УДК 517.518.87

Решение интегральных уравнений на многопроцессорных вычислительных системах

М.Д. Рамазанов*, Д.Я. Рахматуллин, Л.З. Валеева, Е.Л. Банникова
Институт математики с ВЦ УНЦ РАН
450077 Россия, Уфа, Чернышевского, 112[1]

Получена 24.06.2008, окончательный вариант 28.07.2008, принята к печати 10.03.2009

Рассматриваются уравнения Фредгольма по областям различных форм. Приводится описание алгоритма приближенного вычисления интегралов с использованием многопроцессорной техники.

Ключевые слова: многопроцессорные системы, приближенное вычисление интегралов, многопроцессорная техника.

1. Введение

Мы рассматриваем уравнения Фредгольма второго рода

$$u(x) - \int\limits_{\Omega} dyK(x,y)u(y) = f(x), \tag{1}$$

записанные для функций двух переменных в ограниченной области произвольной формы, $\overline{\Omega} \Subset \mathbb{R}^2$. Граница области $\Gamma = \partial\Omega$, функции $K(x,y)$, $f(x)$ и решение предполагаются гладкими, M раз непрерывно дифференцируемыми по своим аргументам. Норма интегрального оператора в пространстве $C(\overline{\Omega})$ считается малой, чтобы можно было применить метод последовательных приближений, $\max\limits_{x\in\overline{\Omega}} \int\limits_{\Omega} dy|K(x,y)| = \theta < 1$. Тогда существует единственное решение – непрерывная функция, $u \in C(\overline{\Omega})$, и

$$u(x) = \lim_{s\to\infty} u_s(x), \quad u_s(x) = f(x) + \int\limits_{\Omega} dyK(x,y)u_{s-1}(y), \quad u_0 = f(x). \tag{2}$$

Из самого вида интегрального уравнения (1) следует гладкость этого решения, $u \in C^M(\overline{\Omega})$. А итерационное равенство (2) позволяет обосновать равномерную ограниченность $C^M(\overline{\Omega})-$норм последовательности $\{u_s(x)|s = 1, 2...\}$.

$$\|u_s\|_{C^M} \le \|u_{s-1}\|_C \cdot \max_{x\in\overline{\Omega}} \int\limits_{\Omega} dy \sum_{|\alpha\le m|} |D_x^\alpha K(x,y)| + \|f\|_{C^M} \le \mathcal{K}.$$

Это дает возможность в вычислениях интегрального члена итерационного равенства (2) использовать кубатурные формулы с равномерными оценками в $C^M(\overline{\Omega})-$ пространствах. Мы применим решетчатые кубатурные формулы с ограниченным пограничным слоем порядка M (ОПС-формулы). При наложенных ограничениях они являются асимптотически

* Corresponding author E-mail address: ramazanov@yandex.ru

оптимальными в нормах пространств W_p^m и C^m с любым $m < M$ (в конкретной, одной из эквивалентных, нормировке и, конечно, с $m > n/p$). Порядок приближения интегрального слагаемого этими формулами – $O(h^m)$ с соответствующей примененному пространству наилучшей константой при этом порядке.

Отметим также, что эти формулы условно ненасыщаемы для всех $m < M$, то есть формулы остаются асимптотически оптимальными при любой промежуточной гладкости $m \in (0, M)$, и мы можем работать с ними для $K(x, y)$ и $f(x)$ из классов C^m, получая соответственно решения этой же гладкости.

В общем случае область произвольной формы задается равенством
$\Omega = \{x|\Phi(x) > 0\}$. Мы предполагаем, что $\Phi \in C^M(\mathbb{R}^2)$, множество $\{x|\Phi(x) \geqslant 0\}$ является ограниченным, а $D\Phi(x) \neq 0$ при $\Phi(x) = 0$, что обеспечивает гладкость границы. Для удобства программной реализации мы накладываем дополнительные условия $|D\Phi(x)| = 1$ при $\Phi(x) = 0$, достигая его заменой первоначального $\Phi(x)$ на $\Phi(x)/|D\Phi(x)|$.

Решетку узлов кубатурной формулы возьмем кубической $\{hk|k \in \mathbb{Z}^n, \ \ h \in \mathbb{R}_+, h \to 0\}$. Интегральное слагаемое итерационного процесса (2) заменяется кубатурной формулой

$$h^n \sum_{k \in \mathbb{Z}^n, \ \rho(hk,\Omega) \leq Lh} c_k(h) K(hj, hk) u_s(hk)$$

с некоторым $L \geqslant 4M$.

ОПС-свойство этой формулы означает, что для $\rho(hk, \mathbb{R}^2 \setminus \Omega) \geqslant Lh$ коэффициенты равны единице, $c_k(h) \equiv 1$. Заметим, что если $f \in C_0^M(\Omega)$, $K \in C_0^M(\Omega \times \Omega)$, то все коэффициенты формулы можно взять равными единице, при этом пограничный слой будет отсутствовать. Тогда и решение получится из класса $C_0^M(\Omega)$. Мы применяем этот вариант в вычислительных экспериментах при оценке объема работы по вспомогательным расчетам.

Выбранный нами алгоритм последовательных приближений позволяет полностью распараллелить расчеты на каждой отдельной итерации. Поэтому мы можем применить многопроцессорные вычислительные системы. Конкретно расчеты проводились на МВС-15000ВМ и МВС-50К Межведомственного Суперкомпьютерного Центра РАН.

Идея применения решетчатых кубатурных формул для приближения интегралов по областям произвольных форм заключается в локализации вычислений при помощи разбиения области на более простые и преобразовании границ этих локальных областей в плоские – координатные гиперплоскости. Алгоритм локализации (предложенный А.Н. Игнатьевым [1]) и реализующая алгоритм программа автоматического разбиения области описаны в п. 3. Эта программа составлена Л.З. Валеевой [2]. А программа для вычисления интегрального слагаемого составлена Е.Л. Банниковой [3], ее описание и анализ работы приведены в п. 4. Алгоритм приближенного вычисления интегралов дан Д.Я. Рахматуллиным [4], описание приведено в п. 2. Следует отметить, что в описании алгоритма Д.Я. Рахматуллина наложено условие выпуклости области интегрирования. Однако оно было использовано им только для упрощения разбиения области на подобласти более простых форм. Эта проблема локализации решается в общем виде в п. 3 без требования выпуклости области $\Omega = \{x|\Phi(x) > 0\}$, что и применено в п. 4 в алгоритме и программе численного решения интегральных уравнений. В основе решетчатых кубатурных формул лежит алгоритм формул С.Л. Соболева – кубатурных формул с регулярным пограничным слоем (РПС-формул) [5] и алгоритм построения ОПС-формул для локальных областей М.Д. Рамазанова [6].

Метод последовательных приближений разбивает вычисление решения интегрального уравнения на отдельные временные такты. Это мешает полному распараллеливанию алгоритма (естественному для вычисления интеграла по ОПС-формулам). Впоследствии мы

намереваемся преодолеть эту трудность записью приближенных решений рядами Неймана с одновременными расчетами членами этих рядов, обеспечивающими заданную точность приближения решения уравнения (1).

В п. 5 мы подводим итоги исследований по результатам вычислительных экспериментов и даем общие рекомендации по применению разработанных программ.

2. Алгоритм и программа приближенного вычисления интегралов в n-мерном случае

2.1. Постановка задачи

Пусть задана ограниченная область $\Omega \subset \mathbb{R}^n$ с гладкой границей $\Gamma \in C^m$.

Рассмотрим пространство Соболева $\widetilde{W}_p^m(Q),\ p > 1,\ m > \frac{n}{p}$, периодических функций, интегрируемых с p-ой степенью вместе с производными до m-го порядка включительно, в одной из эквивалентных норм:

$$\|g\|_{\widetilde{W}_p^m(Q)} := \left(|g_0|^p + \int_Q dx \left| \sum_{k \in \mathbb{Z}^n \setminus \{0\}} g_k \left(1 + |2\pi k|^2\right)^{m/2} e^{2\pi i x k} \right|^p \right)^{\frac{1}{p}},$$

$$g_k := \int_Q g(x) e^{-2\pi i x k} dx,$$

где Q — единичный гиперкуб: $Q := \{x :\ x_i \in [0;1),\ i = \overline{1,n}\}$. Взяв это пространство за основу, построим новое пространство $\widetilde{W}_p^m(\Omega)$ с нормой

$$\|f\|_{\widetilde{W}_p^m(\Omega)} := \inf_{\substack{g|_\Omega = f|_\Omega, \\ g \in \widetilde{W}_p^m(Q)}} \|g\|_{\widetilde{W}_p^m(Q)}.$$

Требуется вычислить интеграл произвольной функции $f \in \widetilde{W}_p^m(\Omega)$ по области Ω. Мы решаем эту задачу путем приближения интеграла решетчатыми кубатурными формулами с ограниченным пограничным слоем (ОПС-формулами). Поясним эти термины.

Решетчатыми кубатурными формулами называются кубатурные формулы, узлы которых совпадают с узлами некоторой решетки. Мы будем рассматривать последовательности формул, соответствующие последовательностям кубических решеток со сгущающимися узлами $\{hk\}_{k \in \mathbb{Z}^n},\ h \to 0$:

$$K_h^\Omega(f) = h^n \sum_{hk \in \Omega} c_k(h) f(hk), \quad h \to 0.$$

Здесь за знак суммы вынесен масштабный множитель h^n, равный объему элементарной (наименьшей) ячейки, образуемой узлами решетки.

Кубатурная формула обладает ограниченным пограничным слоем, если ее коэффициенты равномерно ограничены по k и h :

$$\exists\, L_1 : \quad \sup_{k,\,h} |c_k(h)| < L_1\,,$$

а в глубине области все коэффициенты равны единице:

$$\exists\, L_2 : \quad \forall h,\ k : \rho(hk, \mathbb{R}^n \setminus \Omega) \geqslant L_2 h \Rightarrow c_k(h) = 1.$$

Для решения данной задачи создан алгоритм, позволяющий вычислить коэффициенты кубатурной формулы, приближающей точное значение интеграла с погрешностью порядка h^m.

2.2. Алгоритм

Зададимся произвольным целым числом M, таким что $M > \frac{n}{p}$. Ниже мы приведем алгоритм (см. [4–9]) вычисления кубатурной формулы, оптимальной по порядку на каждом из пространств $\widetilde{W}_p^m(\Omega)$ с $m = \left(\frac{n}{p}, M\right)$, т.е. универсально оптимальной по порядку на этом классе пространств.

Учитывая оценки

$$\exists C_1: \quad \left\|l_h^\Omega\right\|_{(\widetilde{W}_p^m(\Omega))^*} \geqslant C_1 \left\|l_\infty\right\|_{(\widetilde{W}_p^m(Q))^*} \geqslant C_1 h^m$$

для оптимальности по порядку достаточно иметь оценки сверху:

$$\exists C_2: \quad \|l_h^\Omega\|_{(\widetilde{W}_p^m(\Omega))^*} \leqslant C_2 h^m, \quad h \to 0.$$

По следующей теореме универсально оптимальные по порядку кубатурные формулы с ОПС являются *универсально асимптотически оптимальными.*

Теорема (М. Д. Рамазанов). *Кубатурная формула с ОПС асимптотически оптимальна на каждом пространстве из множества*

$$\left\{\widetilde{W}_p^m(\Omega))\right\} \begin{array}{l} m \in (m_1, m_2), \\ p \in (p_1, p_2) \end{array}$$

тогда и только тогда, когда она оптимальна на каждом из них по порядку, при $\frac{n}{p} < m_1 < m_2 < M,\ 1 < p_1 < p_2 < \infty$.

В технических целях наложим дополнительные ограничения на условия задачи. Пусть Ω — выпуклая область, лежащая внутри единичного гиперкуба вместе с замыканием: $\overline{\Omega} \subset \subset \text{int } Q$, а параметр h стремится к нулю по одной из последовательностей, каждый член которой можно представить в виде N^{-1}, где $N \in \mathbb{N}_+$.

Алгоритм основан на сведении задачи интегрирования по области с гладкой границей к нескольким однотипным задачам интегрирования по единичному гиперкубу. Для нашей области всегда можно подобрать специальное разбиение единицы — конечный набор функций (будем называть их *срезывающими*), удовлетворяющий условиям:

1) они являются периодическими с основным периодом Q и в сумме составляют функцию, тождественно равную единице в фиксированной окрестности $\hat{\Omega}$ области Ω, целиком лежащей в гиперкубе Q (периодически продолженной с основным периодом Q);

2) они принадлежат пространству $\widetilde{C}^M(Q)$;

3) носитель одной из срезывающих функций лежит полностью внутри области Ω, не пересекаясь с ее границей; пересечение носителя каждой из остальных срезывающих функций с границей $\partial\,\Omega$ непусто и проектируется взаимно однозначно и гладко (класса C^M) на одну из координатных гиперплоскостей;

Примем количество срезывающих функций за $\mathrm{T}+1$. Обозначим их как

$$\{\varphi_\tau(x)\}_{\tau=0}^{\mathrm{T}}, \qquad \sum_{\tau=0}^{\mathrm{T}} \varphi_\tau(x)\big|_{x\in\hat{\Omega}} \equiv 1.$$

Функцию $\varphi_0(x)$ подберем так, чтобы множество точек E_{φ_0}, где она равна единице, находилось не ближе, чем на некотором расстоянии ε_0 от границы области:

$$\rho(E_{\varphi_0}, \mathbb{R}^n \backslash \Omega) \geqslant \varepsilon_0, \quad E_{\varphi_0} := \{x:\ \varphi_0(x) \equiv 1\}.$$

Конкретный способ построения функций разбиения единицы приведем ниже.

Введем обозначения $\Upsilon_\tau := \operatorname{supp}\varphi_\tau \cap \overline{\Omega}$ и $\Gamma_\tau := \operatorname{supp}\varphi_\tau \cap \Gamma$.

Интеграл $\int_\Omega dx f(x)$ можно представить в виде суммы интегралов:

$$\int_\Omega dx f(x) \equiv \sum_{\tau=0}^{\mathrm{T}} \int_{\Upsilon_\tau} dx\ \varphi_\tau(x) f(x).$$

Для удобства и универсальности программной реализации область Ω будем задавать неявно — как множество точек, где неотрицательна гладкая функция $\Phi(x)$:

$$\Omega := \{x :\ \Phi(x) \geqslant 0\}, \quad \Phi(x) \in C^M(Q).$$

При этом граница задается как

$$\Gamma := \{x :\ \Phi(x) = 0\}.$$

Пусть градиент функции $\Phi(x)$ на границе не обращается в нуль:

$$\nabla\Phi(x)|_\Gamma \neq 0.$$

Наложенных на $\Phi(x)$ условий достаточно для того, чтобы уравнение для каждого из участков границы в достаточно малой окрестности каждой точки границы можно было разрешить относительно координаты, задающей направление, ортогональное гиперплоскости, на которую мы гладко проектируем Γ_τ :

$$\forall\tau\ \exists j_\tau,\ \exists\gamma_\tau(\hat{x}_{j_\tau}) : \quad \Gamma_\tau \equiv \{x \in \Upsilon_\tau :\ x_{j_\tau} = \gamma_\tau(\hat{x}_{j_\tau})\},$$

где $\hat{x}_{j_\tau} := (x_1, \ldots, x_{j_\tau-1}, x_{j_\tau+1}, \ldots, x_n)$.

Если каждой области Υ_τ сопоставить функционал погрешности

$$l_h^{\Upsilon_\tau}(x) := \chi_{\Upsilon_\tau}(x) - \sum_{\substack{hk \in \Upsilon_\tau, \\ k \in \mathbb{Z}^n}} c_k^\tau(h)\delta(x - hk),$$

то общий функционал погрешности можно собрать из таких функционалов так:

$$l_h^\Omega(x) := \sum_{\tau=0}^{\mathrm{T}} \varphi_\tau(x) l_h^{\Upsilon_\tau}(x).$$

При этом коэффициенты $c_k(h)$ функционала погрешности $l_h^\Omega(x)$ получаются следующим образом:

$$c_k(h) = \sum_{\tau=0}^{\mathrm{T}} \varphi_\tau(x) c_k^\tau(h).$$

Так как в определении функционала погрешности $l_h^\Omega(x)$ каждый функционал $l_h^{\Upsilon_\tau}(x)$ умножен на срезывающую функцию $\varphi_\tau(x)$, мы можем в дальнейшем произвольно доопределять $l_h^{\Upsilon_\tau}(x)$ вне $\operatorname{supp}\varphi_\tau(x)$, не изменяя итоговой суммы.

Для того чтобы функционал погрешности $l_h^\Omega(x)$ был оптимальным по порядку в $\widetilde{W}_p^m(\Omega)$, достаточно оптимальности по порядку каждого из функционалов $l_h^{\Upsilon_\tau}(x)$. Необходимо подобрать для каждого τ коэффициенты $c_k^\tau(h)$ так, чтобы выполнялись оценки $\|l_h^{\Upsilon_\tau}\|_{(\widetilde{W}_p^m(\Omega))^*} \asymp O(h^m)$.

Приведем результат, полученный в предположении, что уравнение для границы Γ_τ задается функцией, выражающей последнюю координату вектора x через остальные:

$$\Gamma_\tau := \{x \in \Upsilon_\tau : \ x_n = \gamma(x')\}, \quad x' := (x_1, \ldots, x_{n-1}),$$

и выполняется $x_n \geqslant \gamma(x'), \quad \forall x \in \Upsilon_\tau$.

Искомые коэффициенты:

$$c_k^\tau := \begin{cases} \sum_{p=1}^{M+1} \frac{1}{p} \sum_{i=1}^{M+1} \eta^{i-1} \sum_{z=1}^{\min\{k_n-\sigma-1, M+1\}} v_{zi} \sum_{r=1}^{\min\{k_n-\sigma-z, M+1\}} v_{rp}, & k_n \geqslant 2+\sigma \\ 0, & k_n \leqslant 1+\sigma. \end{cases},$$

где v_{ij} — элементы матрицы, обратной к матрице, составленной из элементов определителя Вандермонда.

Эту формулу для ускорения вычислений лучше переписать в виде:

$$c_t^\tau(h) = \begin{cases} \sum_{i=0}^{\min\{t_n, M\}} R_{t_n - i} \sum_{l=0}^{M} \eta^l \tilde{v}_{il}, & t_n \geqslant 0 \\ 0, & t_n < 0 \end{cases},$$

$$t' := k', \ t_n := k_n - \sigma - 2; \quad \tilde{v}_{ij} := v_{i+1,\, j+1}, \ i, j = \overline{0, M}; R_i := \sum_{p=0}^{M} \frac{1}{p+1} \sum_{j=0}^{i} \tilde{v}_{jp}, \quad i = \overline{0..M}.$$

Для практической реализации алгоритма необходимо построить $\mathrm{T}+1$ срезывающих функций $\{\varphi_\tau(x)\}_{\tau=0}^{\mathrm{T}}, \ \sum_{\tau=0}^{\mathrm{T}} \varphi_\tau(x)|_{x \in \hat{\Omega}} \equiv 1$, удовлетворяющих наложенным выше условиям.

Так как в алгоритме фактически используются лишь внутренние узлы области Ω, можно не следить за разбиением единицы вне области и не соблюдать условие периодичности срезывающих функций. То же касается и подынтегральной функции $f(x)$ — она не обязана быть периодичной и определенной вне области Ω.

Разбиение единицы произведем в два этапа. Вначале построим центральную срезывающую функцию $\varphi_0(x)$, затем построим разбиение единицы для гиперкуба Q: $\{\psi_\tau(x)\}_{\tau=1}^{\mathrm{T}}, \quad \sum_{\tau=1}^{\mathrm{T}} \psi_\tau(x)|_{x \in Q} \equiv 1$. И, наконец, получим окончательный вид срезывающих функций: $\varphi_\tau(x) := \psi_\tau(x)(1 - \varphi_0(x)), \quad \tau = \overline{1..\mathrm{T}}$.
При этом $\sum_{\tau=0}^{\mathrm{T}} \varphi_t(x) \equiv \varphi_0(x) + \sum_{\tau=1}^{\mathrm{T}} \varphi_\tau(x) \equiv \varphi_0(x) + (1 - \varphi_0(x)) \sum_{\tau=1}^{\mathrm{T}} \psi_\tau(x) \equiv 1, \ x \in Q$.

Нам понадобится вспомогательная функция $\xi(t)$:

$$\xi(t) := \begin{cases} 0, \ t < 0 \\ \int_0^t z(t)dt \,/\, \int_0^1 z(t)dt, \quad z(t) = (t(1-t))^M, \qquad 0 \leqslant t < 1. \\ 1, \ 1 \leqslant t. \end{cases}$$

Центральную срезывающую функцию $\varphi_0(x)$ зададим так:

$$\varphi_0(x) := \xi\left(\frac{\Phi(x)}{\varepsilon_2 - \varepsilon_1} - \frac{\varepsilon_1}{\varepsilon_2 - \varepsilon_1}\right),$$

где $\Phi(x)$ — функция, задающая область Ω.

Тогда $\varphi_0(x) = \begin{cases} 0, \ \Phi(x) \leqslant \varepsilon_1 \\ p \in (0,1), \ \varepsilon_1 < \Phi(x) < \varepsilon_2. \\ 1, \ \varepsilon_2 \leqslant \Phi(x). \end{cases}$

Таким образом, конкретный вид функции $\varphi_0(x)$ зависит как от способа задания области Ω, так и от параметров ε_1 и ε_2.

Построим функции $\{\psi_\tau(x)\}_{\tau=1}^{\mathrm{T}}$. Потребуем, чтобы центр гиперкуба Q принадлежал области Ω. Тогда, учитывая выпуклость области, можно построить ровно $2n$ срезывающих функций (по числу граней гиперкуба), удовлетворяющих условиям, наложенным на них выше. Далее будем считать $\mathrm{T} \equiv 2n$.

Для удобства будем нумеровать координатные оси от 0 до $n-1$, а систему функций $\{\psi_\tau(x)\}_{\tau=1}^{2n}$ заменим на систему

$$\{\tilde{\psi}_{dim,dir}(x)\}_{dim=0..n-1,\ dir=\pm1}.$$

Первый индекс dim (от dimension) означает размерность, второй индекс dir (от direction) — направление (отрицательное или положительное). Будем строить систему срезывающих функций так, чтобы функция $\tilde{\psi}_{dim,dir}(x)$ «опиралась» на грань гиперкуба Q, ортогональную оси x_{dim}, и такой, чтобы $x_{dim} \equiv \frac{dir+1}{2}$ для любой точки x, принадлежащей этой грани. Под выражением «$\tilde{\psi}_{dim,dir}(x)$ опирается на грань» имеется в виду то, что соответствующая срезывающая функция $\tilde{\varphi}_{dim,dir}(x)$, определяемая из

$$\tilde{\varphi}_{dim,dir}(x) := \tilde{\psi}_{dim,dir}(x)(1-\varphi_0(x)), \quad dim = 0..n-1,\ dir = \pm1,$$

должна гладко и взаимнооднозначно проектировать вырезаемый ей участок границы области Ω на координатную плоскость, содержащую эту грань.

Зададим вначале разбиение единицы для двумерного случая. Введем вспомогательные функции:

$$\hat{\psi}(t,x) := \xi(xt)\xi((1-t)x),$$
$$A_{-1}(t,b,c) := -\frac{b}{c}t + b, \quad A_1(t,b,c) := \frac{b}{c}t - \frac{(1-c)b}{c}.$$

Рассмотрим плоскость $\{x_0, x_1\}$. Зададим разбиение единицы:

$$\tilde{\psi}_{0,-1}(x_0,x_1) := \hat{\psi}(x_1, A_{-1}(x_0,b,c)),$$
$$\tilde{\psi}_{0,1}(x_0,x_1) := \hat{\psi}(x_1, A_1(x_0,b,c))\tilde{\psi}_{1,-1}(x_0,x_1) + \tilde{\psi}_{1,1}(x_0,x_1) :=$$
$$:= 1 - (\tilde{\psi}_{0,-1}(x_0,x_1) + \tilde{\psi}_{0,1}(x_0,x_1)).$$

Основой для построения разбиения единицы в многомерном случае будут двумерные разбиения во всевозможных координатных плоскостях. В n-мерном пространстве таких плоскостей ровно $\frac{n(n-1)}{2}$ — число сочетаний из n по 2.

Каждую из плоскостей $\{x_i,\ x_j\}$ будем разбивать двумя функциями:

$$\breve{\psi}_{i,j} := \tilde{\psi}_{i,-1}(x_i,x_j) + \tilde{\psi}_{i,1}(x_i,x_j),$$
$$\breve{\psi}_{j,i} := \tilde{\psi}_{j,-1}(x_i,x_j) + \tilde{\psi}_{j,1}(x_i,x_j).$$

Имеем:

$$\begin{cases} \breve{\psi}_{0,1} + \breve{\psi}_{1,0} = 1 \\ \dots \\ \breve{\psi}_{0,n-1} + \breve{\psi}_{n-1,0} = 1 \\ \breve{\psi}_{1,2} + \breve{\psi}_{2,1} = 1 \\ \dots \\ \breve{\psi}_{1,n-1} + \breve{\psi}_{n-1,1} = 1 \\ \dots \\ \dots \\ \breve{\psi}_{n-2,n-1} + \breve{\psi}_{n-1,n-2} = 1. \end{cases}$$

Перемножив левые части этих уравнений, получим выражение с $2^{\frac{n(n-1)}{2}}$ слагаемыми. Каждое из этих слагаемых представляет собой функцию n переменных и может быть использовано в качестве одной из многомерных функций разбиения единицы. Однако количество слагаемых может быть очень велико — в десятимерном случае их $2^{45} \approx 35 \cdot 10^{12}$ штук. Сгруппируем их в n слагаемых следующим образом:

$$\begin{cases} \tilde{\breve{\psi}}_0 := \breve{\Psi}^0_{n-1}, \\ \tilde{\breve{\psi}}_1 := \breve{\Psi}^1_{n-1}, \\ \tilde{\breve{\psi}}_2 := 1 - \breve{\Psi}^0_2 - \breve{\Psi}^1_2, \\ \tilde{\breve{\psi}}_3 := \breve{\Psi}^0_2 + \breve{\Psi}^1_2 - \breve{\Psi}^0_3 - \breve{\Psi}^1_3, \\ \dots \\ \tilde{\breve{\psi}}_{n-1} := \breve{\Psi}^0_{n-2} + \breve{\Psi}^1_{n-2} - \breve{\Psi}^0_{n-1} - \breve{\Psi}^1_{n-1}, \end{cases} \quad , \text{ где } \begin{array}{l} \breve{\Psi}^0_i := \breve{\psi}_{0,1}\breve{\psi}_{0,2}\dots\breve{\psi}_{0,i}, \\ \breve{\Psi}^1_i := \breve{\psi}_{1,0}\breve{\psi}_{1,2}\dots\breve{\psi}_{1,i}, \\ i = \overline{0..n-1}. \end{array}$$

Каждая из полученных функций $\tilde{\breve{\psi}}_i$ легко разбивается на два слагаемых в зависимости от знака выражения $(x_i - 0.5)$. Таким образом, получаем необходимые $2n$ функций ψ_τ разбиения единицы для гиперкуба Q, дающие искомую систему срезывающих функций. Для наглядности приведем их на одном графике для $n = 2,\ M = 2,\ b = 6,\ c = 0.5$ (рис. 1).

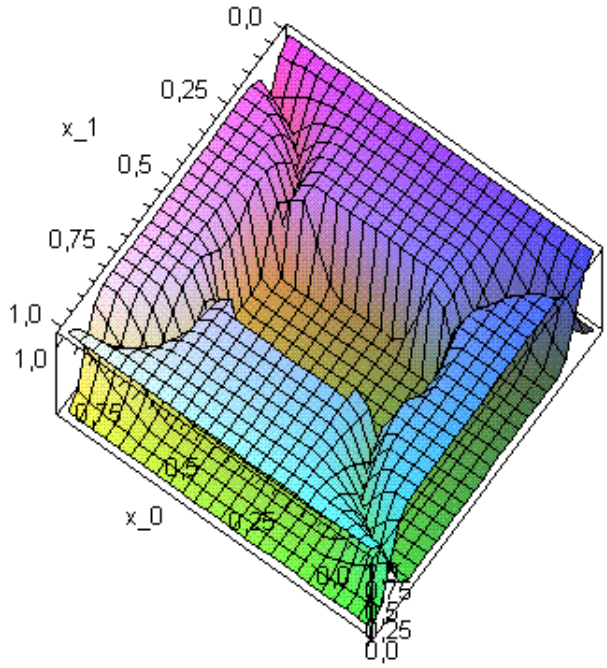

Рис. 1. Функции $\{\varphi_\tau\}_{\tau=1}^{2n}$

2.3. Программа

Программа "CubaInt" [10], предназначенная для вычисления многомерных интегралов по ограниченным выпуклым областям с гладкими границами, тестировалась при следующих параметрах:

1. n от 2 до 10.
2. $f(x) = \sum_{i=1}^{n} a_i x_i^{b_i}$.
3. M от 2 до 6.
4. N от 10 до 10^5, где N – количество точек решетки на ребре единичного куба Q. При этом шаг $h = N^{-1}$.
5. $\Omega = \{x : \Phi(x) = 0,\ \Phi(x) = 1 - \sum_{i=1}^{n} c_i (x_i - 0.5)^{d_i}\}$.
6. P от 1 до 1000.

Для примера возьмем $a = (2, 1, 2, 1, ..., 2, 1)$, $b = (2, 4, 2, 4..., 2, 4)$,$c = (6.25, 39.0625, ..., 6.25, 39.0625)$, $d = (2, 4, 2, 4..., 2, 4)$,

Точность вычислений рассчитывалась по устойчивости десятичных знаков в ответе при уменьшении параметра h. Независимыми параметрами являются n, M, N и P. Приведем некоторые результаты экспериментов.

Таблица 1. $n = 2$, порядки теоретических и экспериментальных погрешностей

N\M	2	3	4	5	6	N\M	2	3	4	5	6
50	3	3	2	2	1	50	4	6	7	9	11
100	4	4	3	2	2	100	4	6	8	10	12
200	7	5	4	5	3	200	5	7	10	12	14
400	8	9	11	7	7	400	6	8	11	14	16
800	8	10	12	13	14	800	6	9	12	15	18
1600	9	11	13	15	16	1600	7	10	13	17	20
3200	10	12	15	16	17	3200	8	11	15	18	22
6400	11	14	16	18	18	6400	8	12	16	20	23
12800	12	15	17	18	17	12800	9	13	17	21	25

Таблица 2. $n = 3$, порядки теоретических и экспериментальных погрешностей

N\M	2	3	4	5	6	N\M	2	3	4	5	6
50	3	3	2	2	1	50	4	6	7	9	11
100	4	4	3	3	3	100	4	6	8	10	12
200	6	7	5	4	4	200	5	7	10	12	14
400	8	8	8	8	7	400	6	8	11	14	16
800	9	9	10	9	9	800	6	9	12	15	18
1600	9	10	11	11	10	1600	7	10	13	17	20

Таблица 3. $n = 5$, порядки теоретических и экспериментальных погрешностей

N\M	2	3	4	5	6	N\M	2	3	4	5	6
25	4	3	3	3	2	25	3	5	6	7	9
50	4	4	4	3	3	50	4	6	7	9	11
75	5	4	4	3	2	75	4	6	8	10	12
100	5	4	4	4	3	100	4	6	8	10	12
125	6	5	4	4	4	125	5	7	9	11	13
150	7	6	5	5	4	150	5	7	9	11	14
175	7	7	5	5	4	175	5	7	9	12	14
200	7	7	6	5	5	200	5	7	10	12	14

В табл. 1–3 показаны порядки абсолютных погрешностей, полученных как при непосредственных вычислениях, так и при теоретической оценке. Как видно, результаты вычислений примерно соответствуют ожидаемым теоретически, уступая им, в основном, по двум причинам. Во-первых, при малом количестве точек N и большом M фактическая толщина пограничного слоя не обеспечивает нужных для обоснования теоретической точности свойств (вмещать $2M$ узлов). Во-вторых, при использовании типа long double с 18 значащими цифрами нет возможности увеличить точность выше 18-го порядка. Также не следует забывать, что при расчете теоретической погрешности мы не учитывали норму подынтегральной функции, которая может ухудшить результат.

Перейдем теперь к анализу быстроты счета и качества распараллеливания программы. Для этого введем понятия ускорения и эффективности программы:

$$S_P = \frac{T_1}{T_P}, \qquad E_P = \frac{S_P}{P},$$

где T_P — время, за которое задача выполняется на P процессорах.

На рис. 2 показаны отклонения экспериментальных ускорений S_P (темные ломаные) от идеальных ускорений (светлые прямые).

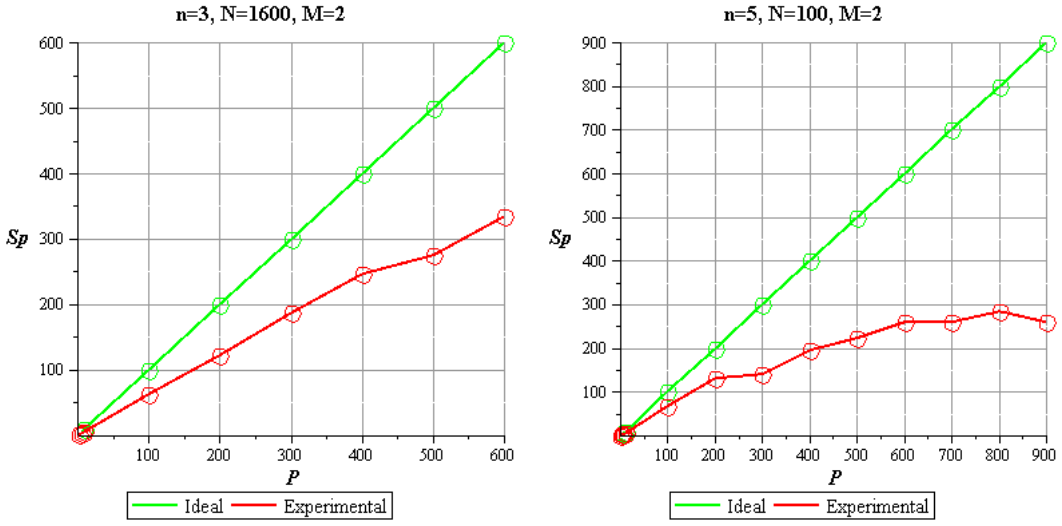

Рис. 2. Ускорение вычислений

Как видно, эффективность распараллеливания постепенно падает с увеличением числа процессоров. Это связано, в первую очередь, с особенностями распределения вычислительной работы по процессорам. Каждый процессор получает примерно одинаковое количество узлов, однако вычислительная сложность в разных узлах различна. В результате процессоры не всегда загружаются равномерно, особенно при сложной форме области.

3. Построение разбиения области интегрирования

3.1. Постановка задачи

Проблема вычисления интегралов по многомерным областям решается путем приближения интеграла решетчатыми кубатурными формулами с ограниченным пограничным слоем

(ОПС–формулы [4,5]). Этот алгоритм предусматривает вычисление интеграла как суммы локальных интегралов по областям более простых форм, которые являются подобластями Ω.

Пусть семейство $\{\Omega_\tau\}_{\tau=0}^{\mathrm{T}}$ образует покрытые области $\overline{\Omega}$. Это взаимно пересекающиеся открытые области с кусочно-гладкими границами, удовлетворяющие главному свойству: часть границы Γ, попавшая в Ω_τ может быть задана явной формулой, выражающей одну из координат однозначной функцией класса C^M от остальных координат, $x_{s_j} = \gamma_j(x_1, ..., x_{s_{j-1}}, x_{s_{j+1}}, ...x_n)$.

Такую систему подобластей можно выбрать с помощью разбиения единицы. Это конечный набор финитных функций $\{\varphi_\tau\}_{\tau=0}^{\mathrm{T}}$, сумма которых тождественно равна единице: $\sum\limits_{\tau=0}^{\mathrm{T}} \varphi_\tau(x) \equiv 1$, для $x \in Q$. У нас они принадлежат пространству $\tilde{C}^M(Q)$.

Для каждой функции φ_τ строится своя кубатурная формула.

Таким образом, итоговая кубатурная формула для области Ω выглядит следующим образом: $K_h^\Omega(h) = \sum\limits_{\tau=0}^{\mathrm{T}} \sum\limits_{kh\in\Omega} c_k^\tau(h)\varphi_\tau(hk)f(hk)$, $\Omega_\tau = \Omega \cap \mathrm{supp}\varphi_\tau$.

То есть приближение интеграла кубатурной формулой сводится к приближению интеграла суммированием кубатурных формул более простого вида.

Наша цель – создать программу, которая автоматически разбивает области произвольных форм на нужные подобласти. Эта задача решена для произвольных двумерных ограниченных областей с гладкими границами.

3.2. Описание алгоритма

Дадим описание конструктивного алгоритма разбиения единицы для такой области Ω.

В общем случае расположенная в единичном кубе $Q = [0,1)^n$ область Ω задается следующим образом: $\overline{\Omega} = \{x|\Phi(x) \geqslant 0\} \subset Q$, $\Phi(x) \in C^M([0,1]^n)$. Ее граница Γ=$\{x|\Phi(x) = 0\}$. Достаточным условием гладкости границы является условие $D\Phi(x)| \neq 0$.

Для дальнейших построений будем использовать вспомогательную функцию $\xi(t)$, определенную в п. 2.

Разбиение единицы проведем в два этапа. Сначала построим центральную срезывающую функцию $\varphi_0(x)$, затем приграничные срезывающие функции. Получим окончательный вид срезывающих функций: $\varphi_\tau(x) = \psi_\tau(x)(1-\varphi_0(x))$, $\tau = \overline{1..T}$ и $\sum\limits_{\tau=0}^{\mathrm{T}} \varphi_\tau \equiv 1$ в некоторой окрестности $\overline{\Omega}$.

Центральную срезывающую функцию зададим как в п. 2: $\varphi_0(x) = \xi(\frac{\Phi(x)}{\varepsilon_2-\varepsilon_1} - \frac{\varepsilon_1}{\varepsilon_2-\varepsilon_1})$. Параметры ε_1 и ε_2 подбираются исходя из условия, чтобы функция $\varphi_0(x)$ была сосредоточена в Ω и равна 0 в некоторой окрестности границы Γ.

Мы решаем задачу для двумерных областей Ω.

Возьмем простейшую область – круг, лежащий в единичном квадрате, со следующими значениями параметров *ε1=0.2, ε2=0.5, α =0.3, β=0.05, M=3.*

Наше разбиение единицы имеет следующий вид: $\varphi_0(y) + \sum\limits_{\tau=1}^{4} \varphi_\tau(y) = \varphi_0(y) + (1-$ $-\varphi_0(y))(\psi_1(y_2) + \psi_2(y_2) + \psi_3(y) + \psi_4(y)) = 1$, где функции $\psi_\tau(y)$ можно задать следующим образом: $\psi_1(y_2) = \xi(y_2 - \frac{1-\alpha-\beta}{\beta})$, $\psi_2(y_2) = \xi(-y_2 - \frac{1-\alpha-\beta}{\beta})$, $\psi_3(y) = \xi(y_1 - \frac{1-\alpha-\beta}{\beta}) \cdot (1-$ $-\psi_1(y_2) - \psi_2(y_2))$, $\psi_4(y) = (1 - \xi(y_1 - \frac{1-\alpha-\beta}{\beta})) \cdot (1 - \psi_1(y_2) - \psi_2(y_2))$, где α и β подбираются в зависимости от ширины пограничного слоя. Причем $\sum\limits_{\tau=1}^{4} \psi_\tau = 1$ на носителе функции

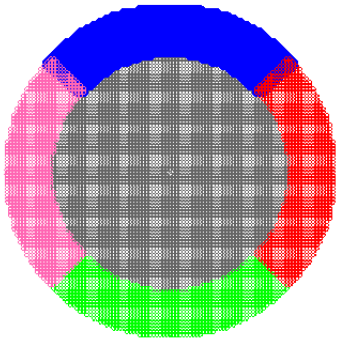

Рис. 3. Область $\{y|\Phi_1(y) > 0\}$

$(1 - \varphi_0(E))$, тогда и $\sum\limits_{\tau=0}^{4} \varphi_\tau(y) = 1$, $y \in [0,1]^2$.

Таким образом, функции $\{\varphi_\tau\}_{\tau=0}^4$ – искомые срезывающие функции для круга.

Если бы это был n-мерный шар, то количество срезывающих функций было бы равно *2n+1*, где n – размерность пространства. Для круга их было 4 (вместе с центральной – 5).

Перейдем к построению разбиения единицы для произвольной двумерной области с гладкой границей.

Сделаем замену переменных $x = \frac{\nabla\Phi(y)}{|\nabla\Phi(y)|}$. Она является гладким отображением приграничной полоски, в которой $1 - \varphi_0(x) \neq 0$, на единичную сферу, $|y| = 1$. А в переменных y применим функции $\psi_\tau(y)$, $\sum \psi_\tau(y)|_{|y|=1} = 1$.

Приведем примеры разбиения единицы для некоторых невыпуклых областей. На рис. 4 изображено разбиение для области $\{x|\Phi_2(x) > 0\}$ с границей $\Phi_2(x) = 1 - \sqrt{8(x_1 - 0.5)^2 + 8(x_2 - 0.5)^2} + \frac{2(x_1-0.5)^2(x_2-0.5)^2}{((x_1-0.5)^2+(x_2-0.5)^2)^2} = 0$ с параметрами *ε1=0.2, ε2=0.5, α =0.3, β=0.05, M=3.*

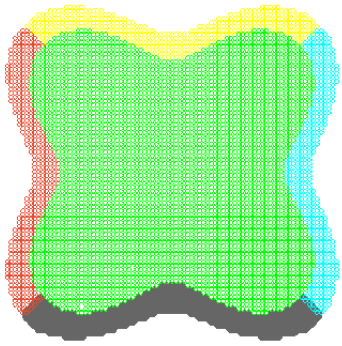

Рис. 4. Область $\Phi_2(x) \geqslant 0$

Следует отметить, что предложенный алгоритм получения разбиения единицы универсален. Он не зависит от формы и расположения области.

Например, повернем область, заданную на рис. 4, на 1 радиан. Тогда уравнение границы запишется в следующем виде:

$$\Phi_3(x) = 1 - \sqrt{8(x_1 - 0.5)^2 + 8(x_2 - 0.5)^2} +$$

$$+\frac{2(x_1 - 0.5)(x_2 - 0.5)\sin(1) + \cos(1)((x_1 - 0.5)^2 - (x_2 - 0.5)^2)^2}{((x_1 - 0.5)^2 + (x_2 - 0.5)^2)^2} = 0.$$

Заметим, что разбиение единицы автоматически подстроилось под новую форму области (относительно декартовой системы координат) (рис. 5).

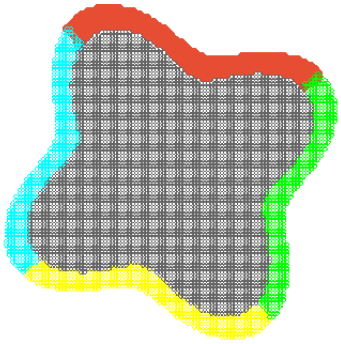

Рис. 5. Область $\Phi_3(x) \geqslant 0$

Проиллюстрируем результат еще несколькими примерами разбиения невыпуклых областей более сложных форм. На рис. 6 изображена область с границей $\Phi_4(x) = \frac{1}{16} - 625((x_1 - 0.5)^2 + (x_2 - 0.7)^2)((x_1 - 0.2)^2 + (x_2 - 0.2)^2) \cdot ((x_1 - 0.4)^4 + (x_2 - 0.9)^2) + + (x_1 - 0.4)^2 + (x_2 - 0.6)^2 = 0.$

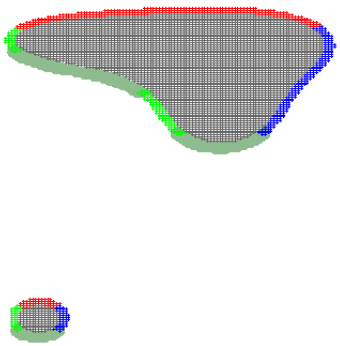

Рис. 6. Двусвязная область $\{x | \Phi_4(x) \geqslant 0\}$

$$\Phi_5(x) = \frac{1}{16} - 150((\mathrm{x}_1 - 0.3)^2 + (\mathrm{x}_2 - 0.7)^2)((\mathrm{x}_1 - 0.2)^4 + (\mathrm{x}_2 - 0.2)^4) = 0$$

На рис. 8 и 9 изображены соответственно области

$$\Phi_6(x) = 16\sqrt{(\mathrm{x}_1 - 0.5)^2 + (\mathrm{x}_2 - 0.5)^2} - 5 +$$

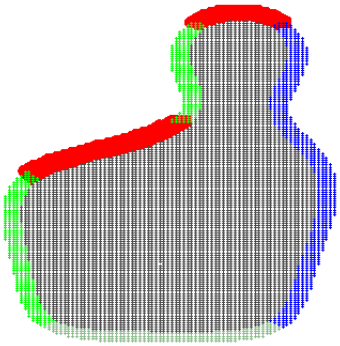

Рис. 7. Область $\{x|\Phi_5(x) \geqslant 0\}$

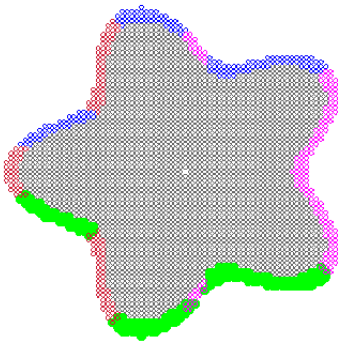

Рис. 8. Область $\{x|\Phi_6(x) \geqslant 0\}$

$$+\frac{(\mathrm{x}_1-0.5)((\mathrm{x}_1-0.5)^4+-10(\mathrm{x}_1-0.5)^2(\mathrm{x}_2-0.5)^2+5(\mathrm{x}_2-0.5)^4)}{((\mathrm{x}_1-0.5)^2+(\mathrm{x}_2-0.5)^2)^{\frac{5}{2}}}=0$$

и

$$\Phi_7(x)=((\mathrm{x}_1-0.4)^2+(\mathrm{x}_2-0.4)^2)-\frac{16}{3}((\mathrm{x}_1-0.4)^2+(\mathrm{x}_2-0.4)^2)-((4\mathrm{x}_1-0.9)^2+$$

$$+(4\mathrm{x}_2-0.9)^2--1)\cdot\frac{9}{2}((4\mathrm{x}_1-0.9)^2+(4\mathrm{x}_2-0.9)^2)=0$$

3.3. Описание программы

По изложенному алгоритму была создана программа на языке программирования C++, подробно закомментированная. Перед ее использованием необходимо провести вспомогательные действия: подсчитать производные границы области интегрирования Ω, которые необходимы для вычисления градиента.

Входными данными в программе являются гладкость функции M, ширина пограничного слоя α и β, параметры ε_1 и ε_2 и количество точек N на стороне единичного квадрата. Главный цикл программы пробегает по всем узлам решетки, и для каждой срезывающей функции рисуется свой график по точкам. Координаты точек записываются в соответствующий текстовый файл, а рисунок выводится на экран.

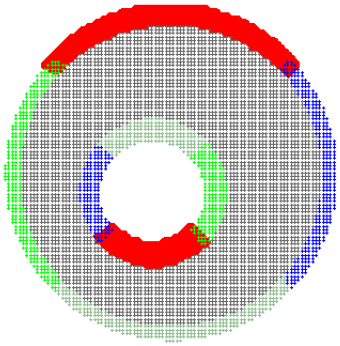

Рис. 9. Область $\{x|\Phi_7(x) \geqslant 0\}$

4. Решение интегральных уравнений

4.1. Алгоритм

Напомним постановку задачи.

Пусть в области $\bar{\Omega} \subset R^2$ задана функция $f(x)$ и функция $K(x,y)$. Наша задача – найти функцию $u(x)$, удовлетворяющую уравнению

$$u(x) - \int\limits_{\Omega} K(x,y)u(y)dy = f(x), x \in \bar{\Omega} \subset R^2.$$

Здесь Ω – ограниченная замкнутая область с гладкой границей, лежащая внутри единичного квадрата $(0,1)^2$, $K \in C^M(\bar{\Omega} \times \bar{\Omega})$ и $f \in C^M(\bar{\Omega})$.

В связи с тем, что изначально планировалось применение многопроцессорной вычислительной техники, необходимо было использовать такие алгоритмы численного решения интегральных уравнений, при распараллеливании которых не возникнут трудности. Для решения данного уравнения был выбран итерационный метод, позволяющий получить относительно простые численные алгоритмы решения интегральных уравнений. Предположим, что $||K||_{C(\bar{\Omega}\times\bar{\Omega})} = \theta < 1$. Можно применить метод последовательных приближений. За начальное приближение берется $u_0(x) = f(x)$. Последующие приближения строятся по рекуррентной формуле $u_{s+1}(x) = f(x) + \int\limits_{\Omega} K(x,y)u_s(y)dy$, $s = 1, 2, \ldots$. Последовательность функций $\{u_s(x)\}$ является приближением к искомому решению данного уравнения в норме пространства $C(\bar{\Omega})$. Было показано, что решение интегрального уравнения $u(x)$ обладает той же гладкостью, что ядро интегрального оператора и правая часть уравнения.

При численной реализации этого метода для вычисления интеграла на каждой итерации используется решетчатая кубатурная формула с ограниченным пограничным слоем (ОПС-формула) (см. п. 2). Для приближенного вычисления интеграла создан алгоритм (см. [5,7,8]), который позволяет вычислить коэффициенты кубатурной формулы, приближающей точное значение интеграла с погрешностью порядка $(h/\alpha)^m$, m – гладкость интегрируемой функции, α – параметр локализации области интегрирования (см. п. 3).

Отметим, что выбор решетчатых кубатурных формул позволяет достичь необходимой универсальности и единообразия в интегрировании по областям произвольных форм вследствие того, что последовательности решеток не зависят от форм областей интегрирования.

А использование ОПС–формул позволяет использовать алгоритмы вычисления коэффициентов формул с хорошими и теоретически обоснованными аппроксимационными свойствами.

4.2. Описание программы

В этой части рассмотрена возможность применения решетчатых кубатурных формул к решению интегральных уравнений с произвольной формой области интегрирования. Исходя из изложенных алгоритмов была написана программа численного решения интегрального уравнения вида (1). Программа написана на языке C++ с использованием библиотеки параллельных функций MPI. Методом последовательных приближений вычисляется последовательность $\{u_s\}$, сходящаяся к точному решению данного интегрального уравнения. Процесс вычисления заканчивается при выполнении условия $||u_s - u_{s-1}|| \leq \varepsilon$, где $||u|| = \max\limits_x |u(x)|$, ε – заранее заданная точность.

Входные данные программы:

1. Область интегрирования Ω, которая задается в неявном виде: $\bar{\Omega} = \{x|\Phi(x) \geqslant 0\}$, $\Phi(x) \in C^M(Q)$, $\bar{\Omega} \subset (0,1)^2$.

2. Функция $K(x_1, y_1, x_2, y_2)$, $\max\limits_{x,y} |K(x,y)| < 1$.

3. Функция $f(x_1, x_2)$.

4. Шаг кубической решетки $h < 0.01$, он задается как $h = 1/N$, где N – натуральное число.

5. Параметры срезывающих функций.

6. Параметр гладкости M.

7. Количество процессоров P.

Приближенным решением интегрального уравнения является результат последней итерации, который выводится в виде табличного значения функции $u_s(x)$ в узлах $x = hk \in \in \bar{\Omega} \subset R^2$.

Программа тестировалась на суперкомпьютерах МВС-15000ВМ и МВС-50К Межведомственного Суперкомпьютерного Центра РАН.

Приведем результаты вычислительного эксперимента решения интегрального уравнения с выпуклой областью интегрирования [3]. Для того чтобы иметь возможность сравнения с точным решением уравнения, мы задавались функцией, которая должна быть точным решением, и по ней вычислялась правая часть уравнения. Вычислительный эксперимент проводился с этой правой частью.

Расчеты проводились со следующими данными. В качестве точного решения была взята функция $\bar{u}(x) = (x_1 - x_2)^5$.

1. $K(x,y) = (x_1 y_1 + x_2 y_2)^3$.

2. $f(x) = (x_1 - x_2)^5 - (0.00099x_1^3 + 0.0007x_2x_1^2 - 0.00059x_2^2x_1 - 0.00093x_2^1)$.

3. Область интегрирования $\bar{\Omega} = \{x|\Phi_1(x) \geqslant 0\}$, $\Phi_1(x) = 1 - ((x_1 - 0.5)/0.4)^2 -$
$- ((x_2 - 0.5)/0.4)^4$, (рис. 10).

4. Шаг кубической решетки $h = 1/N$, $N = 200, 300$.

5. Гладкость $M = 2, 3$.

6. Параметры срезывающих функций , $\varepsilon_2 = 0.5$, $\alpha = 0.1$, $\beta = 0.05$.

7. Количество процессоров 100-1000.

Теоретическая ожидаемая точность – $O(h^M)$, без учета влияния ширины пограничного слоя, в данном примере M =2, $h = 1/200$, ожидаемая точность – 10^{-5}.

Точность вычислений рассчитывалась двумя способами. Сравнением с известным точным решением (погрешность обозначена ε') и по правилу Рунге – по устойчивости десятич-

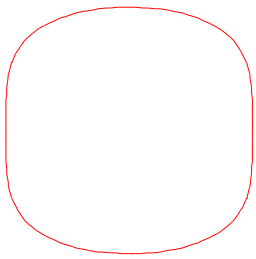

Рис. 10. Область $\Phi_1(x) \geqslant 0$

ных знаков в ответе при уменьшении h (погрешность обозначена ε).

В табл. 4 приведены результаты вычислительного эксперимента.

Таблица 4. Достигнутая точность при учтенной гладкости $M = 2$

k - номер итерации	$h = 1/200$	$h = 1/300$
	ε	ε
1	*0.0987*	*0.0988*
2	*0.000776*	*0.000778*
3	*0.0000268*	*0.0000269*
4	*0.00000310*	*0.00000310*

В табл. 4 точность рассчитывалась по устойчивости десятичных знаков. При сравнении результата последней итерации с точным решением $\bar{u}(x) = (x_1 - x_2)^5$ получили совпадение 5-6 знаков после запятой.

Следует отметить, что программа предназначена для интегральных уравнений вида (1) с произвольной формой области интегрирования. В связи с чем был проведен вычислительный эксперимент для невыпуклой области. Приведем результаты тестирования для области $\Phi_2(x) \geqslant 0$, $\Phi_2(x) = 1 - \sqrt{8(x_1 - 0.5)^2 + 8(x_2 - 0.5)^2} + +(2(x_1 - 0.5)(x_2 - 0.5)\sin(1) + \cos(1)((x_1 - 0.5)^2 - (x_2 - 0.5)^2)^2)/((x_1 - 0.5)^2 + (x_2 - 0.5)^2)^2$, изображенной на рис. 5, с параметрами срезывающих функций: $\varepsilon_1 = 0.1$, $\varepsilon_2 = 0.5$, $\alpha = 0.1$, $\beta = 0.05$.

Точность рассчитывалась по устойчивости десятичных знаков. Получили, что теоретически ожидаемая точность $(h/\alpha)^M$ порядка 10^{-5} достигается на пятой итерации при $N = 250$, $M = 3$, $\alpha = 0.1$.

Далее приведем анализ эффективности распараллеливания. В табл. 5 указано время работы программы на разном количестве процессоров и подсчитаны ускорение и эффективность.

Таблица 5. Время счета (с) при разном числе P процессоров, $h = 1/200$, $M = 2$. Достигнутая точность 10^{-5}

P	***10***	***20***	***30***	***40***	***50***	***100***
T_P	*230*	*127*	*93*	*75*	*63*	*36*
S_p	*10*	*18*	*24.7*	*36*	*42.6*	*63.8*
E_p	*1*	*0.9*	*0.8*	*0.8*	*0.72*	*0.63*

На графике (рис. 11) изображена зависимость ускорения от количества процессоров.

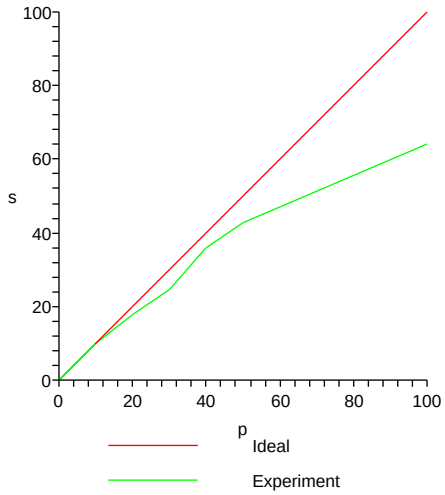

Рис. 11. Зависимость ускорения от количества процессоров

5. Заключение

Суммируем выводы по описанным выше программам.

Создана программа приближенного интегрирования “CubaInt”. Анализ скорости работы программы показал, что эффективность распараллеливания снижается с увеличением числа процессоров, и это связано с, вообще говоря, различной сложностью вычисления коэффициентов кубатурной формулы в разных узлах. Отметим, что даже в случае идеальной эффективности программы нет смысла увеличивать размерность пространства выше 10 и количество узлов выше 10^{12}.

Разработан алгоритм автоматического разбиения двумерных областей произвольных форм с гладкими границами на подобласти и создана программа, реализующая этот алгоритм. Этот алгоритм позволяет построить разбиение и в n-мерном случае (даже для несвязных и не односвязных областей). Однако метод не подходит для областей с кусочно-гладкими границами.

Для наглядности показаны картинки только для двумерного случая, для размерностей больше 3 осложняется иллюстрация результатов, но алгоритм также работает. Результаты опубликованы в [2].

И наконец, оправдано применение ОПС-формул для численного решения интегральных уравнений. Применение итерационного метода в сочетании с решетчатой кубатурной ОПС-формулой позволяет достичь точности порядка 10^{-5} за 5-6 итераций. Отметим, что этот алгоритм решения интегральных уравнений позволяет хорошо распараллелить вычислительную программу для применения многопроцессорных вычислительных систем. На данный момент эффективность распараллеливания около 63 %. Это связано с тем, что количество последовательных операций становится сравнимым с количеством параллельных операций и значительная часть процессоров простаивает. В дальнейшем планируется повысить эту эффективность путем внесения изменений в алгоритм.

Работа выполнена при поддержке Программы № 14 Президиума РАН, Программы целевых расходов Президиума РАН «Поддержка молодых ученых» и Российского фонда фун-

даментальных исследований (№ 06-01-00597).

Список литературы

[1] Игнатьев А.Н. Универсальный алгоритм вычисления интегралов по ограниченным областям с гладкими границами. Кубатурные формулы и их приложения. Уфа: ИМВЦ УНЦ РАН, 1996. С. 21–31.

[2] Валеева Л.З. Применение локализации алгоритма кубатурных формул для решения интегральных уравнений. Сб. статей II Межд. конференции "Математическое и компьютерное моделирование естественнонаучных и социальных проблем". Пенза, 2008. С. 14-17.

[3] Банникова Е.Л. Программа численного решения интегральных уравнений Фредгольма второго рода “IntUr”. Свидетельство №10418 от 15.04.2008.

[4] Рахматуллин Д.Я. Интегрирование функций по выпуклым областям на многопроцессорных вычислительных системах: Автореф. дис... канд. физ.-мат. наук. - Уфа: ИМВЦ УНЦ РАН, 2006. - 22 с.

[5] Рамазанов М.Д. Теория решетчатых кубатурных формул с ограниченным пограничным слоем. Препринт 2007-1 – Уфа: ИМВЦ УНЦ РАН, 2007. – 105 с.

[6] Рамазанов М.Д. Лекции по теории кубатурных формул. Уфа: Изд-во БашГУ, 1973. 177 с.

[7] Соболев С.Л., Васкевич В.Л. Кубатурные формулы. Новосибирск: Изд-во ИМ СО РАН, 1996. 484 с. С. 254—263.

[8] Рахматуллин Д.Я. Вычисление интегралов по многомерным областям на многопроцессорных вычислительных системах // Вычислительные технологии. Т. 11, 3, 2006. С. 118-125.

[9] M.D.Ramazanov To the L_p-Theory of Sobolev Formulas //Siberian advances in mathematics. 1999. Vol. 9, N 1. P. 99-125.

[10] Рахматуллин Д.Я. Программа интегрирования по многомерным областям “CubaInt”. Свидетельство № 2007614331 от 10.10.2007.

Solving of Integral Equations on Multiprocessing Computing Systems

Marat D. Ramazanov, Djangir Y. Rakhmatullin, Leysan Z. Valeeva and Ekaterina L. Bannikova

Institute of Mathematics with Computing Centre
112 Chernyshevsky str., Ufa, 450077 Russia

It is described Fredholm equations for the various forms of domains. It is reduced the description of the algorithm of approximate calculation of integrals using multiprocessing technology.

Key words: multiprocessing systems, approximate calculation of integrals, multiprocessing technology.