Bi-recurrence法の諸特性

成　富　　　敬

あらまし

三重対角連立一次方程式の求解問題は，科学技術計算やCAD(Computer Aided Design：コンピュータ支援設計）あるいは派生証券の価格付けなどさまざまな分野であらわれる基本的な問題のひとつである。本稿では，三重対角連立一次方程式の新しい並列解法であるbi-recurrence法の数値的安定性，並列計算性，負荷分散性などの特徴について考察している。

1　はじめに

従来，地球気象や宇宙構造あるいは生態系の変動を理解したり経済変動の要因を解明したりする等，自然や社会の中のさまざまな現象を理解するには，理論的手法と実験的手法とが用いられてきた。近年，コンピュータ技術の飛躍的な進歩により第三の手法すなわち数値的手法が出現した。この数値的手法では，現象を記述する数学モデルである方程式を，差分近似等の離散化手法によって計算モデルへ変換し，この計算モデルの数値解を求めることにより，現象の再現を試みる。数値的手法を用いることにより，計算モデルに対する解析解が得られない種類の問題に対しても数値解を導出でき，さらに実験的手法では手にいれる事が不可能な知見を得ることができる等々の利点がある。このため，数値的手法の重要性は日々増大しており，数値的手法をより有効なものとするためのハードウェアの開発や問

題を解くための計算手法の開発が急務とされている。

ところで，三重対角連立一次方程式やブロック三重対角連立一次方程式の求解問題は，流体や熱伝導などの科学技術計算をはじめとして，コンピュータを用いた曲線や曲面の設計あるいは派生証券の価格付けに関する数値計算などさまざまな分野であらわれる基本的かつ重要な問題のひとつである。この三重対角連立一次方程式の求解法の開発はそのままブロック三重対角連立一次方程式の求解法[1]へと拡張できるばあいがあり，コンピュータによるシミュレーションがより大規模なものとなりつつある今日，新しい求解法を開発する意味は大きいといえる。

これまで，三重対角連立一次方程式の代表的な解法としてガウスの消去法やLU分解法が知られているが，これらはもともと逐次型計算機に適した解法である。大規模問題を解くためには，多数の処理要素を同時に働かせて問題を高速に解くことのできる並列計算機に適した並列計算手法の開発が不可欠であり，本質的並列性を持つ解法のひとつにbi-recurrence法[2]がある。

Bi-recurrence法は並列計算機に適した本質的並列計算性を持つ解法であり，その有効性はスーパーコンピュータをもちいた数値実験[3]により確認されている。本稿では，bi-recurrence法の数値的安定性，並列計算性，負荷分散性などの特性について考察している。特に，bi-recurrence法の負荷分散性に関して，共有メモリ型並列計算機や分散メモリ型並列計算機を対象に解析し，負荷分散のパラメータを決定している。

2　Bi-recurrence法

三重対角連立一次方程式$\boldsymbol{Ax}=\boldsymbol{h}$はつぎのように表現される。

$$\begin{pmatrix} d_1 & e_1 & & & \mathbf{0} \\ c_2 & d_2 & e_2 & & \\ & \cdots & \cdots & \cdots & \\ & & & d_{N-1} & e_{N-1} \\ \mathbf{0} & & & c_N & d_N \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{pmatrix} = \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_{N-1} \\ h_N \end{pmatrix}.$$

ここで，$\boldsymbol{A}$はN行N列の正則な実三重対角行列とし対角優位であるとする。行列$\boldsymbol{A}$において，d_1，...，d_Nは対角要素ですべて非零であるとする。また，c_2，...，c_Nとe_1，...，e_{N-1}とは，それぞれ係数行列$\boldsymbol{A}$の下対角要素と上対角要素とであり，$c_1 = e_N = 0$とする。

三重対角連立一次方程式$\boldsymbol{Ax}=\boldsymbol{h}$を解くということは，既知の$\boldsymbol{A}$，$\boldsymbol{h}$に対して，未知の$\boldsymbol{x}$を求めることであり，$\boldsymbol{Ax}=\boldsymbol{h}$を解くbi-recurrence法は，つぎのAttractiveステージ，InteractiveステージそしてRepulsiveステージの三つのステージで構成される[2]。

Attractiveステージ

まず始めに，$\alpha_1 = -e_1/d_1$，$\beta_1 = h_1/d_1$，$\alpha_N = -c_N/d_N$，$\beta_N = h_N/d_N$，を計算する。次に，式（1a）をもちいて，$i = 2, 3, \ldots, m$に対してそれぞれα_i，β_iを計算し，式(1b)をもちいて，$k = N-1, N-2, \ldots, m+1$に対してそれぞれ$\alpha_k$，$\beta_k$を計算する。ここで，$\alpha_0 = \beta_0 = \alpha_{N+1} = \beta_{N+1} = 0$とし，$m$をバランサーと呼ぶ。

$$\alpha_i = -\frac{e_i}{d_i + c_i\alpha_{i-1}}, \quad \beta_i = -\frac{h_i - c_i\beta_{i-1}}{d_i + c_i\alpha_{i-1}}, \tag{1a}$$

$$\alpha_k = -\frac{c_k}{d_k + e_k\alpha_{k+1}}, \quad \beta_k = -\frac{h_k - e_k\beta_{k+1}}{d_k + e_k\alpha_{k+1}}. \tag{1b}$$

Interactiveステージ

式（2a），式（2b）により，x_m，x_{m+1}をそれぞれ計算する。

$$x_m = \frac{\beta_m + \alpha_m \beta_{m+1}}{1 - \alpha_m \alpha_{m+1}}, \tag{2a}$$

$$x_{m+1} = \frac{\beta_{m+1} + \beta_m \alpha_{m+1}}{1 - \alpha_m \alpha_{m+1}}. \tag{2b}$$

Repulsiveステージ

式（3a），式（3b）により，x_i，x_kをそれぞれ計算する。

$$x_i = \alpha_i x_{i+1} + \beta_i,\ (i = m-1,\ m-2,\ \dots,\ 1), \tag{3a}$$

$$x_k = \alpha_k x_{k-1} + \beta_k,\ (k = m+2,\ m+3,\ \dots,N). \tag{3b}$$

すなわちbi-recurrence法では，逐次型解法であるガウスの消去法やLU分解法と違い，Attractiveステージでα_i，α_k，β_i，β_kなどのパラメータを計算した後，Interactiveステージで式（2a）と式（2b）とにより，最初に解x_mと解x_{m+1}とがそれぞれ求まる。これらの解x_mと解x_{m+1}とをもちいて，式（3a）と式（3b）より解系列x_{m-1}，x_{m-2}，...，x_1と解系列x_{m+2}，x_{m+3}，...，x_Nとをそれぞれ求めることができる。

3 数値的安定性

Bi-recurrence法において，式（1a）における$d_i + c_i\alpha_{i-1}$，式（1b）における$d_k + e_k\alpha_{k+1}$，式（2a），式（2b）における$1 - \alpha_m\alpha_{m+1}$のそれぞれが零にならないことを証明する。

係数行列$\boldsymbol{A}$は対角優位であるので，

$$|d_1| > |e_1|, \tag{4a}$$

$$|d_i| > |c_i| + |e_i|,\ (i = 2,\ 3,\ \dots,N-1), \tag{4b}$$

$$|d_N| > |c_N|. \tag{4c}$$

最初に，$1 \le i \le m$に対して，$|\alpha_i| < 1$かつ$d_i + c_i\alpha_{i-1} \neq 0$，であることの証明をおこなう。

いま，$i = 1$のとき，$d_1 \neq 0$，$\alpha_0 = 0$より$d_1 + c_1\alpha_0 \neq 0$。また，式（4a）

の条件より，$|\alpha_1| = |-e_1/d_1| = |e_1| / |d_1| < 1$。

ところで，あるi（$2 \leq i \leq m$）について，$|\alpha_i| < 1$かつ$d_i + c_i\alpha_{i-1} \neq 0$，とする。このとき，$d_{i+1} + c_{i+1}\alpha_i = 0$，とすると，

$$\begin{aligned} |d_{i+1}| &= |-c_{i+1}\alpha_i| \\ &= |c_{i+1}||\alpha_i| \\ &< |c_{i+1}|, \end{aligned} \qquad (5)$$

となり，これは式（4b）の条件に反し，矛盾である。したがって，

$$d_{i+1} + c_{i+1}\alpha_i \neq 0. \qquad (6)$$

また，

$$\begin{aligned} |\alpha_{i+1}| &= \left|-\frac{e_{i+1}}{d_{i+1} + c_{i+1}\alpha_i}\right| \\ &= \frac{|e_{i+1}|}{|d_{i+1} + c_{i+1}\alpha_i|} \\ &\leq \frac{|e_{i+1}|}{|d_{i+1}| - |c_{i+1}\alpha_i|} \\ &= \frac{|e_{i+1}|}{|d_{i+1}| - |c_{i+1}||\alpha_i|} \\ &< \frac{|e_{i+1}|}{|d_{i+1}| - |c_{i+1}|} \\ &< \frac{|e_{i+1}|}{|e_{i+1}|} \\ &= 1, \end{aligned} \qquad (7)$$

となり，すべてのi（$1 \leq i \leq m$）に対して，$|\alpha_i| < 1$かつ$d_i + c_i\alpha_{i-1} \neq 0$，であることが証明された。

いっぽう，$|\alpha_k| < 1$かつ$d_k + e_k\alpha_k + 1 \neq 0$，の証明は，上記の証明プロセスと同様であり，ここでは省略する。

つぎに，

$$\begin{aligned} |1 - \alpha_m\alpha_{m+1}| &\geq 1 - |\alpha_m\alpha_{m+1}| \\ &= 1 - |\alpha_m||\alpha_{m+1}| \\ &> 0, \end{aligned} \qquad (8)$$

であり，$1-\alpha_m\alpha_{m+1}\neq 0$，が証明された。

したがって，三重対角連立一次方程式$\boldsymbol{Ax}=\boldsymbol{h}$において，係数行列$\boldsymbol{A}$が対角優位であれば，bi-recurrence法の数値計算は安定的におこなわれる。

4　並列計算性

Bi-recurrence法の特徴はその本質的な並列計算性にある。Bi-recurrence法はバランサーmを$m=N$とおくことによりトーマス法[4]へと自然に帰着されるが，bi-recurrence法の計算量はN次元の問題に対して$8N-4$であり，計算量が$8N-6$のガウスの消去法やトーマス法[4]と比べると，殆ど計算量の増加を伴うことなく並列度の向上を可能にしている。

ところで，三重対角連立一次方程式の解法としては，ガウスの消去法やLU分解法などがあるが，これらの解法はもともと逐次計算機しかなかった時代の解法であり，並列計算機での利用を前提としていない。もちろん，ベクトル型スーパーコンピュータでベクトル化手法を工夫することによりそこそこの高速化は達成できるが，問題の本質的解決にはならない。

そこで，並列計算機向けの解法として，分割統治法に基づく並列解法が考案されてきた[5],[6]。これらの並列解法は，大規模な三重対角連立一次方程式の求解問題を多数の小規模な求解問題に分割して解き，最終的にそれらの部分解を統合する方法である。しかしながら，分割の結果得られた小規模な三重対角連立一次方程式の求解に対しては，従来の逐次型解法であるガウスの消去法やLU分解法などを用いるしかないという欠点がある。

いっぽう，並列計算機での使用を念頭に置いた並列求解アルゴリズムとして，Evans[7]の解法があげられるが，この解法は行列の次元数Nの偶奇によるアルゴリズムの区別が必要であるという欠点がある。しかしながら，bi-recurrence法ではそのような区別が不要であり，よりシンプルな解法といえる。Bi-recurrence法のもつこのようなシンプルさは，VLSI（Very Large Scale Integration：超大規模集積回路）として実現するのに適して

おり，じっさい専用ハードウェアのアーキテクチャが提案されている[2]。

Bi-recurrence法の持つ本来の並列度は2であり，式（1a），式（2a），式（3a）の処理と式（1b），式（2b），式（3b）の処理とを並列に進めることができる。また，分割統治法と組み合わせることにより，いっそう並列度の高い解法を構成できる。

ところで，このような大規模連立一次方程式を解くための並列計算環境としては，一台のコンピュータ内に複数のプロセッサが存在しこれらのプロセッサがひとつのメモリを共有する共有メモリ型と，複数のプロセッサがそれぞれ局所メモリを持ち他のプロセッサの局所メモリにはネットワークを経由してアクセスする分散メモリ型とがある。さらに分散メモリ型並列計算機には，同種のコンピュータをネットワーク接続した均質な計算形態 (homogeneous computing) の場合と，異種のコンピュータをネットワーク接続した不均質な計算形態 (heterogeneous computing) の場合とが考えられる。

計算アルゴリズムを実行する計算機アーキテクチャとしてどれが適しているかは，そのアルゴリズムの性質による。Bi-recurrence法の場合はInteractiveステージにおいて，計算の同期を取らなければならない。すなわち，式（2a）と式（2b）より明らかなように，x_mとx_{m+1}との計算にそれぞれα_{m+1}，β_{m+1}とα_m，β_mとを必要とし，これらの値は他方のプロセッサから手にいれる必要がある。そのためx_mやx_{m+1}を計算するとき，共有メモリ型並列計算機の場合はメモリ空間を共有しているため通信遅延は特に問題にならないと考えられるが，分散メモリ型並列計算機の場合は他のプロセッサの局所メモリからネットワークを経由して転送してもらうことになり，その分の通信遅延が大きくなることが予想される。したがって，bi-recurrence法にもっとも適しているのは，共有メモリ型並列計算機であり，問題の次元数が大きくなり，計算時間に対し通信遅延の占める割合が小さくなるにつれ，分散メモリ型並列計算機に対してもbi-recurrence法は有効に機能するようになると考えられる。

5　負荷分散性

2個の演算プロセッサを持つ並列計算機あるいはネットワークで接続された2台の逐次計算機をもちいた場合の，bi-recurrence法の負荷分散性について考察する。ただし，ここでは求解処理を始める前に負荷の分散を決めておき，求解中は負荷の分散を変えない静的な負荷分散の場合を考える。また，求解処理の実行中には，他のプロセスは実行されていないものとする。

ところで，bi-recurrence法の処理時間Tは，式（1a），式（2a），式（3a），式（1b），式（2b），式（3b）の計算量を評価することにより，次式で与えられる。

$$
\begin{aligned}
T = & \, max\left\{\frac{6m-4}{P_i},\ \frac{6N-6m-4}{P_k}\right\} \\
& + max\left\{\frac{5}{P_i}+2t_c,\ \frac{5}{P_k}+2t_c\right\} \\
& + max\left\{\frac{2(m-1)}{P_i},\ \frac{2(N-m-1)}{P_k}\right\}.
\end{aligned}
\tag{9}
$$

ここで，式（9）において，P_i，P_k（$=rP_i$とし，rは実数とする）は2個の演算プロセッサそれぞれの演算処理能力（1秒間に実行可能な浮動小数点演算の回数），t_cは2個のプロセッサ間あるいは2台のコンピュータ間で，1個のデータを伝送するのに必要な通信時間を表し，$max\{\}$は大小比較演算を表す。

式（9）において，右辺第一項は式（1a）を計算する時間と式（1b）を計算する時間との比較，また右辺第二項は式（2a）を計算する時間と式（2b）を計算する時間との比較であり，$2t_c$の項は式（2a）におけるα_{m+1}，β_{m+1}と式（2b）におけるα_m，β_mとをそれぞれ他方のプロセッサあるいはコンピュータから転送するのに要する時間である。また，右辺第三項は式（3a）を計算する時間と式（3b）を計算する時間との比較である。

以下，式（9）で表される処理時間を最小にするバランサーmを，共有メモリ型並列計算の場合と分散メモリ型並列計算の場合とについて考える。

5．1　共有メモリ型並列計算

2個の演算プロセッサを持つ共有メモリ型並列計算機で並列処理をおこなう場合，それぞれのプロセッサはメモリ空間を共有しておりInteractiveステージでの通信時間は無視できると考えられるので，$t_c = 0$とおき，式（9）の右辺第二項は無視する。

このとき式（9）の右辺第一項を最小にするのは，

$$\frac{6m-4}{P_i} = \frac{6N-6m-4}{P_k}, \tag{10}$$

のとき，すなわち

$$m = \frac{N}{r+1} + \frac{2(r-1)}{3(r+1)}. \tag{11}$$

いっぽう，右辺第三項を最小にするのは，

$$\frac{2(m-1)}{P_i} = \frac{2(N-m-1)}{P_k}, \tag{12}$$

のとき，すなわち

$$m = \frac{N}{r+1} + \frac{(r-1)}{(r+1)}. \tag{13}$$

したがって，$N \gg r$で2個のプロセッサの演算能力に著しい違いがないとき，式（11）と式（13）の右辺第二項はおのおの無視でき，

$$m = \frac{N}{r+1}, \tag{14}$$

のときTが最小となる。

5．2　分散メモリ型並列計算

2台の逐次計算機がネットワークで接続された計算環境で並列処理をお

こなう場合，コンピュータ間でのデータの通信遅延があり，式（9）の右辺第二項の$2t_c$の項が無視できなくなる。

ところで，bi-recurrence法では，問題の規模に依らず2個の数値データを伝送すればよい。そのため，均質な分散メモリ型並列計算機では，問題の規模が大きくなればなるほど，アルゴリズム自体の計算に必要な処理時間が大きなものとなり，それだけ通信遅延の占める割合は小さなものとなる。したがって，このときのバランサーは共有メモリ型並列計算機のばあいと同じく，式（14）にしたがって決めれば良い。

いっぽう，不均質な分散メモリ型並列計算機では，通信遅延がアルゴリズム自体の計算時間に比べて無視できず，

$$\frac{8N-4}{P_i} \approx t_c, \text{ あるいは } \frac{8N-4}{P_k} \approx t_c,$$

となるばあいもあり，負荷を分散すること自体が意味をもたなくなる。

6　考察とまとめ

本稿では，bi-recurrence法の数値的安定性，共有メモリ環境や分散メモリ環境での並列計算性，そして静的な負荷分散性について考察した。特に，静的負荷分散という条件下での，負荷分散の具体的な指標を与えた。

ところで，これまで負荷分散性を考えるとき，2個のプロセッサの能力に著しい違いがないばあいについて考察してきたが，2個のプロセッサの能力に著しい違いがあるときは，式（11）と式（13）より，バランサーmの値は$m=1$（$r\to\infty$）あるいは$m=N-1$（$r\to 0$）となり，並列計算をする積極的意味はなく，能力の大きいプロセッサのみをもちいて処理したほうが処理時間が短くなると思われる。

また，本稿で議論しなかったこととして，動的な負荷分散性があげられる。一般に，このような連立一次方程式$\boldsymbol{Ax}=\boldsymbol{h}$を解くときには，係数行列$\boldsymbol{A}$は同じままで，いくつもの$\boldsymbol{h}$に対して求解をおこなう必要がある。このよ

うなばあいには，求解前に負荷の分散を決めておき，求解中は負荷の分散を変えない静的負荷分散方式は有効ではないと考えられる。そのため，求解プロセスに起因する負荷あるいは他のプロセスに起因する負荷等のプロセッサのさまざまな負荷の状態を監視して，得られた負荷情報を次の求解の負荷分散に反映させる動的負荷分散が必要となる。このような，動的負荷分散アルゴリズムの開発は今後の課題である。

参考文献

(1) 成富敬，阿曽弘具：ブロック三重対角連立一次方程式の並列解法，情処研報，97, 21, 97-HPC65-7, pp. 33-38, 1997.

(2) Naritomi, T. and Aso, H.: A highly parallel systolic tridiagonal solver, IEICE Trans. Inf. & Syst., E79-D, 9, pp. 1241-1247, 1996.

(3) 成富敬，阿曽弘具：三重対角連立一次方程式の並列解法，電子情報通信学会総合大会講演論文集，D-3-6, 1997.

(4) Ames, W. F.: Numerical methods for partial differential equations (2nd ed.) ," Academic Press, London, 1977.

(5) Bondeli, S.: Divide and conquer: A parallel algorithm for the solution of tridiagonal linear systems of equations, Parallel Comput., 17, pp. 419-434, 1991.

(6) Sun, X. H., Zhang, H. and Ni, L. M.: Efficient tridiagonal solvers on multicomputers, IEEE Trans. Comput., C-41, pp. 286-296, 1992.

(7) Evans, D. J.: Parallel numerical algorithms for linear system," in Parallel Processing Systems, ed. Evans, D. J., pp. 357-384, 1982.