В.В. Мареев Е.Н. Станкова

МНОГОСЕТОЧНЫЕ МЕТОДЫ. ВВЕДЕНИЕ В СТАНДАРТНЫЕ МЕТОДЫ.

Санкт-Петербург
2012

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

В. В. Мареев, Е. Н. Станкова

МНОГОСЕТОЧНЫЕ МЕТОДЫ. ВВЕДЕНИЕ В СТАНДАРТНЫЕ МЕТОДЫ.

Издательство С.-Петербургского университета
2012

УДК 51-7 : 519.632 : 519.688 : 004.021
ББК 22
А65

Р е ц е н з е н т ы: д-р физ.-мат. наук *А. В. Богданов* (СПбГЭТУ "ЛЭТИ"),
д-р физ.-мат. наук *А. И. Водяхо* (СПбГЭТУ "ЛЭТИ")

Печатается по постановлению
Редакционно-издательского совета
факультета прикладной математики–процессов управления
Санкт-Петербургского государственного университета

Мареев В. В., Станкова Е. Н.

А65 Многосеточные методы. Введение в стандартные методы. — СПб.: Изд-во С.-Петерб. ун-та, 2012. 61 с.

ISBN 5-983-40073-8

В предлагаемом учебном пособии даются основные представления о стандартных многосеточных методах и стратегии их разработки; показывается эффективность применения методов для решения задач повышенной сложности, требующих использования высокопроизводительных численных алгоритмов и значительных вычислительных ресурсов.

Пособие содержит основы теории многосеточных алгоритмов, описание применения этой теории для решения нелинейных уравнений (схема полной аппроксимации), а также описание метода вложенных итераций и техники многоуровневой адаптации. Отдельный раздел пособия посвящен использованию многосеточного метода для проведения параллельных вычислений.

Пособие предназначено для студентов старших курсов (магистров и специалистов) для самостоятельных и лабораторных работ по численным методам и информатике. Пособие может быть полезно для научных работников и инженеров, занимающихся решением задач с использованием сеточных методов, требующих большого объема вычислений и заинтересованных в повышении их эффективности.

В пособии приведены конкретные вычислительные примеры, легко реализуемые в виде программ.

Библиогр. 50 назв. Ил. 10.

ББК 22

ISBN 5-983-40073-8

Введение

Одним из критериев выбора алгоритма, используемого при численном моделировании той или иной физической задачи, является объем вычислительной работы, который требуется для его реализации. Существует правило, что этот объем должно быть пропорционален реальным физическим изменениям, происходящим в моделируемой системе. Если алгоритм требует большого количество тяжелой вычислительной работы для расчета слабого эффекта или очень медленного физического процесса, от такого «затратного» алгоритма следует по возможности отказаться, выбрав более эффективный.

Примером «затратных» алгоритмов являются обычные итерационные методы для решения алгебраических уравнений, возникающих при численном решении уравнений в частных производных или интегро-дифференциальных уравнений. Так практически единственным, но наиболее существенным недостатком методов Якоби и Зайделя, используемых для решения эллиптических задач методом сеток, является их низкая скорость сходимости. Другим примером могут служить решения нестационарных задач, с шагом по времени (выбор которого диктуется условиями устойчивости) много меньшим масштаба реального изменения решения. Т.е. в общем случае, «затратным» можно назвать такой алгоритм, который требует использования очень подробных сеток там, где на большей части расчетной области величина шага по пространству или по времени много меньше, чем реальный масштаб изменения решения.

В этом случае эффективным решением проблемы является использование многосеточного алгоритма, который позволяет преодолеть главную трудность, возникающую при решении такого рода задачи – ее «жесткость». Жесткость задачи заключается в существовании нескольких компонент решения, которые имеют разный

масштаб и конфликтуют друг с другом. Например, гладкие компоненты, которые можно эффективно аппроксимировать на грубых сетках, но которые медленно сходятся на подробных сетках, конфликтуют с высокочастотными компонентами, которые необходимо аппроксимировать с помощью подробных сеток. Используя несколько уровней дискретизации, многосеточный алгоритм решает конфликты такого рода, позволяя достигать большой эффективности, путем снижения объема вычислений, необходимых для получения численного решения. В идеале многосеточные методы позволяют довести объем вычислительной работы до уровня, который определяется исключительно реальным масштабом изменения решения.

Основоположником многосеточного метода считается Федоренко Р. П. [3]. В 1961 году он сформулировал многосеточный алгоритм для стандартной пятиточечной дискретизации уравнения Пуассона в единичном квадрате, который позволял получить численное решение, выполняя $O(N)$ арифметических операций (N — число узлов сетки). Позднее его идеи получили развитие в работах Н.С. Бахвалова и Г.П. Астраханцева [1,2], а также в работах западных математиков: Бранта [7,8], Хакбуша [9,10], Троттенберга [13] и Йоппиха [11]. Последние внесли большой вклад в перенесении идей многосеточного алгоритма на область решения нелинейных уравнений, в разработку техники многоуровневой адаптации и метода вложенный итераций (полный многосеточный метод).

Обладая высокой эффективностью, многосеточные методы допускают наиболее естественное распараллеливание и векторизацию приложений, что позволяет отнести их к наиболее перспективному и быстро развивающемуся разделу современных высокопроизводительных алгоритмов.

В настоящее время многосеточные алгоритмы эффективно применяются для решения задач динамики плазмы и гидродинамики, расчета собственных значений и собственных функций дифференциальных операторов, для расчета нейтронных полей в ядерном энергетическом реакторе, для решения задач теории упругости, в задачах обтекания тел достаточно сложной формы, а также для решения медицинских проблем, связанных с реконструкцией электрической активности головного мозга или сердца по измерениям электрических потенциалов вне мозга или сердца.

Глава 1
Общие сведения по многосеточным методам

§ 1.1. Общие свойства многосеточных методов и стратегия их разработки

Многосеточные методы используются, в основном, для решения эллиптических уравнений типа:

$$\sum_{i,j} \frac{\partial}{\partial x_j} \left(a_{i,j} \frac{\partial u}{\partial x_j} \right) = -f$$

в произвольной области Ω с краевыми условиями, для определенности, первого рода:

$$u\big|_{\partial\Omega} = \varphi$$

Здесь f, φ — заданные функции, $a_{i,j}(x)$ — известные функции, удовлетворяющие следующим естественным условиям:

1. $a_{ji} = a_{ij}$ (симметричность).
2. для $\forall$ вещественного ξ (эллиптичность уравнения)

$$\sum_{i,j} a_{i,j} \xi_i \xi_j \geq \alpha \sum_i \xi_i^2.$$

Индексы i, j меняются от 1 до N, где N — размерность пространства, в котором решается задача.

Основные преимущества многосеточного метода заключаются в следующем:

1. метод предлагает общую стратегию построения солвера;
2. он обладает естественным параллелизмом;
3. характеризуется высокой эффективностью (см. таблицу 1.1).

Таблица 1.1. Сравнительная таблица эффективности различных численных методов при решения двумерных задач ($N \times N$)

Численные методы	**Число операций**
Прямые методы (четно-нечетное исключение, преобразование Фурье)	$O(N^2 \log_2 N)$
Итерационные методы (метод Якоби, Гаусса-Зайделя)	$O(N^4 \ln \varepsilon^{-1})$
Метод сопряженных градиентов	$O(N^3)$
Многосеточный метод	$O(N^2 \ln \varepsilon^{-1})$

где N — число узлов сетки, ε — скорость сходимости.

Алгоритм решения задачи с помощью многосеточного метода

1. анализ проблемы
2. построение численной модели
3. исследование поведения основных параметров задачи (граничные условия, расчетная область, переменные коэффициенты, нелинейность)
4. дискретизация задачи
5. определение структуры данных: функции, сетки, массивы, архитектура компьютера
6. построение программ для итераций и невязки: сходимость, аппроксимация, сглаживание

7. проведение Фурье анализа
8. прогноз сходимости
9. отладка на простых примерах и грубых сетках
10. отладка отдельных компонент многосеточных методов
11. визуализация решения

§ 1.2. Основные свойства итерационных методов

Модельная задача

Рассмотрим уравнение Пуассона:

$$Lu = -f,$$
$$\text{где} \quad L \equiv \triangle = \left(\frac{\partial^2}{\partial x^2}\,, \frac{\partial^2}{\partial y^2} \right)$$
$$L(\Omega) \longrightarrow R, \qquad \Omega = (0,1) \times (0,1), \quad u\big|_{\partial\Omega} = \varphi$$

Введем основные объекты метода сеток.

Сетка. Область покрываем сеткой из точек (k, m) с координатами

$$x_k = kh, \quad y_m = mh,$$
$$\text{где} \quad k,\, m = \{0, 1, \ldots, N\}, \quad h = 1/N$$

Сеточная функция. Приближенное решение ищем в виде сеточной функции $u_{k,m}$, которую трактуем как приближенное значение $u(x_k, y_m)$. Функция $u_{k,m}$ определена во всех узлах сетки: $\{u_{k,m}\}$, где $k, m \in 0, 1, \ldots, N$.

Аппроксимация уравнения. Сеточную функцию будем искать как решение системы уравнений, полученных простейшим способом, — прямой заменой входящих в уравнение производных на соответствующие разностные отношения:

$$\frac{u_{k-1,m} - 2u_{k,m} + u_{k+1,m}}{h^2} + \frac{u_{k,m-1} - 2u_{k,m} + u_{k,m+1}}{h^2} = -f_{k,m}$$

Это уравнение имеет крестообразный шаблон и называется простейшей пятиточечной аппроксимацией уравнения Пуассона. Оно определено только в так называемых внутренних узлах сетки, т.е. при $k, m = 1, \ldots, N-1$. В дальнейшем мы будем использовать более компактные формы записи этого уравнения:

$$L_h u_h = -f_h, \quad \text{где} \quad u_h = \{u_{k,m}\}, \quad k, m = 0, 1, \ldots, N$$

Основным средством решения больших систем линейных алгебраических уравнений, возникающих при разностной аппроксимации краевых эллиптических задач, являются методы итераций (последовательных приближений). В этих методах, начиная с какой-то сеточной функции u_h^0 (верхний индекс означает номер итерации), по тем или иным правилам находят $u_h^1, u_h^2, \ldots$. Если $u_h^i \to u_h$ при $i \to \infty$), то метод называют сходящимся.

Однако одного факта сходимости мало, необходимо оценить скорость сходимости.

Обычно она имеет вид:

$$\|u_h^i - u_h\| < C_q^i; \quad q < 1$$

Числа $q < 1$ свои для разных методов: чем меньше q, тем лучше метод — тем быстрее он сходится.

Пусть $U(x, y)$ — решение исходной дифференциальной задачи, а $U_h \equiv U_{k,m}$ — ограничение решения на сетку. Поскольку u_h совпадает с решением U_h с точностью до $\varepsilon = O(h^p)$, итерации следует проводить только до тех пор, пока u_h^i не совпадет с u_h с той же точностью $\varepsilon = O(h^p)$. Дальнейшие итерации особого смысла не имеют. Поэтому обычно назначается некоторое $\varepsilon = O(h^p)$ и делается такое число $i(\varepsilon)$ итераций, которое обеспечивает оценку:

$$\|u_h^i - u_h\| \le \varepsilon, \quad \text{т.е.} \quad C_q^i = \varepsilon \quad \Longrightarrow \quad i(\varepsilon) = \frac{\ln(\varepsilon/c)}{\ln q}.$$

Кроме числа итераций, следует учитывать и число операций, затрачиваемых на выполнение одной итерации. Пусть T — время выполнения одной итерации, τ — время, необходимое для получения ε-решения. Очевидно,

$$\tau(\varepsilon) = i(\varepsilon)T = T[\ln(\varepsilon/c)/\ln q]$$

Метод простой итерации

Итерации задаются в следующей форме:

$$u_h^{i+1} = u_h^i + \tau(L_h u_h^i + f_h)$$

- Время выполнения одной итерации $\tau \sim N^2$.
- Погрешность $\nu_i^h \equiv u_h - u_i^h$ заранее неизвестна.
- Уравнение для погрешности $\nu_h^{i+1} = (E + \tau L_h)\nu_h^i$.
- Невязка $r_h^i = -f_n - L_h u_h^i$ удобна тем, что ее всегда можно вычислить.
- Норма для оценки погрешности $\|\nu_h^{i+1}\| = \|(E + \tau L_h)\|^i \cdot \|\nu_h^0\|$.

§ 1.3. Спектральные свойства дискретного оператора Лапласа с граничными условиями Дирихле

Лемма 1 *Сеточные функции* $\varphi_k^p = \sin(pk\pi/N)$ *суть собственные векторы разностного оператора* $\partial^2/\partial x^2$, *которым соответствуют собственные значения*

$$\lambda^p = \frac{4\sin^2(p\pi/2N)}{h^2}.$$

Здесь p *— номер собственной функции* ($p = 1, 2, \ldots, N-1$), k *— номер узла.*

Доказательство состоит в простой проверке, которая опускается.

Аналогично введем собственные векторы и собственные числа оператора $\partial^2/\partial y^2$, соответственно:

$$\varphi_m^q = \sin\frac{qm\pi}{N} \quad \text{и} \quad \lambda^p = \frac{4\sin^2(q\pi/2N)}{h^2}.$$

Лемма 2 *Собственными функциями оператора L являются функции*

$$\varphi_{k,m}^{p,q} = \sin\frac{pk\pi}{N}\,\sin\frac{qm\pi}{N},$$

где $p,q = 1,\ldots,N-1$, которым соответствуют собственные значения $\lambda^{p,q} = \lambda^p + \lambda^q$.

Доказательство состоит в прямом вычислении $L\varphi_{k,m}^{p,q}$.

В дальнейшем особую роль будут играть границы спектрального интервала:

$$\gamma_1 \le \lambda^{p,q} \le \gamma_2$$

$$\gamma_1 = \lambda^{1,1} = \frac{8}{h^2}\,\sin^2\frac{\pi}{2N} \approx 2\pi^2 - \frac{1}{6}\,\pi^4 h^2$$

$$\gamma_2 = \lambda^{N-1,N-1} = \frac{8}{h^2}\,\cos^2\frac{\pi}{2N} \approx \frac{8}{h^2} - 2\pi^2 + \frac{1}{6}\,\pi^4 h^2$$

Глава 2

Основы многосеточных методов

§ 2.1. Основы многосеточных методов: коррекция невязки на грубой сетке

Основные понятия

Точное решение u_h:

$$L_h u_h = -f_h \tag{1}$$

Приближенное решение u_h^i получается после i итераций. Погрешность $\nu_h^i = u_h - u_h^i$. Невязка $r_h^i = -f_h - L_h u_h^i$.

Заменяя в уравнении (1) u_h на $\nu_h^i + u_h^i$, и используя линейность оператора L_h, получим уравнение для погрешности

$$L_h \nu_h^i = r_h \tag{2}$$

Сравним $L_h u_h^i = -f_h$ и $L_h \nu_h^i = r_h$.

Тот же самый оператор, но другие неизвестные и правая часть. Если бы можно было решить уравнение $L_h \nu_h^i = r_h$, то функция ν_h^i была бы поправкой в том смысле, что точное решение $u_h = u_h^i + \nu_h^i$.

Но нет никакого преимущества решать уравнение $L_h \nu_h^i = r_h$ по сравнению с уравнением $L_h u_h = -f_h$, поскольку в обоих уравнениях оператор L_h тот же самый.

Коррекция невязки

Получить итерацию u_h^{i+1} можно следующим образом.

Решим уравнение $L_h \nu_h^i = r_h$ приближенно, с помощью оператора L'_h, сходного с L_h:

$$L'_h \nu_h'^{\,i} = r_h^i$$

Затем проведем коррекцию решения:

$$u_h^{i+1} = u_h^i + \nu_h^{'\,i},$$

где $\nu_h'^{\,i} = (L'_h)^{-1} r_h^i$

Тогда:

$$\begin{aligned} u_h^{i+1} = u_h^i + \nu_h'^{\,i} = u_h^i + (L'_h)^{-1}\, r_h^i = \\ = u_h^i - (L'_h)^{-1}\, f_h - (L'_h)^{-1} L_h u_h^i = \\ = [E - (L'_h)^{-1} L_h] u_h^i - (L'_h)^{-1} f_h \end{aligned}$$

Можно легко показать, что

$$\begin{aligned} \nu_h^{i+1} &= [E - (L'_h)^{-1} L_h]\, \nu_h^i \\ r_h^{i+1} &= [E - (L'_h)^{-1} L_h]\, r_h^i \end{aligned}$$

Коррекция невязки осуществляется с помощью операторов $-L'_h$, $(L'_h)^{-1}$, которые в вычислительном смысле ничем не лучше L_h. Задача состоит в аппроксимации L'_h более простым оператором.

Для этого воспользуемся тем, что погрешность и невязка после проведения некоторого количества итераций становятся гладкими функциями и уравнение для них можно решать на другой, более грубой сетке.

Действительно, поскольку погрешность на границе имеет значение равное нулю, для нее справедливо следующее представление:

$$\nu_{k,m} = \sum_{p,q} c_{p,q} \varphi_{k,m}^{p,q}$$

где $\varphi_{k,m}^{p,q}$ - собственные функции оператора L_h.

При этом

$$\|\nu\| = \sqrt{\sum_{k,m} \nu_{k,m}^2} = \sqrt{\sum_{p,q} c_{p,q}^2}$$

Разлагая в ряд Фурье начальную погрешность, и используя уравнение для эволюции погрешности в методе простой итерации:

$$\nu_h^{i+1} = \nu_h^i + \tau(L_h\nu_h^i) = (E + \tau L_h)^i \nu_h^0$$

получим

$$\nu_h^{i+1} = \sum_{p,q} c_{p,q}(E + \tau L_h)^i \varphi_{k,m}^{p,q} = \sum_{p,q} c_{p,q}(1 - \tau\lambda^{p,q})^i \varphi_{k,m}^{p,q}\,,$$

где $\lambda_{p,q}$ — собственные числа оператора L_h.

Анализ скорости сходимости показывает, что при значении итерационного параметра $\tau = \tau_{\text{opt}} = h^2/4$ сходимость метода простой итерации будет максимально быстрой.

Проанализируем уравнение:

$$\nu_h^{i+1} = \sum_{p,q} c_{p,q}(1 - \tau_{\text{opt}}\lambda^{p,q})^i \varphi_{k,m}^{p,q}$$

Из него видно, что погашение Фурье-компонент погрешности происходит неравномерно: в средней части спектра (при $\lambda \approx \gamma_2/2$, где γ_2 — верхняя граница спектрального интервала) существенно быстрее, чем на его краях, и результат определяется именно скоростью погашения на краях спектра.

Метод простой итерации гасит компоненту погрешности (p, q), умножая ее за один шаг на $(1 - \tau\lambda^{p,q})$. Высокие частоты расположены, очевидно, между $\lambda^{1,N/2} = (4/h^2)\sin^2(\pi N/4N) \approx 2/h^2$ и $\lambda^{N-1,N-1} \approx 8/h^2$.

Выбирая τ оптимальным для этой части спектра, получим убывание негладких компонент погрешности (невязки) в процессе итераций со скоростью $(0.6)^i$, т.е. достаточно быстрое.

На остальной части спектра сходимость, конечно, очень медленная. Так компонента $(1, 1)$ погрешности убывает с показателем $1 - 2\pi^2\tau = 1 - 0.4(\pi/N)^2$ за шаг.

В целом итерационный процесс оказывается неэффективным. Однако небольшое число таких итераций "сглаживает"невязку: высокие гармоники в ней гасятся, основную роль играет гладкая компонента.

Введем вспомогательную, более грубую сетку G_H, $(H > h)$. После проведения i итераций невязка $r_h^i = -f_h - L_h u_h^{i-1}$ стала гладкой функцией. Возьмем ограничение невязки на H-сетку $r_H^i = I_h^H r_h^i$ и

решим на грубой сетке уравнение $L_H \nu_H^i = r_H^i$. Здесь L_H — аппроксимация оператора L_h на H-сетке. Эта задача заметно проще исходной хотя бы потому, что в ней меньше число неизвестных и число обусловленности также имеет меньшее значение. Затем, интерполируя функцию ν_H^i на h-сетку, получим $\nu_h^i = I_H^h \nu_H^i$. Вычитая ее из u_h^{i-1}, приходим к новому приближению

$$u_h^i = u_h^{i-1} + \nu_h^i$$

Выражая ν_h^i через операторы I_H^h, L_H^{-1}, I_h^H и учитывая уравнение для r_h^i, получим:

$$\begin{aligned} u_h^i = u_h^{i-1} + I_H^h L_H^{-1} I_h^H (-f_h - L_h u_h^{i-1}) = \\ = (E - I_H^h L_H^{-1} I_h^H L_h) u_h^{i-1} - I_H^h L_H^{-1} I_h^H f_h \end{aligned}$$

Введем оператор $M_h^H(\mathrm{CGC}) = E - I_H^h L_H^{-1} I_h^H L_h$ (CGC — *Coarse Grid Correction* — коррекция на грубой сетке). Тогда искомый нами оператор L_h' для коррекции невязки, будет иметь вид

$$(L_h')^{-1} = I_H^h L_H^{-1} I_h^H$$

Метод $M_h^H(\mathrm{CGC})$ не сходится, поскольку невозможно осуществить коррекцию для тех значений r_h^i, при которых $I_h^H r_h^i = 0$. Причина обращения $I_h^H r_h^i$ в ноль заключается в том, что при переходе с сетки h, на сетку $H = 2h$, некоторые собственные функции оператора L обращаются в ноль.

$$\varphi^{N/2,q}(kH, mH) = \varphi^{N/2,q}(2kh, 2mh) = \sin(k\pi)\sin(2mq\pi h) = 0$$

Поэтому важно проанализировать, что происходит собственными функциями при переходе с одной сетки на другую.

Классификация функций по длинам волн

Длина волны $l = 2 \times (\text{номер гармоники})^{-1} = 2/p$, $h = 1/N$. Следовательно можно считать, что

- Низкочастотные: $p < N/2 \Longrightarrow l > 4h$.
- Высокочастотные: $p \geq N/2 \Longrightarrow l \leq 4h$.
- Неразрешимые: $p \geq N \Longrightarrow l \leq 2h$.

Действительно, при этом

$$\sin[(2N-p)\pi x] = -\sin(p\pi x) \quad \text{для} \quad x \in \Omega_{1/N}.$$

Эффекты переноса на грубую сетку

- Высокочастотные: $p \geq N/2 \Longrightarrow l \leq 4h$.
- Сетка $G_H : H = 2/N,\ x = kH,\ 0 \leq k \leq N/2$.

Низкие частоты: $1 \leq p,\ q \leq N/2 - 1$, длина волны $\geq 4h = 2H$ видны на G_h, G_H. Высокие частоты: $p,\ q \geq N/2$, длина волны $\leq 4h = 2H$ видны на G_h, но не на G_H. Частоты $p, q \geq N/4$ становятся высокочастотными на G_H.

Трансформация собственных функций

$$\begin{array}{cc} \text{на} \quad G_H & \varphi_H^p = -\varphi_h^{N-p} \\ \sin(kp\pi H) = -\sin[2k(N-p)\pi h] & \end{array}$$

Таким образом, высокочастотные компоненты *невозможно* скорректировать посредством грубой сетки, но коррекция на грубой сетке имеет смысл, если низкие частоты доминируют в разложении погрешности, а для этого, перед коррекцией на грубой сетке необходимо провести сглаживание невязки.

Нужна хорошая процедура сглаживания, которая позволила бы быстро погасить негладкие компоненты невязки и превратить ее в достаточно гладкую функцию. В качестве таких процедур целесообразно вместо метода простой итерации использовать взвешенный метод Якоби или метод Гаусса-Зайделя, скорость сходимости которых выше.

Комбинация сглаживания с последующей коррекцией на грубой сетке приводит к двухсеточной схеме коррекции — CS (*Correction Scheme*), которая является теоретической основой многосеточного метода.

Схема коррекции (CS)

Обозначим

$$u_h^i = \mathrm{RELAX}^\nu(u_h^{i-1}, L_h, f_h) \Longleftrightarrow u_h^i = S_h^\nu u_h^{i-1}.$$

Таблица 2.1. Схема коррекции

1. Предварительное сглаживание	$u_h^i = \mathrm{RELAX}^\nu(u_h^{i-1}, L_h, f_h)$
2. Вычисление невязки	$r_h^i = -f_h - L_h u_h^i$
3. Ограничение невязки	$r_H^i = I_h^H r_h^i$
4. Точное решение задачи на грубой сетке	$L_H \nu_H^i = r_H$
5. Перенос поправки	$\nu_h^i = I_H^h \nu_H$
6. Уточнение решения	$u_h^* = u_h^i + \nu_h^i$
7. Послесглаживание	$u_h^i = \mathrm{RELAX}^{\nu 1}(u_h^*, L_h, f_h)$

Общий оператор одного шага схемы CS:

$$M_h^H(\mathrm{CS}) = S_h^{\nu 1} M_h^H(\mathrm{CGC}) S_h^{\nu 2}$$

§ 2.2. Многосеточный метод: V-цикл

Основная идея многосеточного метода заключается в том, что на грубой сетке не обязательно решать уравнение $L_H \nu_H^i = r_H$ точно. Достаточно провести несколько сглаживаний и получить приближенное решение $\tilde{\nu}_H^i$. ведет к многосеточному методу.

При этом мы эффективно подавляем высокочастотные для данной сетки компоненты погрешности решения, а остающиеся низкочастотные составляющие можно эффективно сгладить при переходе на следующую, еще более грубую сетку $2H$. Вспомним (см. предыдущий параграф), что компоненты ошибки эффективно подавляются сглаживанием на той сетке, на которой они разрешимы и высокочастотны, а при переходе на более грубую сетку низкие частоты становятся высокими. На сетке $2H$ проводим очередную коррекцию решения: сглаживаем высокочастотную составляющую, а для подавления низких частот используем сетку $4H$ и т.д. Таким образом, рекурсивное использование схемы коррекции (CS)

Введем последовательность сеток

$$\{G_k\}_{k=0}^{k=l}, \quad h_l = h_{l-1}/2$$

и соответствующие операторы

$$L_l,\ S_l,\ I_l^{l-1},\ I_{l-1}^{l}\ .$$

Тогда один V-цикл расчетов можно графически изобразить в следующем виде:

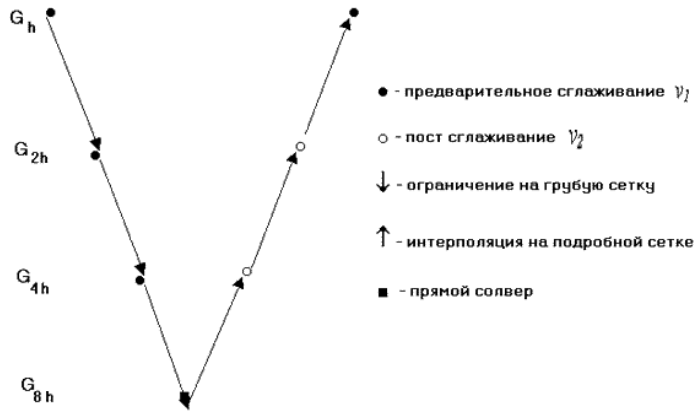


Рис. 2.1. Схематическое изображение V-цикла

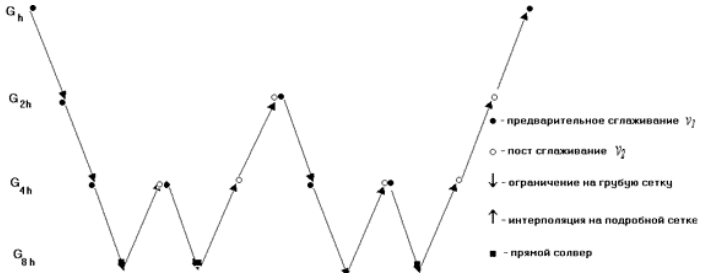


Рис. 2.2. Схематическое изображение W-цикла

Рекурсивное определение оператора шага многосеточного метода:

$$M_1 = S_1^{\nu 2}(E - I_0^1 L_0^{-1} I_1^0 L_1)\, S_h^{\nu 1}\,,$$
$$M_{k+1} = S_{k+1}^{\nu 2}(E - I_k^{k+1}(E - M_k^{\gamma}) L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{\nu 1}\,,$$

ведет к многосеточному методу, где $k = 2, \ldots, M-1$.

$(E - M_k^{\gamma})$ отражает решение задачи на сетке G_k, полученное с помощью γM_k многосеточных итераций, γ — рекурсивный параметр, который влияет на качество коррекции на грубой сетке, для сеток разного уровня может быть разным.

Стандартный случай: $\gamma \leq 4$

- V-цикл при $\gamma = 1$ наиболее прост в вычислительном смысле;
- W-цикл при $\gamma = 2$ более сложен для вычислений, но и более надежен (см. рис 2.2).

Глава 3

Применение многосеточного метода

§ 3.1. Применение многосеточного метода к решению нелинейных задач: схема полной аппроксимации (FAS)

Ранее мы обсуждали использование многосеточных методов применительно к решению линейных задач. Ясно, что если линейный многосеточный метод использовать в сочетании с каким-либо методом линеаризации, например, методом Ньютона, то это сочетание можно будет применить и для решения нелинейных задач. Этот "непрямой"подход использования линейного многосеточного метода для решения нелинейной задачи более менее очевиден.

Рассмотрим нелинейное эллиптическое уравнение $Lu = -f$ и его дискретный аналог:

$$L_h u_h = -f_h$$

Линеаризуем уравнение по методу Ньютона.

Напомним суть метода Ньютона.

Если x_0 есть приближенное значение корня уравнения $F(x) = 0$, то в качестве более точного приближения берется

$$x_1 = x_0 - \frac{F(x_0)}{F'(x_0)}$$

Заменой x_0 на x_1 может быть получено следующее приближение и т. д.

Процесс последовательных приближений всегда сходится, если только корень, а не кратный (т.е. $F'(a) \neq 0$).

В нашем случае,

$$L_h u_h = -f_h$$
$$F = L_h u_h + f_h$$
$$F' = L'_h u_h + \frac{\partial f_h}{\partial u_h} = L' u_h$$
$$u_h^{k+1} = u_h^k - \frac{L_h u_h^k + f_h}{L'_h u_h^k}$$
$$L'_h u_h^k \left(u_h^{k+1} - u_h^k\right) + L_h u_h^k = -f_h$$
$$u_h^{k+1} = u_h^k + \nu_h^{k+1}, \text{ где } \nu - \text{погрешность}$$
$$L'_h \nu_h^{k+1} + L_h^k = -f_h$$

Таким образом, мы получили систему линейных уравнений:

$$L'_h \nu_h^{k+1} + L_h^k = -f_h$$
$$\nu_h^{k+1} = u_h^{k+1} - u_h^k, \text{ где } k = 0, 1, 2, \ldots$$

Для решения каждого из этих линейных уравнений можно применить метод многосеточных итераций, причем применить двояко.

1-й способ. Адаптировать число многосеточных итераций под каждую Ньютоновскую итерацию, т.е. мы должны вычислить значение u_h^{k+1} с одинаковой точностью, как по методу Ньютона, так и многосеточным методом. Если метод Ньютона имеет сходимость порядка h^2, а многосеточный метод сходится линейно, то число многосеточных итераций должно удваиваться на каждую следующую итерацию Ньютона, т.е. следует удваивать число многосеточных циклов. Чтобы это осуществить, мы должны ввести некоторый механизм контроля за скоростью сходимости метода Ньютона, что не всегда бывает легко.

2-ой способ. Фиксировать число многосеточных итераций на каждый шаг метода Ньютона. Например, можно делать одну многосеточную итерацию на каждый шаг Ньютона. Для этого метод Ньютона нужно ограничить так, чтобы его скорость сходимости стала линейной. Это можно сделать, заморозив $L'_h u_h^k$ на нулевой

итерации, т.е. считая $L'_h u_h^k = L_h u_h^0$. Недостатком этого метода является большой объем работы, связанный с линеаризацией.

Схема полной аппроксимации

Наряду с таким "непрямым"подходом существует прямой метод использования метода многосеточных итераций для решения нелинейных задач. Он обладает рядом преимуществ перед "непрямым наивным"подходом:

1. не требуется согласовывать внутренние и внешние итерации;
2. структура цикла та же, что и для линейных задач;
3. иногда быстрее сходится;
4. линеаризацию проводим внутри цикла на самых грубых сетках.

Вспомним использование многосеточного метода для решения линейных задач, где использовалась линейность оператора L_h. В схеме коррекции при написании уравнения для погрешности ν_h^i:

$$L_h u_h = -f_h$$
$$L_h(u_h^i + \nu_h^i) = -f_h$$

Тогда имеем следующее уравнение для невязки r_h:

$$L_h \nu_h^i = -f_h - L_h u_h^i = r_h \ .$$

Ограничивая невязку на более грубую сетку, получаем:

$$L_H \nu_H = I_h^H r_h$$

Идея состоит в изменении схемы коррекции таким образом, чтобы туда вместо линейного входил нелинейный оператор.

Пусть теперь оператор L_h является нелинейным. Тогда

$$L_h u_h = -f_h$$
$$L_h(u_h^i + \nu_h^i) = -f_h$$

Вычтем из правой и левой части последнего уравнения $L_h u_h^i$, тогда

$$L_h(u_h^i + \nu_h^i) - L_h u_h^i = -f_h - L_h u_h^i = r_h$$

Ограничиваем решение на более грубую сетку:

$$L_H(\hat{I}_h^H u_h^i + \nu_H^i) - L_H \hat{I}_h^H u_h^i = I_h^H r_h$$

может быть, что $\hat{I}_h^H \neq I_h^H$.

Пусть точное решение на грубой сетке

$$u_h = \hat{I}_h^H w_h + \nu_H^i \ ,$$

тогда

$$L_H u_H = I_h^H r_h + L_H \hat{I}_h^H u_h^i$$

Пусть u_H^i — новая аппроксимация решения на грубой сетке,

$$u_H^i = \tilde{\nu}_H + \hat{I}_h^H u_h^i \, ,$$

где $\tilde{\nu}_H$ — аппроксимация погрешности.

Считаем, что $u_H^i \approx u_H = \hat{I}_h^H u_h^i + \tilde{\nu}_H$, отсюда

$$\tilde{\nu}_H = u_H - \hat{I}_h^H u_h^i$$

Интерполируем погрешность на более подробную сетку:

$$\tilde{\nu}_h = I_H^h \tilde{\nu}_H$$

Корректируем решение на более подробной сетке:

$$\tilde{u}_h = u_h^i + I_H^h \tilde{\nu}_H = u_h^i + u_H^i - \hat{I}_h^H u_h^i$$

Переносить сразу $\tilde{u}_h = I_H^h u$ будет неправильно, т.к. мы внесем большую ошибку интерполяции полной аппроксимации решения вместо того, чтобы интерполировать достаточно гладкую функцию погрешности решения $\tilde{\nu}_H$.

Существенное замечание. На более грубой сетке мы решаем уравнение для полной аппроксимации решения (отсюда название метода — схема полной аппроксимации). Но на подробную сетку мы переносим погрешность и невязку, а не решение, поскольку именно погрешность и невязка являются гладкими функциями (сглаживание происходит при итерациях на подробной сетке методами Якоби-Ньютона, Якоби-Пикара).

Схема полной аппроксимации (FAS) для двух сеток

Алгоритм реализации схемы полной аппроксимации для двух сеток состоит из следующих действий:

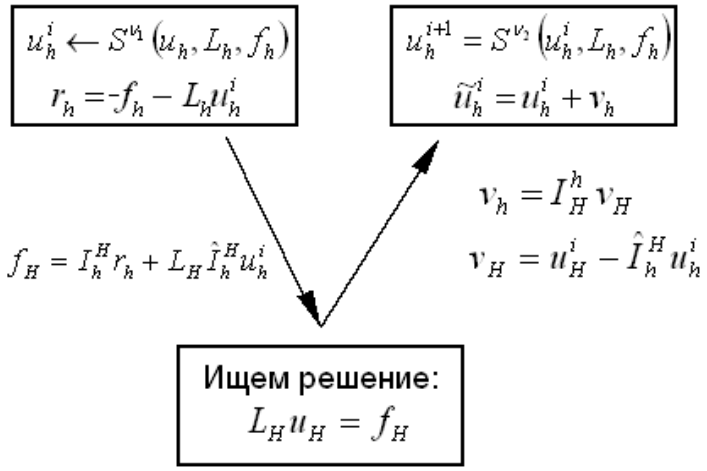


Рис. 3.1. Схематическое представление схемы полной аппроксимации (FAS) для двух сеток

1. предварительное сглаживание погрешности и невязки на подробной сетке с шагом h

$$u_h'^{(n)} = \mathrm{RELAX}^{\nu 1}(u_h^{(n-1)}, L_h, f_h)$$

2. расчет невязки на подробной сетке:

$$r_h^{(n)} = -f_h - L_h u_h'^{(n)}$$

3. ограничение (пересчет) невязки на более грубую сетку:

$$r_H^{(n)} = I_h^H r_h^{(n)}$$

4. получение точного решения на грубой сетке:

$$L_H u_H^{(n)} = L_H \hat{I}_h^H u_h'^{(n)} + r_H^{(n)}$$

5. вычисление погрешности на грубой сетке и ее последующий пересчет путем интерполяции на более подробную сетку:

$$\nu_H'^{(n)} = u_H^{(n)} - \hat{I}_h^H u_h'^{(n)}$$

$$\nu_h'^{(n)} = I_H^h \nu_h'^{(n)}$$

6. коррекция аппроксимации решения на подробной сетке:

$$u_h''^{(n)} = u_h'^{(n)} + \nu_h^{(n)}$$

7. окончательное сглаживание решения путем осуществления нескольких итераций с помощью стандартного итерационного метода, выбирая в качестве начального приближения скорректированное решение, полученное в п.6:

$$u_h^{(n)} = \mathrm{RELAX}^{\nu 2}(u_h''^{(n)}, L_h, f_h)$$

Таким образом, подводя итоги, можно сказать, что при решении нелинейных уравнений многосеточные методы можно использовать тремя способами:

1. Линеаризация по Ньютону с адаптацией числа многосеточных итераций на каждую итерацию Ньютона.

2. Линеаризация по Ньютону с фиксированным числом многосеточных итераций на каждый шаг по Ньютону. По сути этот метод Ньютона упрощается так, чтобы его скорость сходимости была линейной.

3. Нелинейный многосеточный метод.

Расчеты на конкретных задачах показали, что все три метода сходятся примерно одинаково, но применение FAS позволяет:

1. избежать глобальной линеаризации;

2. не производить расчет больших Якобианов;

3. производить линеаризацию как можно позже;

4. легко трансформировать схему коррекции для линейного метода в схему коррекции для нелинейного метода, изменяя только правую часть;

5. использовать разнообразные алгоритмы сглаживания.

Рассмотрим некоторую особенность схемы полной аппроксимации, которая позволяет получить ошибку ограничения при переходе с подробной на более грубую сетку. Это нам понадобится в дальнейшем для определения погрешности ограничения перехода от сетки к сетке для оценки погрешности аппроксимации в разделе "Техника многоуровневой адаптации"(MLAT):

$$L_h u_h = -f_h$$
$$L_h(u_h^i + \nu_h^i) = -f_h$$

Вычитаем $L_h u_h^i$ из правой и левой части, тогда:

$$L_h(u_h^i + \nu_h^i) - L_h u_h^i = -f_h - L_h u_h^i$$
$$L_h(u_h^i + \nu_h^i) = r_h + L_h u_h^i$$

При переходе на более грубую сетку получим:

$$L_H u_H = -f_H = I_h^H r_h + L_H \hat{I}_h^H u_h^i$$

Подставляя явное выражение для r_h, имеем:

$$L_H u_H = -I_h^H f_h - I_h^H L_h u_h^i + L_H \hat{I}_h^H u_h^i$$
$$L_H u_H = -I_h^H f_h + \tau_h^H = -f_H,$$

где $\tau_h^H = L_H \hat{I}_h^H u_h^i - I_h^H L_h u_h^i$ является поправкой перехода с более подробной на более грубую сетку, поправкой к уравнению на грубой сетке, сделанной для того, чтобы решение на грубой сетке совпадало с решением на подробной сетке.

Замену $L_H u_H = f_H$ на $L_H u_H^i = f_H + \tau_H^h$ необходимо провести для того, чтобы гарантировать, что решение на грубой сетке совпадает с решением на подробной сетке.

Аналогично можно ввести $\tau = \tau_{\text{cont}}^h$ — погрешность перехода с непрерывного решения на дискретное

$$\tau_{\text{cont}}^h = L_h(u)_h - (Lu)_h = L_h I_{\text{cont}}^h u - I_{\text{cont}}^h Lu$$

Теперь можно пересмотреть нашу точку зрения на весь многосеточный процесс: вместо того, чтобы рассматривать грубую сетку как инструмент для увеличения скорости сходимости решения на подробной сетке, мы можем рассматривать подробную сетку как

инструмент для вычисления поправки τ_h^H к уравнению на грубой сетке.

Так как эта поправка зависит от негладкой компоненты решения, мы получаем ее, интерполируя решение на более подробную сетку, и корректируя там негладкие компоненты путем сглаживания. Получив поправку τ_h^H путем "визита"на более подробную сетку, мы продолжаем процесс решения на сетке . Затем мы опять "посещаем"h и считаем τ_h^H. В этом случае аппроксимацию решения получаем следующим образом: $u_h^{\prime\prime(n)} = u_h^{\prime(n)} + I_h^H \nu_h^{\prime h(n)}$, интерполируем на подробную сетку погрешность, чтобы не потерять негладкие компоненты решения, уже полученные путем сглаживания в предыдущих "визитах".

Весь процесс в итоге дает решение на сетке H, которое можно улучшить, посетив более подробную сетку h. Эта точка зрения отражает тот факт, что полное решение на подробной сетке представлено во всех более грубых сетках, что открывает широкие алгоритмические возможности, о которых будет сказано в дальнейшем.

§ 3.2. Полный многосеточный метод

Можно существенно увеличить эффективность многосеточных методов, применяемых для решения эллиптических краевых задач, используя их в форме полного многосеточного метода (*Full Multigrid* — FMG), также называемого методом вложенных или гнездовых (*nested*) итераций. Хотя FMG применяется для решения как линейных, так и нелинейных задач (используя FAS), мы ограничимся рассмотрением только линейных случаев.

Полный многосеточный подход можно рассматривать с различных точек зрения. Мы здесь рассмотрим его в "узком"смысле, а именно как "приближение"прямого солвера (для данной эллиптической задачи, аппроксимированной на сетке с шагом h), который характеризуется следующими свойствами:

1. Аппроксимация решения $u_h^{(n)}$ рассчитывается до тех пор, пока "алгебраическая"погрешность становится меньше или равна погрешности дискретизации:

$$\|u - u_h^{(n)}\|_\infty \approx \|u - u_h\|.$$

2. Число арифметических операций, необходимых для достижения первого условия пропорционально числу точек N на самой подробной сетке.

Основные проблемы, связанные с итерационными методами:

1. как выбрать начальное приближение;

2. каков критерий прекращения расчета.

По **1-му пункту** используем аппроксимацию решения на грубой сетке в качестве начального приближения при расчете на подробной сетке, так называемая "вложенная итерация".

По **2-му пункту** очевидно, что если алгебраическая погрешность $\|u-u_h^{(n)}\|$ приблизительно равна погрешности аппроксимации $\|u-u_h\|$, то расчет далее проводить бессмысленно.

Основные принципы многосеточного метода

Алгоритм многосеточного метода можно кратко описать набором следующих действий:

1. Проведение сглаживания погрешности и невязки на подробной сетке.

2. Осуществление коррекции решения на грубых сетках.

3. Реализация вложенных итераций.

Если ограничиться выполнением первых двух операций, то в результате мы реализуем простую многосеточную итерацию, которую можно графически представить, например V- циклом. Если осуществить еще и третью операцию, т.е. сделать несколько вложенных итераций, то будет реализован алгоритм полного многосеточного метода. Для того, чтобы понять механизм его действия, вспомним сначала как соотносятся друг с другом решения, полученные на сетках с разным шагом.

Вернемся к схеме полной аппроксимации (FAS). Решаем уравнение $L_h u_h = -f_h$ на последовательных сетках

$$h_1 > h_2 > ... > h_{M-1} > h,$$

начиная с сетки

$$G_M(L_M u_M \ = \ -f_M).$$

Решение на грубых сетках используется для коррекции аппроксимации решения на подробной сетке.

Решения на разных сетках соотносятся друг с другом через относительную локальную погрешность аппроксимации:

$$L_M u_M = -f_M$$
$$\dots$$
$$L_{l-1} u_{l-1} = -f_{l-1} + \tau_{l-1}^{l}, \quad \text{где} \quad f_{l-1} = I_l^{l-1} f_l$$
$$\dots$$
$$L_1 u_1 = -f_1 + \tau_1^2, \quad \text{где} \quad f_1 = I_l^{l-1} f_l$$

τ — гарантирует, что решения, полученные на грубых сетках, будут иметь ту же точность, что и решения на подробной сетке.

Решаем задачу $Lu = -f$ на последовательности сеток

$$h_1 > h_2 > ... > h_{M-1} > h.$$

На каждой сетке имеем своё разностное уравнение:

$$L_M u_M = -f_M$$
$$\dots$$
$$L_{l-1} u_{l-1} = -f_{l-1}$$
$$\dots$$
$$L_1 u_1 = -f_1$$

В отсутствии согласованного решения имеем относительную локальную погрешность аппроксимации:

- u_1 аппроксимирует u с точностью $O(h_1^p)$,
- u_2 аппроксимирует u с точностью $O(h_2^p)$,
- u_M аппроксимирует u с точностью $O(h_M^p)$.

Хорошая аппроксимация, когда u_{l-1} близко к u_l по крайней мере с точностью до $O(h_{l-1}^P)$. Переход от сетки к сетке u_{l-1} служит начальным приближением u_l.

Полный многосеточный метод или метод вложенных (гнездовых) итераций (Full Multigrid)

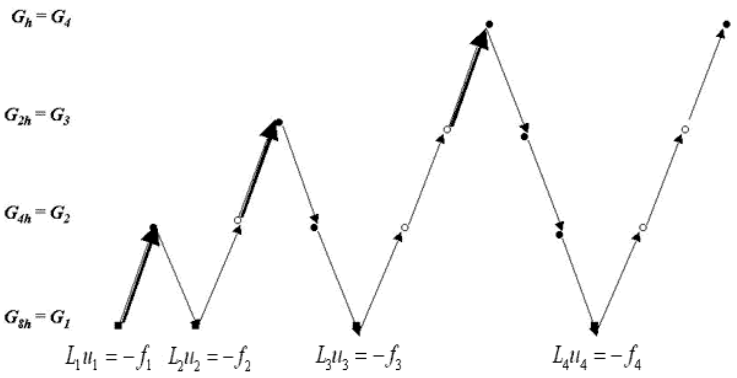


Рис. 3.2. Схематическое представление FMG метода для четырех сеток

Обозначения: • — ν_1 итераций по сглаживанию (предварительное сглаживание), ∘ — ν_2 - постсглаживание, ↑ — перенос ошибки на более подробную сетку (интерполяция ошибки), ↓ — ограничения на более грубую сетку (FMG интерполяция), ■ — точные решения на самой грубой сетке.

Полный многосеточный метод более эффективен, чем просто многосеточный метод, за счет комбинированного использования решения на более грубых сетках в качестве начального приближения и многосеточного метода для расчета самого решения на подробных сетках. Такой подход применим для решения как линейных, так и нелинейных уравнений.

- Решаем уравнение $L_h u_h = -f_h$ на последовательности сеток
$$h_1 > h_2 > \cdots > h_{M-1} > h\,.$$
- Ищем решение на самой грубой сетке G_1 : $L_1 u_1 = -f_1$.
- На подробных сетках при $l = 2, \ldots, M$ интерполируем начальную аппроксимацию решения
$$u_l^{(0)} = I_{l-1}^l u_{l-1}\ ,$$

рассчитываем значение u_l, выполняя стандартный многосеточный цикл на сетках $G_1, G_2, \ldots, G_l$, начиная с уровня l.

Новые свойства FMG

1. FMG рассчитывает значения решения вплоть до уровня ошибки аппроксимации.

2. Объем вычислительных операций при этом имеет порядок $O(n)$, где n — число узлов подробной сетки.

Можно показать простыми оценками, что после полного многосеточного цикла при достаточно большом количестве сеток

$$\|u_h - u_h^{(n)}\| \leq \|u - u_h\|$$

алгебраическая ошибка становится меньше или равна ошибке аппроксимации.

Для простоты мы ограничимся прямоугольными сетками.

Предположим:

1. Пусть нормы итерационного M_l и интерполяционного операторов ограничены:

$$\|M_l\| \leq \eta \leq 1$$

$$\|I_{l-1}^l\| \leq C \quad (l = 1, 2, \ldots).$$

2. Считаем, что ошибка аппроксимации и ошибка интерполяции имеют порядок k_1 и k_2 соответственно:

$$\|u - u_l\| \leq K_1 h_l^{k_1} \text{ — ошибка аппроксимации}$$

$$\|u - I_{l-1}^l u - w_l\| \leq K_2 h_l^{k_2}$$

$$I_{l-1}^l u + w_l = u_l^0 \text{ — начальное приближение}$$

w_l — некая сеточная функция, добавленная, чтобы включить в интерполяционный процесс граничные условия, k_1, k_2, K_1, K_2 не зависят от l.

Считаем, что $h_{l-1}/h_l = \xi$ $(l = 1, 2, \ldots)$, $\xi > 1$. Получим, учитывая предположения, сделанные выше, утверждение.

Лемма 1 *Пусть $K_1 > K_2$ и $\eta^r < 1$, где r — число итераций.*

$$A = C\xi^{K_1}$$

Тогда для всех $l \geq 1$ справедливы следующие оценки:

$$\|\tilde{u}_l - u_l\| \leq \delta^* h_l^{K_1} \quad \text{, где} \quad \delta^* = \frac{\eta^r K}{1-\eta^r A} + O(1) \quad \textit{при} \quad l \to \infty \ (1) \quad ,$$

где $\tilde{u}_l$ — окончательное FMG-решение $L_1 u_1 = -f_1$.

$$K = \begin{cases} K_1(1+A) + K_2 & (1) \\ K_1(1+A) \end{cases}$$

Если мы дополнительно имеем нижнюю границу для ошибки аппроксимации

$$\|u - u_l\| \geq \hat{K}_1 h_l^{k_1}, \quad \hat{K}_1 > 0$$

не зависящую от l, то тогда $h_l^{K_1} = \|u - u_l\|/\hat{K}_1$ в (1) и получаем

$$\|\tilde{u}_l - u_l\| \leq \beta^* \|u - u_l\|$$

$$\textit{где} \quad \beta^* = \frac{K\eta^r}{\hat{K}_1(1-\eta^r A)} + O(1), \quad l \to \infty$$

Доказательство.
По определению FMG метода для всех $l \geq 1$ имеем

$$\tilde{u}_l - u_l = M_l^r(u_l^0 - u_l) = M_l^r u_l^0 - M_l^r u_l$$
$$M_l^r u_l = u_l$$
$$u_l^0 = I_{l-1}^l \tilde{u}_{l-1} + w_l u_l^0 - u_l = I_{l-1}^l(\tilde{u}_{l-1} - u_{l-1}) +$$
$$+ I_{l-1}^l(u_{l-1} - u) + (I_{l-1}^l u + w_l - u) + (u - u_l)$$

После раскрытия скобок:

$$u_l^0 - u_l = I_{l-1}^l \tilde{u}_{l-1} + w_l - u_l$$
$$\|\tilde{u}_l - u_l\| \leq \|M_l^r\| \|u_l^0 - u_l\| = \eta^r \|u_l^0 - u_l\|$$

Пусть $\delta_l = \dfrac{\|\tilde{u}_l - u_l\|}{h_l^{K_1}}$, тогда

$$I_{l-1}^{l}(\tilde{u}_{l-1}-u_{l-1}) \le \|I_{l-1}^{l}\|\|\tilde{u}_{l-1}-u_{l-1}\| = C\delta_{l-1}h_{l-1}^{K_1} =$$
$$= A\xi^{-K_1}h_{l-1}^{K_1}\delta_{l-1} = Ah_{l-1}^{-K_1}h_l^{K_1}h_{l-1}^{K_1}\delta_{l-1}$$
$$I_{l-1}^{l}(u_{l-1}-u) \le \|I_{l-1}^{l}\|\|u_{l-1}-u\| = CK_1h_{l-1}^{K_1} =$$
$$= A\xi^{-K_1}K_1h_{l-1}^{K_1} = Ah_{l-1}^{-K_1}h_l^{K_1}K_1h_{l-1}^{K_1}$$
$$I_{l-1}^{l}u + w_l - u \le \|I_{l-1}^{l}u + w_l - u\| = K_2h_l^{K_2}$$
$$u - u_l \le \|u-u_l\| \le K_1h_l^{K_1}$$
$$u_l^0 - u_l \le \|u_l^0 - u_l\| \le \delta_l h_l^{K_1}$$
$$\|\tilde{u}_l - u_l\| \le \delta h_l^{K_1} \le \eta^r(Ah_l^{K_1}\delta_{l-1} + Ah_l^{K_1}K_1 + K_2h_l^{K_2} + K_1h_l^{K_1})$$
$$\delta_l \le \eta_r(A\delta_{l-1} + K_1(A+1) + K_2h_l^{k_2-K_1})$$

Отсюда легко получить

$$\|\tilde{u}_l - u_l\| \le \delta^* h_l^{K_1} \quad \text{где} \quad \delta^* = \frac{\eta^r K}{1-\eta^r A}$$

$$K = \begin{cases} K_1(A+1) + K_2\ , & K_1 = K_2 \\ K_1(A+1)\ , & K_2 > K_1 \end{cases}$$

Оценим эффективность полного многосеточного метода (количество вычислительной работы, которая требуется для получения решения):

$$W_M^{FMG} \approx \sum_{l=2}^{l=M} \left(n_l W_l + W_{l-1}^{FMG-INT}\right) \quad ,$$

где W_l — объем вычислительной работы на уровне l, n_l — число узлов сетки l, $W_{l-1}^{FMG-INT}$ — работа по интерполяции решения на грубой сетке.

Считаем, что $n_l = 2^d n_l$, где d — размерность.

Если мы распишем сумму, начиная с самой подробной сетки M, при $l \to \infty$, то получим бесконечную геометрическую прогрессию.

$$W_M^{FMG} \approx (n_M W_M + W_{M-1}^{FMG-INT}) +$$
$$+(n_M W_M + W_{M-1}^{FMG-INT})(1/2^d) +$$
$$+(n_M W_M + W_{M-1}^{FMG-INT})\left(1/2^d\right)^2 + \ldots$$

Знаменатель геометрической прогрессии равен $1/2^d$.

Используя выражение для суммы бесконечной геометрической прогрессии:

$$\sum_{n=0}^{\infty} aq^n = \frac{a}{1-q} \quad , \quad \text{где} \quad q = \frac{1}{2^d},$$

мы получим следующее выражение

$$W_M^{FMG} \approx (n_M W_M + W_M^{FMG-INT}) \frac{1}{1-1/2^d} = \\ = (n_M W_M + W_M^{FMg-INt}) \frac{2^d}{2^d-1},$$

где W_M — это объем вычислительной работы на самой подробной сетке.

Выражение для него можно записать в следующем виде:

$$W_M = \sum_{l=2}^{M} \gamma^{M-l} W_l^{l-1},$$

где W_l^{l-1} — объем работы на двух сетках, γ — рекурсивный параметр, зависящий от типа цикла.

$$W_M \approx \sum_{l=2}^{M} \gamma^{M-l} (\nu W_l(S) + W_l(R) + W_l(P)) n_l \approx \\ \approx (\nu W_M(S) + W_M(R) + W_M(P)) n_M \sum_{l=2}^{M} \frac{\gamma^{M-1}}{2^d} \approx \\ \approx (\nu+1) W_M(S)\, n_M \sum_{l=2}^{M} \left(\frac{\gamma}{2^d}\right)^{M-1} = \\ = (\nu+1) W_M(S)\, n_M \, \frac{1}{1-\gamma/2^d} = \\ = (\nu+1) W_M(S)\, n_M \, \frac{2^d}{2^d-\gamma} = \\ = (\nu+1) W_{M-1}^{FMG-INT} n_M \, \frac{2^d}{2^d-\gamma},$$

где $W_M(S) \approx W_{M-1}^{FMG-INT}$, ν — сглаживание, $W_l(R)$ — предварительное сглаживание, $W_l(P)$ — окончательное сглаживание.

Подставляем

$$W_M^{FMG} \approx \frac{2^d}{2^d-1}\left[n_M W_M + W_{M-1}^{FMG-INT}\right] = \\ = \frac{2^d}{2^d-1}\left[n_M(\nu+1)W_{M-1}^{FMG-INT}\frac{2^d}{2^d-1} + W_{M-1}^{FMG-INT}\right]$$

Пусть $W_M^{FMG} = 1\,\mathrm{RU}$ единица сглаживания (равна *one relaxation unit*), тогда:

$$W_M^{FMG} \cong \frac{2^d}{2^d-1}\left[n_M(\nu+1)\frac{2^d}{2^d-\gamma}+1\right]\mathrm{RU}$$

Если $\gamma = 1$, $d = 2$

$$W_M^{FMG} \cong \frac{4}{3}\left[n_M(\nu+1)\frac{4}{3}+1\right]\mathrm{RU}$$

Тем самым мы доказали, что объем вычислительных операций FMG метода имеет порядок пропорциональный числу узлов на самой подробной сетке.

Можно показать также, что при использовании FMG алгоритма итерационная работа, необходимая для уменьшения погрешности начального приближения в $1/\varepsilon$ раз, определяется только итерационной работой на самой подробной сетке.

§ 3.3. Техника многоуровневой адаптации

Неоднородное разрешение по пространству требуется во многих, возможно, в большинстве практических задач. Набор все более подробных сеток нужен для описания областей вблизи различных особенностей, около негладких границ, в пограничных слоях, около фронтов, ударных волн, в областях с резким изменением градиентов расчетных величин. Набор все более грубых сеток необходим для решения внешних задач в неограниченных областях.

Техника многоуровневой адаптации (*Multi Level Adaptive Technique* — MLAT) в сочетании с FAS (схемой полной аппроксимации) представляет удобный способ создавать неоднородные адаптивные структуры, которые к тому же являются весьма гибкими. MLAT позволяет быстро проводить локальное уточнение и локальную трансформацию координат и при этом решать задачу с эффективностью, характерной для многосеточных методов.

Итак, рассмотрим технику многосеточной адаптации.

Цель этого метода состоит в обеспечении требуемой точности на минимуме узлов сетки или обеспечении максимальной точности на данном количестве узлов сетки.

Идея метода состоит в том, что ошибка должна быть равно распределенной (т.е. на подробной сетке мы должны решать задачу с максимальной точностью и обеспечить ту же точность при решении на самой грубой сетке).

Реализация метода состоит в сочетании генерации сеток и движения сетки, подстраивания сетки к границам расчетной области и особенностям решения.

При этом могут возникнуть следующие проблемы:

1. возникновение нестационарной сеточной структуры;
2. необходимость выработки критерия перехода на более подробную сетку.

Обсуждение двух подходов:

1. Механически добавляем или убираем узлы сетки;
2. Перемещаем текущие узлы сетки в соответствии с эволюцией решения.

Критерий ввода более подробной сетки

С **физической** точки зрения надо учитывать

1. резкое изменение градиента расчетной величины;
2. резкое изменение отношения концентраций (например, при расчете химических процессов).

С **вычислительной** точки зрения надо учитывать возрастание локальной или глобальной ошибки ограничения.

Рассмотрим первый подход. Будем вносить или удалять дополнительные узлы на имеющуюся сетку. Введем набор сеток вблизи локальной подобласти. Схематическое представление FMG метода для четырех сеток представлено на рис. 3.3.

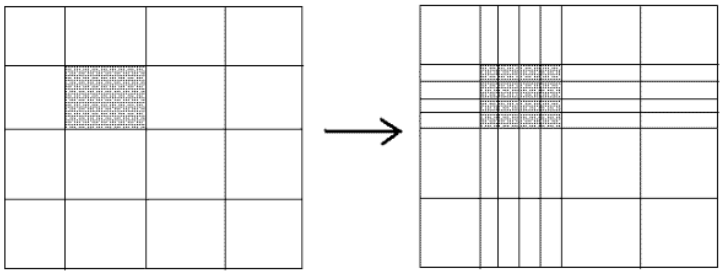


Рис. 3.3. Первый подход к введению дополнительного набора сеток

Преимущество такого подхода состоит в том, что структура сетки остается неизменной.

В качестве **недостатка** мы получаем дополнительные узлы сетки в тех областях, где уточнение нам не нужно.

Рассмотрим второй подход. Наблюдения показали, что при расчетах многосеточным методом различные сетки не требуется вводить во всей расчетной области. Отсюда родилась идея так называемой "неоднородной сеточной аппроксимации". Область, покрываемая подробной сеткой, может быть только частью области, покрытой более грубой сеткой. Каждая сетка h может быть распространена на те подобласти, где требуемый размер шага сетки должен быть меньше, чем $2h$.

Рассмотрим некую расчетную область. Допустим, что вблизи угла наблюдаются некоторые особенности решения. Вводим в этой подобласти набор более подробных сеток. Рисунок 3.4 иллюстрирует введение набора сеток вблизи локальной подобласти.

Такая структура является очень гибкой, т.к. локальное сеточное уточнение (или загрубленное) делается с помощью однородных сеток, которые относительно легко и "дешево"можно ввести. Более того, такая структура одновременно обеспечивает очень эффек-

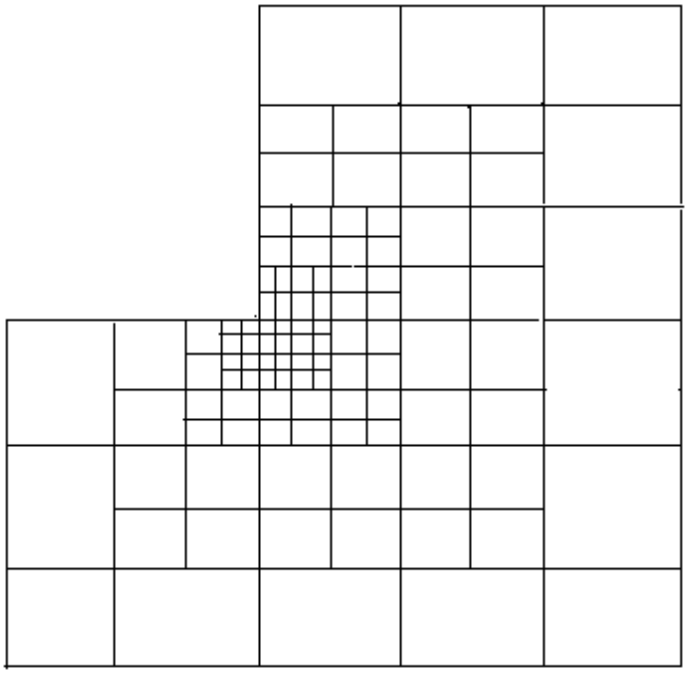


Рис. 3.4. Второй подход к введению дополнительного набора сеток

тивный процесс решения дифференциальных уравнений, используя имеющиеся сетки в качестве основы многосеточного солвера. При этом мы не вводим дополнительные узлы там, где они не нужны.

Рассмотрим двухсеточную структуру (см. рис. 3.5).

На рис. 3.5. введены обозначения: • — общие узлы сеток G_h и G_H, ∘ — узлы принадлежат только сетке G_h, □ — узлы принадлежат только сетке G_H, ■ — граничные узлы сетки G_h.

Попробуем использовать эти две сетки для построения многосеточного алгоритма решения. Для этого узлы двух последовательных сеток G_h и G_H будут выполнять различные функции.

1. Те точки G_H, которые одновременно являются и точками G_h, будем использовать для получения коррекции решения на сет-

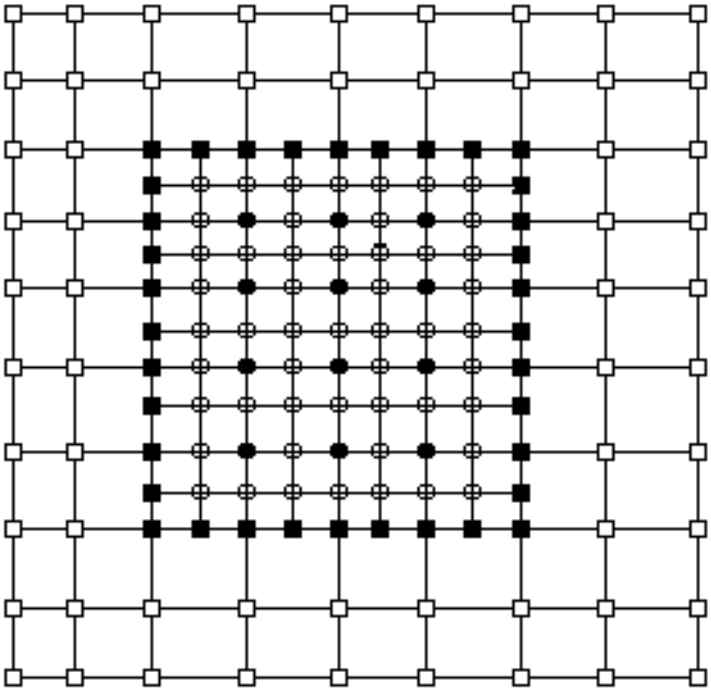


Рис. 3.5. Композиционная сетка, состоящая из двух подсеток $G_H = G_H \cup G_h$

ке G_H.

2. Узлы, принадлежащие только сетке G_H, будем использовать для получения решения на сетке с шагом H, причем будем считать, что H - самый подробный шаг.

3. Чтобы провести склейку решения на обеих сетках (решение на композитной сетке), необходимо подобрать подходящую аппроксимацию граничных узлов сетки G_h.

Будем использовать модифицированный метод полной аппроксимации (FAS), поскольку в тех областях расчетной области, которые не покрыты подробной сеткой h, на грубой сетке $H = 2h$, мы

должны иметь не только погрешность, но и полное решение. В этом случае нам подходит именно FAS. Мы используем сетку H для решения уравнений $L_H u_H = -F_H$ и $F_H = L_H \hat{I}_h^H u_h + \hat{I}_h^H r_h$, где есть общие узлы сеток h и $2h$. В других точках сетки $F_H = f_H$. Другими словами, мы считаем, что относительная локальная ошибка аппроксимации τ_h^H просто равна 0 там, где сетка h не определена.

Применяя FAS алгоритм в рамках такой структуры и уравнений, мы получим решение, которое будет удовлетворять уравнениям на подробной сетке, в то же время на внутренних границах (в точках подробной сетки, не совпадающих с узлами более грубой сетки) решение будет автоматически получаться путем интерполяции узлов более грубой сетки.

Решение на подробной сетке: $L_h u_h = -f_h$ на G_h — в точках подробной сетки, $u_h = \hat{I}_h^h u_H$ — в граничных точках.

Причем там, где точки G_h и G_H совпадают, делаем коррекцию решения.

Модифицированная схема полной аппроксимации для двух сеток

1. Предварительное сглаживание

$$\tilde{u}_h^{(n)} = \text{RELAX } V_1^{\nu}[u_h^{(n-1)}, L_h, f_h, u_H^{(n-1)}] .$$

2. Вычисление невязки

$$r_h^{(n)} = -f_h - L_h u_h^{(n)}$$

обладает естественным параллелизмом.

3. Ограничение невязки на более грубую сетку

$$r_H^{(n)} = I_h^H r_h^{(n)} .$$

4. Решение задачи на грубой сетке

$$L_h u_h^{(n)} = -F_H .$$

С учетом того, что там, где есть более подробная сеть

$$F_H = -I_h^H f_h + \tau_h^h ,$$

а там, где нет

$$-F_h = f_H .$$

5. Интерполируем погрешность на более подробную сетку

$$\tilde{\nu}_H^{(n)} = u_H^{(n)} - \hat{I}_h^H \tilde{\nu}_H ,$$
$$\tilde{\nu}_h^{(n)} = I_h^H \tilde{\nu}_H^{(n)} .$$

6. Проводим коррекцию решения

$$\tilde{u}_h^{(n)} = \tilde{u}_h^{(n)} + \tilde{\nu}_h^{(n)} .$$

7. Окончательное сглаживание

$$\tilde{u}_h^{(n)} = \text{RELAX } V^{\nu_2}[\tilde{u}_h^{(n)}, L_h, f_h, u_H^{(n)}]$$

Для идеального решения задачи необходимо правильно находить значения решения на внутренних границах подробной сетки.

Относительная локальная погрешность аппроксимации

С помощью такой погрешности можно получить критерий перехода с более грубой наболее подробную сетку.

Вспомним, что

$$L_H u_H = L_H \hat{I}_h^H u_h + I_h^H(-f_h - L_h u_h) = \\ = L_H \hat{I}_h^H u_h - I_h^H f_h - I_h^H L_h u_h = \tau_h^H - I_h^H f_h = -f_H$$

Эта формула определяет относительную ошибку аппроксимации при переходе с сетки h на сетку H.

Запишем локальную ошибку аппроксимации при переходе с непрерывного решения на дискретное на сетках h и H.

$$\tau_h = L_h(\hat{I}_{\text{cont}}^h u) - I_{\text{cont}}^h(Lu)$$
$$\tau_H = L_H(\hat{I}_{\text{cont}}^H u) - I_{\text{cont}}^H(Lu)$$

Но τ_h одновременно определяет ошибку аппроксимации, пусть она будет порядка $O(h^q)$, τ_H — также ошибка аппроксимации, пусть она будет порядка $O(H^q)$

$$\frac{\tau_H}{\tau_h} = \left(\frac{H}{h}\right)^q \tag{1}$$

Отметим аналогию между τ_H и τ_h^H:

$$\tau_H = L_H(\hat{I}_{\text{const}}^H u) - I_{\text{const}}^H(Lu)$$
$$\tau_h^H = L_H(\hat{I}_h^H u_h) - I_h^H L_h u_h$$

где τ_H — поправка к правой части уравнения $L_H u_H = -f_H$, которая делает аппроксимацию решения на грубой сетке равной точному решению; τ_h^H — поправка к правой части уравнения $L_H u_H = -f_H$, которая делает решение на грубой сетке равным решению на подробной сетке.

Отсюда можно предположить, что асимптотически верно следующее:

$$\tau_H \cong \tau_h^H + I_h^H \tau_h$$

Но, поскольку $\tau_h = \tau_H(h/H)^q$, считаем, что $I_h^H = (h/H)^q$.

Тогда

$$\tau_h^H = \tau_H - \tau_H \left(\frac{h}{H}\right)^q = \frac{\tau_H(H^q - h^q)}{H^q}$$

Пусть $H = 2h$, $q = 2$, тогда $\tau_H = (4/3)\tau_h^H$. τ_H можем легко вычислить с помощью FAS, и, значит, можем получить оценку для τ_H.

Если τ_h^H становится слишком большой, можно осуществить переход на более подробную сетку и вносить коррекцию в решение на более грубой сетке, добавляя τ_h^H.

§ 3.4. Использование многосеточного метода для проведения параллельных вычислений

1. Основные понятия параллелизма
   - модель параллельных вычислений,
   - распараллеливание по пространству,
   - производительность параллельных вычислений.
2. Параллелизация многосеточных методов
   - основной подход,
   - сглаживание с помощью метода Якоби.
3. Теоретический анализ эффективности параллельного многосеточного алгоритма.

Модель параллельных вычислений

Параллелизм вычислений достигается путем одновременного запуска нескольких вычислительных процессов, каждый и которых работает в своем адресном пространстве. Отдельный процесс, проводя операции с данными, действует независимо от других процессов. Процессы производят обмен информацией друг с другом посредством коммуникаций.

Такая модель параллелизма лучше всего соответствует системам с распределенной памятью, в которых каждый процесс запускается на одном их узлов вычислительной системы. Узел состоит из одного или нескольких процессоров плюс локальный банк операционной памяти.

Эту модель можно использовать и для систем с разделяемой памятью, где локальное адресное пространство может соответствовать как частным данным для отдельного потока, так и данным, используемым для обмена или синхронизации.

Распараллеливание по пространству (разделение расчетной области)

Для P процессоров сетку Ω можно разделить на P не перекрывающихся подсеток Ω_q.

Каждый процессор отвечает за вычисления, которые приводят к изменениям приближенного решения на его подсетке. Для проведения этих вычислений процессу могут понадобиться данные от других процессов.

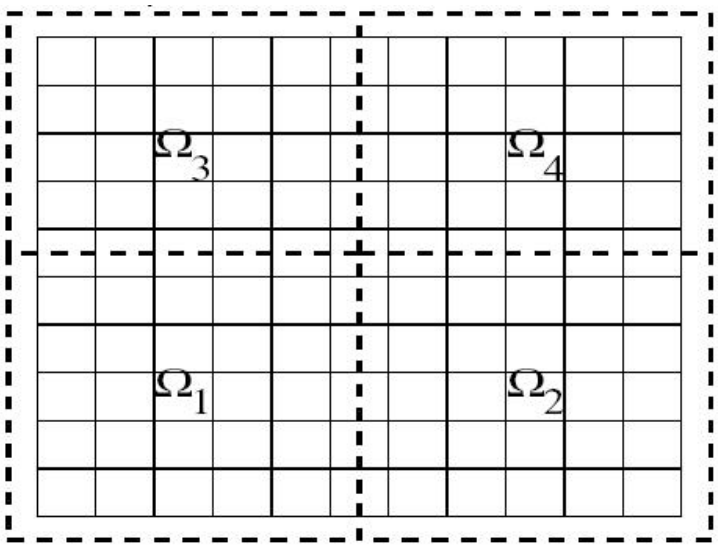


Рис. 3.6. Пример разделения расчетной области

Балансировка нагрузки. Для равномерного распределения работы, было бы желательно, чтобы количество точек подсетки для каждого процессора было бы одинаковым.

Чтобы избежать пересылки множества мелких сообщений между процессами, необходимо, чтобы помимо данных со своей подсетки каждый процесс имел доступ к данным из области перекрытия, которая содержит копии неизвестных от "соседних" процессов.

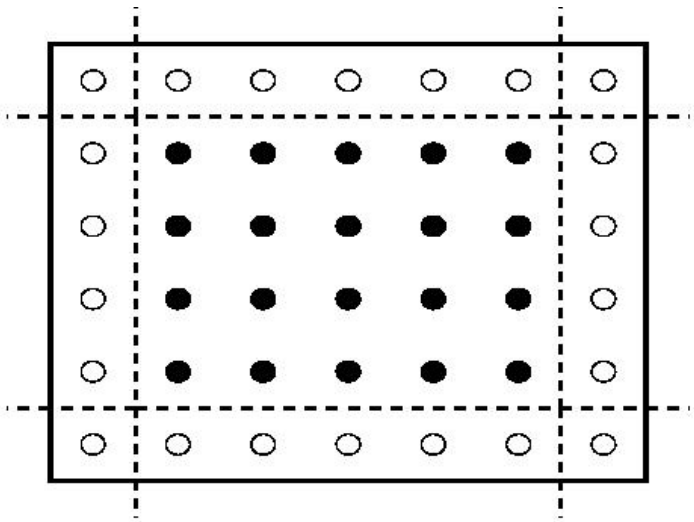


Рис. 3.7. Пример подсетки с выделенной областью перекрытия

Производительность параллельных вычислений

Пусть $T(N,P)$ время, которое требуется для решения задачи с N неизвестными на вычислительной системе, имеющей P процессоров.

Ускорением (*speedup factor*) параллельного алгоритма в P процессорной системе называется величина $S(N,P)$

$$S(N,P) = \frac{T(N,1)}{T(N,P)}$$

где $T(N,1)$ — время выполнения задачи на одном процессоре однопроцессорной системы, $T(N,P)$ — время выполнения задачи на многопроцессорной системе с P процессорами.

Максимальный фактор ускорения $S(N,P) = N$ (линейное ускорение). В общем случае $S(N,P) < N$, так как обычно невозможно обеспечить идеальную балансировку нагрузки процессоров.

Эффективность масштабирования E показывает, во сколько раз больше время выполнения задания одним процессором $T(N,1)$, чем время выполнения того же задания многопроцессорной системой $T(N,P)$, умноженное на число процессоров P:

$$E(N,P) = \frac{T(N,1)}{T(N,P)P}$$

Эффективность характеризует ту часть времени, которую процессоры используют на вычисление.

Наилучший результат достигается при $E(N,P) = 1$.

Код считается масштабируемым, если

$$E(N,P) = (N) \geq 0, \quad \text{при } P \to \infty,$$

где (N) — эффективность масштабирования, при $P \to \infty$.

Параллелизация многосеточных методов

Основной подход — разделение расчетной области

Разделение расчетной области происходит на подробной и грубых сетках, при этом точки i на грубой сетке Ω_{2h} принадлежат подсетке процессора q только в том случае, если они принадлежат также точкам на подробной сетке Ω_h.

При таком подходе не требуется вносить никаких изменений в сам вычислительный алгоритм. Используется стандартный многосеточный цикл: предварительное сглаживание с помощью методов Якоби или Гаусса-Зайделя, коррекция невязки на грубой сетке, интерполяция ошибки на подробную сетку и последующее сглаживание решения. Параллельный код дает те же результаты, что и последовательный. Результат не зависит от количества процессоров.

Сглаживание с помощью метода Якоби

Пусть i изменяется от 1 до n, где n число итераций.

1. каждый процесс выполняет сглаживание внутри подсетки Ω_q;
2. посылает сообщения, чтобы обновить область перекрытия с копиями данных от соседних процессоров.

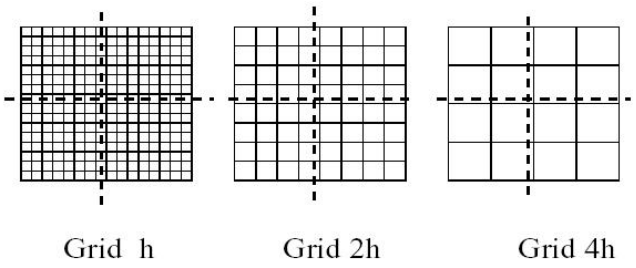


Рис. 3.8. Пример параллелизации по пространству для многосеточного алгоритма

На каждой подсетке Ω_q введем последовательность сеток $\{\Omega_q\}_{k=0}^{k=l}$, $h_l = h_{l-1}/2$ и рассмотрим стандартный многосеточный V цикл для решения уравнения $L_h u_h = -f_h$:

1. Предварительное сглаживание
$$u_h^i = \mathrm{RELAX}^{\nu 1}(u_h^{i-1}, L_h, f_h)\,.$$
2. Вычисление невязки
$$r_h^i = -f_h - L_h u_h^i\,.$$
3. Ограничение невязки
$$r_H^i = I_h^H r_h^i\,.$$
4. Точное решение задачи на грубой сетке
$$L_H e_H^i = r_H\,.$$
5. Перенос поправки
$$e_h^i = I_H^h e_H\,.$$

6. Уточнение решения

$$u_h^* = u_h^i + e_h^i \,.$$

7. Послесглаживание

$$u_h^i = \mathrm{RELAX}^{\nu 2}(u_h^*, L_h, f_h) \,.$$

Анализ эффективности параллельного многосеточного алгоритма

Теоретическая оценка масштабируемости

Полное время выполнения параллельных вычислений является суммой времени, затраченного на вычисления T_{comp}, и времени, затраченного на коммуникации (обмен данных между процессорами) T_{comm}.

Обозначим время запуска (*startup*), иногда называемое временем скрытого состояния сообщений (*message latency*), как α, а время, требуемое для передачи данных между парой процессоров как β.

Тогда

$$T_{\mathrm{comm}} = \alpha + n\beta \,,$$

где n — число пар процессоров, между которыми передаются данные. Широта пропускания канала связи вычисляется как $1/\beta$.

Пусть T_{comp} — время, затрачиваемое для выполнения n операций с плавающей точкой. Тогда $T_{\mathrm{comp}} = fn$, где $1/f$ – число операций с плавающей точкой в секунду, выполняемое системой.

Представим, что двумерная задача с числом точек $(pN)^2$ распределена на p^2 процессорах таким образом, что подсетка каждого процессора имеет размерность N^2. Используем 5-ти точечную аппроксимацию разностного оператора. Рассчитаем время, затрачиваемое на коммуникации и вычисления для этапа сглаживания.

Время, затрачиваемое на сглаживание на самой подробной сетке (нулевой уровень):

$$T_0 \approx 4\alpha + 4N\beta + 5N^2 f$$

(принимаем во внимание 4 области перекрытия для обмена данными с другими процессорами и пятиточечную аппроксимацию разностного оператора).

Время, затрачиваемое на сглаживание на первой грубой сетке (первый уровень):

$$T_1 \approx 4\alpha + 4(N/2)\beta + 5(N/2)^2 f$$

Время, затрачиваемое на сглаживание на уровне l:

$$T_l \approx 4\alpha + 4(N/2^l)\beta + 5(N/2^l)^2 f$$

Для V цикла:

$$\begin{aligned} T_\nu \approx \sum_l 2T_l \approx 2\sum_l [4\alpha + 4(N/2^l)\beta + (N/2^l)^2 f] \approx \\ \approx 8\alpha(1+1+1+\ldots) + 8N\beta(1+1/2+1/4+\ldots) + \\ + 10N^2 f(1+1/4+1/16+\ldots) \approx \\ \approx 8L\alpha + 16N\beta + 40/3N^2 f, \end{aligned}$$

где L – общее число многосеточных уровней: $L \approx \log_2(pN)$.

Масштабируемость.

$$\lim_{p\to\infty} T_\nu(N,1)/T_\nu(PN,P) = O(1/\log(P)).$$

Глава 4

Примеры решения задач

§ 4.1. Решение задачи Дирихле для уравнения Пуассона

Рассмотрим уравнение Дирихле, определенное на единичном квадрате [1].

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -f, \qquad \Omega = (0,1) \times (0,1)$$

$$u = g, \quad (x,y) \in \partial\Omega$$

Зададим сетку.

$$G_h = \{(x_k, y_m) \mid x_k = kh,\ y_m = mh,\ h = 1/N,\ 0 \le k,\ m \le N\}$$

Конечно-разностное представление имеет вид:

$$\frac{u_{k-1,m} - 2u_{k,m} + u_{k+1,m}}{h^2} + \frac{u_{k,m-1} - 2u_{k,m} + u_{k,m+1}}{h^2} = -f_{k,m}$$

Зададим граничные условия.

$$u_{0,j} = g_{0,j}, \quad u_{N,j} = g_{N,j}, \quad \text{для} \quad 0 \le j \le N$$
$$u_{i,0} = g_{i,0}, \quad u_{i,N} = g_{i,N}, \quad \text{для} \quad 1 \le j \le N-1$$

[1] В данном разделе использовались задачи из интернет-пособия [22].

Форма разностного уравнения для метода Якоби:

$$u_{k,m}^{i+1} = \frac{u_{k-1,m}^{i} + u_{k+1,m}^{i} + u_{k,m-1}^{i} + u_{k,m+1}^{i}}{4} + \frac{h^2}{4} f_{k,m}$$
$$0 \le k,\, m \le N-1$$

Форма разностного уравнения для метода Гаусса-Зейделя:

$$u_{k,m}^{i+1} = \frac{u_{k-1,m}^{i+1} + u_{k+1,m}^{i} + u_{k,m-1}^{i+1} + u_{k,m+1}^{i}}{4} + \frac{h^2}{4} f_{k,m}$$
$$0 \le k,\, m \le N-1$$

Тестовая задача

Граничные условия. $g(x,y) = x^4 y^3$.
Правая часть. $f(x,y) = -6x^2 y(x^2 + 2y^2)$.
Указания. Используйте $N = 64$ и для сравнения $N = 128$. Напечатайте норму невязки после каждой итерации. Сравните методы Якоби и Гаусса-Зейделя.

Во второй тестовой задаче задайте нулевые граничные условия и в качестве начального приближения возьмите функции:

$$\varphi^{p,q}(x,y) = \sin p\,\pi x \, \sin p\,\pi y$$

при $p = q = 2$, $p = q = 32$ и $p = q = 63$ для $N = 64$.

Рекомендуемая структура программы

1. Распределить память для искомой функции u и правой части уравнения f.

2. Ввести максимальное число итераций `itmax` и требуемую точность `epsi`.

3. Задать на подробной сетке правую часть f и граничные условия $u = g$. Начальное значение u должно быть положено везде равным 0. Определить начальную невязку для того, чтобы проверять изменение нормы невязки после каждой из `itmax` итераций.

4. Начать выполнять итерационный цикл расчета искомой функции u методом Якоби или Гаусса-Зайделя.

5. Если отношение значения нормы текущей невязки к величине нормы начальной невязки становится меньше `epsi`, предусмотреть выход из цикла и напечатать число итераций, за которые метод сошелся.

6. Условия выхода из цикла из-за расходимости:

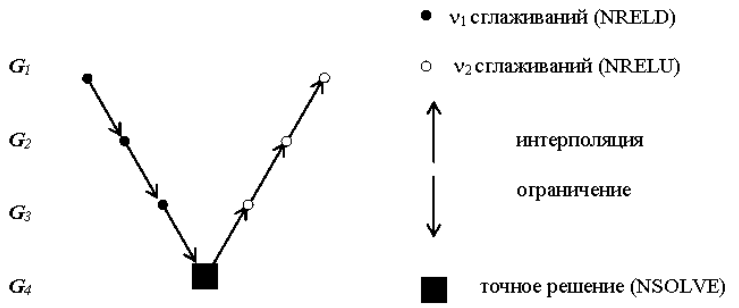
   (a) число итераций превышает значение `itmax`;

   (b) отношение значения нормы текущей невязки к величине нормы начальной невязки превышает `epsi`.

   Следует также предусмотреть возможность печати причины, по которой итерации не сошлись.

§ 4.2. Многосеточный метод для уравнения Пуассона

На основе предыдущего упражнения надо создать многосеточный алгоритм. Для сглаживания используйте метод Гаусса-Зейделя. Иерархия сеток должна быть создана методом стандартного удвоения. Ограничение невязки на грубую сетку производится простым переносом, для обратного переноса надо использовать билинейную интерполяцию. Для решения используйте V-цикл:



Стандартный метод ограничения $G_h \to G_H$– это простой перенос:

$$I_h^H u_h = u_h \mid G_H$$

Билинейная интерполяция при использовании стандартного удвоения определяется формулами:

1. Для $(x_k, y_m) \in G_h \cap G_H$ положим $u_h(x_{2k}, y_{2m}) = u_{2h}(x_k, y_m)$.

2. Для узлов подробной сетки (x_{2k+1}, y_{2k}) и $u_{2h}(x_k, y_m)$, для которых нет узлов грубой сетки — обычная линейная интерполяция:

$$u_h(x_{2k+1}, y_{2m}) = 0.5[u_{2h}(x_k, y_m) + u_{2h}(x_{k+1}, y_m)]$$
$$u_h(x_{2k}, y_{2m+1}) = 0.5[u_{2h}(x_k, y_m) + u_{2h}(x_k, y_{m+1})]$$

3. Для узлов подробной сетки (x_{2k+1}, y_{2m+1});

$$u_h(x_{2k+1}, y_{2m+1}) = 0.25[u_{2h}(x_k, y_m) + u_{2h}(x_{k+1}, y_m) + \\ + u_{2h}(x_k, y_{m+1}) + u_{2h}(x_{k+1}, y_{m+1})]$$

Рекомендуемая структура программы

Задать на подробной сетке правую часть f и граничные условия $u = g$. Начальное значение u должно быть положено везде равным 0. Определить начальную невязку для того, чтобы проверять изменение нормы невязки после каждой из `NCYC` итераций.

Сам цикл состоит из следующих шагов, начинающихся с самой подробной сетки ($l = 1$):

1. Выполнить `NRELD` шагов сглаживания методом Гаусса-Зейделя на сетке G_l.

2. Вычислить невязку на сетке G_l и ограничить ее на следующую грубую сетку G_{l+1}.

3. Задать нулевые начальные и граничные условия на G_{l+1}.

4. Выполнить пункты 1-3 пока не будет достигнута самая грубая сетка ($l = M - 1$).

5. Решить задачу на этой (G_M) сетке используя `NSOLVE` итераций метода Гаусса-Зейделя.

6. Результаты с грубой сетки проинтерполировать на более подробную и прибавить к полученным ранее значениям сеточной функции на этой сетке (G_{l-1}).

7. Выполнить `NRELU` шагов метода Гаусса-Зейделя на сетке (G_{l-1}).

8. Повторить шаги 6 и 7 пока не будет достигнут уровень G_l.

Сравнить норму невязки, полученной после такого цикла с начальным ее значением. Закончить `MG`-итерации, если норма невязки стала составлять `EPSI` от начального значения или счетчик циклов достиг значения `NCYC`.

Литература

[1] Г.П. Астраханцев. Об одном итерационном методе. // ЖВМиМФ. 1971. Т.11, N.2.

[2] Н.В. Бахвалов. О сходимости одного релаксационного метода. // ЖВМиМФ. 1966. Т.6, N.5.

[3] Р.П. Федоренко. Введение в вычислительную физику. - М.: Изд. МФТИ,1994.

[4] Р.П. Федоренко. О скорости сходимости одного итерационного процесса. // ЖВМиМФ. 1964. Т.4, N.3.

[5] Р.П. Федоренко. Релаксационного метод решения разностных эллиптических уравнений. ЖВМиМФ. 1961. Т.1, N.5.

[6] В. Шайдуров. Многосеточные методы конечных элементов. - М.: Наука,1972.

[7] A. Brandt. Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics. - GMD-Studien, N.85, 1984.

[8] A. Brandt. Multi-level Adaptive Computations in Fluid Dynamics. AIAA J. 18 (1980), 1165-1172.

[9] V. Hackbusch. Multigrid method and applications. - Berlin etc.: Springer - Verlag, 1985.

[10] V. Hackbusch , U. Trottenberg. Multigrid Methods. - Lecture Notes in Math. 960, Springer Verlag, 1982.

[11] S. Mijalkovic, W. Joppich. Multigrid method for Process Simulation. - Series "Computational Microelectronics"edited by S. Selberherr, Springer Verlag, New York, Vienna, 1993.

[12] W.J.A. Mol. Numerical Solutions of the Navier-Stokes Equations by Means of a Multigrid Method and Newton-iterations. Proc. Seventh Int. Conf. on Num. Methods in Fluid Dynamics. (W.C. Reynolds, R.W. MacCormack, eds.), Lecture Notes in Physics 141, Springer-Verlag, 1981.

[13] K. Stuben, U. Trottenberg. Multigrid Methods: Fundamental Algorithms, Model Problem Analysis and Applications. - GMD-Studien, N.96, 1984.

[14] MGNet http://www.mgnet.org/

[15] Pieter Wesseling, An Introduction to Multigrid Methods, published by John Wiley & Sons, Chilchester, 1992 http://www.mgnet.org/mgnet-books-wesseling.html

[16] Preprints about Multigrid http://www.mgnet.org/mgnet-papers.html

[17] Multigrid Tutorials http://www.mgnet.org/mgnet-tuts.html

[18] Multigrid Bibliography http://www.mgnet.org/mgnet-bib.html

[19] Мартыненко С.И. Распараллеливание универсальной многосеточной технологии, //Интернет-журнал - Вычислительные методы и программирование, 2003, Том 4, стр. 45 (http://www.srcc.msu.su/num-meth/zhurnal/tom_2003/v4r106.html)

[20] Мартыненко С.И. Универсальная многосеточная технология для численного решения дифференциальных уравнений в частных производных на структурированных сетках // Вычислительные методы и программирование, 2000, Том 1, стр. 83 (http://www.srcc.msu.su/num-meth/zhurnal/tom_2000/art1_7.html)

[21] Мартыненко С.И. Универсальная многосеточная технология для численного решения систем дифференциальных уравнений в частных производных// Вычислительные методы и программирование, 2001, Том 2, стр. 1 (http://www.srcc.msu.su/num-meth/zhurnal/tom_2001/art1_1.html)

[22] Затевахин М.А., Станкова Е.Н. Многосеточные методы. Интернет-пособие (http://www.csa.ru/ stan/multigrid/index.html).

Оглавление