УДК 519.633.9

ПРИМЕНЕНИЕ МНОГОПРОЦЕССОРНЫХ СИСТЕМ ДЛЯ РЕШЕНИЯ ДВУМЕРНЫХ ИНТЕГРАЛЬНЫХ УРАВНЕНИЙ ФРЕДГОЛЬМА I РОДА

М. П. Васильев[1], **А. Г. Ягола**[1]

Рассматриваются особенности численной реализации решения двумерных интегральных уравнений Фредгольма I рода с применением многопроцессорных систем. Для решения этой некорректной задачи применяется алгоритм, основанный на минимизации функционала Тихонова. В качестве метода минимизации рассмотрен метод сопряженных градиентов. Предлагаются схемы распараллеливания задачи, показывается эффективность данного подхода.

1. Введение. Многие физические задачи сводятся к решению двумерных, а иногда и трехмерных уравнений Фредгольма I рода. Например, обратные задачи астрофизики, геофизики, компьютерной томографии, радиофизики приводят к необходимости решать многомерные интегральные уравнения [1, 2]. Еще совсем недавно такие многомерные задачи были неприступны для обычных последовательных компьютеров. Сегодня они решаются с использованием современных многопроцессорных машин [1]. Мы рассмотрим нелинейное двумерное уравнение Фредгольма I рода. Отметим, что этим же методом можно решать и линейные задачи.

2. Постановка задачи и метод решения. Рассмотрим нелинейное двумерное интегральное уравнение Фредгольма I рода без ограничений

$$\int\limits_{L_x}^{R_x}\int\limits_{L_y}^{R_y} K\big(s,t,x,y,z(x,y)\big)\,dx\,dy = u(s,t), \tag{1}$$

где $L_s \leqslant s \leqslant R_s,\ L_t \leqslant t \leqslant R_t$. Обозначим

$$P = \big\{(x,y):\ L_x \leqslant x \leqslant R_x,\ L_y \leqslant y \leqslant R_y\big\},$$
$$Q = \big\{(s,t):\ L_s \leqslant s \leqslant R_s,\ L_t \leqslant t \leqslant R_t\big\}.$$

Считаем, что $z(x,y) \in W_2^2(P),\ u(s,t) \in L_2(Q)$. Ставится задача нахождение функции $z(x,y)$ по заданной $u(s,t)$.

При выписанных условиях задача является некорректной, для ее решения необходимо строить регуляризирующий алгоритм. Воспользуемся алгоритмом, основанным на минимизации функционала Тихонова $M^\alpha[z] = \Phi[\tau] + \alpha\Omega[\tau]$, который в нашем случае принимает вид

$$M^\alpha[z] = \int\limits_{L_s}^{R_s}\int\limits_{L_t}^{R_t} ds\,dt\left\{u(s,t) - \int\limits_{L_x}^{R_x}\int\limits_{L_y}^{R_y} K\big(s,t,x,y,z(x,y)\big)\,dx\,dy\right\}^2 + \alpha\Omega[z],$$

где $\Omega[z]$ — стабилизатор: $\Omega[z] = \int\limits_{L_x}^{R_x}\int\limits_{L_y}^{R_y}\left[\left(\dfrac{\partial^2 z}{\partial x^2}\right)^2 + \left(\dfrac{\partial^2 z}{\partial y^2}\right)^2\right] dx\,dy.$

Для выбора параметра регуляризации можно использовать алгоритм конечномерного обобщенного принципа невязки [3]. В качестве метода минимизации функционала Тихонова применяется метод сопряженных градиентов.

3. Структура алгоритма и возможности для распараллеливания. Для численного решения задачи переходим к конечномерным пространствам. Введем сетки по x, y, s, t с шагами h_x, h_y, h_s, h_t и

[1] Московский государственный университет им. М. В. Ломоносова, физический факультет, Воробьевы горы, 119992, Москва; e-mail: ecmu@mail.ru; yagola@inverse.phys.msu.su; ill-posed@mail.ru

числом узлов N_x, N_y, N_s, N_t соответственно:

$$x_i = L_x + (i-1)h_x, \quad i = \overline{1, N_x}, \quad h_x = \frac{R_x - L_x}{N_x - 1},$$
$$y_j = L_y + (j-1)h_y, \quad j = \overline{1, N_y}, \quad h_y = \frac{R_y - L_y}{N_y - 1},$$
$$s_k = L_s + (k-1)h_s, \quad k = \overline{1, N_s}, \quad h_s = \frac{R_s - L_s}{N_s - 1},$$
$$t_l = L_t + (l-1)h_t, \quad l = \overline{1, N_t}, \quad h_t = \frac{R_t - L_t}{N_t - 1}.$$

При решении задачи минимизации методом сопряженных градиентов необходимо вычислять значения функционала $\widehat{M}^{\alpha}[z]$ и его градиента $\operatorname{grad} \widehat{M}^{\alpha}[z]$. Конечно-разностная аппроксимация функционала Тихонова выглядит следующим образом:

$$\widehat{M}^{\alpha}[\widehat{z}] = \widehat{\Phi}[\widehat{z}] + \alpha\widehat{\Omega}[\widehat{z}],$$

где

$$\widehat{\Phi}[\widehat{z}] = \sum_{k=1}^{N_s}\sum_{l=1}^{N_t} h_s h_t \left[\sum_{i=1}^{N_x}\sum_{j=1}^{N_y} h_x h_y K_{klij}(z_{ij}) - u_{kl}\right]^2, \tag{2}$$

$\widehat{\Omega}[\widehat{z}]$ — конечно-разностная аппроксимация стабилизатора, $z_{ij} = z(x_i, y_j)$, $K_{klij}(z_{ij}) = K(s_k, t_l, x_i, y_j, z_{ij})$.

Рассмотрим численный пример для линейного случая. Возьмем отрезок $[0, 1]$ по x и y и отрезок $[0, 2]$ по s и t. Пусть ядро оператора $K = e^{-20(s-x-0.5)^2-20(t-y-0.5)^2} z(x, y)$. Возьмем сетку $N_x = Ny = N_s = N_t = 100$ и рассмотрим первую итерацию метода сопряженных градиентов, посчитанную на последовательном компьютере. Время вычисления функционала — 52.5 сек, градиента — 225.0 сек. На первой итерации функционал считается шесть раз, градиент — один раз. Таким образом, полное время вычисления первой итерации составляет $6 \cdot 52.5 + 225.0 = 315 + 225 = 540$ сек, или 9 мин. Далее мы покажем, насколько сокращается время вычисления функционала и градиента при распараллеливании.

Из формулы (2) видно, что невязка состоит из несвязных между собой групп слагаемых (по k и l). Аналогичная ситуация имеет место и для градиента функционала Тихонова:

$$(\operatorname{grad} \widehat{M}^{\alpha})_{ij} = \frac{\partial \widehat{M}^{\alpha}(z_{ij})}{\partial z_{ij}} =$$
$$= 2h_x h_y \sum_{k=1}^{N_s}\sum_{l=1}^{N_t} h_s h_t \frac{\partial K_{klij}(z_{ij})}{\partial z_{ij}} \left[\sum_{m=1}^{N_x}\sum_{n=1}^{N_y} h_x h_y K_{klmn}(z_{mn}) - u_{kl}\right] + \frac{\partial\widehat{\Omega}(z_{ij})}{\partial z_{ij}}.$$

Это дает возможность применения многопроцессорной системы. Задачу можно распараллелить, т.е. переписать программу таким образом, чтобы независимые части программы выполнялись на разных процессорах.

4. Распараллеливание задачи минимизации. Исходная задача решается с помощью N взаимодействующих параллельных процессов с номерами $0, 1, \ldots, N-1$. Каждый процесс выполняется на отдельном процессоре. Нулевой процесс (процесс с номером 0) выполняет все нераспараллеливаемые действия. Распараллеливание происходит только при минимизации функционала $\widehat{M}^{\alpha}[\widehat{z}]$: до и после минимизации ненулевые процессы не задействованы. Нулевой процесс рассматривается только как процесс, передающий и получающий информацию от других процессов.

Распараллеливание задачи вычисления функционала Тихонова поясним на примере, в котором $N_x = N_y = N_s = N_t = 2$. Шаги h_s, h_t, h_x, h_y для наглядности опущены. Так как время вычисления стабилизатора пренебрежимо мало по сравнению с временем вычисления невязки, то распараллеливаем только

невязку:

$$\widehat{\Phi}[\widehat{z}] = \sum_{k=1}^{2}\sum_{l=1}^{2}\left[\sum_{i=1}^{2}\sum_{j=1}^{2} K_{klij}(z_{ij}) - u_{kl}\right]^2 = \tag{3}$$
$$= \left[K_{1111}(z_{11}) + K_{1112}(z_{12}) + K_{1121}(z_{21}) + K_{1122}(z_{22}) - u_{11}\right]^2 +$$
$$+ \left[K_{1211}(z_{11}) + K_{1212}(z_{12}) + K_{1221}(z_{21}) + K_{1222}(z_{22}) - u_{12}\right]^2 +$$
$$+ \left[K_{2111}(z_{11}) + K_{2112}(z_{12}) + K_{2121}(z_{21}) + K_{2122}(z_{22}) - u_{21}\right]^2 +$$
$$+ \left[K_{2211}(z_{11}) + K_{2212}(z_{12}) + K_{2221}(z_{21}) + K_{2222}(z_{22}) - u_{22}\right]^2.$$

Вычисление $\widehat{M}^{\alpha}$ происходит по следующей схеме. Каждый процесс (кроме нулевого) считает свой квадрат выражения (3) и посылает его нулевому процессу, который суммирует все значения в невязку $\widehat{\Phi}$, после чего считает стабилизатор $\widehat{\Omega}[\widehat{z}]$.

Алгоритм распараллеливания задачи вычисления градиента $\operatorname{grad}\widehat{M}^{\alpha}[\widehat{z}]$ следующий. Каждый ненулевой процесс вычисляет p-ый элемент вектора градиента и передает нулевому процессу элемент $\operatorname{grad}_p \widehat{M}^{\alpha}[\widehat{z}]$ вместе с номером p. Нулевой процесс суммирует все компоненты градиента в единый вектор (двумерный градиент с индексами i, j свернут в одномерный с индексом p):

$$\operatorname{grad}\widehat{M}^{\alpha}[\widehat{z}] = \sum_{p}\operatorname{grad}_p\widehat{M}^{\alpha}[\widehat{z}] =$$
$$= \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\left\{2h_xh_y\sum_{k=1}^{N_s}\sum_{l=1}^{N_t}h_sh_t\frac{\partial K_{klij}(z_{ij})}{\partial z_{ij}}\left[\sum_{m=1}^{N_x}\sum_{n=1}^{N_y}h_xh_yK_{klmn}(z_{mn}) - u_{kl}\right] + \frac{\partial\widehat{\Omega}(z_{ij})}{\partial z_{ij}}\right\}.$$

5. Функционал Тихонова. При решении модельных задач использовался вычислительный кластер SCI (Pentium III/500 МГц) Научно-исследовательского вычислительного центра МГУ [4]. Последовательные вычисления проводились на одном процессоре того же кластера. При составлении параллельных версий программ использовалась библиотека MPI (Message Passing Interface) [4]. В качестве языка программирования был выбран Fortran 90.

Рассмотрим задачу (1) при следующих параметрах: $L_s = L_t = L_x = L_y = 0$, $R_s = R_t = 2$, $R_x = R_y = 1$, $K\left(s,t,x,y,z(x,y)\right) = e^{-20(s-x-0.5)^2-20(t-y-0.5)^2}z(x,y)$.

Для этого примера посмотрим, насколько эффективным является использование параллельных процессов для вычисления функционала Тихонова $\widehat{M}^{\alpha}[\widehat{z}]$. Результаты, полученные в режиме последовательных и параллельных вычислений, приведены для сравнения в табл. 1.

Таблица 1

Вычисление функционала

N_x	$N=1$	$N=2$	$N=3$	$N=4$	$N=5$	$N=6$	$N=8$	$N=10$	$N=16$
40	1.25	1.26	0.65	0.44	0.33	0.26	0.19	0.15	0.10
50	3.0	3.0	1.48	1.16					
60	6.2	6.2	3.07	2.12					
70	11.5	11.5	5.84	3.87		2.58			
80	19.5								
90	34.4								
100	52.5						7.56		3.54

Размерность сетки по всем направлениям одинакова: $N_x = Ny = N_s = N_t$. Второй столбец "$N = 1$" показывает время вычисления функционала на обычной (последовательной) машине. Столбцы, начиная с третьего ($N = 2$), содержат информацию о времени вычисления функционала на кластере. Единица измерения времени — секунда. Напомним, что реально в параллельных вычислениях используются $N - 1$ процессов, поскольку нулевой процесс только собирает значения.

Из табл. 1 видно, что эффективность распараллеливания близка к 100 % (почти линейна). Аналитическое выражение времени вычисления функционала на q процессорах дается формулой $t_q = \dfrac{t_2}{q-1}$, где t_2 — время вычисления функционала на двух процессорах. Отсюда можно сделать вывод, что структура функционала Тихонова идеальна для вычисления на параллельной машине.

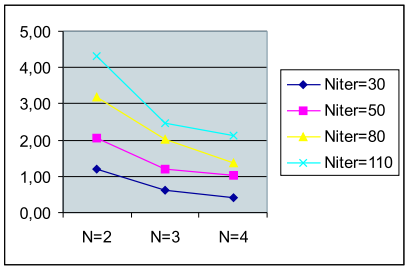
Аналогичная ситуация имеет место и для градиента функционала. Таким образом, видна эффективность применения многопроцессорных систем для вычисления функционала Тихонова и его градиента.

6. Метод сопряженных градиентов. В условиях примера из п. 5 рассмотрим полный алгоритм метода сопряженных градиентов для минимизации функционала Тихонова без ограничений. Для сравнения с примером из п. 3 посчитаем первую итерацию метода на четырех процессорах ($N = 4$). Напомним, что размерность сеток равна 100. Время счета функционала — 17.4 сек, градиента — 75 сек. Следовательно, время, затраченное на первую итерацию, составляет $6 \cdot 17.4 + 75 = 179.4$ сек, или 3 мин. Вспомним, что на последовательной машине время счета равнялось 9 мин. Таким образом, на одной итерации три процессора считают в три раза быстрее.

Рассмотрим несколько итераций (см. табл. 2 и рисунок). Здесь размерность сетки $N_x = Ny = N_s = N_t$ и N_{iter} — число итераций метода. Времена вычисления представлены в минутах.

Таблица 2
Метод сопряженных градиентов

N_x	N_{iter}	$N = 2$	$N = 3$	$N = 4$
30	30	1.20	0.60	0.41
30	50	2.05	1.19	1.03
30	80	3.20	2.03	1.36
30	110	4.33	2.46	2.11
40	50	6.26	3.53	3.02
40	80	10.07	6.07	4.47



Метод сопряженных градиентов, $N_x = 30$

Очевидно ухудшение распараллеливания. Сказываются следующие факторы.

1) Кроме вычисления функционала Тихонова и его градиента все остальные вычисления производятся последовательно. При увеличении количества итераций доля последовательных вычислений возрастает. Однако при больших размерностях сетки время работы последовательного кода много меньше, чем время распараллеленного кода.

2) Даже в распараллеленных блоках программы сказывается неравномерность загрузки процессоров. Из-за того, что числа N_xN_y и N_sN_t в общем случае не являются кратными числа $q - 1$, каждый процесс должен выполнить разный объем вычислений, при этом чем больше итераций, тем больше рассогласование в загруженности процессоров.

7. Выводы. Применение многопроцессорных компьютеров при решении двумерных уравнений Фредгольма I рода позволяет существенно сократить время работы программы. Алгоритм метода сопряженных градиентов для минимизации функционала Тихонова является пригодным для распараллеливания, что обеспечивает эффективность данного подхода.

Авторы благодарят РФФИ за частичное финансирование данной работы (грант № 02–01–00014).

СПИСОК ЛИТЕРАТУРЫ

1. *Ягола А.Г., Титаренко В.Н., Васильев М.П., Шимановская Е.В.* Особенности решения задач картирования распределения химических элементов по поверхностям звезд как некорректных задач с использованием многопроцессорных систем // Вычислительные методы и программирование. 2002. **3**, № 1. 5–17.
2. *Гончарский А.В., Черепащук А.М., Ягола А.Г.* Некорректные задачи астрофизики. М.: Наука, 1985.
3. *Тихонов А.Н., Леонов А.С., Ягола А.Г.* Нелинейные некорректные задачи. М.: Наука, 1995.
4. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. С.-П.: БХВ-Петербург, 2002.

Поступила в редакцию
07.10.2003