УДК 681.326

О РЕШЕНИИ РАЗРЕЖЕННЫХ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ ПРИ ПОМОЩИ СТАБИЛИЗИРОВАННОГО МЕТОДА БИСОПРЯЖЕННЫХ ГРАДИЕНТОВ

ЧАДОВ С.Н., асп.

Рассматривается один из итеративных методов решения систем линейных уравнений. Приводятся описание метода, некоторые способы его распараллеливания, а также преимущества данного метода перед более распространенными.

Ключевые слова: система линейных уравнений, системы большой размерности, матрица предобуславливания, параллельная реализация, метод бисопряженных градиентов.

THE SOLUTION OF LOW-DENSITY LINEAR EQUATION SYSTEMS WITH THE HELP OF STABILIZED BI-CONJUGATE GRADIENT METHOD

CHADOV S.N., postgraduate.

The article concerns one of the iterative methods of linear equation systems solution. The article contains the description of the method, some ways of its paralleling and the advantages of this method against more common ones.

Key words: linear equation system, high dimensionality systems, fore-stipulation matrix, parallel realization, bi-conjugate gradient method.

Введение. Решение систем линейных уравнений является задачей, имеющей применение как в чистом виде, так и в качестве примитива в более сложных численных алгоритмах. В качестве примера можно привести решение систем дифференциальных уравнений с использованием неявных схем [1, 2].

Существует множество методов решения систем линейных уравнений, каждый из которых имеет свои достоинства и недостатки. Наиболее часто на практике используется один из вариантов метода LU-разложения (см., например, [1]). Однако в реальных задачах при моделировании сложных систем (например, в энергетике) приходится иметь дело с системами очень большой размерности (десятки и сотни тысяч элементов и более), но при этом часто эти системы являются достаточно разреженными. Для таких систем распространенные методы не всегда являются самыми оптимальными, поэтому часто используют итерационные методы [3]. Рассмотрим один из таких методов, известный под названием BiCG-Stab [3]. Этот метод был предложен H. van der Vorst в 1992 г. и является модификацией метода бисопряженных градиентов, обеспечивающей лучшую сходимость [3, 4] .

Поскольку итерационные методы часто находят применение именно для решения задач большой размерности, важной задачей является их параллельная реализация.

Постановка задачи и описание метода. Итак, пусть требуется решить систему уравнений

$$Ax = b, \qquad (1)$$

где x – искомый вектор; A – некоторая матрица.

Общая идея метода бисопряженных градиентов для решения линейной системы уравнений (1) заключается в итеративной генерации пары остатков $\hat{r}$ и r, вычисляемых как

$$r^{(i)} = r^{(i-1)} - \alpha_i A p^{(i)}$$

и $\hat{r}^{(i)} = \hat{r}^{(i-1)} - \alpha_i A^T \hat{p}^{(i)}$,

где p – векторы направлений: $p^{(i)} = r^{(i-1)} + \beta_{i-1} p^{(i-1)}$, $\hat{p}^{(i)} = \hat{r}^{(i-1)} + \beta_{i-1} \hat{p}^{(i-1)}$; α и β выбираются таким образом, чтобы выполнялись отношения ортогональности $(\hat{r}^{(i)})^T r^{(j)} = (p^{(i)})^T A p^{(j)} = 0$ для $i \neq j$:

$$\alpha_i = \frac{\hat{r}^{(i-1)T} r^{(i-1)}}{\hat{p}^{(i)T} A p^{(i)}}, \quad \beta = \frac{\hat{r}^{(i)T} r^{(i)}}{\hat{r}^{(i-1)T} r^{(i-1)}}.$$

Можно заметить, что r(j) и $\hat{r}^{(j)}$ выразимо в виде

$$r^{(j)} = P^{(j)}(A) r^{(0)},$$

$$r^{(j)} = P^{(j)}(A^T) \hat{r}^{(0)}.$$

Из отношения ортогональности получаем

$$r^{(j)} \hat{r}^{(i)} = (P^{(j)}(A) r^{(0)})(P^{(i)}(A^T) \hat{r}^{(0)}) =$$

$$= (P^{(j)}(A) P^{(i)}(A\ ) r^{(0)}) \hat{r}^{(0)} = 0,$$

где (·) – оператор скалярного произведения; P – некоторый полином.

Таким образом, используя последнее выражение, возможно полностью избежать транспонирования матрицы A, а также вычисления вектора $\hat{r}^{(j)}$.

H. van der Vorst в 1992 г. развил эту идею, предложив представить r(j) в виде

$$r^{(j)} = Q^{(j)}(A) P^{(j)}(A) r^{(0)},$$

где $Q^{(j)}(x) = (1 - \omega_1 x)(1 - \omega_2 x)...(1 - \omega_j x)$, а коэффициенты ω выбираются так, чтобы минимизировать r(j) [4]

Такой метод имеет преимущество перед классическим методом бисопряженных градиентов в скорости сходимости [3].

Следует отметить использование матрицы M в качестве матрицы предобусловливания (preconditioner). Использование предобусловливания во многих случаях позволяет на порядки сократить время вычислений. Причина заключается в том, что скорость сходимости итеративных методов зависит от спектральных характеристик матрицы коэффициентов. Соответственно, можно попытаться так преобразовать систему, чтобы она имела те же решения, что и исходная, но при этом имела бы лучшую (относительно метода решения) спектральную характеристику. Выбор матрицы предобусловливания часто является достаточно сложной творческой задачей, поскольку на настоящий момент неизвестно удобных на практике формальных алгоритмов для такого выбора (хотя существует несколько наиболее распространенных вариантов [3]). Одним из наиболее распространенных является реализация предобусловливателя на основе алгоритма неполного LU-разложения [3, 5]

Вычислить $r^{(0)} = b - Ax^{(0)}$ для некоторого начального значения $x^{(0)}$

$\hat{r} = r^{(0)}$

для i = 1,2,…

$\rho_{i-1} = \hat{r}^T r^{(i-1)}$

если $\rho_{i-1} == 0$ -> вернуть ошибку

если i==1, $p^{(i)} = r^{(i-1)}$

иначе

$\beta_{i-1} = (\rho_{i-1} / \rho_{i-2})(\alpha_{i-1} / \omega_{i-1})$

$p^{(i)} = r^{(i-1)} + \beta_{i-1}(p^{(i-1)} - \omega_{i-1} v^{(i-1)})$

Решить $Mp' = p^{(i)}$

$v^{(i)} = Ap'$

$\alpha_i = \rho_{i-1} / \hat{r}^T v^{(i)}$

$s = r^{(i-1)} - \alpha_i v^{(i)}$

если (норма s достаточно мала), $x^i = x^{i-1} + \alpha_i p'$ и остановиться

Решить $Ms' = s$

$t = As'$

$\omega_i = t^T s / t^T t$

$x^{(i)} = x^{(i-1)} + \alpha_i p + \omega_i s'$

$r^{(i)} = s - \omega_i t$

Проверить на сходимость и, если она не достигнута и $\omega_i \neq 0$, выполнить следующую итерацию

Алгоритм метода Bicg-STAB

Предобусловливание может не использоваться вообще, тогда в алгоритме следует заменить шаги «Решить $Mp' = p^{(i)}$» и «Решить $Ms' = s$» на шаги «p'=p» и «s'=s», соответственно.

Параллельная реализация. Возможно, самым простым способом распараллеливания данного алгоритма является выполнение вычислений в режиме SIMD, то есть запуск одной и той же программы в разных потоках исполнения, каждый из которых производит операции над своим участком данных. Преимущества такого подхода следующие:

1) не требует серьезной модификации существующего программного кода, при условии, что этот код грамотно спроектирован;

2) реализация обладает высокой степенью переносимости между программными и аппаратными платформами, не теряется возможность работы в однопоточном режиме;

3) масштабируется практически на любое количество потоков.

Описанный подход, ввиду своей простоты, находит достаточно широкое применение (см., например, [2]).

Перспективной представляется реализация такого алгоритма на машине с общей памятью и использованием векторных команд процессора (например, современные процессоры персональных компьютеров поддерживают наборы SIMD-инструкций SSE, SSE2 и так далее, позволяющие выполнять две операции с числами с двойной точностью или 4 команды с числами с одинарной точностью одной командой).

Данный алгоритм можно улучшить, уменьшив количество необходимых операций синхронизации, преобразовав соответствующим образом уравнения оригинального алгоритма. Например, в [6] предлагается следующее преобразование:

$$v^{(i)} = Ap = A(r^{(i-1)} + \beta_{i-1}(p^{(i-1)} - \omega_{i-1} v^{(i-1)})) =$$

$$= u^{(i-1)} + \beta_{i-1} v^{(i-1)} - \beta_{i-1}\omega_{(i-1)} q^{(i-1)},$$

где $u^{(i-1)} = Ar^{(i-1)}$; $q^{(i-1)} = Au^{(i-1)}$; для остальных используемых векторов – аналогично.

Такой подход может облегчить написание эффективной параллельной версии алгоритма, однако в приведенном виде он не пригоден для решения с использованием предобусловливателя.

Отдельную задачу представляет собой параллельная реализация предобусловливателя. Эксперименты показывают, что на решение предобусловливающей системы уравнений уходит не менее 30 % времени вычислений. Разработать какой-либо универсальный алгоритм распараллеливания в данном случае невозможно, поскольку шаг вычисления предобусловливателя может выполняться совершенно произвольным образом, зависящим от конкретной решаемой задачи. Однако приведем вариант параллельной реализации наиболее часто используемого метода неполного LU-разложения.

Для параллельной реализации разобьем исходную матрицу на перекрывающиеся блоки, сопоставив один блок каждому потоку исполнения [7]. Вычисления в каждом блоке выполняются без каких-либо межпроцессных взаимодействий и синхронизации, что способствует масштабируемости такого решения.

Заключение

Рассмотренный метод решения систем линейных уравнений, а именно, стабилизированный метод бисопряженных градиентов, имеет несомненные преимущества перед более распространенными (в первую очередь, LU-разложением) на определенном классе задач (задачи большой размерности с разреженными матрицами коэффициентов). Использование предобусловливания позволяет еще более ускорить решение.

Рассмотренный метод поддается распараллеливанию и соответственно может быть использован в пакетах параллельных программ. Возможным слабым местом в реализации метода может являть-

ся реализация предобусловливателя, которая также должна быть параллельной. Однако наиболее распространенный вариант – предобусловливание при помощи неполной LU-факторизации – поддается геометрическому распараллеливанию.

Данный вариант метода BICG-Stab реализован как часть комплекса программ для решения систем жестких дифференциальных уравнений (применяется для вычисления корректора неявной схемы интегрирования) как в последовательном, так и в параллельном вариантах. Параллельные варианты были разработаны для многопроцессорных систем с общей памятью (с использованием OpenMP) и на основе SIMD-инструкций процессоров x86.

Список литературы

1.**William H. Press ... [et al].** Numerical recipes in C: the art of scientific computing . 2nd ed ISBN 0-521-43108-5.

2.**Hindmarsh A.C., Serban R.** User Documentation for cvode v2.4.0. Center for Applied Scientific Computing Lawrence Livermore National Laboratory, 2006.

3.**Barrett R., Berry M., Chan T.F., et al.** Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition.

4.**Van der Vorst H.** Linear System Solvers: Sparse Iterative methods.

5.**Chan T.F., Van der Vorst H.** Approximate and incomplete factorizations.

6.**Ouarraui C., Kaeli D.** Developing object-oriented parallel iterative methods.Department of Electrical and Computer Engineering,Northeastern University, Boston, MA 02115.

7.**Gonzalez P., Cabalero J., Pena T.** Parallel Incomplete LU factorization as a Preconditioner for Krylov Subspace methods.

Чадов Сергей Николаевич,
ГОУВПО «Ивановский государственный энергетический университет имени В.И. Ленина»,
аспирант кафедры высокопроизводительных вычислительных систем,
e-mail: sergei.chadov@gmail.com