AP1000 における BiCGStab(ℓ) 法の有効性について

野寺 隆[†] 野口 雄一郎[†]

大規模で疎な非対称行列を係数とする連立1次方程式を解く積型反復解法の1つである BiCGStab 法は，1次の MR 多項式を使用して BCG 法の残差ノルムの収束を滑らかにしたものである．この算法は様々な問題に対して有効であり，一般にもよく知られている．近年，この方法を改良した BiCGStab2 法や GP-BiCG 法が提案されている．Sleijpen と Fokkema[13] により提案された BiCGStab(ℓ) 法はこれらの算法を一般化し，計算量を減らすように改良された算法である．本稿は分散メモリ型の並列計算機 AP1000（富士通）を用いて，2階の偏微分方程式の境界値問題などの数値実験により，BiCGStab(ℓ) 法の有効性について検証し，この算法が優れた反復解法であることを確かめるとともに，並列計算に適した方法であることを述べる．

Effectiveness of BiCGStab(ℓ) Method on AP1000

TAKASHI NODERA[†] and YUUICHIROU NOGUCHI[†]

For solving the large and sparse non-symmetric linear systems of equations, BiCGStab method is known as one of the product type of iterative solvers. This method smoothes the residual norm of BCG method, using degree one MR (minimal residual) polynomial. The BiCGStab method is efficient in many cases and has been used for actual problems. Recently, BiCGStab2 method and GP-BiCG method which improved by the BiCGStab method, have been proposed. The BiCGStab(ℓ) method, which is proposed by Sleijpen and Fokkema[13], is generalized by these methods. This algorithm is also improved to decrease the amount of computational cost per iteration. In this paper, the BiCGStab(ℓ) method and other related methods are parallelized on distributed memory machine Fujitsu AP1000. Results obtained from the numerical experiments, i.e. boundary value problems of 2nd order partial differental equations, etc., show that BiCG-Stab(ℓ) algorithm is effective iterative method and suitable for parallel computing.

1. はじめに

移流拡散方程式の境界値問題を有限差分法や有限要素法で離散化すると大規模な連立1次方程式

$$Ax = b \tag{1}$$

が得られ，この方程式の解を求めることが重要な問題となる．ただし，係数行列 A は $n \times n$ の非対称な正則行列とする．式(1)の解法には，係数行列の LU 分解を用いて解を求める直接法と，適当な初期値から始めて近似解の系列を計算してゆく反復解法がある．しかし，行列の次元 n が大きくなると，直接法は反復法と比較して計算コストやメモリ空間の面で採算がとれないことが多い．

式(1)の係数行列 A が疎で対称な正定値行列の場合には，Hestenes と Stiefel[1] によって提案さた共役勾配法（Congugate Gradient method，CG 法ともいう）は，有効な反復解法である[4),15)]．しかし，係数行列 A が非対称な正則行列の場合には，式(1)の近似解を求める解法として共役勾配法を用いることは正規方程式を解くことになり，方程式の条件数を悪化させるので，ある特殊な問題を除いてあまり有効な手段ではない．通常，非対称行列系の反復解法としては，一般共役勾配法（Generalized Conjugate Gradient method，GCG 法ともいう）のクラスに属する反復解法を用いることが多い[15),18),19),23)]．

Lanczos[2] と Fletcher[3] によって提案された BCG 法（Bi-Conjugate Gradient Method）は大規模な疎行列を係数とする連立1次方程式の近似解を求めるための反復解法である[4)]．特に，行列 A が非対称な場合にも，その算法は見かけ上対称行列を係数とする方程式を解く共役勾配法に似た反復解法になる．それゆえ，BCG 法の反復は非常に単純でメモリが少なくてすむという利点を持つが，残差ノルムの収束はしば

† 慶應義塾大学理工学部
Faculty of Science and Technology, Keio University

しば不規則で，反復の途中で停滞（stagnate）したり，破綻（break down：ブレイクダウン）をしたりして，うまく収束しないことがある[5]．しかし，BCG 法の算法は単純で上手に利用すると精度の良い近似解が得られることも多い．

Sonneveld[6]は BCG 法の算法を再構成して，残差ノルムの収束が BCG 法のそれと比較してほぼ 2 倍となる CGS 法（Conjugate Gradient Squared method）を提案した．しかし，この算法も BCG 法と同様，残差の収束にむらがあり，残差ノルムの発散や反復の途中で算法の破綻を招く場合も多い[5),6]．この点を改良し，残差ノルムの収束を滑らかにしたのが BiCGStab 法[8]や QMR 法[7]である．

1992 年，van der Vorst[8]は，BCG 法の残差ベクトルとそれを加速する 1 次の MR 多項式（Minimal Residual polynomial）との積を構成し，この新しい残差を 2–ノルムに関して最小化し，BCG 法を安定かつ高速に収束させる BiCGStab 法を提案した．BiCGStab 法は多くの問題に対して優れた残差ノルムの収束性を示し，現在，大規模な非対称行列問題の反復解法として一般によく使われている[9]．しかし残念ながら，係数行列 A がほぼ純粋に虚数部のみからなる複素固有値を持つような問題に対しては，残差ノルムの停滞や発散，反復途中での算法の破綻などが起こることが多い．そのような問題に対処する 1 つの解決策として，1993 年，Gutknecht[12]は，2 次の MR 多項式の導入によって BiCGStab 法に改良を加えた BiCGStab2 法を開発した．その後，Sleijpen と Fokkema[13]によってそれらの算法を一般化し，残差ノルムの停滞や発散に対してより頑強な BiCGStab(ℓ) 法が提案された．この算法では，高次の MR 多項式を使用しても記憶領域や計算量を最小限にとどめる工夫がなされている．また，わが国においてもランチョス過程に基づく積型反復法の理論から Zhang[14]や張と藤野[16]によって GP-BiCG 法が考案された．現在では，このように BiCGStab 系に関連する算法は，戦国時代という感じを受ける[23]．

従来，連立 1 次方程式 (1) に対する行列の前処理（preconditioning）は，係数行列の固有値分布を改め，それにより行列の条件数を改善することで，反復解法の収束性を向上させ，計算時間を短縮するための有効な手段として開発されてきた[15]．しかし，並列計算機を用いて計算を行う場合に，行列 A が非対称なときには，ある特殊なケースを除いて有用な前処理は近年あまり提案されていない[15]．また，各々の問題にマッチする前処理行列（preconditioner）を見つけ出すこともそれほど簡単なことではない．そのため最近では，非対称行列を係数とする式 (1) の近似解を求めるには，前処理なしの反復解法が用いられることが多い．それゆえ，1 回の反復ごとに行う計算量が多少増大しても，それに見合うだけの強力な収束性を示す算法が必要となることはいうまでもない．

BiCGStab 法を改良して作られた BiCGStab2 法[12]，GP-BiCG 法[16]，BiCG-Stab(ℓ) 法[13]などの算法は，BCG 法や BiCGStab 法に比べて 1 回の反復にかかる平均的な計算量は大きい．しかし，これらの算法は様々な問題に対して BCG 法や BiCGStab 法よりも優れた収束を示す．その中でも BiCGStab(ℓ) 法は，各反復における計算量や残差ノルムの収束性などの点から他の算法に比べて優れていると考えられる[13),21),22]．本稿では，この BiCGStab(ℓ) 法とその他の BiCGStab 法に関連する算法を並列化し，いくつかのモデル問題に適用して，BiCGStab(ℓ) 法が優れた算法であることを示すとともに，分散メモリ型の並列計算に適した算法であることを示す．

2 章ではこれらの BiCGStab 法に関連する算法を紹介し，それぞれの算法の特徴を示す．3 章では分散メモリ型の並列計算機 AP1000 への実装方法について述べる．4 章では並列計算機 AP1000 上で，2 章で述べた BiCGStab 法に関連する算法を 2 階の偏微分方程式の離散化から得られる代表的なモデル問題に適用し，それぞれの算法の残差ノルムの収束の様子について比較検討を行う．最後に前述の数値実験から得られた結論について述べることにする．

2. BiCGStab 法関連の方法

本章では，BiCGStab 法に関連するいくつかの算法の特徴について述べる．

2.1 BiCGStab 法

van der Vorst[8]によって提案された BiCGStab 法（**図 1** を参照）は，BCG 法で得られる残差ベクトルと 1 次の MR 多項式の積を作り，この新しい残差ベクトルの 2–ノルムを各々の反復で最小化する算法である．こうすることにより，BCG 法の不規則な残差ノルムの収束性を改善し，いくぶん滑らかな収束を期待するものである．すなわち，BiCGStab 法の k 回目の反復における残差ベクトルは

$$r_k = (I - \alpha A) q_{k-1}(A) \tilde{r}_k \qquad (2)$$

という数式で表され，$\|r_k\|_2$ が最小になるように変数 α を決定することになる．ただし，$\tilde{r}_k$ は BCG 法の k 回目の反復における残差ベクトルであり，$q_k(x) = (1 - \alpha x) q_{k-1}(x)$ とする．各反復で必要となる行列と

[BiCGStab Algorithm]

choose $\boldsymbol{x}_0$, $\tilde{\boldsymbol{r}}_0$

compute $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$

take $\rho_0 = \alpha = \omega_0 = 1$, $\boldsymbol{v}_0 = \boldsymbol{p}_0 = 0$

repeat until $\|\boldsymbol{r}_i\|_2$ is small enough

For $i = 1, 2, 3, \ldots$

$$
\begin{aligned}
\rho_i &= (\hat{\boldsymbol{r}}_0, \boldsymbol{r}_{i-1}) \\
\beta &= (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1}) \\
\boldsymbol{p}_i &= \boldsymbol{r}_{i-1} + \beta(\boldsymbol{p}_{i-1} - \omega_{i-1}\boldsymbol{v}_{i-1}) \\
\boldsymbol{v}_i &= \boldsymbol{A}\boldsymbol{p}_i \\
\alpha &= \rho_i/(\hat{\boldsymbol{r}}_0, \boldsymbol{v}_i) \\
\boldsymbol{s} &= \boldsymbol{r}_{i-1} - \alpha\boldsymbol{v}_i \\
\boldsymbol{t} &= \boldsymbol{A}\boldsymbol{s} \\
\omega_i &= (\boldsymbol{t}, \boldsymbol{s})/(\boldsymbol{t}, \boldsymbol{t}) \\
\boldsymbol{x}_i &= \boldsymbol{x}_{i-1} + \alpha\boldsymbol{p}_i + \omega_i\boldsymbol{s} \\
\boldsymbol{r}_i &= \boldsymbol{s} - \omega_i\boldsymbol{t}
\end{aligned}
$$

図1 BiCGStab 法の算法
Fig. 1 The BiCGStab algorithm.

ベクトルとの積の数は BCG 法と同じであるが，この算法では BCG 法に現れる $A^T v$ の計算は必要とせず，2 回の Av の計算のみが必要となる．

BiCGStab 法は様々な偏微分方程式の境界値問題に対して良い収束性を示している[8),9)]．しかし，BiCGStab 法においても 1 次の MR 多項式の変数 α が零ではないが零に近い値をとるとき，この算法の収束は停滞したり，ときには反復の続行に破綻を来すこともある．実際，連立 1 次方程式 (1) の係数行列 A の行列要素がすべて実数で，複素数の固有値を持ち，さらにその固有値の実数部と比較して虚数部が相対的に大きな値をとるような問題に対して，この算法を適用すると，残差ノルムの収束が悪くなり，ある時点から収束の停滞や発散が起こる場合がある[12)]．これは BiCGStab 法が使用する 1 次の MR 多項式の退化に起因するものであると考えられる．この 1 次の MR 多項式が退化すると，BCG 反復の係数計算に誤差の混入を招き，反復を繰り返せば繰り返すほどそれが拡大するからである．そこで，このような状況を打開するために，BiCGStab 法の算法に 2 次の MR 多項式を導入したのが，Gutknecht[12)] により考案された BiCGStab2 法である．

2.2 BiCGStab2 法

Gutknecht[12)] により提案された BiCGStab2 法は，BiCGStab 法で用いている 1 次式に加え，次のステップで 2 次の MR 多項式を利用して残差ノルム $\|r_k\|_2$ の最小化を行う算法である．これにより，連立 1 次方程式 (1) の係数行列 A の行列要素がすべて実数ではあるが，(虚数部が実数部と比較して極端に大きな値をとる) 複素固有値を持つ問題に対しても，残差ノルムの停滞や発散をある程度防ぐことが可能で，収束の安定性を得ることができる．ただし，本稿では紙面の都合上，BiCGStab2 法の算法の記述を省略したので，その詳細に関しては文献 12), 23) を参照してほしい．

BiCGStab2 法の k 回目の反復における残差ベクトル r_k は

$$
r_k = \begin{cases} (I - \alpha A)q_{k-1}\tilde{r}_k, & \text{if} \quad k = 2m \\ ((1-\beta)q_{k-2} + (\beta + \gamma A)q_{k-1})\tilde{r}_k, & \text{if} \quad k = 2m+1 \end{cases} \tag{3}
$$

という式で表される．ただし，変数 α, β, γ はいずれも実数である．すなわち，奇数回目の残差をこのように決めることで $q_k(0) = 1$ が保証され，β, γ を変数とする $q_k(0) = 1$ を満たす任意の 2 次の多項式が選べることになる．

ここで，式 (3) は，次式のように書き変えることができる．

$$
r_k = (I - \omega A)(I - \tilde{\omega} A)q_{k-2}(A)\tilde{r}_k \tag{4}
$$

ただし，ω, $\tilde{\omega}$ はともに実数，もしくは共役な複素数である．つまり実際の計算は実数で行っているが，複素数を扱っていることになる．

この BiCGStab2 法の奇数回目の反復では，BCG 法の残差ベクトルと 2 次の MR 多項式を組み合わせた残差ノルム $\|r_k\|_2$ の最小化を行っている．実際には式 (3) を展開して，次のような式に直してこれを最小化することになる．

$$
\begin{aligned}
r_k = (&q_{k-2} + \beta(q_{k-1} - q_{k-2}) \\
&+ \gamma(Aq_{k-1}))\tilde{r}_k
\end{aligned} \tag{5}
$$

この式に対して大きさ $n \times 2$ の行列 B_{m+1} を

$$
B_{m+1} = [q_{k-1} - q_{k-2} | Aq_{k-1}], \qquad k = 2m+1 \tag{6}
$$

のように定めると式 (5) の最小 2 乗解は，$k = 2m+1$ に対して

$$
\begin{pmatrix} \beta \\ \gamma \end{pmatrix} = -(B_{m+1}^T B_{m+1})^{-1} B_{m+1}^T q_{k-2} \tag{7}
$$

のようにして求められる．ここで，$x = q_{k-1} - q_{k-2}$, $y = Aq_{k-1}$ として式 (7) を書き直すと，β, γ は次のような式で計算できることになる．

$$
\beta = \frac{(x,x)(x,q_{k-1}) - (x,y)(y,q_{k-1})}{(x,x)(y,y) - (x,y)(x,y)} \tag{8}
$$

$$
\gamma = \frac{-(x,q_{k-1})(x,y) + (x,x)(y,q_{k-1})}{(x,x)(y,y) - (x,y)(x,y)} \tag{9}
$$

このようにして変数 β, γ を決定すると，内積の計算

が数多く現れることになる．

この算法の奇数回目の反復では，BCG 法の残差ベクトルと 2 次の MR 多項式を組み合わせた残差ノルム $\|r_k\|_2$ を最小化しているが，偶数回目の反復では BiCGStab 法と同じ 1 次の MR 多項式を組み合わせた残差ノルム $\|r_k\|_2$ を最小化することになる．すなわち，BiCGStab2 法の 1 ステップの反復は，前述の奇数回目の反復と偶数回目の反復をペアで行うことになる．ここで，1 次の MR 多項式について考えると，偶数回目のステップで，その前のステップの 1 次の MR 多項式を 2 次の多項式に修正する操作を行うことになる．しかし，奇数回目のステップで 1 次の MR 多項式がほぼ退化する現象が起こると，これは BCG 反復の（内積計算による）係数計算にも悪影響を与えることになる．これにより，その次の 2 次の MR 多項式の計算にも大きな誤差の混入を招き，算法の続行を不安定なものにする．このため，BiCGStab2 法においても BiCGStab 法と同様に，この 1 次の MR 多項式の退化が原因で，反復途中で残差ノルムの停滞や発散が起こり，算法の破綻を招く可能性を残している[13]．

2.3 GP-BiCG 法

GP-BiCG 法[14]は，ランチョス過程に基づく積型反復解法の理論から生まれたものであるが，BiCGStab 法や BiCGStab2 法を一般化した方法として考えることもできる．ただし，本稿では紙面の都合上，この算法の記述を省略したので，算法の詳細は文献 14)，16) を参照してほしい．

GP-BiCG 法は BiCGStab2 法の奇数回目の反復を偶数回目の反復でも行う算法である．すなわち，各々の反復で 2 次の MR 多項式と BCG 法の残差ベクトルとの積を作り，その残差ノルム $\|r_k\|_2$ の最小化を行うことになる．ただし，BiCGStab2 法で用いた 1 次の MR 多項式は使わない．

GP-BiCG 法の k 回目の反復における残差ベクトルは

$$r_k = ((1-\beta)q_{k-2} + (\beta+\gamma A)q_{k-1})\tilde{r}_k \quad (10)$$

という式で表され，反復ごとにこの残差ノルム $\|r_k\|_2$ が最小になるように変数 β, γ を決定することになる．すなわち，GP-BiCG 法は毎回の反復で前節の式 (5)～(9) と同様の方式で β, γ を決定することになる．しかし，この計算方式では，変数 β, γ の決定に式 (7) の中に現れる 2×2 の逆行列の計算を毎回行う必要があり，それにともない内積計算の数が増え，BiCGStab2 法よりも 1 回の反復あたりの計算量が増加することになる（後述の表 1 を参照せよ）．

2.4 BiCGStab(ℓ) 法

Sleijpen と Fokkema[13] によって提案された BiCGStab(ℓ) 法（**図 2** を参照）は，ℓ 回の BCG 法反復で得られた残差ベクトルと ℓ 次の MR 多項式を組み合わせ，この残差ノルムを局所的に最小化する算法である．今，$k = m\ell + \ell$ とすると，BiCGStab(ℓ) 法の k 回目の反復での残差ベクトルは

$$r_k = p_m(A)q_{k-\ell}(A)\tilde{r}_k \quad (11)$$

と表される．ただし，$q_{k-\ell}(A) = p_{m-1}(A)p_{m-2}(A)\ldots p_0(A)$ であり，$p_m(0) = 1$ を満たす．また，$p_0(A), p_1(A), \ldots, p_m(A)$ はいずれも ℓ 次の MR 多項式であり，$q_{k-\ell}(A)$ は $k(=m\times\ell)$ 次の多項式となる．なお，$m\times\ell$ 回目のステップで使われる ℓ 次の MR 多項式 $p_m(A)$ は，$\|r_k\|_2$ が最小になるように選ばれる．

BiCGStab(ℓ) 法は，BiCGStab2 法を一般化したものと考えられ，ℓ 次の MR 多項式を利用する算法である．しかし，BiCGStab2 法と同じ方法でこれを最小化すると，算法が複雑になり，計算時間と実装に必要な記憶領域が増えてしまう．そこで BiCGStab(ℓ) 法では計算方法に工夫をこらすことで，記憶領域や計算量の増加を最小限にとどめ，ℓ 次の MR 多項式を組み合わせた残差ノルム $\|r_k\|_2$ の最小化を行えるようにしたものである．

BiCGStab(ℓ) 法では，$k = m\ell + \ell$ ($m = 0, 1, 2, \ldots$) に関して，残差ベクトル r_k，探索方向ベクトル u_{k-1}，近似解 x_k の系列を計算することになる．通常，この反復のことを外部反復と呼んでいる．1 回の外部反復（すなわち，$k = m\ell$ から $k = m\ell + \ell$ に進む過程）は，内部反復（ステップ）と呼ばれる処理を行うことになる．この内部反復では，2 つの処理を行う過程がある．すなわち，新しい BCG 法のベクトルを計算する部分と ℓ 次の MR 多項式を用いて局所的に最小な残差ベクトルを求める部分である．以下，この 2 つの過程の概略を示すことにする．

(1) **BCG 法を使う部分**

前回のステップで得られた残差ベクトル r_k に対し，BCG 法の反復を ℓ 回適用する．これで得られたベクトルを $\tilde{r}_k$ とすると，計算の途中で副産物として $A^i\tilde{r}_k$ ($i = 0, \ldots, \ell$) が得られる．通常の BCG 法では，1 回の反復で Av と A^Tv の計算を必要とするが，この BCG 法の反復を ℓ 回行う BiCGStab(ℓ) 法は，$2\times\ell$ 回の Av の計算を必要とする．すなわち，行列とベクトルとのかけ算に関して，BiCGStab(ℓ) 法は BCG 法と同じだけの計算量しか必要としない．

[BiCGStab(ℓ) Algorithm]

$k = -\ell$

choose x_0 , $\tilde{r}_0$

compute $\mathbf{r}_0 = b - Ax_0$

take $\mathbf{u}_{-1} = 0$, $\mathbf{x}_0 = x_0$, $\rho_0 = 1$, $\alpha = 0$, $\omega - 1$

repeat until $\|\mathbf{r}_{k+\ell}\|_2$ is small enough

$k = k + \ell$

$\hat{u}_0 = \mathbf{u}_{k-1}$, $\hat{r}_0 = \mathbf{r}_k$, $\hat{x}_0 = \mathbf{x}_k$, $\rho_0 = -\omega\rho_0$

For $j = 0, \ldots, \ell - 1$ (BCG part)

$\rho_1 = (\hat{r}_j, \tilde{r}_0)$, $\beta = \alpha\frac{\rho_1}{\rho_0}$, $\rho_0 = \rho_1$

For $i = 0, \ldots, j$

$\hat{u}_i = \hat{r}_i - \beta\hat{u}_i$

end

$\hat{u}_{j+1} = A\hat{u}_j$, $\gamma = (\hat{u}_{j+1}, \tilde{r}_0)$, $\alpha = \frac{\rho_0}{\gamma}$

For $i = 0, \ldots, j$

$\hat{r}_i = \hat{r}_i - \alpha\hat{u}_{i+1}$

end

$\hat{r}_{j+1} = A\hat{r}_j$, $\hat{x}_0 = \hat{x}_0 + \alpha\hat{u}_0$

end

For $j = 1, \ldots, \ell$ (modified Gram-Schmidt) (MR part)

For $i = 1, \ldots, j - 1$

$\tau_{ij} = \frac{1}{\sigma_i}(\hat{r}_j, \hat{r}_i)$, $\hat{r}_j = \hat{r}_j - \tau_{ij}\hat{r}_i$

end

$\sigma_j = (\hat{r}_j, \hat{r}_j)$, $\gamma'_j = \frac{1}{\sigma_j}(\hat{r}_0, \hat{r}_j)$

end

$\gamma_\ell = \gamma'_\ell$, $\omega = \gamma_\ell$

For $j = \ell - 1, \ldots, 1$

$\gamma_j = \gamma'_j + \sum_{i=j+1}^{\ell} \tau_{ji}\gamma_i$

end

For $j = 1, \ldots, \ell - 1$

$\gamma''_j = \gamma_{j+1} + \sum_{i=j+1}^{\ell} \tau_{ji}\gamma_{i+1}$

end

$\hat{x}_0 = \hat{x}_0 + \gamma_1\hat{r}_0$, $\hat{r}_0 = \hat{r}_0 + \gamma'_\ell\hat{r}_\ell$, $\hat{u}_0 = \hat{u}_0 + \gamma_\ell\hat{u}_\ell$

For $j = 1, \ldots, \ell - 1$

$\hat{u}_0 = \hat{u}_0 - \gamma_j\hat{u}_j$, $\hat{x}_0 = \hat{x}_0 + \gamma''_j\hat{r}_j$, $\hat{r}_0 = \hat{r}_0 - \gamma'_j\hat{r}_j$

end

$\mathbf{u}_{k+\ell-1} = \hat{u}_0$, $\mathbf{r}_{k+\ell} = \hat{r}_0$, $\mathbf{x}_{k+\ell} = \hat{x}_0$.

図 2 BiCGStab(ℓ) 法の算法
Fig. 2 The BiCGStab(ℓ) algorithm.

(2) ℓ 次の MR 多項式を用いる部分

BCG 法を用いた部分で作られた $A^i\tilde{r}_k$ $(i = 0, \ldots, \ell)$ を用いて，局所的に最小な残差ベクトル（locally minimal residual vector）r_k を見つけることである．つまり，$\|r_k\|_2$ が局所的に最小になるように

$$r_k = \tilde{r}_k + a_1A\tilde{r}_k + a_2A^2\tilde{r}_k + \cdots + a_\ell A^\ell\tilde{r}_k \quad (12)$$

の係数 $a_1, a_2, \ldots, a_\ell$ を決定することになる．この係数を GMRES(ℓ) 法を利用して求めるには，行列とベクトルとのかけ算が ℓ 回必要になる．しかし，BiCGStab(ℓ) 法では修正グラムシュミットの直交化法を用いることにより，行列とベクトルとのかけ算をこの部分の計算では必要としない．

以下，$A^i\tilde{r}_k$ $(i = 0, \ldots, \ell)$ から局所的に最小な残差ベクトル r_k の求め方を示す．ただし，$\hat{r}_i = A^i\tilde{r}_k$ とする．

$$R = (\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_\ell) \quad (13)$$

とする．これを修正グラムシュミットの直交化法で直交化したものを

$$Q = (q_1, q_2, \ldots, q_\ell) \quad (14)$$

とする．また T は $R = QT$ となるような大きさ $\ell \times \ell$ の下三角行列とし，大きさ $\ell \times \ell$ の行列 D を

$$D = Q^T Q$$
$$= \mathrm{diag}\,(\|q_1\|_2, \|q_2\|_2, \ldots, \|q_\ell\|_2)$$

と定める．ここで明らかに行列 D と T は正則であり，Q と R は非正則な行列である．このとき

$$\|r_k\|_2 = \|\hat{r}_0 - R\vec{\gamma}\|_2 \quad (15)$$

が最小になるような $\vec{\gamma}$ を求めればよい．式 (15) は

$$\|r_k\|_2 = \|\hat{r}_0 - QT\vec{\gamma}\|_2 \quad (16)$$

と書き変えることができるので，これを最小化することにする．この $\vec{\gamma}$ は

$$QT\vec{\gamma} = \hat{r}_0 \quad (17)$$

の最小 2 乗解として，次の式で与えられる．

$$\vec{\gamma} = T^{-1}D^{-1}Q^T\hat{r}_0 \quad (18)$$

よって，最小となる残差ベクトル r_k は

$$r_k = \hat{r}_0 - R\vec{\gamma}$$
$$= \hat{r}_0 - QD^{-1}Q^T\hat{r}_0 \quad (19)$$

として求められることになる．

この算法では，修正グラムシュミットの直交化を利用して，ℓ 本のベクトルの直交化を行っている．そのため ℓ が大きくなると直交化部分の計算量が $O(\ell^2)$ で増えていく．計算時間を短くするためには ℓ をできるだけ小さくすべきではあるが，逆に ℓ を大きくした方が早い収束を得られることもある．そのような場合には ℓ を大きくすることは有益となる．

2.5 各算法の計算量

表 1 には，それぞれの算法が 1 回の反復で必要となる平均的な計算量を示した．ただし，1 回の反復の定義であるが，Sleijpen と Fokkema[13] の反復の定義を用いて，ここでは Av（行列 × ベクトル）の計算を 1 回行う場合に（すなわち，クリロフ部分空間の計算を 1 回行うと），反復を 1 回行ったものと見なすことにする．たとえば，BiCGSTab(4) 法は，内部反復の前半で BCG 法（1 回の反復で，2 回の行列 × ベクトルの計算を必要とする）を 4 回行うために 8 回の行列 × ベクトルの計算を行わねばならない．そこで，BiCGSTab(4) 法の外部反復 1 回の計算量を考える場合には計算の総量を 8 で割り，反復 1 回の平均的な計算量を算出した．この他の算法に関しても同様の換算が必要なものは，この処理を行い平均的な計算量を算出した．ここで，表 1 の中の Av は行列 × ベクトル計算の回数を示し，$dots$ は内積計算の回数，αv はベクトルのスカラ倍の計算の回数を示す．また，4 章の数値実験で用いた行列に対して，各算法が 1 回の反復に必要とする浮動小数点数のかけ算の回数を示した．行列の大きさ（次元）は，いずれも 16384 次元である．この表を見ると，BiCGStab(2) 法の計算量が BiCGStab2 法や GP-BiCG 法に比べて少ないことが分かる．

表 1 1 回の反復に必要となる平均的な計算量
Table 1 The average amount of computational cost per iteration.

算法	Av	$dots$	αv	乗算の回数 * 例題 1	乗算の回数 * 例題 2
BiCGStab	1	2	3	131072	163840
BiCGStab2	1	2.75	5.5	184320	217098
GP-BiCG	1	3.5	8.0	248988	270336
BiCGStab(2)	1	2.25	3.75	147456	180224
BiCGStab(4)	1	2.75	5.25	180224	212992

* 4 章における数値実験の各例題で用いた行列に対して 1 回の反復で必要となる浮動小数点数の平均的な乗算の回数．行列の大きさはいずれも 16384 次元である．

3. 並列化の方法

BiCGStab 法関連の算法は，行列 × ベクトル，ベクトルの内積，スカラ × ベクトル演算の組合せで構成されている．今回これらの算法を並列化するにあたり，算法そのものを並列化するのではなく，行列とベクトルとの積，内積などの関数をそれぞれ並列化した．各々のセルに同じ長さのベクトルの計算を割り当てることで，ベクトルどうしの加算，減算，乗算などは容易に行うことができる．それに対し，並列計算において最も実装方法を工夫しなければならないのは行列 × ベクトルの計算である．以下，行列とベクトルとの乗算の並列化の方法について述べる．

矩形領域 $[0,1] \times [0,1]$ において 5 点中心差分法で離散化して得られる行列について考える．離散化する場合のメッシュの大きさを $m \times m$ とし，使用するセルを 2 次元に配置してセルの数を $p = x \times y$ とする．ただし，m は x，y の両方で割り切れる値をとるものとする．

与えられた矩形領域を重複しない同じ形をした正方形または長方形に区切る．$x = y$ のとき，区切られた領域は正方形になる．そしてこの領域を並列計算機上で 2 次元に配列されたセルにそれぞれ割り当てる．その分けられた領域ごとに整合順序で番号付けをして計算を行う．このとき，1 個のセルが担当するベクトルの大きさは $(m/x) \times (m/y)$ である．

計算時の通信量に関しては，隣のセルとの境界部分の要素を必要とするから x 軸方向に接している領域との通信については m/y の通信量を，y 軸方向については m/x の通信量を必要とする．特に $x = y$ のときの 4 方向の通信量を考えると $4 \times (m/x)$ の通信量が必要となる．一方，領域全体を単純に整合順序で番号付けして計算を並列化すると，その行列とベクトル

表 2 AP1000 の仕様
Table 2 Specification of AP1000.

プロセッサ間のネットワーク	ブロードキャストネットワーク（50 MB/s） 2 次元トーラスネットワーク（25 MB/s/port） 同期ネットワーク
セルプロセッサ	SPARC IU+FPU
セルキャッシュ	128 KB
セルローカルメモリ	16 MB
性能（倍精度）	5.56 MFLOPS

のかけ算に要する通信量は $2 \times m$ となる[21]．今回はセルの数を 64 個（$x = y = 8$）にしているので，通信量の点では今回の並列化の方法のほうが優れている．

4. 数値実験

BiCGStab 法に関連するいくつかの方法を並列計算機 AP1000 上で実行し，その残差ノルムの収束の様子を調べた．4.1 節では，特殊な複素固有値の分布を持つ Toeplitz 行列を係数とする連立 1 次方程式に対して，BiCGStab 系の算法を適用した場合に，各々の残差ノルムの収束について考察する．4.2 節では，偏微分方程式の境界値問題に対して数値実験を行った結果について述べる．4.3 節では，BiCGStab(ℓ) 法で，ℓ の値を変えたときのそれぞれの算法の残差ノルムの収束の様子を比較する．なお，いずれの問題に対しても行列の前処理を行わず，算法の持つ本来の収束特性を比較した．

また，数値実験ではそれぞれの場合について 5 回ずつ実験を行った．そのうえで，表中の収束時間については 5 回の実験の平均の値を示した．収束時間は，プロセッサが 1 回目の反復を開始したときから，プロセッサが計算を終わるまでの時間を計測した．時間と同様に，反復回数についても 5 回の平均の値をとり，この場合小数点以下の部分については切り捨てた．また実行時間を示したグラフについては 5 回実行した中で 3 番目に実行時間の短かったものを示した．

今回実験に使用した富士通の AP1000 はメッセージパッシングをベースとする MIMD 型並列計算機である．この AP1000 は高スループット/低レイテンシの通信ネットワークを持ち，高速メッセージ処理を行うことができるなどの特徴を持つ．今回用いた最大のプロセッサ数は 64 である．このシステムの仕様を**表 2** に示したので参考にしてほしい．

4.1 Toeplitz 行列に対する各算法の残差ノルムの収束性の比較

例題 1 次の連立 1 次方程式 $Ax = b$ の近似解を求める問題を考える．これは，Reichel と Trefethen[10]，Gutknecht[12]，および阿部ら[20]の文献で用いられている例題である．なお，行列の次元は 16384 とする．

$$A = \begin{pmatrix} 2 & 1 & & & \\ 0 & 2 & 1 & & \\ \eta & 0 & 2 & 1 & \\ & \eta & 0 & 2 & \ddots \\ & & \ddots & \ddots & \ddots \end{pmatrix} \tag{20}$$

ただし，右辺は $b = (1, 1, 1, \ldots, 1)^T$ とする．この問題の反復解法として BiCGStab 法，BiCG-Stab2 法，GP-BiCG 法，BiCGStab(2) 法を用いて数値実験を行った．各算法の初期近似解は零ベクトルとし，プロセッサ数は 64 個，最大反復回数は 2000 回，収束判定条件は $\|r_k\|_2/\|r_0\|_2 \leq 1.0 \times 10^{-12}$ とする．この問題に対し，η の値を様々に変えて数値実験を行い，その結果を**表 3** に示した．なお，この表には，残差ノルム（$\|r_k\|_2/\|r_0\|_2$）が収束条件を満たすまでに必要な計算時間（秒）T と，反復回数 I の値を示した．$\eta = 1.5, 1.7$ のときの BiCGStab 法および $\eta = 1.7$ のときの GP-BiCG 法は 2000 回では収束しなかった．そこで，$\eta = 1.7$ とした場合の計算時間（秒）に対する残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子を**図 3** に，反復回数（400 回まで）に対する残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子を**図 4** に示した．これらの図から，BiCGStab 法の残差ノルムは，初期反復の段階ではしばらくの間収束するが，その後多少振動を繰り返しながら停滞が続き，最終的には発散をすることになる．GP-BiCG 法の残差ノルムに関しては，10^{-10} 程度まで収束し，その後しばらく停滞し，徐々に発散していくことが読み取れる．GP-BiCG 法がこのように発散する原因の 1 つは，その算法の中で内積計算が多く現れ，浮動小数点演算における桁落ちや丸め誤差の影響を受けやすいからである．また，これらの図や表に示された結果からも分かるように，各算法において，収束条件を満たすのに必要な反復回数がまったく同じでも，各算法の計算時間に差異が生じることになる．よって，各算法の収束の良し悪しは，従来から行われてきた反復回数の多い少ないに関する評価ではなく，計算時間に関する評価で行う方がよい．

表3 例題1：残差ノルムの収束時間（T sec）および反復回数（I）
Table 3 Example 1: Computational time (T sec) and iteration count (I).

η	1.0		1.1		1.3		1.5		1.7	
算法	T	I	T	I	T	I	T	I	T	I
BiCGStab	1.883	94	2.014	118	4.788	246	—	—	—	—
BiCGStab2	1.505	56	1.729	64	2.278	86	3.320	124	5.402	192
GP-BiCG	1.711	56	1.872	64	2.742	90	3.819	124	—	—
BiCGStab(2)	1.084	56	1.193	64	1.698	88	2.489	126	3.675	186

注）2000 回の反復で収束しなかったものは “—” と記述した．

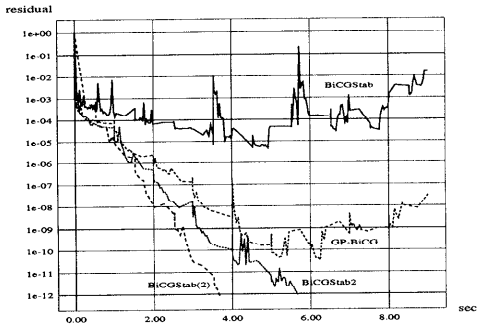


図3 例題1：計算時間に対する BiCGStab 系の残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子の比較（$\eta = 1.7$）
Fig. 3 Example 1: The behaviour of residual norm ($\|r_k\|_2/\|r_0\|_2$) v.s. computational time (sec) for the variants of BiCGStab method ($\eta = 1.7$).

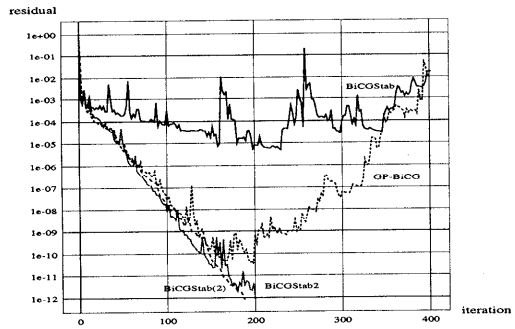


図4 例題1：反復回数に対する BiCGStab 系の残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子の比較（$\eta = 1.7$）
Fig. 4 Example 1: The behaviour of residual norm ($\|r_k\|_2/\|r_0\|_2$) v.s. number of iterations for the variants of BiCGStab method ($\eta = 1.7$).

Reichel と Trefethen[10]，Gutknecht[12]，阿部ら[20] などの文献に示されているように，これらの行列の固有値は複素平面上で点 $(2,0)$ を中心として点対称な三葉のように広がっており，実軸に対して対称な形をしている．固有値の多くは実軸から離れたところに分布しており，複素固有値の実数部の値に比べて虚数部の値が大きくなっている．BiCGStab 法は計算で用いている1次の MR 多項式がこの虚数の固有値に対応できずに残差ノルムの収束を悪くしたものと考えられる．また，$\eta = 1.0$ としたときよりも $\eta = 1.7$ としたときの方が固有値の虚数部分がより大きくなっている．$\eta = 1.7$ とした問題に対して，BiCGStab 法の残差ノルムの収束が悪くなっているのはこのためだと考えられる．いずれの問題に対しても，BiCGStab(2) 法が収束判定条件を満足するまでの実行時間は短い．

4.2 2階の偏微分方程式の境界値問題

例題2 矩形領域 $\Omega = [0,1] \times [0,1]$ における2階の楕円型偏微分方程式のディリクレ境界値問題を考える（Joubert[11] を参照）．

$$-u_{xx} - u_{yy} + Du_x(x,y) = G(x,y),$$
$$u(x,y)\,|_{\partial\Omega} = 1 + xy.$$

この方程式を5点中心差分近似を用いて離散化し，真の解を $u(x,y) = 1 + xy$ と設定し，右辺を決定して数値実験を行った．このとき，メッシュの大きさは 128×128 とした．なお，AP1000 のプロセッサ数は64個とした．解法として BiCGStab 法，BiCGStab2 法，GP-BiCG 法，BiCGStab(ℓ) 法 ($\ell = 2,4,8$) を用いて数値実験を行い，その収束の様子を比較した．各算法の初期近似解は零ベクトル，収束判定条件は $\|r_k\|_2/\|r_0\|_2 \le 1.0 \times 10^{-12}$ とした．

最初に，$h = 1/129$ とし，$Dh = 2^2$ となるように D の値を定め，BiCGStab 法，BiCGStab2 法，GP-BiCG 法，BiCGStab(2) 法について，**図5** には計算時間（秒）に対する残差ノルムの収束する様子を，**図6** には反復回数に対する残差ノルムの収束する様子を示した．

次に，**表4** には，$h = 1/129$ を固定し，様々な Dh の値に対して，それぞれの算法の残差ノルムが収束条件を満たすまでの収束時間（秒）T と反復回数 I の値を示した．この実験では最大反復回数を2000回としたが，$Dh = 2^3, 2^4, 2^5$ のとき，BiCGStab 法は収束しなかった．また，$Dh = 2^5$ のとき GP-BiCG 法は収束はするものの，BiCGStab2 法のほぼ3倍，BiCGStab(2) 法のほぼ4倍の計算時間が必要であった．

表 4 例題 2 の収束時間（T sec）および反復回数（I）
Table 4 Example 2: Computational time (T sec) and iteration count (I).

Dh	2^{-2}		2^{-1}		2^{0}		2^{1}	
算法	T	I	T	I	T	I	T	I
BiCGStab	11.04	550	10.17	468	11.00	474	10.11	496
BiCGStab2	12.92	530	12.29	448	13.09	462	15.81	604
GP-BiCG	12.16	444	12.76	414	14.92	468	17.17	572
BiCGStab(2)	9.88	532	9.02	452	9.69	468	8.47	422
BiCGStab(4)	8.96	480	8.61	438	11.03	566	11.03	580
BiCGStab(8)	10.39	500	11.17	538	13.54	656	14.06	662

Dh	2^{2}		2^{3}		2^{4}		2^{5}	
算法	T	I	T	I	T	I	T	I
BiCGStab	21.05	954	—	—	—	—	—	—
BiCGStab2	15.09	566	12.18	472	13.87	510	13.64	680
GP-BiCG	19.06	634	23.48	796	34.03	1152	47.08	1328
BiCGStab(2)	10.81	548	8.94	468	9.84	500	10.26	792
BiCGStab(4)	11.25	588	9.21	488	8.67	462	9.36	504
BiCGStab(8)	13.40	644	11.35	548	9.97	476	10.07	486

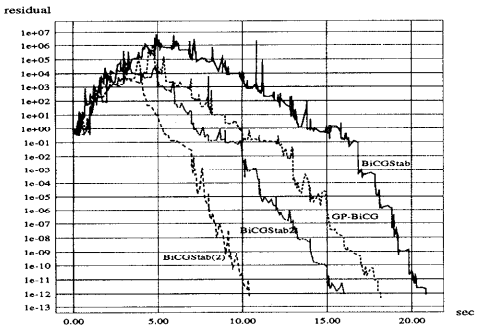


図 5 例題 2：計算時間に対する BiCGStab 系の残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子の比較（$Dh = 2^2$）
Fig. 5 Example 2: The behaviour of residual norm ($\|r_k\|_2/\|r_0\|_2$) v.s. computational time (sec) for the variants of BiCGStab method ($Dh = 2^2$).

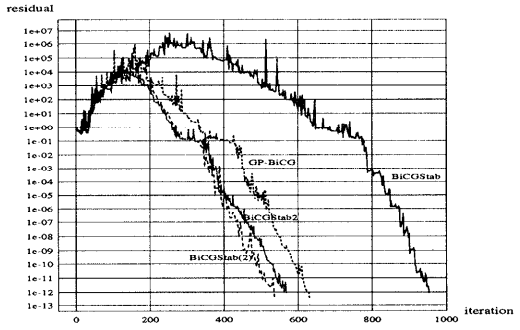


図 6 例題 2：反復回数に対する BiCGStab 系の残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子の比較（$Dh = 2^2$）
Fig. 6 Example 2: The behaviour of residual norm ($\|r_k\|_2/\|r_0\|_2$) v.s. number of iterations for the variants of BiCGStab method ($Dh = 2^2$).

例題 3 矩形領域 $\Omega = [0,1] \times [0,1]$ における下記の 2 階の偏微分方程式のディリクレ境界値問題を考える（Joubert [11] を参照）.

$$-u_{xx} - u_{yy} + D((y - \tfrac{1}{2})u_x(x,y) + (x - \tfrac{1}{3})(x - \tfrac{2}{3})u_y(x,y)) = G(x,y),$$
$$u(x,y)\,|_{\partial\Omega} = 1 + xy.$$

この方程式を例題 2 と同様に 5 点中心差分近似を用いて離散化し，真の解を $u(x,y) = 1 + xy$ と設定して右辺を決定し，数値実験を行った．使用した計算機とそのプロセッサ数，使用した解法および収束条件，初期近似解，メッシュの大きさに関しては，例題 2 と同じものを使用した．ただし，最大反復回数に関しては 6000 回とした．

表 5 に示した結果は，$h = 1/129$ を固定して Dh の値を変化させたときに，それぞれの算法の残差ノルムが収束条件を満たすまでの収束時間（秒）T および反復回数 I を比較したものである．なお，$Dh = 2^4$ のときの BiCGStab 法は 6000 回では収束しなかった．

また，特に $Dh = 2^2$ のときの BiCGStab 法，BiCGStab(2) 法，GP-BiCG 法，BiCGStab(4) 法の残差ノルムの収束の様子を**図 7** に示した．この実験では，いずれの場合においても BiCGStab(4) 法が最も早く収束していることが分かる．

4.3 BiCGStab(ℓ) どうしの比較

4.2 節で用いた偏微分方程式の境界値問題に対し ℓ の値を変えたときの BiCG-Stab(ℓ) どうしの残差ベクトルのノルムの収束の様子を比較した．$h = 1/129$，プロセッサ数は 64 とし，解法として BiCGStab(2) 法，BiCGStab(4) 法，BiCGStab(8) 法を用いて数値実験を行った．例題 2 の問題に対して $Dh = 2^2$ とし

表5 例題3：残差ノルムの収束時間（T sec）および反復回数（I）
Table 5 Example 3: Computational time (T) and Iteration count (I).

Dh	2^{-3}		2^{-2}		2^{-1}		2^{0}	
算法	T	I	T	I	T	I	T	I
BiCGStab	21.15	866	25.02	978	28.10	1098	32.81	1208
BiCGStab2	24.89	864	27.78	966	31.63	1080	35.19	1206
GP-BiCG	27.28	822	31.23	982	34.57	1054	40.18	1222
BiCGStab(2)	18.59	856	20.81	968	23.51	1088	25.89	1200
BiCGStab(4)	15.87	812	18.96	958	20.06	1038	23.40	1204
BiCGStab(8)	17.59	794	20.89	944	22.92	1040	26.66	1210

Dh	2^{1}		2^{2}		2^{3}		2^{4}	
算法	T	I	T	I	T	I	T	I
BiCGStab	36.67	1428	46.53	1778	78.08	3010	—	—
BiCGStab2	40.00	1390	49.73	1744	67.05	2358	107.45	3780
GP-BiCG	45.02	1404	57.04	1752	78.63	2450	133.58	4130
BiCGStab(2)	29.85	1388	36.97	1740	50.51	2356	78.88	3720
BiCGStab(4)	27.43	1404	35.18	1786	46.32	2378	71.00	3598
BiCGStab(8)	32.21	1472	41.13	1868	53.69	2428	79.85	3616

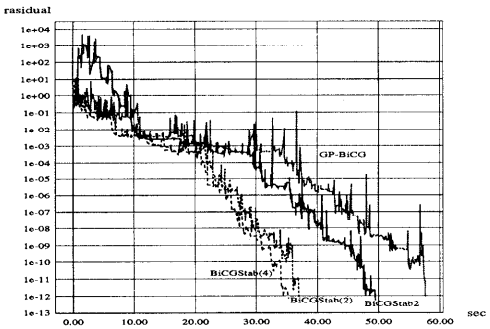


図7 例題3：計算時間（sec）に対するBiCGStab系の残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子の比較（$Dh = 2^2$）
Fig. 7 Example 3: The behaviour of residual norms ($\|r_k\|_2/\|r_0\|_2$) v.s. computational time (sec) for the variants of BiCGStab method ($Dh = 2^2$).

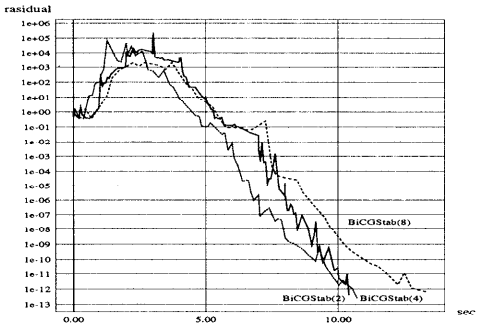


図8 例題2：計算時間に対するBiCGStab(ℓ)法（$\ell = 2, 4, 8$）の残差ノルム（$\|r_k\|_2/\|r_0\|_2$）の収束する様子の比較（$Dh = 2^2$）
Fig. 8 Example 2: The behaviour of residual norms ($\|r_k\|_2/\|r_0\|_2$) v.s. computational time (sec) for the BiCGStab(ℓ) method ($\ell = 2, 4, 8$), $Dh = 2^2$.

たときの結果を**図8**に示し，例題3の問題に対して$Dh = 2^1$としたときの結果を**図9**に示した．

BiCGStab(ℓ)法はℓの値が大きくなると，ベクトルの直交化に要する計算量だけが$O(\ell^2)$で増えていく．その他の部分の計算量は変わらない．そのため，計算量はBiCGStab(2)法が最も少なくなる．図8を見ると，BiCGStab(2)法の残差ノルムは，BiCGStab(4)法の残差ノルムを最終段階の一歩手前で追い抜き，最も早い計算時間で収束判定条件に到達していることが分かる．しかし，反復途中でのBiCGStab(4)法の残差ノルムは，BiCGStab(2)法の残差ノルムよりも少し早い収束をしていることは明らかである．また，図9では，BiCGStab(4)法の残差ノルムが最も早く収束している．表1の中に示した各算法の1回の反復に要する浮動小数点数のかけ算の回数を比べると，BiCGStab(2)法よりもBiCGStab(4)法の方が多くなっている．この実験の結果は明らかにこれに反している．これはそれぞれの算法の計算方法に原因がある．BiCGStab(ℓ)法は1度にℓ回分の反復をまとめて行っている．このとき，内積の計算や行列×ベクトルの計算など，プロセッサ間の同期をとって通信を行わなければならないものは，ℓ回の反復のはじめの部分に集中している．BiCGStab(2)法が2回分の反復のはじめの部分で同期をとるのに対して，BiCGStab(4)法は4回分の反復のはじめの部分で同期をとるだけでよい．つまり，BiCGStab(4)法の方がプロセッサ間で必要とする同期の回数が少なくなり，そのため計算が速くなったのである．

同様のことはBiCGStab(ℓ)法とその他の算法との比較においてもいえる．BiCGStab法，BiCGStab2

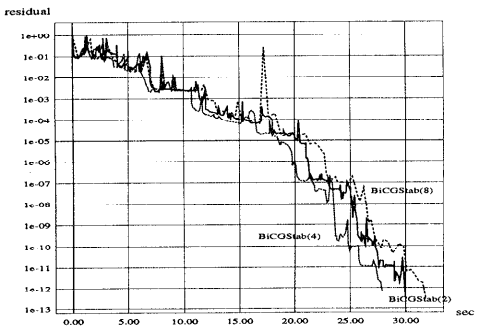


図 9 例題 3：計算時間に対する BiCGStab(ℓ) 法 ($\ell = 2, 4, 8$) の残差ノルム ($\|r_k\|_2/\|r_0\|_2$) の収束する様子の比較 ($Dh = 2^1$)

Fig. 9 Example 3: The behaviour of residual norms ($\|r_k\|_2/\|r_0\|_2$) v.s. computational time (sec) for the BiCGStab(ℓ) method ($\ell = 2, 4, 8$), $Dh = 2^1$.

法，GP-BiCG 法は内積や行列とベクトルとのかけ算などプロセッサ間の同期を必要とする計算が反復の中に一様に分布している．一方，BiCGStab(ℓ) 法は ℓ 回分の反復をまとめて行い，そのはじめの部分だけで同期をとってその他の部分ではプロセッサごとに別々の計算を行えばよい．つまり，BiCGStab 法，BiCGStab2 法，GP-BiCG 法は反復中でプロセッサどうしで同期をとる回数が BiCGStab(ℓ) 法に比べて多くなってしまい，計算が遅くなったのである．このため並列計算機 AP1000 上の数値実験では BiCGStab(ℓ) 法が良い結果を示した．

4.4 虚数部の大きな固有値を持つ行列

例題 4 次の行列 A を考える．これは Weiss[17] の文献で用いられている例題である．なお，行列の次元は 16384 とする．

$$A = \begin{pmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & A_n \end{pmatrix} \tag{21}$$

ただし，$j = 1, 2, \ldots, n$ に対して

$$A_j = \begin{pmatrix} \lambda_{re,j} & \lambda_{im,j} \\ -\lambda_{im,j} & \lambda_{re,j} \end{pmatrix} \tag{22}$$

この行列 A の固有値は

$$\lambda_j = \lambda_{re,j} \pm i\lambda_{im,j} \tag{23}$$

と計算される．$\lambda_{re,j}$，$\lambda_{im,j}$ の値はそれぞれ，$[\lambda_{re,\min}, \lambda_{re,\max}]$，$[\lambda_{im,\min}, \lambda_{im,\max}]$ の範囲内で乱数として生成するものとする．この行列に対し，真の解を $[0,1]$ 内の乱数として右辺を決定し，数値実験を行った．なお，解法として BiCGStab 法，BiCGStab2 法，BiCGStab(2)，BiCGStab(4)，BiCGStab(8) 法を用いた．各算法の初期近似解は零ベクトルとし，プロセッサ数は 64 個，最大反復回数は 4000 回とした．この問題に対し，λ_{im} の存在する範囲は $[-1, 1]$ に固定し，λ_{re} の値を様々に変えて数値実験を行い，その結果を**表 6** に示した．また，$\lambda_{re} \in [10^{-3}, 10^{-2}]$ のときの残差ノルムの収束の様子を図 **10** に示した．ここで用いた行列 A の固有値は実数部に比べて虚数部が非常に大きいという特徴を持っている．BiCGStab 法は 1 次の MR 多項式がこの虚数固有値に対応でなかったためにうまく収束できなかった．また，$\lambda_{im} \in [10^{-3}, 10^{-2}]$ としたときの GP-BiCG 法も定めた最大反復回数内では収束しなかった．

表 6 例題 4：残差ノルムの収束時間（T sec）および反復回数（I）
Table 6 Example 4: Computational time (T sec) and iteration count (I).

λ_{re}	$[10^{-2}, 10^{-1}]$		$[10^{-3}, 10^{-2}]$	
算法	T	I	T	I
BiCGStab	—	—	—	—
BiCGStab2	9.71	364	70.97	2546
GP-BiCG	13.08	422	—	—
BiCGStab(2)	7.59	368	53.38	2552
BiCGStab(4)	8.07	360	53.32	2352
BiCGStab(8)	9.73	368	60.12	2292

注) 4000 回の反復で収束しなかったものは "—" と記述した．

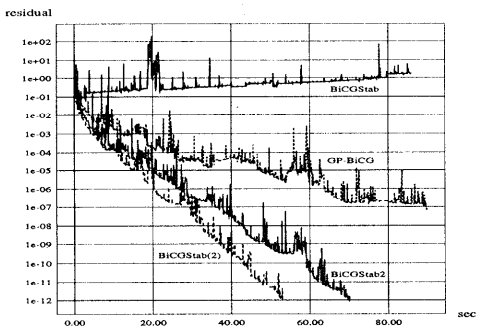


図 10 例題 4：計算時間に対する BiCGStab 系の残差ノルム ($\|r_k\|_2/\|r_0\|_2$) の収束する様子の比較 ($\lambda_{re} \in [10^{-3}, 10^{-2}]$)

Fig. 10 Example 4: The behaviour of residual norm ($\|r_k\|_2/\|r_0\|_2$) v.s. computational time (sec) for the variants of BiCGStab method ($\lambda_{re} \in [10^{-3}, 10^{-2}]$).

5. おわりに

分散メモリ型の並列計算機 AP1000 を用いて数値実験を行い，いくつかの BiCGStab 法に関連する算法の残差ノルムの収束性について比較検討を行った．

連立1次方程式(1)の係数行列 A が複素固有値を持ち，その固有値の虚数部分が実数部分より大きい値をとるような問題に対して，2次のMR多項式とBCG法の残差ベクトル $\tilde{r}_k$ を組み合わせ，その残差ノルム $\|r_k\|_2$ の最小化を行うBiCGStab(2)法，BiCGStab2法，GP-BiCG法は，有効な解法であるといえる．

GP-BiCG法[16]は，ランチョス過程に基づく積型反復法の理論から提案されたものであるが，BiCGStab法やBiCGStab2法を一般化したものと考えることもできる．例題1に示した複素固有値を持つような問題に対して残差ノルムが収束するのに必要な反復回数を比較すると，GP-BiCG法はBiCGStab2法やBiCGStab(2)法とほぼ同程度の反復回数で近似解に収束していることが分かる．しかし，この実験結果が示すようにGP-BiCG法の反復ごとに必要となる計算時間は，BiCGStab2法やBiCGStab(2)法よりも大きい．これは算法が1回の反復で必要とする計算量が多いためである．すなわち，変数 β，γ の決定に多くの内積計算を必要とするからである．GP-BiCG法は各反復における計算時間の面において，ここで述べた他の算法に劣る．また，各反復で内積計算が多いことは，桁落ちや丸め誤差の影響を受けやすく，例題1や例題4の実験結果（図3，4，10を参照）からも分かるように，たとえ残差ノルムが収束していても，ある時点から停滞，発散してしまう危険性を有する算法である．いうなれば，算法の実装をうまく行わないと，計算精度の面で多少問題を起こす可能性があると思われる．

例題1から例題4までの数値実験におけるBiCGStab法とBiCGStab(2)法の比較から明らかであるように，BiCGStab(2)法が良い結果を示すのは2次のMR多項式がうまく働くときである．逆に非常に行列の性質が良く，バランスの良い複素固有値を係数行列とする問題では，1次のMR多項式が十分効果的に働き，計算量の少ないBiCGStab法が良い結果を示すように思われる．

各算法が1回の反復に必要な浮動小数点数のかけ算の回数は，BiCGStab法が最も少ない．しかし，今回行った数値実験では収束に必要な計算時間の点において，BiCGStab法よりもBiCGStab(2)法が優れた結果を示している．また，$\ell = 2$ で十分な収束が得られない場合には，BiCGStab(ℓ)法の次数 ℓ をあげてみることも必要である．たとえば，例題3の結果を見ると，BiCGStab(4)法はBiCGStab(2)よりも計算時間の点において優れた収束を示している．

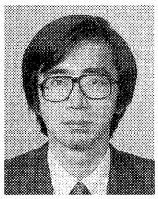
分散メモリ型の並列計算機AP1000を使用した今回の数値実験では，BiCGStab(ℓ)法は本稿で比較した算法の中で最も優れた反復解法であるといえる．

参考文献

1) Hestenes, M.R. and Stiefel, E.: Methods of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards*, Vol.49, pp.409–435 (1952).
2) Lanczos, C.: Solution of systems of linear equations by minimized iteration, *J. Res. Nat. Bur. Standard*, Vol.49, pp.33–53, (1952).
3) Fletcher, R.: Conjugate gradient methods for indefinite systems, *Lecture Notes in Math.*, Vol.506, pp.73–89 (1976).
4) 名取，野寺：大型疎行列計算における反復解法，情報処理，Vol.28, No.11, pp.1452–1459 (1987).
5) Nodera, T.: New variant of BCG method for solving non-symmetric systems, *Advances in Computer Methods for Partial Differential Equations*, IMACS, Vol.6, pp.130–135 (1987).
6) Sonneveld, P.: CGS, a fast Lanczos-type solver for nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.*, Vol.10, pp.36–52 (1989).
7) Freund, R.W. and Nachtigal, N.: QMR: A quasi-minimal residual method for non-Hermitian linear systems, *Numer. Math.*, Vol.60, pp.315–339 (1991).
8) van der Vorst, H.A.: Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems, *SIAM J. Sci. Stat. Comput.*, Vol.13, pp.631–644 (1992).
9) Nodera, T.: A note on BiCGSTAB algorithm, *SANUM*, Vol.18, pp.157–166 (1992).
10) Reichel, L. and Trefethen, L.N.: Eigenvalues and Pseudo-eigenvalues of Toeplitz Matrices, *Lin. Alg. Appl.*, Vol.162, pp.153–185 (1992).
11) Joubert, W.: Lanczos Methods for the solution of non-symmetric systems of Linear Equations, *SIAM. J. Matrix Anal.*, Vol.13, No.3, pp.926–943 (1992).
12) Gutknecht, M.H.: Variants of BiCGSTAB for matrices with complex spectrum, *SIAM J. Sci. Comput.*, Vol.14, pp.1020–1033 (1993).
13) Sleijpen, G.L.G. and Fokkema, D.R.: BiCGSTAB(ℓ) for linear equations involving unsymmetric matrices with complex spectrum, *ETNA*, Vol.1, pp.11 32 (1993).
14) Zhang, S.: GPBi-CG: Generalized product-type method based on Bi-CG for solving nonsymmetric linear systems, (Technical Report 9301, Nagoya University, Japan, 1993), *SIAM J. Sci. Comput.*, Vol.18, No.2, pp.516–536

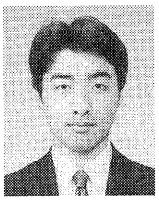
(1997).
15) Bruaset, A.M.: A survey of preconditioned iterative methods, *Pitman Research Notes in Math. Series*, No.328, Longman Science & Technical (1995).
16) 張，藤野：ランチョス・プロセスに基づく積型反復解法，日本応用数理学会論文誌，Vol.5, No.4, pp.345–360 (1995).
17) Weiss, R.: Properties of Generalized Conjugate Gradient Methods, *Numer. Lin. Alg. Appl.*, Vol.1, No.1, pp.45–63 (1994).
18) Schönauer, W. and Weiss, R.: An engineering approach to generalized conjugate gradient methods and beyond, *Appl. Numer. Math.*, Vol.19, No.3, pp.27–34 (1995).
19) Weiss, R.: A theoritical overview of Krylov subspace methods, *Appl. Numer. Math.*, Vol.19, No.3, pp.207–234 (1995).
20) 阿部，張，三井，杉浦：線型連立系に対する積型反復解法の加速多項式の評価，日本応用数理学会論文誌，Vol.6, No.4, pp.405–425 (1996).
21) 野口，稲津，野寺：BiCGStab(ℓ) 法の収束特性について，情報処理学会 HPC 研究会報告，Vol.96, No.22, pp.13–18 (1996).
22) 野口，野寺：BiCGStab(ℓ) 関連のアルゴリズムの収束特性について，PCW'95 Japan, P2-J (1996).
23) Gutknecht, M.H.: Lanczos-type solvers for nonsymmetric linear systems of equations, CSCS/SCSC Technical Report 97-04, ETH (1997).

(平成 9 年 1 月 17 日受付)
(平成 9 年 9 月 10 日採録)



野寺　隆（正会員）

1982 年慶應義塾大学大学院工学研究科博士課程（数理工学専攻）修了．現在，同大学助教授．その間，'86 年より 1 年間米国スタンフォード大学客員教授．大規模な行列計算の算法の研究開発に従事．ハイパフォーマンス・コンピューティングや文書処理に興味を持つ．著書に『楽々LaTeX』（共立出版）などがある．工学博士．エッセイスト．SIAM，日本応用数理学会会員．



野口雄一郎（学生会員）

1996 年慶應義塾大学理工学部数理科学科卒業．現在，同大学大学院理工学研究科数理科学専攻修士課程在学中．非線形現象の数値的解析に興味を持つ．