$ *при* $k = \overline{1, K-1}$. *Числа* $\{\lambda_k^{(l)}\}_{l=1}^n$ *отличаются от* $\{\lambda_0^{(l)}\}_{l=1}^{n-1}$ *и различны при фиксированном* k.

2. *Собственному значению* $\lambda_0^{(l)}$ *отвечает собственный вектор*

$$s_{0,j}^{(l)} = 0, \quad j = \overline{1, K-1}, \qquad s_{0,j-1/2}^{(l)} = (-P)^{j-1} e^{(l)}, \quad j = \overline{1, K}, \tag{2.3}$$

при $l = \overline{1, n-1}$. *Здесь* $(-P)^{j-1} e = (-1)^{j-1} e$ *для четного* e, $(-P)^{j-1} e = e$ *для нечетного* e.

3. *Собственному значению* $\lambda_k^{(l)}$ *отвечает собственный вектор*

$$s_{k,j}^{(l)} = s_{k,j}, \quad j = \overline{1, K-1}, \qquad s_{k,j-1/2}^{(l)} = p_k^{(l)} \sin\frac{\pi k(j-1)}{K} + \check{p}_k^{(l)} \sin\frac{\pi k j}{K}, \quad j = \overline{1, K}, \tag{2.4}$$

где $s_{k,j} := \sin\dfrac{\pi k j}{K}$, *а* $p_k^{(l)} \in \mathbb{R}^{n-1}$ *служит решением невырожденной алгебраической системы уравнений* $\tilde{G}(\lambda_k^{(l)}) p_k^{(l)} = -g(\lambda_k^{(l)})$ *при* $k = \overline{1, K-1}$, $l = \overline{1, n}$.

4. *Введенные собственные векторы являются* $\mathscr{C}$*-ортогональными, т.е.*

$$(C s_k^{(l)}, s_{\tilde{k}}^{(\tilde{l})})_{S_K^{(n)}} = 0 \tag{2.5}$$

для всех $k, \tilde{k} \in \overline{0, K-1}$, $l \in \overline{1, n - \delta_{k0}}$, $\tilde{l} \in \overline{1, n - \delta_{\tilde{k}0}}$ *таких, что* $k \ne \tilde{k}$ *и/или* $l \ne \tilde{l}$.

Как следствие, они образуют базис в $S_K^{(n)}$, *т.е. любой* $w \in S_K^{(n)}$ *может быть однозначно разложен как*

$$w = \sum_{l=1}^{n-1} w_{0l} s_0^{(l)} + \sum_{k=1}^{K-1} \sum_{l=1}^{n} w_{kl} s_k^{(l)}. \tag{2.6}$$

Доказательство. Будем различать два случая.

1. Пусть сначала $\lambda \in \tilde{S}_n$ и e удовлетворяет (1.9). Тогда для любого $j = \overline{1, K}$ с использованием уравнения (1.7) получим

$$0 = v_{j-1/2} \cdot \tilde{G}(\lambda) e = \tilde{G}(\lambda) v_{j-1/2} \cdot e = -[(g(\lambda) \cdot e) v_{j-1} + (\check{g}(\lambda) \cdot e) v_j].$$

Ясно, что

$$G(\lambda) \begin{pmatrix} 0 \\ e \\ 0 \end{pmatrix} = \begin{pmatrix} g(\lambda) \cdot e \\ 0 \\ \check{g}(\lambda) \cdot e \end{pmatrix}.$$

Поскольку $\lambda \notin S_n$ по предположению (A), то имеем $g(\lambda) \cdot e \ne 0$ или $\check{g}(\lambda) \cdot e \ne 0$. В силу предположения (A) и леммы 1.1 λ является простым в $\tilde{S}_n$ и e четен или нечетен. Соответственно или

$\check{g}(\lambda)\cdot e = g(\lambda)\cdot e \neq 0$ и $v_j = -v_{j-1}$, или $\check{g}(\lambda)\cdot e = -g(\lambda)\cdot e \neq 0$ и $v_j = v_{j-1}$. Поскольку $v_0 = 0$, то в обоих случаях получим

$$v_j = 0, \quad j = \overline{0, K}. \tag{2.7}$$

Поэтому уравнение (1.7) сводится к $\tilde{G}v_{j-1/2} = 0$, откуда следует, что $v_{j-1/2} = c_{j-1/2}e$.

Теперь уравнение (1.6) сводится к

$$c_{j-1/2}\,\check{g}(\lambda)\cdot e + c_{j+1/2}g(\lambda)\cdot e = 0, \quad j = \overline{1, K-1},$$

поэтому $c_{j+1/2} = -c_{j-1/2}$ при четном e или $c_{j+1/2} = c_{j-1/2}$ при нечетном e. Следовательно, искомый собственный вектор v удовлетворяет (2.7) и

$$v_{j-1/2} = (-1)^{j-1}e \quad \text{при четном } e, \quad v_{j-1/2} = e \quad \text{при нечетном } e, \quad j = \overline{1, K}$$

(v определен с точностью до ненулевого постоянного множителя). Таким образом приходим к собственным векторам (2.3).

2. Пусть теперь $\lambda \notin \tilde{S}_n$. Тогда из уравнения (1.7) получим

$$v_{j-1/2} = -G^{-1}(\lambda)[v_{j-1}g(\lambda) + v_j\,\check{g}(\lambda)], \quad j = \overline{1, K}. \tag{2.8}$$

Подставляя эту формулу в уравнение (1.6), выведем трехточечное уравнение

$$\hat{g}_n(\lambda)v_{j-1} + 2\hat{g}_0(\lambda)v_j + \hat{g}_n(\lambda)v_{j+1} = 0, \quad j = \overline{1, K-1}, \tag{2.9}$$

где

$$\hat{g}_0(\lambda) = (g_0 - g\cdot\tilde{G}^{-1}g)(\lambda), \quad \hat{g}_n(\lambda) = (g_n - \check{g}\cdot\tilde{G}^{-1}g)(\lambda).$$

Непосредственно проверяются следующие равенства

$$G(\lambda)\begin{pmatrix} 1 \\ -(\tilde{G}^{-1}g)(\lambda) \\ 0 \end{pmatrix} = \begin{pmatrix} \hat{g}_0(\lambda) \\ 0 \\ \hat{g}_n(\lambda) \end{pmatrix}, \quad G(\lambda)\begin{pmatrix} 0 \\ -(\tilde{G}^{-1}\check{g})(\lambda) \\ 1 \end{pmatrix} = \begin{pmatrix} \hat{g}_n(\lambda) \\ 0 \\ \hat{g}_0(\lambda) \end{pmatrix}.$$

Если $\hat{g}_n(\lambda) = \hat{g}_0(\lambda) = 0$, то эти равенства означают, что λ является по крайней мере двукратным собственным значением задачи (1.8), что противоречит предположению (A).

Если $\hat{g}_n(\lambda) = 0$ и $\hat{g}_0(\lambda) \neq 0$, то уравнение (2.9) вместе с (2.8) приводит к $v = 0$, поэтому такое λ не удовлетворяет (1.2).

Значит, $\hat{g}_n(\lambda) \neq 0$ и уравнение (2.9) упрощается до

$$v_{j-1} - 2\hat{\gamma}(\lambda)v_j + v_{j+1} = 0, \quad j = \overline{1, K-1}, \tag{2.10}$$

с функцией $\hat{\gamma}(\lambda) = \hat{g}_0(\lambda)/\hat{g}_n(\lambda)$ (см. ее также в (2.1)). Поскольку $v_0 = v_n = 0$, то можно использовать разложение

$$v_j = \sum_{k=1}^{K-1}\tilde{v}_k s_{k,j}, \quad j = \overline{0, K},$$

и ввести вектор его коэффициентов $\tilde{\mathbf{v}} := (\tilde{v}_1, \ldots, \tilde{v}_{K-1})$. Применение этого разложения в (2.10) дает

$$2\sum_{k=1}^{K-1}\tilde{v}_k(\theta_k - \hat{\gamma}(\lambda))s_{k,j} = 0, \quad j = \overline{1, K-1}. \tag{2.11}$$

Ясно, что это равенство выполнено для некоторого $\tilde{\mathbf{v}} \neq 0$ тогда и только тогда, когда

$$\hat{\gamma}(\lambda) = \theta_k \quad \text{для некоторого} \quad k = \overline{1, K-1}. \tag{2.12}$$

Отметим, что $\tilde{\mathbf{v}} = 0$ эквивалентно $v = 0$ в $S_K^{(n)}$ (с учетом формулы (2.8)). Поэтому λ удовлетворяет (2.12); более того, $\tilde{v}_j = \delta_{kj}$ и, как следствие, $v_j = s_{k,j}$, $j = \overline{0, K}$, а также

$$v_{j-1/2} = -s_{k,j-1}(G^{-1}g)(\lambda) - s_{k,j}(G^{-1}\check{g})(\lambda), \quad j = \overline{1, K},$$

см. (2.8) (все последние три равенства выполнены с точностью до одинакового ненулевого множителя). Таким образом приходим к собственным векторам (2.4).

Общее количество собственных значений $\lambda \notin \tilde{S}_n$ (с учетом возможной кратности) равно $\dim S_K^{(n)} - (n-1) = n(K-1)$. Максимальное количество корней алгебраических уравнений (2.12) при всех k то же самое, поэтому каждое уравнение (2.12) должно иметь ровно n различных корней (при фиксированном k записанный вектор v определяется значением λ однозначно).

3. Свойство (2.5) заведомо выполняется для собственных векторов $s_k^{(l)}$ и $s_{\tilde{k}}^{(\tilde{l})}$, отвечающих различным собственным значениям задачи (1.2), в частности, при $k = 0$ и $\tilde{k} \neq 0$, или $k = \tilde{k}$ и $l \neq \tilde{l}$.

Оставшийся случай будет рассмотрен ниже в следствии 2.1 связанной с этим леммы 2.1.

Отметим, что: 1) векторы $s_0^{(l)}$ используются только для описания алгоритма, и только векторы $e^{(l)}$ применяются в его реализации; 2) $s_{k,j}^{(l)}$ не зависят от l; 3) векторы $p_k^{(l)}$ не зависят от j и могут быть также вычислены в силу леммы 1.2.

Лемма 2.1. *Пусть* $w \in S_K^{(n)}$ *и* $w_{j-1/2} = qw_{j-1} + \check{q}\, w_j$, $j = \overline{1, K}$, *для некоторого* $q \in \mathbb{R}^{n-1}$. *Тогда*

$$(\mathscr{C}s_0^{(l)}, w)_{S_K^{(n)}} = 0, \quad l = \overline{1, n-1}, \tag{2.13}$$

$$(\mathscr{C}s_k^{(l)}, w)_{S_K^{(n)}} = 2\{c_0 + c \cdot p_k^{(l)} + (\tilde{C}p_k^{(l)} + c) \cdot q + \theta_k[c_n + c \cdot \check{p}_k^{(l)} + (\tilde{C}p_k^{(l)} + c) \cdot \check{q}]\}(s_k, w)_{\omega_h} \tag{2.14}$$

при $k = \overline{1, K-1}$, $l = \overline{1, n}$, где $(s_k, w)_{\omega_h} := \sum_{j=1}^{K-1} s_{k,j} w_j$.

Доказательство. 1. Вспомнив обозначение (1.3), для любых $v, w \in S_K^{(n)}$, имеем

$$\begin{aligned}(\mathscr{C}v, w)_{S_K^{(n)}} = &\sum_{j=1}^{K-1}[2c_0 v_j + c_n(v_{j-1} + v_{j+1}) + \check{c} \cdot v_{j-1/2} + c \cdot v_{j+1/2}]w_j + \\ &+ \sum_{j=1}^{K}(cv_{j-1} + \tilde{C}v_{j-1/2} + \check{c}\, v_{j+1}) \cdot w_{j-1/2}.\end{aligned} \tag{2.15}$$

2. Согласно формулам (2.15) и (2.3) при четном $e^{(l)}$ получим

$$\begin{aligned}(\mathscr{C}s_0^{(l)}, w)_{S_K^{(n)}} &= \sum_{j=1}^{K-1}(-1)^j(c - \check{c}) \cdot e^{(l)} w_j + \sum_{j=1}^{K}(-1)^{j-1}\tilde{C}e^{(l)} \cdot (qw_{j-1} + \check{q}\, w_j) = \\ &= \tilde{C}e^{(l)} \cdot (q - \check{q}) \sum_{j=1}^{K-1}(-1)^j w_j = 0,\end{aligned}$$

поскольку $(c - \check{c}) \cdot e^{(l)} = 0$ и $\tilde{C}e^{(l)} \cdot (q - \check{q}) = 0$ для всех $c, q \in \mathbb{R}^{n-1}$, а также $w_0 = w_K = 0$.

При нечетном $e^{(l)}$ аналогично получим

$$(\mathscr{C}s_0^{(l)}, w)_{S_K^{(n)}} = \sum_{j=1}^{K-1}(c + \check{c}) \cdot e^{(l)} w_j + \sum_{j=1}^{K}\tilde{C}e^{(l)} \cdot (qw_{j-1} + \check{q}\, w_j) = \tilde{C}e^{(l)} \cdot (q + \check{q}) \sum_{j=1}^{K-1} w_j = 0,$$

поскольку $(c + \check{c}) \cdot e^{(l)} = 0$ и $\tilde{C}e^{(l)} \cdot (q + \check{q}) = 0$ для всех $c, q \in \mathbb{R}^{n-1}$, а также $w_0 = w_K = 0$. Равенство (2.13) доказано.

3. Формула (2.15) вместе с (2.4) и (2.5) влекут равенство

$$(\mathscr{C}s_k^{(l)}, w)_{S_K^{(n)}} = \sum_{j=1}^{K-1}[2(c_0 + c\cdot p_k^{(l)})s_{k,j} + (c_n + c\cdot \check{p}_k^{(l)})(s_{k,j-1} + s_{k,j+1})]w_j + \\ + \sum_{j=1}^{K-1}[(\tilde{C}p_k^{(l)} + c)s_{k,j-1} + (\tilde{C}\,\check{p}_k^{(l)} + \check{c})s_{k,j}]w_{j-1/2} =: \sum{}' + \sum{}''. \tag{2.16}$$

В силу формулы

$$s_{k,j-1} + s_{k,j+1} = 2\theta_k s_{k,j} \tag{2.17}$$

сначала выводим

$$\sum{}' = 2[c_0 + c\cdot p_k^{(l)} + \theta_k(c_n + c\cdot \check{p}_k^{(l)})](s_k, w)_{\omega_k}. \tag{2.18}$$

Затем с использованием формул (2.5) и (2.10) получим

$$\sum{}'' = \sum_{j=1}^{K-1}(\tilde{C}p_k^{(l)} + c)\cdot q(s_{k,j-1}w_{j-1} + s_{k,j}w_j) + (\tilde{C}p_k^{(l)} + c)\cdot \check{q}(s_{k,j-1}w_j + s_{k,j}w_{j-1}).$$

В силу свойств $s_{k,0} = s_{k,K} = 0$, $w_0 = w_K = 0$ и формул (2.17) далее выводим

$$\sum{}'' = 2(\tilde{C}p_k^{(l)} + c)\cdot q(s_k, w)_{\omega_h} + (\tilde{C}p_k^{(l)} + c)\cdot \check{q}\sum_{j=1}^{K-1}(s_{k,j-1} + s_{k,j+1})w_j = \\ = 2[(\tilde{C}p_k^{(l)} + c)\cdot q + \theta_k(\tilde{C}p_k^{(l)} + c)\cdot \check{q}](s_k, w)_{\omega_h}. \tag{2.19}$$

Сложив (2.18) и (2.19), докажем (2.14).

Следствие 2.1. *Верно свойство ортогональности* (2.5) *из теоремы* 2.1, п. 4.

Доказательство. Остается рассмотреть случай $k, \tilde{k} \in \overline{1, K-1}$ и $k \neq \tilde{k}$. Поскольку тогда $(s_k, s_{\tilde{k}})_{\omega_h} = 0$, то результат следует из (2.14).

Вычисление $w \in S_K^{(n)}$ по коэффициентам w_{kl} разложения (2.6) назовем *обратным* F_n-преобразованием, а вычисление коэффициентов w_{kl} по $w \in S_K^{(n)}$ – *прямым* F_n-преобразованием. Рассмотрим также связанное с этим разложение $y \in S_K^{(n)}$ вида

$$y = \sum_{l=1}^{n-1}\tilde{y}_{0l}\mathscr{C}s_0^{(l)} + \sum_{k=1}^{K-1}\sum_{l=1}^{n}\tilde{y}_{kl}\mathscr{C}s_k^{(l)} \tag{2.20}$$

и вычисление коэффициентов $\tilde{y}_{kl}$ по $y \in S_K^{(n)}$, которое назовем *прямым* FC_n-преобразованием.

Опишем их быструю БДПФ-реализацию.

Теорема 2.2. 1. *Обратное* F_n*-преобразование может быть реализовано в соответствии со следующими формулами*

$$w_j = \sum_{k=1}^{K-1}\left(\sum_{l=1}^{n}w_{kl}\right)\sin\frac{\pi kj}{K}, \quad j = \overline{1, K-1}, \tag{2.21}$$

$$w_{j-1/2} = (-P)^{j-1}\sum_{l=1}^{n-1}w_{0l}e^{(l)} + 2\sum_{k=1}^{K-1}d_{k,e}\cos\frac{\pi k}{2K}\sin\frac{\pi k(j-1/2)}{K} - \\ - 2\sum_{k=1}^{K-1}d_{k,o}\sin\frac{\pi k}{2K}\cos\frac{\pi k(j-1/2)}{K}, \quad j = \overline{1, K}, \tag{2.22}$$

где $d_{k,e}$ *и* $d_{k,o}$ *– соответственно четные и нечетные компоненты вектора* $d_k := \sum_{l=1}^{n}w_{kl}p_k^{(l)}$. *Заметим, что* $(-P)^{j-1}e = e$ *при нечетном* j *и* $(-P)^{j-1}e = -\check{e}$ *при четном* j *для любого* $e \in \mathbb{R}^{n-1}$.

Набор $\{w_j\}_{j=1}^{K-1}$ *можно вычислить с помощью стандартного обратного БДПФ по синусам. Набор* $\{w_{j-1/2}\}_{j=1}^{K}$ *можно вычислить с помощью* $n-1$ *модифицированного обратного БДПФ, связанного с центрами элементов, в количестве* $[n/2]$ *по синусам и* $[(n-1)/2]$ *по косинусам с использованием продолжений* $d_{K,e} := 0$ *и* $d_{0,o} := 0$*, см. алгоритмы DST-I, DST-III и DCT-III в* [6].

2. *Прямое* FC_n*-преобразование можно реализовать на основе стандартной формулы*

$$\tilde{y}_{kl} = (y, s_k^{(l)})_{S_K^{(n)}} / \left\| s_k^{(l)} \right\|_C^2. \tag{2.23}$$

Здесь, во-первых, при $k = 0$, $l = \overline{1, n-1}$ *верны формулы*

$$(y, s_0^{(l)})_{S_K^{(n)}} = \left(\sum_{j=1}^{K} (-P)^{j-1} y_{j-1/2} \right) e^{(l)}, \quad \left\| s_0^{(l)} \right\|_C^2 = K. \tag{2.24}$$

Во-вторых, при $k = \overline{1, K-1}$, $l = \overline{1, n}$ *верны следующие формулы*:

$$(y, s_k^{(l)})_{S_K^{(n)}} = \sum_{j=1}^{K-1} y_j \sin\frac{\pi kj}{K} + p_{k,e}^{(l)} \cdot \sum_{j=1}^{K-1} (y_{j-1/2} + y_{j+1/2})_e \sin\frac{\pi kj}{K} + p_{k,o}^{(l)} \cdot \sum_{j=1}^{K-1} (y_{j+1/2} - y_{j-1/2})_o \sin\frac{\pi kj}{K}, \tag{2.25}$$

$$\left\| s_k^{(l)} \right\|_C^2 = K\{c_0 + (\tilde{C} p_k^{(l)} + 2c) \cdot p_k^{(l)} + \theta_k [c_n + (\tilde{C} p_k^{(l)} + 2c) \cdot p_k^{(l)}]\}. \tag{2.26}$$

Отметим, что суммы в формуле (2.25) *не зависят от* l*. Набор всех этих коэффициентов можно вычислить с помощью* n *стандартных прямых БДПФ по синусам.*

3. *Аналогично п.* 2*, прямое* F_n*-преобразование можно реализовать на основе стандартной формулы*

$$w_{kl} = (\mathscr{C} w, s_k^{(l)})_{S_K^{(n)}} / \left\| s_k^{(l)} \right\|_C^2. \tag{2.27}$$

Здесь при $k = 0$, $l = \overline{1, n-1}$ *верна следующая формула*:

$$(\mathscr{C} w, s_0^{(l)})_{S_K^{(n)}} = \left(\tilde{C} \sum_{j=1}^{K} (-P)^{j-1} w_{j-1/2} \right) e^{(l)}. \tag{2.28}$$

При $k = \overline{1, K-1}$, $l = \overline{1, n}$ *применима формула* (2.25) *с* $y := \mathscr{C} w$*. С другой стороны, верна также следующая формула*

$$\begin{aligned} (\mathscr{C} w, s_k^{(l)})_{S_K^{(n)}} &= 2[c_0 + c \cdot p_k^{(l)} + \theta_k (c_n + c \cdot \check{p}_k^{(l)})] \sum_{j=1}^{K-1} w_j \sin\frac{\pi kj}{K} + \\ &+ q_{k,e}^{(l)} \cdot \sum_{j=1}^{K-1} (w_{j-1/2} + w_{j+1/2})_e \sin\frac{\pi kj}{K} + q_{k,o}^{(l)} \cdot \sum_{j=1}^{K-1} (w_{j+1/2} - w_{j-1/2})_o \sin\frac{\pi kj}{K}, \end{aligned} \tag{2.29}$$

где $q_{k,e}^{(l)}$ и $q_{k,o}^{(l)}$ *— соответственно четная и нечетная компоненты вектора* $q_k^{(l)} := \tilde{C} p_k^{(l)} + c$*. Снова все эти коэффициенты можно вычислить с помощью* n *стандартных прямых БДПФ по синусам.*

Доказательство. 1. Пусть коэффициенты w_{kl} разложения (2.6) известны. В соответствии с первыми из формул (2.3) и (2.4) значения w для целых индексов в (2.6) сводятся к (2.21).

Чтобы вычислить w для полуцелых индексов, преобразуем вторую сумму в (2.6). В силу разложения (1.4) перепишем вторую формулу (2.4) в виде

$$\begin{aligned} s_{k,j-1/2}^{(l)} &= p_{k,e}^{(l)} \left(\sin\frac{\pi k(j-1)}{K} + \sin\frac{\pi kj}{K} \right) - p_{k,o}^{(l)} \left(\sin\frac{\pi kj}{K} - \sin\frac{\pi k(j-1)}{K} \right) = \\ &= 2\cos\frac{\pi k}{2K} p_{k,e}^{(l)} \sin\frac{\pi k(j-1/2)}{K} - 2\sin\frac{\pi k}{2K} p_{k,o}^{(l)} \cos\frac{\pi k(j-1/2)}{K}, \quad j = \overline{1, K}. \end{aligned}$$

Затем, применив также вторую формулу (2.3), получим формулу (2.22).

2. Теперь рассмотрим вычисление коэффициентов в разложении (2.20) при заданном $y \in S_K$. В силу свойства ортогональности (2.5), они сначала могут быть выражены в виде (2.23) при $k = 0$, $l = \overline{1, n-1}$ и $k = \overline{1, K-1}$, $l = \overline{1, n}$.

Формулы (2.15) и (2.3) влекут равенства

$$\left\|s_0^{(l)}\right\|_C^2 = K\tilde{C}e^{(l)} \cdot e^{(l)} = K, \quad l = \overline{1, n-1}.$$

Лемма 2.1 немедленно влечет формулу (2.26), поскольку $(s_k, s_k)_{\omega_h} = K/2$.

В силу формул (2.3) для числителя формулы (2.23) при $k = 0$ можно записать

$$(y, s_0^{(l)})_{S_K^{(n)}} = \sum_{j=1}^{K} y_{j-1/2} \cdot (-P)^{j-1} e^{(l)} = \left(\sum_{j=1}^{K} (-P)^{j-1} y_{j-1/2}\right) \cdot e^{(l)}.$$

В силу формул (2.4) для того же числителя при $k = \overline{1, K-1}$ получим

$$(y, s_k^{(l)})_{S_K^{(n)}} = \sum_{j=1}^{K-1} y_j s_{k,j} + \sum_{j=1}^{K} y_{j-1/2} s_{k,j-1} \cdot p_k^{(l)} + \sum_{j=1}^{K} y_{j-1/2} s_{k,j} \cdot \check{p}_k^{(l)}. \tag{2.30}$$

Поэтому сдвинув на 1 индекс во второй из этих сумм, применив тождество $a_1 b_1 + a_2 b_2 = 0.5(a_1 + a_2)(b_1 + b_2) + 0.5(a_1 - a_2)(b_1 - b_2)$ и вспомнив разложение (1.4), выведем

$$(y, s_k^{(l)})_{S_K^{(n)}} = \sum_{j=1}^{K-1} y_j s_{k,j} + p_{k,e}^{(l)} \cdot \sum_{j=1}^{K-1} (y_{j-1/2} + y_{j+1/2}) s_{k,j} + p_{k,o}^{(l)} \cdot \sum_{j=1}^{K-1} (y_{j+1/2} - y_{j-1/2}) s_{k,j}.$$

Поскольку также

$$p_e \cdot q = p_e \cdot q_e, \quad p_o \cdot q = p_o \cdot q_o \quad \text{для всех} \quad q, p \in \mathbb{R}^{n-1},$$

то в итоге установим формулу (2.25).

3. Благодаря свойству ортогональности (2.5) справедлива формула (2.27).

В силу формул (2.3), (2.15), а также $\tilde{C}^{\mathrm{T}} = \tilde{C}$ и $P = P^{\mathrm{T}}$ для ее числителя при $k = 0$ можно записать

$$(\mathscr{C}w, s_0^{(l)})_{S_K^{(n)}} = (\mathscr{C}s_0^{(l)}, w)_{S_K^{(n)}} = \sum_{j=1}^{K} \tilde{C}(-P)^{j-1} e^{(l)} \cdot w_{j-1/2} = \left(\tilde{C}\sum_{j=1}^{K} (-P)^{j-1} w_{j-1/2}\right) \cdot e^{(l)}.$$

В силу формул (2.16), (2.18) для того же числителя при $k = \overline{1, K-1}$ получим

$$(\mathscr{C}w, s_k^{(l)})_{S_K^{(n)}} = (\mathscr{C}s_k^{(l)}, w)_{S_K^{(n)}} = 2[c_0 + c \cdot p_k^{(l)} + \theta_k(c_n + c \cdot \check{p}_k^{(l)})](s_k, w)_{\omega_h} + \sum_{j=1}^{K} (q_k^{(l)} s_{k,j-1} + \check{q}_k^{(l)} s_{k,j}) w_{j-1/2},$$

где $q_k^{(l)} = \tilde{C}p_k^{(l)} + c$. Преобразовав последнюю сумму таким же образом, как второе и третье слагаемые в (2.30), установим (2.29).

3. ПРИЛОЖЕНИЯ К ОБОБЩЕННОМУ УРАВНЕНИЮ ПУАССОНА

Для начала обратимся к простой 1D краевой задаче для ОДУ

$$-u''(x) + \alpha u(x) = f(x) \quad \text{на} \quad [0, X], \quad u(0) = u(X) = 0, \tag{3.1}$$

где $\alpha \neq -(\pi i/X)^2$, $i \in \mathbb{N}$. Ее МКЭ-дискретизация имеет операторную форму

$$4h^{-2}\mathscr{A}v + \alpha\mathscr{C}v = f^h, \quad v \in S_K^{(n)}, \tag{3.2}$$

где $f^h \in S_K^{(n)}$ – это МКЭ-усреднение f. Ее решение можно записать в виде

$$v = \sum_{k=0}^{K} \sum_{l=1}^{n-\delta_{k0}} \frac{\tilde{f}_{kl}^h}{4h^{-2}\lambda_k^{(l)} + \alpha} s_k^{(l)} \tag{3.3}$$

разложения типа (2.6), где $\tilde{f}^h_{kl}$ – коэффициенты разложения типа (2.20) вектора f^h; напомним, что δ_{k0} – символ Кронекера. Формула корректна для невырожденных знаменателей, что имеет место при всех h, если $\alpha \in \mathbb{R}$ и $\alpha = \text{const} > -(\pi/X)^2$, или по крайней мере для достаточно малых h в противном случае, или если $\alpha \in \mathbb{C}\backslash\mathbb{R}$. В последнем случае коэффициенты разложения становятся комплексными, а функцию f, а поэтому и вектор f^h и его коэффициенты разложения $\tilde{f}^h_{kl}$ также можно считать комплексными. Подобное относится и к многомерному случаю ниже.

Теперь рассмотрим детально решение N-мерной ($N \geq 2$) краевой задачи

$$-\Delta u + \alpha u = f \quad \text{в} \quad \Omega = (0, X_1)\times\ldots\times(0, X_N), \quad u\big|_{\partial\Omega} = 0, \tag{3.4}$$

где Δ – оператор Лапласа, а $\alpha = \text{const}$ не принадлежит его спектру (при тех же краевых условиях).

Введем пространство $H^{(n_1)}_{h_1}[0, X_1]\otimes\ldots\otimes H^{(n_N)}_{h_N}[0, X_N]$ кусочно-полиномиальных функций в $\bar{\Omega}$, где $h_i = X_i/K_i$ и $n_i \geq 2$, $i = \overline{1, N}$. Пусть $\mathbf{K} = (K_1, \ldots, K_N)$ и $\mathbf{n} = (n_1, \ldots, n_N)$.

Введем также пространство вектор-функций $S^{(\mathbf{n})}_{\mathbf{K}} = S^{(n_1)}_{K_1}\otimes\ldots\otimes S^{(n_N)}_{K_N}$. Например, при $N = 2$, эти функции являются числами для индексов (j_1, j_2), $j_1 = \overline{0, K_1}$, $j_2 = \overline{0, K_2}$ и векторами из $\mathbb{R}^{n_1-1}$, $\mathbb{R}^{n_2-1}$ и $\mathbb{R}^{(n_1-1)\times(n_2-1)}$ соответственно для индексов

$$(j_1 - 1/2, j_2), \quad j_1 = \overline{1, K_1}, \quad j_2 = \overline{0, K_2}; \quad (j_1, j_2 - 1/2), \quad j_1 = \overline{0, K_1}, \quad j_2 = \overline{1, K_2} \quad \text{и}$$
$$(j_1 - 1/2, j_2 - 1/2), \quad j_1 = \overline{1, K_1}, \quad j_2 = \overline{1, K_2},$$

а также нулевыми векторами для $j_1 = 0, K_1$ и $j_2 = 0, K_2$. Аналогично 1D случаю существует естественный изоморфизм между функциями из $H^{(n_1)}_{h_1}[0, X_1]\otimes\ldots\otimes H^{(n_N)}_{h_N}[0, X_N]$ и векторами из $S^{(\mathbf{n})}_{\mathbf{K}}$.

МКЭ-дискретизация задачи (3.4) может быть записана в следующей операторной форме

$$(4h_1^{-2}\mathscr{A}_1\mathscr{C}_2\ldots\mathscr{C}_N + \ldots + 4h_N^{-2}\mathscr{A}_N\mathscr{C}_1\ldots\mathscr{C}_{N-1})v + \alpha\mathscr{C}_1\ldots\mathscr{C}_N v = f^h, \quad v \in S^{(\mathbf{n})}_{\mathbf{K}}, \tag{3.5}$$

где $\mathscr{A}_i$ и $\mathscr{C}_i$ – это версии введенных выше операторов $\mathscr{A}$ и $\mathscr{C}$, действующие по переменной x_i (зависящие от K_i и n_i), $i = \overline{1, N}$, а $f^h \in S^{(\mathbf{n})}_{\mathbf{K}}$ – МКЭ-усреднение f.

Напомним, что случай неоднородных краевых условий Дирихле $u(x) = b(x)$ на $\partial\Omega$ в (3.4) мог бы быть легко охвачен сведением к (3.5) с модифицированным вектором f^h, зависящим от аппроксимации b^h для b.

Чтобы вычислить ее решение, F_n-преобразования из теоремы 2.2 могут быть применены двояко.

(а) Рассмотрим кратное разложение $f^h \in S^{(\mathbf{n})}_{\mathbf{K}}$ типа (2.20)

$$f^h = \sum_{i=1}^{N}\sum_{k_i=0}^{K_i-1}\sum_{l_i=1}^{n_i-\delta_{k_i0}} \tilde{f}^h_{k_1l_1,\ldots,k_Nl_N}\mathscr{C}_1 s^{(l_1)}_{1,k_1}\ldots\mathscr{C}_N s^{(l_N)}_{N,k_N}. \tag{3.6}$$

Тогда разложение решения имеет следующий вид:

$$v = \sum_{i=1}^{N}\sum_{k_i=0}^{K_i-1}\sum_{l_i=1}^{n_i-\delta_{k_i0}} \frac{\tilde{f}^h_{k_1l_1,\ldots,k_Nl_N}}{4h_1^{-2}\lambda^{(l_1)}_{1,k_1} + \ldots + 4h_N^{-2}\lambda^{(l_N)}_{N,k_N} + \alpha} s^{(l_1)}_{1,k_1}\ldots s^{(l_N)}_{N,k_N}. \tag{3.7}$$

Здесь $\{\lambda^{(l_i)}_{i,k_i}, s^{(l_i)}_{i,k_i}\}$ являются версиями введенных выше собственных пар $\{\lambda^{(l)}_k, s^{(l)}_k\}$ по отношению к x_i. Формула корректно определена при всех $\mathbf{h} := (h_1, \ldots, h_N)$, если $\alpha \in \mathbb{R}$ и $\alpha > -\pi^2(X_1^{-2} + \ldots + X_N^{-2})$, или по крайней мере при достаточно малых $\mathbf{h}$ в противном случае, или если $\alpha \in \mathbb{C}\backslash\mathbb{R}$.

Алгоритм (а) состоит из двух достаточно стандартных шагов:

(1) нахождение коэффициентов разложения (3.6) для f^h (посредством прямых FC_n-преобразований по $x_1, \ldots, x_N$);

(2) нахождение v по коэффициентам его разложения (3.7) (посредством обратных F_n-преобразований по $x_1, \ldots, x_N$).

(б) Рассмотрим разложение f^h типа (2.20) по $x_2,\ldots, x_N$, т.е.

$$f^h = \sum_{i=2}^{N}\sum_{k_i=0}^{K_i-1}\sum_{l_i=1}^{n_i-\delta_{kj0}} \tilde{f}^h_{k_1 l_1,\ldots,k_N l_N} \mathscr{C}_2 s^{(l_2)}_{2,k_2} \ldots \mathscr{C}_N s^{(l_N)}_{N,k_N}, \tag{3.8}$$

теперь с коэффициентами $\tilde{f}^h_{k_2 l_2,\ldots,k_N l_N} \in S^{(n_1)}_{K_1}$. Тогда коэффициенты $v_{kl} \in S^{(n_1)}_{K_1}$ аналогичного разложения решения $v \in S^{(\mathbf{n})}_{\mathbf{K}}$, а именно

$$v = \sum_{i=2}^{N}\sum_{k_i=0}^{K_i-1}\sum_{l_i=1}^{n_i-\delta_{kj0}} v_{k_2 l_2,\ldots,k_N l_N} s^{(l_2)}_{2,k_2} \ldots s^{(l_N)}_{N,k_N}, \tag{3.9}$$

служат решениями 1D задач по x_1

$$[4h_1^{-2}\mathscr{A}_1 + (4h_2^{-2}\lambda^{(l_2)}_{k_2} + \ldots + 4h_N^{-2}\lambda^{(l_N)}_{k_N} + \alpha)\mathscr{C}_1]v_{k_2 l_2,\ldots,k_N l_N} = \tilde{f}^h_{k_2 l_2,\ldots,k_N l_N}. \tag{3.10}$$

Их матрицы симметричны и положительно определены, если $\alpha \in \mathbb{R}$ и $\alpha > -\pi^2(X_1^{-2} + \ldots + X_N^{-2})$, либо невырожденны по крайней мере при достаточно малых $\mathbf{h}$ в противном случае, или если $\alpha \in C\backslash\mathbb{R}$.

Алгоритм (б) состоит из трех достаточно стандартных шагов:

(1) нахождение коэффициентов разложения (3.8) для f^h (посредством прямых FC_n-преобразований по $x_2, \ldots, x_N$);

(2) решение набора независимых 1D задач (3.10) для коэффициентов разложения v;

(3) нахождение v по коэффициентам его разложения (3.9) (посредством обратных F_n-преобразований по $x_2, \ldots, x_N$).

Реализация алгоритмов (а) и (б) требует соответственно $O(K_1 \ldots K_N \log_2(K_1 \ldots K_N))$ и $O(K_1 \ldots K_N \log_2(K_2 \ldots K_N))$ арифметических действий. Здесь не анализируется зависимость от $\mathbf{n}$, но она умеренная согласно численным экспериментам в следующем разделе.

Важно, что они могут быть применены для решения разнообразных эволюционных уравнений в частных производных (УрЧП) таких, как уравнение теплопроводности, волновое уравнение или уравнение Шрёдингера, поскольку для их неявных по времени дискретизаций обычно получаются задачи типа (3.5) на верхнем слое по времени.

Кроме того, алгоритм (б) непосредственно обобщается на случаи более общих уравнений, чем в (3.4), с коэффициентами зависящими от x_1 (что существенно, в частности, в полярных или цилиндрических координатах), различных краевых условий при $x_1 = 0, X_1$ и неравномерной сетки по x_1 [3]. Он может быть также применен для сведения 3D задач в цилиндрической области к набору независимых 2D задач в основании цилиндра.

4. ЧИСЛЕННЫЕ ЭКСПЕРИМЕНТЫ

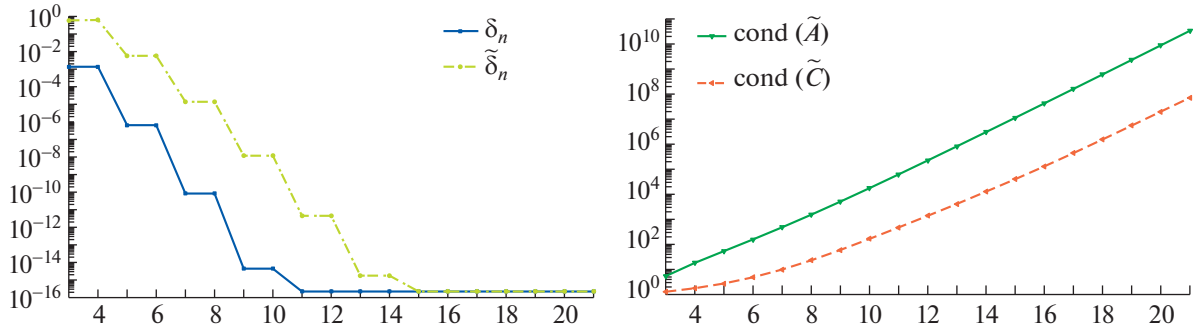
1. Сначала проверим, что собственные значения каждой из задач (1.8) и (1.9) хорошо разделены. Определим их спектральные щели как

$$\min_{1\le l\le n}(\lambda_{0(l+1)} - \lambda_{0(l)}) = \frac{\pi^2}{4} + \delta_n, \qquad \min_{1\le l\le n-2}(\lambda_0^{(l+1)} - \lambda_0^{(l)}) = \frac{3\pi^2}{4} + \tilde{\delta}_n,$$

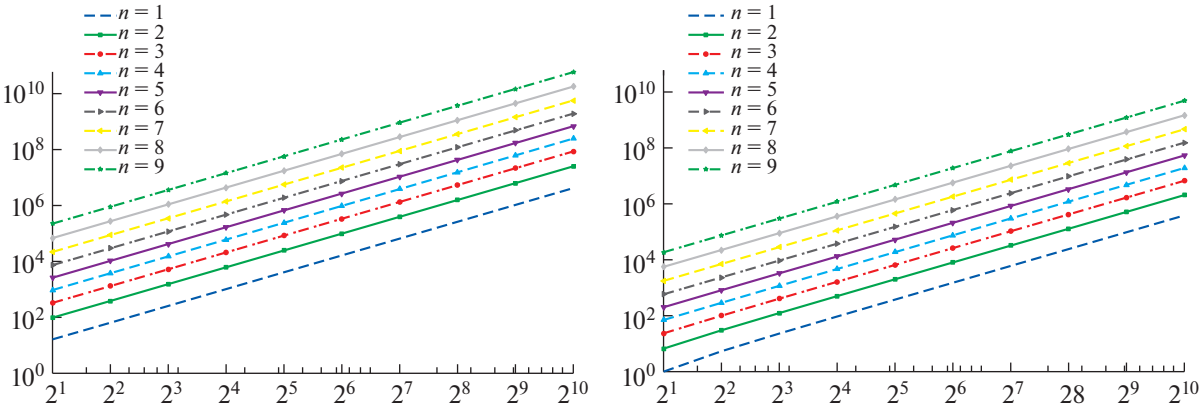
где $S_n =: \{\lambda_{0(l)}\}_{l=1}^{n+1}$, и представим δ_n и $\tilde{\delta}_n$ на фиг. 1 (слева). Слагаемые $\pi^2/4$ и $3\pi^2/4$ являются спектральными щелями (фактически щелями между двумя минимальными собственными значениями) соответствующих задач для ОДУ, см. [11]. Для обеих величин δ_n и $\tilde{\delta}_n$ наблюдается убывание и быстрое стремление к 0 с возрастанием n. Проверено также, что $S_n \cap \tilde{S}_n = \emptyset$ при всех $2 \le n \le 21$.

Представим также спектральные числа обусловленности $\tilde{A}$ и $\tilde{C}$ на фиг. 1 (справа) и обратим внимание на их быстрый рост с возрастанием n (к сожалению).

Отметим, что все вычисления были выполнены на ординарном ноутбуке ASUS-U36S с Intel Core i3-2350M CPU 2.3 GHz, 8 Gb, Win 10 x64. Для реализации алгоритмов были разработаны ко-



Фиг. 1. Числа δ_n и $\tilde{\delta}_n$, связанные с минимальной спектральной щелью в задачах на собственные значения (1.8) и (1.9) (слева), а также cond $\tilde{A}$ и cond $\tilde{C}$ (справа) для задачи (1.9), в зависимости от n.



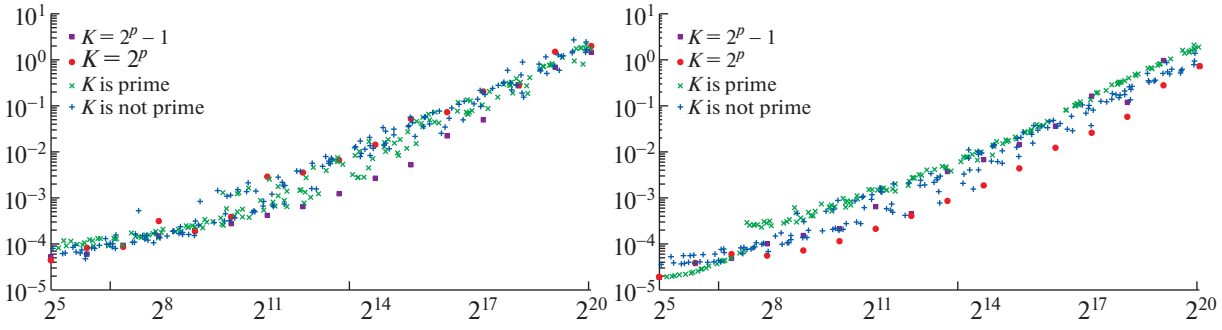
Фиг. 2. Числа обусловленности локальных (слева) и глобальных (справа) матриц в 1D задаче (3.2) при $\alpha = 1$ в зависимости от $K = 2, 4, \ldots, 1024$ для $n = \overline{1,9}$.

ды на Matlab R2016a, и подчеркнем, что при этом были применены несколько базовых и продвинутых методов векторизации кодов [13], чтобы заметно ускорить их выполнение.

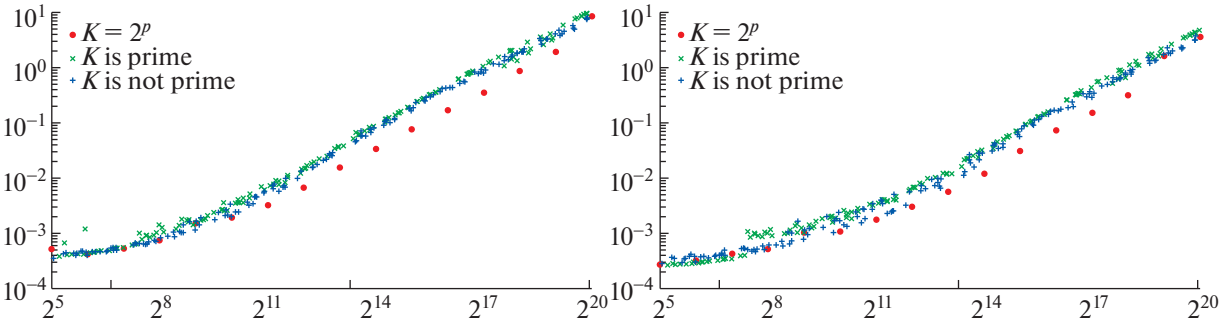
В случае 1D задачи для ОДУ (3.2) при $\alpha = 1$ дадим числа обусловленности глобальной матрицы МКЭ $4h^{-2}\mathcal{A} + \alpha\mathcal{C}$ при $\alpha = 1$ и ее локальной версии в зависимости от $K = 2, 4, \ldots, 1024$ при $n = \overline{1,9}$ на фиг. 2 для последующих ссылок. Отметим их быстрый рост с возрастанием K.

Далее проанализируем время выполнения F_n-преобразований в зависимости от выбора K с его возрастанием до весьма высоких значений 2^{20}. Рассмотрим три способа выбора K: степени двойки ($K = 2^p$), простые числа и составные числа (не степени двойки). Простые и составные значения K выбираются случайным образом между двумя последовательными степенями двойки, и сравнение выполняется в стиле [14]. Используются внешние функции `comp_dst` для DST-I и DST-III и `comp_dct` для DCT-III из Large Time-Frequency Analysis Toolbox [15], основанном на библиотеке FFTW [16], которая также используется в **Matlab**-функции `fft` для БДПФ.

Времена выполнения для DST-I и DST-III, а также наших прямого и обратного F_n-преобразований в случае $n = 5$ (для определенности) даны соответственно на фиг. 3 и 4. Время выполнения для нахождения пар $\{\lambda_k^{(l)}, p_k^{(l)}\}$ не принимается во внимание, поскольку оно становится пренебрежимо малым при многократном применении преобразований при фиксированном K, как это требуется ниже. Для наших преобразований времена выполнения соответственно выше (чем для DST-I и DST-III) из-за кратного использования БДПФ и некоторых дополнительных вычислений. Обратное F_n-преобразование требует меньше времени, чем прямое. Кроме того, наилучшие



Фиг. 3. Времена выполнения (в секундах) для DST-I (слева) и DST-III (справа) для нескольких способов выбора K между 2^5 и 2^{20}.



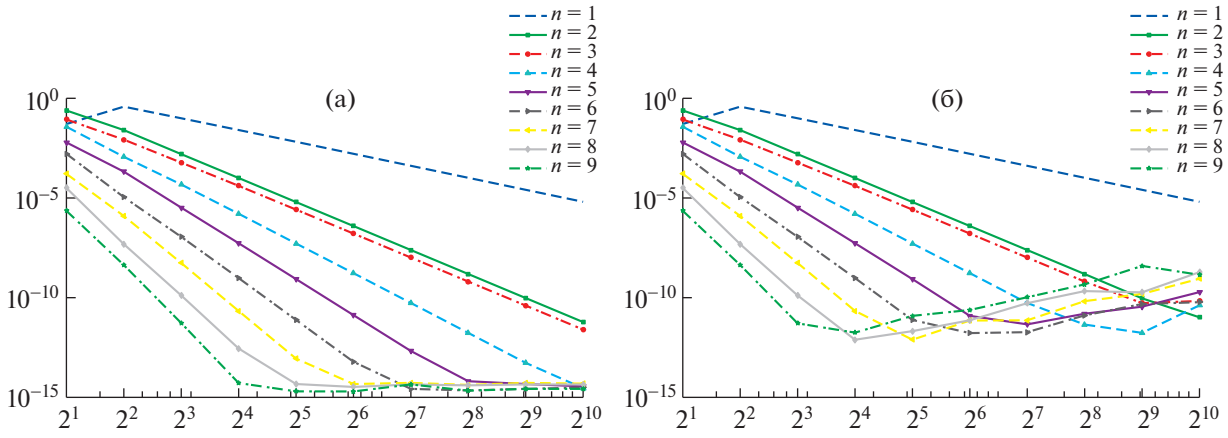
Фиг. 4. Времена выполнения (в секундах) для прямого (слева) и обратного (справа) F_n-преобразований, $n = 5$, для нескольких способов выбора K между 2^5 и 2^{20}.

результаты получаются в основном для $K = 2^p$, хотя это не так для DST-I (к счастью, оно применяется в оптимальном случае $K = 2^p - 1$). Для двух других выборов K времена выполнения похуже, но близки, и разница между всеми ними меньше, чем в случае как DST-I, так и DST-III. Эти результаты положительного плана. Отметим, что данные на фиг. 4 могут быть аппроксимированы линейными функциями при $2^5 \le K \le 2^{10}$ и $2^{10} \le K \le 2^{20}$, но с довольно плоским наклоном в первом случае и заметно более крутым во втором.

Замечание 4.1. Последнее явление объясняется продвинутой архитектурой современных процессоров, включая кэш-память, потоковые SIMD-расширения, продвинутые векторные расширения набора команд и т.д., а также применением указанных выше реализаций БДПФ высокого качества.

2. Наши основные вычислительные результаты относятся к решению задачи (3.4) как для $N = 2$, так и $N = 3$, при $\alpha = 1$ и $X_i = 1$. Возьмем для простоты $K_i = K$ и $n_i = n$. Для вычисления f^h применим кратные квадратурные формулы Гаусса с $n+1$ узлами по x_i. Собственные пары $\{\lambda_k^{(l)}, p_k^{(l)}\}$ 1D задач вычисляются с четверной точностью (с использованием Mathematica) для повышения устойчивости по отношению к ошибкам округления. Отметим, что система (3.5) содержит $(Kn-1)^N$ неизвестных. Для сравнения включим в анализ простейший и известный случай $n = 1$, реализованный в коде единым образом.

Рассмотрим сначала 2D случай ($N = 2$) и возьмем гладкое точное решение $u(x) := \sin(2\pi x_1)\sin(3\pi x_2)\cosh(\sqrt{2}x_1 - x_2)$. Вычислим МКЭ-решение для различных значений n и K для того, чтобы изучить поведение его (абсолютной) погрешности, см. фиг. 5 и табл. 1 (где значения R_C меньше 3 опущены). Поведение погрешности в равномерной сеточной норме для алгоритма (а) стандартно: скорость убывания R_C в основном пропорциональна $(n+1)^2$, за исключе-



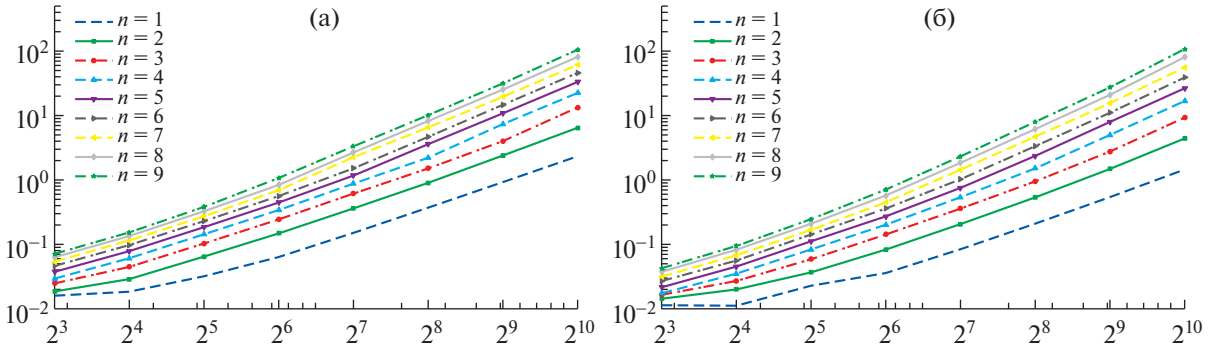
Фиг. 5. 2D случай. Погрешности в равномерной сеточной норме для алгоритмов (а) и (б) в зависимости от $K = 2, 4, \ldots, 1024$ при $n = \overline{1, 9}$.

нием случая $n = 2$, когда она выше и аналогична $n = 3$ (такое исключение ранее уже отмечалось в [17]). Любопытно, что при $n = 9$ и минимальном $K = 2$ погрешность уже меньше, чем при $n = 1$ и максимальном $K = 1024$. Разумеется, погрешность не может быть лучше, чем уровень ошибок округления, который достигается тем быстрее, чем выше $5 \le n \le 9$. Наблюдается почти полное отсутствие влияния ошибок округления.

Таблица 1. 2D случай. Погрешности в равномерной сеточной норме и их отношения R_C в зависимости от $K = 2, 4, \ldots, 1024$ и $n = \overline{1, 9}$ для алгоритма (а)

K	$n = 1$	R_C	$n = 2$	R_C	$n = 3$	R_C	$n = 4$	R_C	$n = 5$	R_C
2	5.1×10^{-2}	–	2.4×10^{-1}	–	8.7×10^{-2}	–	3.7×10^{-2}	–	6.1×10^{-3}	–
4	3.8×10^{-1}	–	2.5×10^{-2}	9.6	8.4×10^{-3}	10.3	1.2×10^{-3}	32.2	2.1×10^{-4}	29.3
8	1.0×10^{-1}	3.8	1.6×10^{-3}	16.1	5.9×10^{-4}	14.2	4.7×10^{-5}	24.5	3.3×10^{-6}	63.8
16	2.6×10^{-2}	3.9	1.0×10^{-4}	15.7	4.1×10^{-5}	14.6	1.6×10^{-6}	29.3	5.4×10^{-8}	60.8
32	6.6×10^{-3}	3.9	6.2×10^{-6}	16.1	2.6×10^{-6}	15.6	5.2×10^{-8}	31.1	8.5×10^{-10}	63.2
64	1.6×10^{-3}	4.0	3.9×10^{-7}	15.9	1.6×10^{-7}	15.9	1.7×10^{-9}	30.6	1.3×10^{-11}	64.0
128	4.1×10^{-4}	4.0	2.4×10^{-8}	16.0	1.0×10^{-8}	16.0	5.4×10^{-11}	31.4	2.1×10^{-13}	63.1
256	1.0×10^{-4}	4.0	1.5×10^{-9}	16.0	6.4×10^{-10}	16.0	1.7×10^{-12}	31.7	6.4×10^{-15}	32.8
512	2.6×10^{-5}	4.0	9.6×10^{-11}	16.0	4.0×10^{-11}	16.0	5.4×10^{-14}	31.7	4.7×10^{-15}	–
1024	6.4×10^{-6}	4.0	6.0×10^{-12}	16.0	2.5×10^{-12}	16.0	2.9×10^{-15}	18.6	3.6×10^{-15}	–

K	$n = 6$	R_C	$n = 7$	R_C	$n = 8$	R_C	$n = 9$	R_C
2	1.6×10^{-3}	–	1.6×10^{-4}	–	3.2×10^{-5}	–	2.3×10^{-6}	–
4	1.1×10^{-5}	148.2	1.3×10^{-6}	129.3	4.8×10^{-8}	657.8	4.3×10^{-9}	521.5
8	1.1×10^{-7}	98.0	5.5×10^{-9}	229.1	1.3×10^{-10}	373.8	5.3×10^{-12}	817.1
16	9.6×10^{-10}	117.1	2.2×10^{-11}	254.6	2.8×10^{-13}	459.0	5.2×10^{-15}	1019.3
32	7.6×10^{-12}	125.6	8.8×10^{-14}	245.2	4.7×10^{-15}	60.4	2.0×10^{-15}	–
64	6.1×10^{-14}	125.5	4.7×10^{-15}	18.8	3.3×10^{-15}	–	2.0×10^{-15}	–
128	2.7×10^{-15}	22.8	5.3×10^{-15}	–	4.4×10^{-15}	–	4.4×10^{-15}	–
256	2.2×10^{-15}	–	4.2×10^{-15}	–	4.0×10^{-15}	–	2.2×10^{-15}	–
512	2.7×10^{-15}	–	5.3×10^{-15}	–	4.4×10^{-15}	–	2.7×10^{-15}	–
1024	3.1×10^{-15}	–	4.9×10^{-15}	–	4.4×10^{-15}	–	2.7×10^{-15}	–



Фиг. 6. 2D случай. Времена выполнения (в секундах) для алгоритмов (а) и (б) в зависимости от $K = 8,16,...,1024$ при $n = \overline{1,9}$.

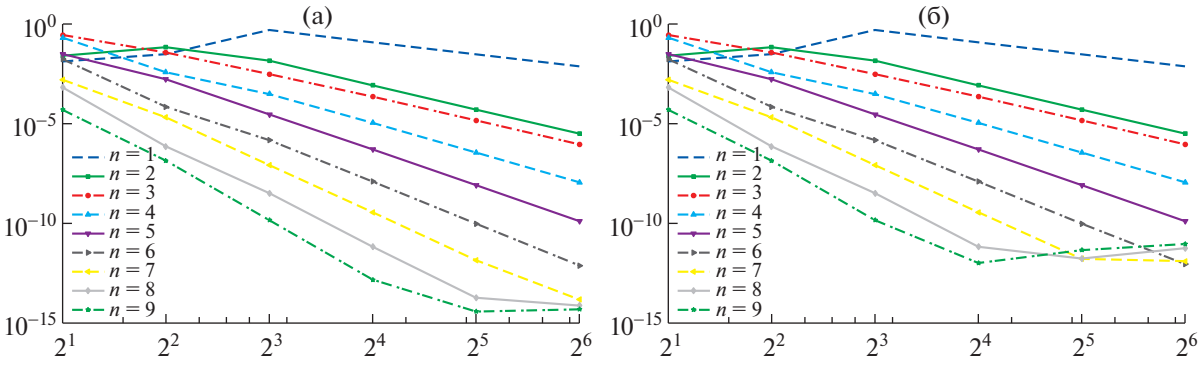
Для алгоритма (б) ситуация аналогична, но только до уровня погрешности $\sim 10^{-11}$. Как только это значение достигается при фиксированном $3 \le n \le 9$, далее наблюдается рост погрешности с возрастанием K, что означает ощутимое влияние ошибок округления. Оно происходит из-за соответствующего роста чисел обусловленности в системе (3.10) с возрастанием K или n, см. выше фиг. 2. Как следствие, алгоритм (а) предпочтительнее (б), если требуется очень высокая точность. Отметим, что при использовании только двойной точности вычислений уровень наилучшей погрешности становится $\sim 10^{-12}$–10^{-13} и результаты остаются устойчивыми для алгоритма (а), но они практически не меняются для алгоритма (б). Поэтому двойная точность всех вычислений возможна, если указанная точность достаточна (что имеет место в большинстве приложений).

Таблица 2. 2D случай. Отношения времен выполнения для алгоритма (а) в зависимости от $K = 8,16,\ldots,1024$ при $n = \overline{1,9}$

K	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$
8	–	–	–	1.46	1.6	1.97	2	2.02	2.01
16	1.15	1.55	1.8	2.07	2.1	2.1	2.13	2.14	2.15
32	1.74	2.24	2.31	2.35	2.35	2.35	2.37	2.42	2.49
64	1.99	2.29	2.37	2.38	2.41	2.45	2.55	2.57	2.81
128	2.37	2.44	2.51	2.57	2.62	2.69	3.24	3.19	3.1
256	2.46	2.49	2.46	2.53	3.07	3.08	2.9	3.08	3.05
512	2.49	2.66	2.66	3.33	3.02	3.15	2.99	3.08	3.11
1024	2.52	2.69	3.33	3.04	3.07	3.13	3.14	3.22	3.34

Таблица 3. 2D случай. Отношения времен выполнения для алгоритма (б) в зависимости от $K = 8,16,\ldots,1024$ при $n = \overline{1,9}$

K	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$
8	–	–	–	1.17	1.28	1.97	1.99	1.99	2.01
16	–	1.39	1.64	2.02	2.09	2.08	2.13	2.2	2.22
32	2.04	1.84	2.17	2.38	2.47	2.54	2.49	2.53	2.61
64	1.57	2.26	2.42	2.42	2.44	2.52	2.67	2.74	2.88
128	2.34	2.49	2.52	2.67	2.77	2.85	3.21	3.21	3.27
256	2.5	2.6	2.64	2.8	3.13	3.27	3.25	3.34	3.44
512	2.58	2.76	2.91	3.31	3.37	3.29	3.31	3.4	3.43
1024	2.68	2.98	3.36	3.36	3.33	3.53	3.58	3.84	3.94



Фиг. 7. 3D случай. Погрешности в равномерной сеточной норме для алгоритмов (а) и (б) в зависимости от $K = 2, 4, \ldots, 64$ при $n = \overline{1,9}$.

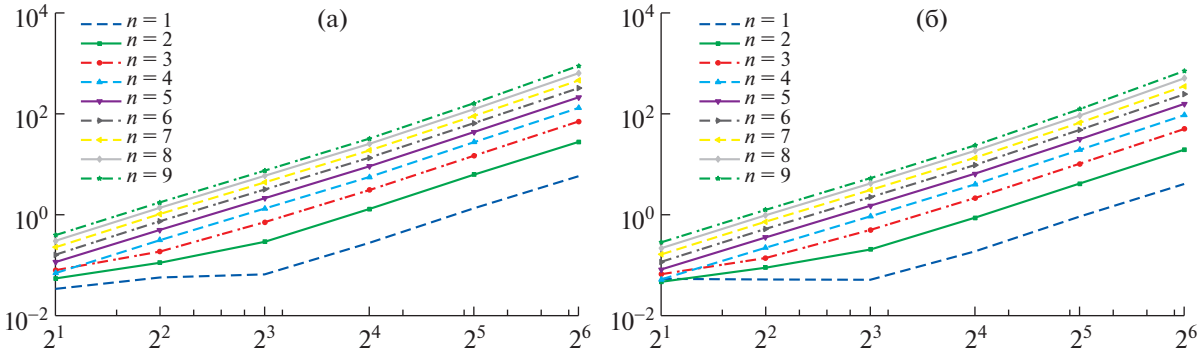
Отметим, что устойчивость метода должен повысить переход в разделе 1 от лагранжева базиса к ортогональным многочленам.

Проанализируем также времена выполнения обоих алгоритмов (с использованием кратного запуска кода и медианного времени их выполнения), см. фиг. 6 и табл. 2 и 3 для одних и тех же K и n. Очевидно, что они не зависят от указанного выше специального выбора u. Время выполнения для вычисления пар $\{\lambda_k^{(l)}, p_k^{(l)}\}$ во внимание не принимается, поскольку рассматривается случай, когда они вычисляются заранее и хранятся (напомним, что они не зависят от данных задачи для УрЧП (3.4)). Обратим внимание на слабо сверхлинейную зависимость времен от K и их слабый монотонный рост по n. Отметим, что все отношения последовательных времен исполнения в обеих таблицах даже меньше, чем нижняя граница 4 для соответствующих теоретических отношений (отношения меньше 1 опущены); см. в этой связи замечание 4.1. Вопреки теоретическим ожиданиям почти все отношения для алгоритма (а) меньше, чем для (б). Отношения растут

Таблица 4. 3D случай. Погрешности в равномерной сеточной норме и их отношения R_C в зависимости от $K = 2, 4, \ldots, 64$ и $n = \overline{1,9}$ для алгоритма (а)

K	$n=1$	R_C	$n=2$	R_C	$n=3$	R_C	$n=4$	R_C	$n=5$	R_C
2	1.3×10^{-2}	–	2.6×10^{-2}	–	2.8×10^{-1}	–	2.1×10^{-1}	–	3.1×10^{-2}	–
4	3.1×10^{-2}	–	6.9×10^{-2}	–	3.7×10^{-2}	7.6	3.8×10^{-3}	54.6	1.7×10^{-3}	18.5
8	5.0×10^{-1}	–	1.5×10^{-2}	4.7	3.1×10^{-3}	12.2	3.0×10^{-4}	12.5	2.9×10^{-5}	57.8
16	1.2×10^{-1}	4.2	8.4×10^{-4}	17.4	2.3×10^{-4}	13.4	1.1×10^{-5}	27.7	5.1×10^{-7}	56.8
32	3.0×10^{-2}	4.0	5.1×10^{-5}	16.4	1.5×10^{-5}	15.5	3.6×10^{-7}	30.6	8.3×10^{-9}	62.0
64	7.5×10^{-3}	4.0	3.2×10^{-6}	16.0	9.2×10^{-7}	16.0	1.2×10^{-8}	31.1	1.3×10^{-10}	64.0

K	$n=6$	R_C	$n=7$	R_C	$n=8$	R_C	$n=9$	R_C
2	1.7×10^{-2}	–	1.6×10^{-3}	–	6.6×10^{-4}	–	5.0×10^{-5}	–
4	7.0×10^{-5}	245.2	2.1×10^{-5}	77.3	7.2×10^{-7}	909.9	1.4×10^{-7}	355.4
8	1.5×10^{-6}	46.9	8.4×10^{-8}	248.1	3.3×10^{-9}	221.1	1.4×10^{-10}	961.4
16	1.3×10^{-8}	119.5	3.6×10^{-10}	230.7	6.7×10^{-12}	486.8	1.5×10^{-13}	957.3
32	9.7×10^{-11}	128.9	1.4×10^{-12}	254.2	1.9×10^{-14}	360.8	3.8×10^{-15}	40.1
64	7.8×10^{-13}	124.9	1.5×10^{-14}	94.6	7.5×10^{-15}	–	4.9×10^{-15}	–



Фиг. 8. 3D случай. Времена выполнения (в секундах) для алгоритмов (а) и (б) в зависимости от $K = 2,4,\ldots,64$ при $n = \overline{1,9}$.

с возрастанием K и n, и для алгоритма (б) их наибольшее значение уже близко к 4. При максимальных $K = 1024$ и $n = 9$, система (3.5) содержит почти 85×10^6 неизвестных, но требуется только менее 2 мин для ее решения, что является отличным результатом.

В заключение рассмотрим наиболее интересный 3D случай ($N = 3$) и возьмем точное решение $u(x) := \sin(2\pi x_1)\sin(3\pi x_2)\sin(4\pi x_3)\cosh(\sqrt{2}x_1 - x_2 + x_3/\sqrt{3})$. Опять вычислим МКЭ-решение для различных значений $K = 2,4,\ldots,64$ и $n = \overline{1,9}$ и изучим поведение погрешности, см. фиг. 7 и табл. 4 (где значения R_C меньше 3 опущены). Здесь можно сделать в основном те же самые выводы, что и в 2D случае. Отметим, что более плохие свойства устойчивости алгоритма (б) проявляются только при высоких $n = 8,9$, поскольку берется намного меньшее максимальное значение K.

Времена выполнения в 3D случае представлены на фиг. 8 и в табл. 5 и 6. Опять естественны выводы, аналогичные сделанным в 2D случае. Все отношения в обеих таблицах заметно ниже,

Таблица 5. 3D случай. Отношения времен выполнения для алгоритма (а) в зависимости от $K = 4,8,\ldots,64$ при $n = \overline{1,9}$

K	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$
4	1.68	2.05	2.34	4.51	4.35	4.62	4.58	4.52	4.49
8	1.15	2.6	3.82	4.23	4.23	4.25	4.22	4.29	4.24
16	4.21	4.47	4.31	4.22	4.29	4.27	4.29	4.3	4.31
32	4.85	4.79	4.8	4.89	4.78	4.84	4.83	4.86	5.03
64	4.32	4.46	4.78	4.77	4.88	4.93	5.05	5.16	5.52

Таблица 6. 3D случай. Отношения времен выполнения для алгоритма (б) в зависимости от $K = 4,8,\ldots,64$ при $n = \overline{1,9}$

K	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$
4	–	1.9	2.09	4.24	4.32	4.51	4.47	4.51	4.41
8	–	2.29	3.6	4.19	4.25	4.23	4.23	4.26	4.25
16	3.65	4.28	4.29	4.3	4.32	4.35	4.36	4.44	4.46
32	4.89	4.75	4.78	4.81	4.81	4.96	4.94	5	5.22
64	4.43	4.68	4.87	4.89	5.01	5.08	5.31	5.47	5.71

чем нижняя граница 8 для соответствующих теоретических отношений. Важно, что для максимальных $K = 64$ и $n = 9$ система (3.5) содержит более, чем 190×10^6 неизвестных, но требуется только менее 15 мин для ее решения, что является хорошим результатом (результат менее быстрый, чем в 2D случае, из-за неэффективной Matlab-реализации вложенных циклов).

СПИСОК ЛИТЕРАТУРЫ

1. *Сьярле Ф.* Метод конечных элементов для эллиптических задач. М.: Мир, 1980.
2. *Swarztrauber P.N.* The methods of cyclic reduction, Fourier analysis and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle // SIAM Review. 1977. V. 19. № 3. P. 490–501.
3. *Самарский А.А., Николаев Е.В.* Методы решения сеточных уравнений. М.: Наука, 1978.
4. *Kwan Y.-Y., Shen J.* An efficient direct parallel spectral-element solver for separable elliptic problems // J. Comput. Phys. 2007. V. 225. P. 1721–1735.
5. *Bialecki B., Fairweather G., Karageorghis A.* Matrix decomposition algorithms for elliptic boundary value problems: a survey // Numer. Algor. 2011. V. 56. P. 253–295.
6. *Britanak V., Rao K.R., Yip P.* Discrete cosine and sine transforms: general properties, fast algorithms and integer approximations. Oxford: Academic Press – Elsevier, 2007.
7. *Злотник А.А., Злотник И.А.* Быстрый прямой алгоритм реализации метода конечных элементов высокого порядка на прямоугольниках для краевых задач для уравнения Пуассона // Докл. АН. 2017. Т. 473. № 3. С. 131–137.
8. *Кузнецов Ю.А.* Вычислительные методы в подпространствах // В сб.: Вычислительные процессы и системы. Вып. 2. Под ред. Г.И. Марчука. М.: Наука, 1985. С. 265–350.
9. *Du K., Fairweather G., Sun W.* Matrix decomposition algorithms for arbitrary order C^0 tensor product finite element systems // J. Comput. Appl. Math. 2015. V. 275. P. 162–182.
10. *Дьяконов Е.Г.* Минимизация вычислительной работы. Асимптотически оптимальные алгоритмы для эллиптических задач. М.: Наука, 1989.
11. *Zlotnik A., Zlotnik I.* Finite element method with discrete transparent boundary conditions for the time-dependent 1D Schrödinger equation // Kinetic Relat. Models. 2012. V. 5. № 3. P. 639–667.
12. *Ducomet B., Zlotnik A., Zlotnik I.* The splitting in potential Crank-Nicolson scheme with discrete transparent boundary conditions for the Schrödinger equation on a semi-infinite strip // ESAIM: Math. Model. Numer. Anal. 2014. V. 48. № 6. P. 1681–1699.
13. *Altman Y.M.* Accelerating MATLAB Performance: 1001 Tips to Speed up MATLAB Programs. New York: Chapman and Hall/CRC, 2014.
14. *Eddins S.* Timing the FFT, http://blogs.mathworks.com/steve/2014/04/07/timing-the-fft/
15. *Průša Z., Søndergaard P.L., Holighaus N., Wiesmeyr C., Balazs P.* The large time-frequency analysis toolbox 2.0. Sound, music, and motion // Lecture Notes in Computer Science. V. 2014. P. 419–442.
16. *Frigo M., Johnson S.G.* The design and implementation of FFTW3 // Proc. IEEE. 2005. V. 93. P. 216–231.
17. *Du K., Fairweather G., Nguyen Q.N., Sun W.* Matrix decomposition algorithms for the C^0-quadratic finite element Galerkin method // BIT Numer. Math. 2009. V. 49. P. 509–526.