反復法における直接解法の利用

寒川 光

日本アイ・ビー・エム（株）

FDM で作成された行列を係数とする連立方程式を CG 法で解くとき、適当な節点の順序付けに基づくシュール補元を反復的に用いるほうが高速化に都合がよい場合がある。特に 2 次元直交格子に 5 点差分を用いて作られる対称行列には有効とする報告があるが、3 次元 7 点差分でも有効であることを確かめた。対称行列の場合は、前処理による効果を（計算時間の観点では）上回るものも多いので、並列化には有望である。

Using Direct Elimination in Iterative Solvers

Hikaru Samukawa

IBM Japan, Ltd.

Using Schur complement based on appropriate grid sequences in the CG methods for coefficient matrices generated in the FDM is much better than using original matrices. The effect of this approach has been reported in the five points FDM in the two dimensional problem. We make it sure that the approach is quit effective in the three dimensional problem as well. In case of symmetric matrices, since the approach is faster than that of using preconditioning, it is promising in the parallelization.

1 はじめに

共役勾配法（CG 法）は、ベクトル型スーパーコンピュータに適した連立 1 次方程式の解法として発達したが、これはこの解法が記憶域に対する要求が少ないということに一因があった。最近の HPC 環境では、ベクトル型から RISC プロセッサをベースとしたシステム移行しつつある。この環境（特に分散メモリ型）では、利用可能な主記憶装置の容量に余裕がでてきた。この余裕を CG 法に活かす方法として、適当な節点の順序付けを行った後、行列の一部を直接消去して、縮小された系に対して CG 法反復を行う方法がある。

例えば 3 重対角行列 A を係数行列とする方程

式 \( Ax = b \) は、レッドブラック (RB) 順序により分割並べ換えを行い、

\[
\begin{pmatrix}
A_{rr} & A_{rb} \\
A_{br} & A_{bb}
\end{pmatrix}
\begin{pmatrix}
x_r \\
x_b
\end{pmatrix} =
\begin{pmatrix}
b_r \\
b_b
\end{pmatrix}
\]

Red に対応する未知数を消去して、方程式 \( Sx_b = b_b \) に書き換える。

\[
(A_{bb} - A_{br}A_{rr}^{-1}A_{rb})x_b = b_b - A_{br}A_{rr}^{-1}b_r
\]

シュール補元 \( S \) のシュール補元 \( S \)（3 重対角行列）が得られ、CG
法で扱う問題の規模を半分にすることができる1.

3 重対角行列は問題の領域が 1 次元のときに現れるが、2 次元領域を 5 点有限差分法 (FDM) で離散化して得られる行列 A にもこの方法は有効である [1]。2 次元領域を nx × ny の格子にメッシュ分割した場合、A は 5 本対角行列になり、これを RB 順序により分割してシュール補元を求めると、nx が奇数の場合は 9 本対角行列に、偶数の場合は 11 本対角行列になる。

このような形に縮小されるのは、直交格子に中心差分または風上 1 次の差分スキームを用いた場合に限られる。3 次元 7 点差分の問題でも、直交格子にこれらの差分スキームを組み合わせた場合は同様の形に縮小できるが、シュール補元を陰に求めると計算量が増加すると考えられている [2]。これはシュール補元を数本のベクトルの形のデータ構造で保持しようとすると、(1 次元は 3 本 → 3 本に収まると、2 次元では 5 本 → 9 本または 11 本に増加し、3 次元ではベクトル本数の増加がかなり多くなると考えられているからではないだろうか。

しかし 3 次元領域を nx × ny × nz に分割した場合、nx と ny が奇数の場合は、シュール補元は 19 本対角行列に収まる。未知数が半分に減っているので、元の 7 本対角行列と比較すると 9.5 本分である。行列ベクトル積の計算量で比較すると、30％強の増加にすぎない。シュール補元 S の条件数 κS は O(h⁻¹) で、A の条件数 κA の O(h⁻²) に比較すると大きな改善である (h はメッシュサイズ)。CG 法の反復回数は O(√κ) と考えられるので、連立方程式を解くための計算量は減少が期待できる [2]。

また nx が奇数の場合でも、それに加えて、それに基準行列を挿入すれば、奇数の場合のプログラムで計算可能である。

シュール補元も陰に求めず、CG 法反復の中で運行的に計算する方法も可能である。こうすれば、行列ベクトル積の計算量は増加しないが、前処理を必要としないという点がある。

2 RB 順序とシュール補元

直交格子での RB 順序に基づくシュール補元についてまとめる。

2.1 2 次元 5 点差分

元の自然な番号付けを、“左から右へ、次に “下から上へ” と x 方向を先に回すこともする。x 方向の節点数 nx が奇数の場合、ひとつの線上的節点は Red で始まり Red で終わる、次の線上の節点が Black から始まる。したがって元の節点番号と RB 順序での節点番号との関係は “元の番号が奇数なら Red、偶数なら Black” の関係がある。しかし nx が偶数の場合は、ひとつの線の最後及び次の線の最初が同色になり、この関係は成立しない。

図 1 に奇数の場合の元の節点 9 (RB 順序で 5r) が、この場合のスケミで接続する点を示した。図 2 は自然な順序に対応する行列である。これはブロック 3 重対角行列で、各ブロックは図 1 のひとつの線に対応している。また、対角小行列は 3 重対角行列、対角小行列は対角行列なので、ブロック
クを意識することなく、これを本の1次元配列に格納するデータ構造が一般的に用いられている。非対角元は対角元から、下三角は\(-nx\)と\(-1\)、上三角は\(1\)と\(nx\)要素ストリング。本論文ではこの行列を "5本対角行列\((-nx, -1, 0, 1, nx)\)"と表すことにする。この場合は非ゼロ要素の位置が対称なので、"5本対角行列\((sym, 0, 1, nx)\)"と略す。

図1に示した節点\(S\)の接続関係は、図2の行列の中では黒印の連続（非対角要素）で表される。黒印は行列がRed列位置がBlackの要素である。ひとつの手前の節点\(A_{rb}\)の接続関係は黒印で表され、図1に同色の接続関係が存在しないということが、図2の行列では、手列が同色の位置には、対角元以外に非ゼロ要素が存在しないということにつながる。これがRB順序に並び替えると、\(A_{rb}\)は対角行列になることに対応している。

図3に分割された行列のうちのひとつである\(A_{rb}\)を示す。図2の節点\(S\)の接続関係は図3の黒印に移る。また節点\(A_{rb}\)の接続関係は\(A_{rb}\)に回る。

\(x\)方向の節点数\(nx\)が奇数の場合は図の左のように、\(A_{rb}\)の対角行列は\(nx\)行があり、\(nx\)列が0と\(nx\)列が01と交互に並べられ、非ゼロ要素は2本の直線上に並ぶので、\(A_{rb}\)は4本対角行列\((-nx, -1, 0, nx)\)になる。ただし\(nx = [nx/2], nx = [(nx + 1)/2]\)とする。この4本はRed節点の接続を表し、行列の左側から、\(D\)（Down）、\(L\)（Left）、\(R\)（Right）、\(U\)（Up）側の点との連続関係である。

しかし、\(nx\)が偶数の場合は、\(A_{rb}\)は図の右のように、5本対角行列\((sym, 0, 1, nx)\)になる。これは\(A_{rb}\)についても言えることである。この結果シェール補元は、\(nx\)が奇数の場合は9本対角行列に収まるが、偶数の場合は11本増える。

シェール補元\(S\)をパラメータで求めるには、いったん自然な順序で係数行列を生成した後、上記の手続きで行列の分割並べ替え、演算\(A_{rb} + A_{rb}^{-1}A_{rb}\)を行えばよいので簡単であるが、記憶域は増加する。

図4に5点差分の接続関係と、そのシェール補元の接続関係を比較した。Redが消去されると、

接続関係は、消去された節点が直接接続していた点に延長される。中央の点から見て、例えば右上への点は"R-U"と"U-R"の2つの経路でつながるが、これは\(A_{br}\)と\(A_{rb}\)の積を考えると、\(A_{br}\)の\(R\)項と\(A_{rb}\)の\(U\)項の積と、\(A_{br}\)の\(U\)項と\(A_{rb}\)の\(R\)項との積が、シェール補元では同じ要素に重ね合わせることに対応している。また対角項は"U-D", "D-U", "R-L", "L-R"の4項の重ね合わせになる。このように4×4の16項の積が、対角項で4つ、余りの4点で2つの重なりがあり、7つ減って9本対角行列になる。要するに、シェール補元を"Redを消去された5点差分スキーム"と解釈できる。このことは、行列生成を1節点ごとに行うのではなく、上下左右を含めた5節点の係数を保持して、Red点かBlack点かで処理を分けるれば、シェール補元を直接生成できることを意味している。

2.2 3次元7点差分

3次元7点差分FDMで、RB順序に従って行列\(A\)を分割した場合の、小行列\(A_{rb}\)と\(A_{rb}\)を図5に示した（\(nx\)と\(ny\)は奇数）\(nx\)が奇数の場合は、これらの小行列は列数と行数が1だけ異なる長方形行列になる。

7点差分格子では1点は前後左右上下の6点と

図4: シェール補元の差分スキーム
接続する。元の番号付けが、左から右、次に手前から奥に、最後に下面から上面にという方法によっているとき、図に示された6本のベクトルはそれぞれ、下 (D:Down)，手前 (F:Fore)，左 (L:Left)，右 (R:Right)，奥 (B:Back)，上 (U:Up) 側の点との連続性である。nx, ny ともに奇数の場合は、
A_b は6本対角行列 (-nxy0, -nx0, 0, 1, nx1, nxy1), A_r は6本対角行列 (-nxy1, -nx1, -1, 0, nx0, nxy0) になる。ただし nxy0 = [nx * ny/2], nxy1 = [(nx * ny + 1)/2] とする。

したがってシュール補元は19本対角行列 (sym, 0, 1, nx0, nx1, nx, nxy0 - nx0, nxy0, nxy1, nxy1 + nx0, nxy) である。

差分スキームとしては、図4の2次元9点スキームをxy面、yz面、zx面に重ねた3次元19点スキームを考えればよい。19は、対角項に6つ (U-D, D-U, F-B, B-F, R-L, L-R) の重なり、斜めの12点で2つの重なりがあるので、6×6＝(12×1+1×5) より得られる。

2.3 シュール補元の不完全LU分解

2次元5点差分や3次元7点差分が生成する行

列 A を不完全LU分解 (ILU(0)) すると、対角項以外に更新はしない。シュール補元 S は差分スキームとして見ると、斜め方向と2つ離れた縦横 (前後) 方向の点に接続が延長されているので、不完全LU分解によって非対角項にも更新される項が多い。

これは2次元では図6のように表すことができ、左図の5点差分では口印の行から見ると、上の位置にある節点が消去されると、LU分解ならフィルインとなる3つのO印には口印と重なりがないので、不完全LU分解では無視される。しかし右図の9点差分ではこの重なりが多い。

3次元で19点差分の場合、LU分解を更新型で考えると、消去される節点より大きい番号の9節点の積81項の中で、更新される項は、対角項の9項と非対角項の42項である (図7)。

この性質は前処理としてILU(1)のようなフィルインを部分的に採り入れる分解を考慮しなくても、前処理の効果が大きいという利点になる反面、前処理による記憶域の増加という欠点にもなる。

3 収束性の評価

例題として、解析領域 [0,1] × [0,1] × [0,1] において、次の偏微分方程式の7点中心差分による離
散化を考える。

\[-(Ku_x)_x - (Ku_y)_y - (Ku_z)_z + Gu_x = F\]

3.1 対称行列と CG 法

流動項の係数 \( G \) がゼロの場合はボアソング方程式となり、係数行列 \( A \) は対称である。これを前処理なし（CG 法）と、不完全コレスキー分解による前処理付きの CG 法（ICCG 法）で解き、全体系を反復する場合と、縮小系を反復する場合の収束性を比較した。

生成項 \( F \) は中心に位置する \( 0.1 \times 0.1 \times 0.1 \) の立方体の部分領域で \( F = 100 \) 、その外側の領域ではゼロとした。

解析領域を \( x, y, z \) 方向にそれぞれ \( nx, ny, nz \) に等しく分割した。\( nx, ny \) が倍数の場合、縮小系の係数行列は \( nx + 1, ny + 1 \) に拡大されている。

拡散係数 \( K \) が領域全域で 1 の場合と、図 8 のような分布を持つ場合を解いた。分布は、図で斜線を入れた領域で \( K = 10^4 \)、網掛けした領域で \( K = 10^{-5} \)、その他の領域で \( K = 100 \) とした。左図の斜線断面で左から見た図が右図である。中央の出口は辺が 0.02 の正方形なので、\( nx, nz \) が 50 よりも小さい偶数の場合は出口が塞がれる。

境界条件は領域の 6 つの面のうち、上面を除いて \( u = 1 \)、上面 \( (y = 1) \) の面で \( u = 0 \) のディリクレ条件とした。この問題は Van der Vorst の 4 番目の問題を、3 次元に拡張したものである [3]。

係数行列はどちらの場合も対角項を 1 にスケーリングしている。

収束は相対残差 \( ||r||_2 / ||r_0||_2 < 10^{-5} \) で判定し、初期反復ベクトルは右辺ベクトルを用いた。

収束までの反復回数（上段）と条件数（下段括弧内）を、拡散係数が一定の場合は表 1 に、拡散係数が変化する場合を表 2 に示した。前処理なしの場合は反復回数は、縮小系全体系の半分程度になっている。拡散係数が一定の問題では、固有値の分布が一定であると思われる。これによってどの分割数でも、反復回数を条件数の平方根で割ると、CG 法は 5、ICCG 法は 6 になっている。これは固有値の分布状態が保存されたまま行列が縮小され、条件数 \( \kappa \) が \( O(h^{-2}) \) \( \rightarrow \) \( O(h^{-1}) \) となって、反復回数が \( O(\sqrt{\kappa}) \) に比例して減少したからと考えられる。

拡散係数が変化する問題でも縮小の効果は同様に現れるが、反復回数と条件数の平方根の比は、分割数によって異なる。これは分割数を変えると、係数の異なる部分領域（特に出口部分）に属する節点数が変わるため、固有値の分布が変化するためと思われる。

全体系に前処理を行った場合と、縮小系に前処理を行った場合を比較すると、反復回数は全体系に前処理を行った場合のほうが少ない。プログラムのチューニングを行っていないので、計算時間は制限したが、前処理資金では代入計算のために、反復 1 回あたりの計算時間は 2 倍以上になるのが普通である。したがって全体系に前処理するよりも、縮小系に前処理なしのほうが速い。

これらの結果から、対称行列を CG 法で解く場合は、縮小すること自体が有効な前処理といえる。これは 2 次元 5 要素条件分間問題に対する報告 [1] と一致する結論である。

### 表 1: 反復回数と条件数（対称行列：\( K \) 一定）

<table>
<thead>
<tr>
<th>分割数</th>
<th>全体系</th>
<th>縮小系</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 41^3 )</td>
<td>CG: 135 (714)</td>
<td>ICCG: 52 (73.6)</td>
</tr>
<tr>
<td>( 60^3 )</td>
<td>CG: 196 (1505)</td>
<td>ICCG: 73 (155)</td>
</tr>
<tr>
<td>( 80^3 )</td>
<td>CG: 259 (2656)</td>
<td>ICCG: 96 (272)</td>
</tr>
</tbody>
</table>

---

-11-
### 3.2 非対称行列と Bi-CGSTAB 法

拡散係数 $K$ は図8の分布で、これに移流項 $G = 2 \exp(2(x^2+y^2))$ を加えた非対称行列とした。この問題は前処理なしでは容易には解けず、したがって不完全 $LU$ 分解法がおとめに効果的な Bi-CGSTAB 法で全体系と縮小系の収束性を比較した。なお Bi-CGSTAB 法を選んだ理由は、転置行列に対する行列演算がいかがからである。

結果を表3に示した。収束に至るまでの反復回数を ICCG 法の場合と比較すると、ばらつきはあるものの、縮小の効果は対称行列の場合と同様の範囲にある。

### 4 まとめ

3次元7点差分の問題で、RB 順序に基づくシェール補元を反復法に用いることが有効であることを確かめた。この方法の欠点は記憶量に対する要求が大きくなることであるが、この制限を緩い場合では実用的に期待できる。係数行列が対称正定盤の場合は、全体系を前処理付きで反復計算するよりも、縮小系を前処理なしで反復したほうが速い場合もあり、並列化には期待が持てる。

前処理なしであれば、シェール補元を陽に求めず、行列ベクトル積を $(A_{xy} - A_{xt}A_{xt}^{-1}A_{xy})x$ の形で陰に計算すれば、プログラミングも簡単で、計算量も少なく済む、しかしこの方法では、メーター提供の反復法のライブラリが利用できない欠点がある。プログラム全体をあらたに開発するのであれば、n 本対角行列のデータ構造を、n 本の長さが $nx \times ny \times nz$ の 1 次元配列から、$n \times nx \times ny \times nz$ の 4 次元配列に変更すれば、$nx$ や $ny$ が偶数の場合の調整も避けられる。

このように直交格子に単純な差分スキームを適用した FDM の場合、シェール補元を RB 順序で作成することは、反復法の高速化の有力な手段である。節点の順序付けは、境界適合格子や高次の差分スキームではマルチカラー順序、非構造格子では適当な順序（nested dissection ordering）が、RB 順序に変わるものとして考えられる。これからの順序付けによって、記憶域が許す範囲で係数行列を縮小することは、前処理による代入計算が並列化の障害となりやすいので、分散メモリ型の並列計算機では重要な高速化の手法と考えられる。

### 参考文献

2. R. Barrett 他、長谷川進美、長谷川秀成、藤野 清太郎：反復法 Templates，朝倉書店，1996。