適応的に ℓ を変化させる BiCGStab(ℓ) 法

森屋 健太郎† 野寺 隆††

BiCGStab(ℓ) 法は，非対称の大型疎行列を係数とする連立1次方程式を解く算法の1つである．これは，BiCG 法の残差ベクトルに ℓ 次の MR 多項式（最小残差多項式ともいう）を掛けることで残差ノルムの収束を加速させたものである．一般に，MR 多項式の次数 ℓ の値が大きいと残差ノルムの収束は良くなるが，余分な計算時間を必要としてしまうのが難点である．算法がブレイクダウン（破綻ともいう）しそうになったときに，MR 多項式の次数 ℓ を変化させる算法が，Sleijpen ら[12]によって提案されているが，彼らの算法を従来の BiCGStab(ℓ) 法と比較すると，残差ノルムが収束するまでに余分な計算時間を必要とすることが多い．本稿では，ブレイクダウンが起こりそうなときと残差ノルムの収束が停滞したときの両方の場合に，MR 多項式の次数 ℓ を変化させる算法を提案する．最後に，この新しい算法を富士通の分散メモリ型並列計算機 AP3000 に実装し数値実験を行い，その算法の有効性について，従来の BiCGStab(ℓ) 法と比較検討を行う．

BiCGStab(ℓ) Method with Varying ℓ in Adaption

KENTARO MORIYA† and TAKASHI NODERA††

BiCGStab(ℓ) method is one of iterative methods to solve large and sparse nonsymmetric linear systems of equations. It stablizes the residual norm of BiCG (bi-conjugate gradient) method by adapting ℓ degree MR (minimal residual) polynomial. When the degree of MR polynomial ℓ becomes larger, the convergence of residual norm will be better. However, the computational cost will be increased. Sleijpen et al.[12] proposed the BiCGStab(ℓ) method with varying ℓ in case of a breakdown. However, the BiCGStab(ℓ) method based on their algorithm is often required more computational time than the conventional BiCGStab(ℓ) method. In this paper, it is both of when the convergence of residual norm stagnates and the breakdown is likely to occur, the BiCGStab(ℓ) method for varying ℓ in adaption is proposed. At last, this algorithm is implemented on the distributed memory machine Fujitsu AP3000 and the numerical examples are given.

1. はじめに

数理物理学や工学のあらゆる分野に現れる偏微分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，大型で疎な正則行列 A を係数とする連立1次方程式

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

が得られる．一般に，連立1次方程式 (1) の解法は，未知数の消去に基づく直接法と，適当な近似解から出発して真の解に近づく反復法がある[11]．直接法は大規模な疎行列を係数とする問題では，非常に大きな容量のメモリと計算量を必要としたり，フィルイン（行列の0要素の場所が消去により0でなくなってしまう）などの問題が生じることがあるので，反復法による解法が必要不可欠となる．

非定常な反復法では，式 (1) の係数行列 A が対称な正定値行列であれば，Hestenes ら[1]による CG 法（共役勾配法ともいう）が非常に有効な解法である．しかし，係数行列 A が正則で非対称である場合には，現在さまざまな算法が提案されているが，まだ決定的に有効と呼べる反復解法は提案されていない．

Lanczos[2]と Fletcher[3]によって提案された BiCG 法（双共役勾配法ともいう）は，行列 A が非対称の場合に，疑似残差ベクトル，疑似方向ベクトル，行列 A^T などを用いて，あたかも対称行列を係数とする連立1次方程式を CG 法によって解くような反復解法である[4]．したがって，算法も CG 法に似ており，実装に必要なメモリと計算量も少なくてすむ．しかし，BiCG 法は反復の途中でしばしばブレイクダウンを起こした

† 慶應義塾大学大学院理工学研究科
Graduate School of Science and Technology, Keio University
†† 慶應義塾大学理工学部
Faculty of Science and Technology, Keio University

り，残差ノルムが発散したり，停滞（stagnation）を起こしたりして良い精度の近似解を得られないことが多い．

BiCGStab 法は，Van der Vorst[5]によって提案されたもので，BiCG 法の残差ベクトルに 1 次の MR 多項式（最小残差多項式ともいう）を作用させた積型の反復解法である．この算法は BiCG 法の算法に比べ，良い収束性を示すことがある[5),6)]．しかし，係数行列 A の固有値に関して，複素固有値の虚部が実部よりも相対的に大きい場合には，残差ノルムの発散や収束の停滞が起こりやすい．それに対して Gutknecht[8]は，1 次の MR 多項式と 2 次の MR 多項式を反復過程で交互に適用する BiCGStab2 法を提案した．この算法は，BiCGStab 法を改良したものであるが，1 次の MR 多項式を適用する部分がやはり残差ノルムの収束に悪い影響を与えることがある．また，BiCGStab 法の算法に比べて内積の計算量が多いので，丸め誤差の影響を受けやすいことも難点である[13)]．

Sleijpen ら[9)]は，それらを一般化した BiCGStab(ℓ) 法[9),13)]を提案した．BiCGStab(ℓ) 法は，BiCG 法の残差ベクトルに ℓ 次の MR 多項式を組み合わせることで，残差ノルムの収束性を加速させたものであり，ブレイクダウンや残差ノルムの発散と収束の停滞に関して頑強な算法となっている．また，計算量もできるだけ少なくする工夫がなされている．一般に，ℓ が大きければ残差ノルムの収束性は安定して良くなるが，それに比例して $O(\ell^2)$ で直交化に要する計算量が増えてしまう．それゆえ，ℓ を適切な値に決定することは非常に重要である．反復の途中で MR 多項式の次数 ℓ を変化させる最も簡単な手法は Sleijpen ら[12)]によって提案されている．彼らの提案した算法では，BiCG 法の残差ベクトルと疑似残差ベクトルの内積が 0 に近い値をとるとき，ピボットブレイクダウンと呼び，MR 多項式の次数 ℓ を増大させることによって，ピボットが小さい値になることを防ぐものである．しかし，彼らの手法では，従来の BiCGStab(ℓ) 法に比べて，残差ノルムの収束に要する計算時間が増加することもある．

本稿では，ブレイクダウンを起こしそうなときと，残差ノルムの収束が停滞するときの両方に，BiCGStab(ℓ) 法の MR 多項式の次数 ℓ を適切な値に決定し，残差ノルムの収束を向上させる方法を提案する．

2 章では，BiCG 法，BiCGStab(ℓ) 法の算法を簡単に述べ，その特徴や性質などについて示す．3 章では，MR 多項式の次数 ℓ を変化させる方法について述べる．4 章では，本論文で述べる算法を AP3000 上に実装する方法について記述する．5 章では，3 章で提案した MR 多項式の次数 ℓ を変化させる BiCGStab(ℓ) 法を AP3000 上に実装し，従来の BiCGStab(ℓ) 法[9),12)] などと比較し，その数値実験の結果を示す．6 章では，実験結果について比較検討を行い結論を述べることにする．

2. BiCG 法と BiCGStab(ℓ) 法について

この章では BiCG 法と BiCGStab(ℓ) 法の算法の特徴について簡単に述べる．

2.1 BiCG 法

BiCG 法は，残差ベクトル，疑似残差ベクトル，探索ベクトル，疑似探索ベクトルのそれぞれ 4 つの漸化式によって構成されている．**図 1** に BiCG 法の実装について示す．ここで，r_k，u_k が残差ベクトルおよび探索ベクトルであり，$\bar{r}_k$，$\bar{u}_k$ が疑似残差ベクトルおよび疑似探索ベクトルである．係数 α_k，β_k については，残差ベクトルが双直交で，かつ探索ベクトルが双共役になるように定めている．疑似残差ベクトルと疑似探索ベクトルの漸化式を導入することで，行列 A が非対称であっても見かけ上は対称行列を係数に持つ連立 1 次方程式を CG 法で解くような算法である．したがって，非常に簡単な算法で計算量も少なくてすむが，しばしば残差ノルムの発散や収束の停滞が生じる．また，図 1 におけるスカラー ρ_k や γ_k が 0 に近い値をとるときにブレイクダウンを起こすことも知られている．これらの欠点のいくつかを改善させるために，ℓ 次の MR 多項式を用いることによって，BiCG 法の残差ノルムの収束を加速させるような積型の算法が次に述べる BiCGStab(ℓ) 法である．

2.2 BiCGStab(ℓ) 法

BiCG 法を ℓ 回反復させることで得られた残差ベクトルと ℓ 次の MR 多項式との積をつくって，この新たな

(1) **Choose** an initial guess x_0, and $\bar{r}_0$ is an arbitrary vector such that $\bar{r}_0 = \bar{u}_0 \neq 0$. **Compute** $r_0 = u_0 = b - Ax_0$.

(2) **repeat until** $\| r_k \|$ **is small enough**

for $k = 1, 2, 3, \ldots$

$\alpha_k = (r_{k-1}, \bar{r}_{k-1})/(Au_{k-1}, \bar{r}_{k-1})$
$r_k = r_{k-1} - \alpha_k A u_{k-1}$
$\bar{r}_k = \bar{r}_{k-1} - \alpha_k A^T \bar{u}_{k-1}$
$\beta_k = -(r_k, \bar{r}_k)/(r_{k-1}, \bar{r}_{k-1})$
$u_k = r_k - \beta_k u_{k-1}$
$\bar{u}_k = \bar{r}_k - \beta_k \bar{u}_{k-1}$
$x_k = x_{k-1} + \alpha_k u_{k-1}$

endfor

図 1 BiCG 法
Fig. 1 BiCG method.

```
(1) choose an initial guess x_0, and compute r_0 = b - Ax_0. Take r̄_0 = r_0,  u_0 = 0,
    ρ_0 = 1,  α = 0,  ω = 1.
(2) repeat until ‖ r_k ‖ is small enough
  for k = ℓ, 2ℓ, 3ℓ, ...
      ⟨⟨ BiCG process ⟩⟩
      û_0 = u_{k-ℓ},  r̂_0 = r_{k-ℓ},  x̂_0 = x_{k-ℓ}
      ρ_0 = -ωρ_0
      for j = 0 to ℓ - 1 do
          ρ_1 = (r̂_j, r̄_0),  β_{k+j} = αρ_1/ρ_0,  ρ_0 = ρ_1
          for i = 0 to j do
              û_i = r̂_i - β_{k+j} û_i
          endfor
          û_{j+1} = A û_j
          γ = (û_{j+1}, r̄_0),  α_{k+j} = ρ_0/γ
          for i = 0 to j do
              r̂_i = r̂_i - α_{k+j} û_{i+1}
          endfor
          r̂_{j+1} = A r̂_j,  x̂_0 = x̂_0 + α_{k+j} û_0
      endfor
      ⟨⟨ end of BiCG process ⟩⟩
      ⟨⟨ MR polynomial process ⟩⟩
      for j = 1 to ℓ do
          for i = 1 to j - 1 do
              τ_ij = (r̂_j, r̂_i) / ρ_i
              r̂_j = r̂_j - τ_ij r̂_i
          endfor
          ρ_j = (r̂_j, r̂_j),  γ'_j = (r̂_0, r̂_j) / ρ_j
      endfor
      γ_ℓ = γ'_ℓ,  ω = γ_ℓ
      for j = ℓ - 1 to 1 do
          γ_j = γ'_j - Σ_{i=j+1}^{ℓ} τ_ji γ_i
      endfor
      for j = 1 to ℓ - 1 do
          γ''_j = γ_{j+1} + Σ_{i=j+1}^{ℓ-1} τ_ji γ_{i+1}
      endfor
      x̂_0 = x̂_0 + γ_1 r̂_0,  r̂_0 = r̂_0 - γ'_ℓ r̂_ℓ,  û_0 = û_0 - γ_ℓ û_ℓ
      for j = 1 to ℓ - 1 do
          û_0 = û_0 - γ_j û_j,  x̂_0 = x̂_0 + γ''_j r̂_j,  r̂_0 = r̂_0 - γ'_j r̂_j
      endfor
      u_k = û_0,  r_k = r̂_0,  x_k = x̂_0
      ⟨⟨ end of MR polynomial process ⟩⟩
  endfor
```

図 2 BiCGStab(ℓ) 法
Fig. 2 BiCGStab(ℓ) method.

残差ノルムを局所的に最小にする算法が BiCGStab(ℓ) 法である．一般に，$k = \ell + m\ell$（ただし，m は非負の整数）のとき k 回目における BiCGStab(ℓ) 法の残差ベクトルは

$$r_k = p_m(A) q_{m\ell}(A) \tilde{r}_k \qquad (2)$$

とおくことができる．ただし，

$$q_{m\ell}(A) = p_{m-1}(A) p_{m-2}(A) \ldots p_0(A) \qquad (3)$$

であり，$\tilde{r}_k$ は k 回目の反復における BiCG 法の残差ベクトルとする．また，$p_i(A)\,(1 \le i \le m)$ はいずれも ℓ 次の MR 多項式である．**図 2** に BiCGStab(ℓ) 法の算法を示した．ここでは BiCGStab(ℓ) 法の詳しい説明は省略するが，詳細は文献 9), 13) を参照してほしい．

3. ℓ を変化させる方法

本章では，BiCGStab(ℓ) 法で用いる MR 多項式の次数 ℓ を変化させる方法について提案する．

3.1 ピボットによる ℓ の変化法

BiCG 法や BiCGStab(ℓ) 法において，算法が不安定になってブレイクダウンを起こす原因となるのは，残差ベクトルと疑似残差ベクトルの内積が 0 に近い値をとることである．これは多項式の最大次の係数が小さくなったり，誤差が大きくなったりするためである[12]．したがって，この値が小さくなったときに，MR 多項式の次数 ℓ を増加させて算法を安定させればよい．この算法は Sleijpen ら[12] で提案されており，算法の詳細はそちらの文献を参照してほしい．ここでは実装方法についてのみ示すことにする．

まず，BiCGStab(ℓ) 法の初期残差ベクトルと k 回目の反復における残差ベクトルの内積を正規化したものをピボットと定義し，次のように表すことにする．

$$\sigma_k = (r_k, r_0) / (\| r_k \| \cdot \| r_0 \|) \qquad (4)$$

このピボット σ_k は，BiCG 法の残差ベクトルと疑似残差ベクトルの内積を正規化したものと同値である[12]．

MR 多項式の次数 ℓ は最初 1 に設定して反復を行い，$\sigma_k < \varepsilon$ かつ $\ell < \ell_{max}$ なら ℓ を 1 ずつ増やしていく．ただし，ℓ_{max} は ℓ のとりうる最大値である．ε は計算機の誤差限界の平方根をとる値である．これは，内積計算で 2 乗を計算したときに，誤差限界が平方根をとった値になるためである．たとえば Sun の SPARC チップを搭載したワークステーションなどのように，倍精度の浮動小数点演算が 64 ビットで行われる計算機であるなら，誤差限界はほぼ $O(10^{-16})$ であるから，ε の値としては 1.0×10^{-8} の前後が妥当な値である．

なお，ピボット σ_k を計算するのには内積計算が必要であるが，この値は図 2 の BiCGStab(ℓ) 法における BiCG 法の過程で

$$\rho_1 = (\hat{r}_k, \bar{r}_0) \tag{5}$$

という部分を使えばよい．したがって，ピボットを求めるのに新たに内積計算を行う必要はない．

本稿では，このような算法を取り入れた BiCGStab(ℓ) 法を P-BiCGStab(1 : ℓ_{max}) 法と記述する．さらに，5 章の数値実験では，ℓ の初期値が 2 である P-BiCGStab(2 : ℓ_{max}) 法についても取り扱う．

3.2 残差ノルムの収束停滞時における ℓ の変化法

連立 1 次方程式における係数行列の条件数が悪いと，残差ノルムの収束が停滞することがある．そのような場合に ℓ を一時的に大きな値にして，残差ノルムの収束の停滞が回避できたらまた元の数値に戻すことを考える．具体的には，最初に $\ell = \ell_{min}$ と決め，通常はこの値を使って BiCGStab(ℓ) 法の反復を行う．そして，ある一定の反復回数において残差ノルムの値に変化がみられないときに，残差ノルムの収束が停滞したと見なして ℓ を最大値 $\ell = \ell_{max}$ にあげる．そして，残差ベクトルの停滞を回避できたら，また元の $\ell = \ell_{min}$ の値に戻して反復を続ける．

残差ノルムの収束の停滞を判定する基準としては，

$$\frac{\left| \| r_k \| - \| r_{k-\ell} \| \right|}{\| r_k \|} < \delta \tag{6}$$

を利用する．式 (6) の左辺は，k 回目と $k-\ell$ 回目の反復における BiCGStab(ℓ) 法の相対残差ノルムを表し，これが δ よりも小さくなったら，残差ノルムに変化がないと判定する．そして，式 (6) を連続してある一定の回数だけ満たしたら，残差ノルムの収束が停滞したと見なして ℓ を変化させる．そして，式 (6) の左辺の相対残差ノルムの値が再び右辺の δ 以上になったらまた $\ell = \ell_{min}$ に戻す．

3.3 ハイブリッド法

本稿で提案する方法は，3.1 節と 3.2 節で述べた方法を組み合わせて利用することである．つまり，ピボットのブレイクダウンが発生した場合と残差ノルムの収束が停滞した場合の両方の要因から ℓ を変化させることを考える．ただし，本稿ではピボットがブレイクダウンしたときの ℓ の変化法は，3.1 節で述べた手法では行わず，次に述べる方法で行うものとする．まず最初に，ℓ の通常値を ℓ_{min} としておいて，ブレイクダウンを起こしたときに ℓ を ℓ_{max} に変化させる．そして，ブレイクダウンが回避できたら，ℓ をまた ℓ_{min} に戻す．つまり，$\sigma_k \geq \varepsilon$ のときは $\ell = \ell_{min}$ とし，$\sigma_k < \varepsilon$ のときは $\ell = \ell_{max}$ とする．このようなブレイクダウンによる ℓ の変化法と 3.2 節で述べた残差ノルムの収束の停滞が発生したときにおける ℓ の変化法とを組み合わせて利用する．両方を組み合わせた方法を**図 3** に示す．

```
w_k = | ‖r_k‖ − ‖r_{k−ℓ}‖ | / ‖r_k‖
σ_k = (r_k, r_0)/(‖r_k‖ ‖r_0‖)

if w_k < δ and ℓ = ℓ_min then
    count = count + 1
endif
if w_k > δ and ℓ = ℓ_min then
    count = 0
endif

if count = stag and ℓ = ℓ_min then
    ℓ = ℓ_max
endif
if σ_k < ε and ℓ = ℓ_min
    ℓ = ℓ_max
endif
if w_k ≥ δ and ℓ = ℓ_max and σ_k ≥ ε then
    ℓ = ℓ_min ,   count = 0
endif
```

図 3 ブレイクダウンと残差ノルム停滞による ℓ の変化法
Fig. 3 BiCGStab(ℓ) method varying ℓ in adaption by the breakdown and the stagnation of residual norm.

図 3 における変数 *count* は残差ノルムが変化しない回数を数えるパラメータである．連続で $w_k < \delta$ を満たすとき *count* は増加する．そして，$count = stag$ になったときに残差ノルムの収束が停滞したと見なして $\ell = \ell_{max}$ とする．また，ブレイクダウンを起こしそうなとき，つまり $\sigma_k < \varepsilon$ となるときにも $\ell = \ell_{max}$ とする．そして，$w_k \geq \delta$ かつ $\sigma_k \geq \varepsilon$ となって，残差ノルムの収束の停滞とブレイクダウンのいずれも回避できたら，また元の $\ell = \ell_{min}$ に戻す．なお，*stag* のことを残差ノルムの収束に関する停滞判定回数と呼ぶことにする．

図 3 の算法を図 2 で示した BiCGStab(ℓ) 法における "⟨⟨ **end of MR polynomial process** ⟩⟩" の直後に導入する．つまり，BiCGStab(ℓ) 法のベクトル u_k, r_k, x_k を更新した直後に，この算法が毎回実行されることになる．

表1 AP3000の仕様
Table 1 Specification of AP3000.

プロセッサ間のネットワーク	内部ネットワーク（AP-Net 200 MB/秒×双方向） 同期ネットワーク
セルプロセッサ	UltraSPARC IU+FPU
セルキャッシュ	512 KB
セルローカルメモリ	256 MB

次に，各パラメータの決定法について述べることにする．野寺ら[13),14)]によると，多くの問題において，残差ノルムの収束に要する計算時間が短くなるのは $\ell = 2$ のときなので，ℓ_{min} の値は2に設定する．また，次章の数値実験の結果から，δ は0.10の前後が適切であり，残差ノルムの収束に関する停滞判定回数については，5回から20回まで5ずつ変化させたところ15回が最も適切であることが確認された．

本稿では，このような新しい BiCGStab(ℓ) 法のことを PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法と呼ぶことにする．ここで，MR多項式の次数 ℓ の通常値が ℓ_{min} であり，ℓ を変化させる最大の値が ℓ_{max} である．

4. 算法の AP3000 への実装

本稿で述べた算法を AP3000 に実装するのに際し並列化したのは，ベクトルどうしの和，ベクトルのスカラー倍，ベクトルの内積，行列とベクトルの積である．

行列およびベクトルの要素は，各プロセッサに均等に割り当てた．連立1次方程式の次元が n でプロセッサの数が p （n は p で割り切れる数とする）だとすると，1番目のプロセッサは，ベクトルの1行目から $\tilde{n}$ 行目までの要素を担当する．ただし，$\tilde{n} = n/p$ である．2番目のプロセッサは，$\tilde{n}+1$ 行目から $2\tilde{n}$ 行目までの要素を担当する．同様にして，i 番目のプロセッサは，$\tilde{n}(i-1)+1$ 行目から $\tilde{n}i$ 行目の要素を担当することになる．行列の場合もベクトルの場合と同様に，各プロセッサが上から順番に $\tilde{n}$ 行 n 列の長方行列を担当する．

ベクトルどうしの加算とベクトルのスカラー倍については，プロセッサ間で通信をすることなく，各プロセッサで独立に計算することができる．内積計算は，まず各プロセッサが担当するベクトルの局所的な内積を計算する．次に各プロセッサの持つ局所的な内積をグローバル通信によって足し合わせる．行列とベクトルの積は，行列が疎であるという性質を利用して，両隣のプロセッサのみから必要なデータを通信で受け取ればよい．

5. 数値実験

数値実験はすべて富士通の分散メモリ型並列計算機 AP3000 （セル台数16個）によって行った．**表1** に AP3000 の性能を示す．

また，PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法に関しては，ℓ の通常の値 ℓ_{min} を2に設定して，残差ノルムの収束が停滞したときに，それぞれ4，6，8に変化させる方法の3通りについて数値実験を行った．これらの算法を従来の BiCGStab(ℓ) 法および P-BiCGStab(1 : ℓ_{max}) 法，P-BiCGStab(2 : ℓ_{max}) 法と比較した．

なお，特に指定をしない限り数値実験は以下の条件で行った．

- 収束条件：$\| r_k \| / \| r_0 \| < 1.0 \times 10^{-12}$
- 最大反復回数：6000 回
- 初期近似解：$x_0 = [0, 0, \ldots, 0]^T$
- 計算精度：倍精度
- ε ：1.0×10^{-8}
- 時間計測：各実験について5回計測を行い，それらの平均値を採用した．

計算結果は，収束条件を満たすまでにかかった時間と反復回数を示した．また，反復回数はクリロフ部分空間が1つ進むことを1回の反復と数えることにする．たとえば，BiCGStab(4) 法は，BiCGStab 法の反復を4回行ったと数えることにする．

［数値例1］

正方領域 $\Omega = [0,1] \times [0,1]$ における2階の偏微分方程式の境界値問題を考える[7)]．

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

ただし，厳密解を $u(x,y) = 1 + xy$ と定め，メッシュ幅を $h = 1/513$ として，5点中心差分を用いて離散化した．得られた連立1次方程式の次元は262144となる．まず，この問題における PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法の δ と残差ノルムの収束に関する停滞判定回数を変化させて収束条件を満たすまでに要する計算時間（秒）と反復回数を計測した．その結果を**表2** に示す．3つの δ および4つの $stag$ のうち，最も計算時間が短縮された箇所にアンダーラインを引いた．表2から，$Dh = 2^{-1}$ の $\ell_{max} = 6$，$Dh = 2^{1}$ の $\ell_{max} = 8$ および $Dh = 2^{2}$ の $\ell_{max} = 6, 8$ の4通りを除いたすべての場合において，停滞判定回数が15

表 2 数値例 1 における適切な δ および停滞判定回数
Table 2 The appropriate judgement of iteration counts for the stagnation of residual norm and δ in example 1.

($\delta = 0.20$ の場合)

Dh		2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
ℓ_{max}	$stag$	time	iter	time	iter	time	iter	time	iter	time	iter
4	5	82.914	914	88.916	1014	104.334	1160	118.136	1256	105.593	1122
	10	82.972	926	87.663	994	104.949	1156	116.147	1256	105.432	1122
	15	82.936	936	87.839	994	104.693	1156	117.201	1256	104.986	1122
	20	84.461	954	87.561	994	105.277	1156	117.440	1256	104.901	1122
6	5	90.205	942	107.682	1120	116.848	1196	127.750	1224	136.004	1250
	10	88.224	946	114.372	1172	116.529	1212	129.105	1224	135.786	1250
	15	89.099	952	109.787	1140	118.076	1228	128.893	1224	136.344	1250
	20	91.541	978	108.787	1140	115.642	1208	128.453	1224	136.122	1250
8	5	111.669	1066	130.503	1246	164.467	1292	148.322	1286	145.400	1238
	10	111.158	1078	121.860	1182	132.829	1266	154.444	1330	144.104	1238
	15	105.755	1054	124.165	1218	132.986	1276	154.964	1330	144.723	1238
	20	105.148	1034	123.306	1198	133.957	1276	154.030	1330	145.011	1238

($\delta = 0.10$ の場合)

Dh		2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
ℓ_{max}	$stag$	time	iter	time	iter	time	iter	time	iter	time	iter
4	5	92.066	942	87.251	986	104.337	1166	114.717	1256	102.909	1122
	10	92.136	946	87.449	994	104.661	1156	114.342	1256	102.301	1122
	15	81.809	936	87.196	994	103.871	1156	113.902	1256	102.037	1122
	20	82.509	936	87.474	994	104.167	1156	114.091	1256	102.671	1122
6	5	89.596	976	107.487	1140	115.627	1234	125.574	1224	129.328	1250
	10	92.121	1010	107.264	1136	113.281	1216	125.585	1224	133.007	1250
	15	89.007	962	107.010	1140	112.576	1208	124.126	1224	131.614	1250
	20	90.462	962	106.373	1138	113.314	1208	124.211	1224	129.627	1250
8	5	105.167	1058	126.466	1238	134.199	1310	148.096	1322	143.077	1238
	10	101.198	1020	118.484	1200	129.987	1276	149.413	1330	143.148	1238
	15	99.134	1012	117.402	1198	129.935	1276	149.357	1330	147.711	1238
	20	99.912	1012	118.490	1198	131.021	1276	150.884	1330	178.518	1238

($\delta = 0.05$ の場合)

Dh		2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
ℓ_{max}	$stag$	time	iter	time	iter	time	iter	time	iter	time	iter
4	5	84.621	930	88.970	986	105.565	1156	116.998	1256	108.389	1122
	10	83.723	936	90.152	994	105.623	1156	118.690	1256	103.304	1122
	15	83.209	936	88.702	994	107.235	1156	116.339	1256	104.160	1122
	20	84.209	936	90.877	994	104.499	1156	118.982	1256	105.865	1122
6	5	97.904	1022	109.900	1152	114.949	1194	127.037	1224	133.134	1250
	10	91.201	962	107.640	1136	116.245	1208	127.305	1224	133.182	1250
	15	91.072	962	107.913	1140	116.166	1208	127.848	1224	132.179	1250
	20	91.492	962	107.606	1138	114.983	1208	127.410	1224	133.859	1250
8	5	106.948	1050	126.672	1256	132.649	1276	154.439	1330	148.324	1238
	10	102.256	1012	121.539	1198	132.854	1276	152.323	1330	146.083	1238
	15	101.889	1012	123.043	1198	136.729	1276	154.235	1330	148.301	1238
	20	102.120	1012	123.601	1198	132.294	1276	154.045	1330	148.943	1238

回で δ が 0.10 のときに最も残差ノルムの収束に要する計算時間が少なくなることが確認できた．さらに，3 つの ℓ_{max} の値の中で，最も計算時間が少なかった場合である $\ell_{max} = 4$ の場合について，停滞判定回数を 15 回に固定して，δ を 0 から 0.20 まで 0.025 ずつ変化させたときの計算時間を図 **4** に示した．図 4 からも δ が 0.10 のときに計算時間が最も短縮されると確認された．したがって，この数値例における停滞判定回数は 15 回，δ の値は 0.10 と設定した．

次に，残差ノルムが収束判定条件を満たすまでに要した計算時間と反復回数を各々の算法について比較した．その結果を**表 3** に示す．P-BiCGStab(1 : ℓ_{max}) 法は，ℓ_{max} が 2 のときに BiCGStab(2) 法と，$\ell_{max} = 4$

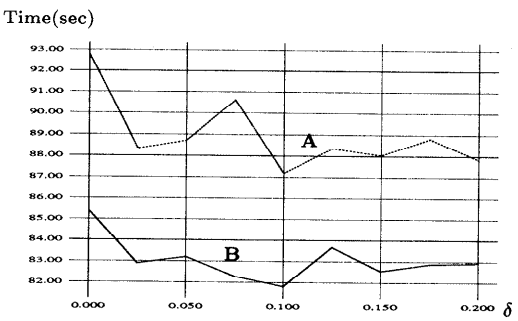
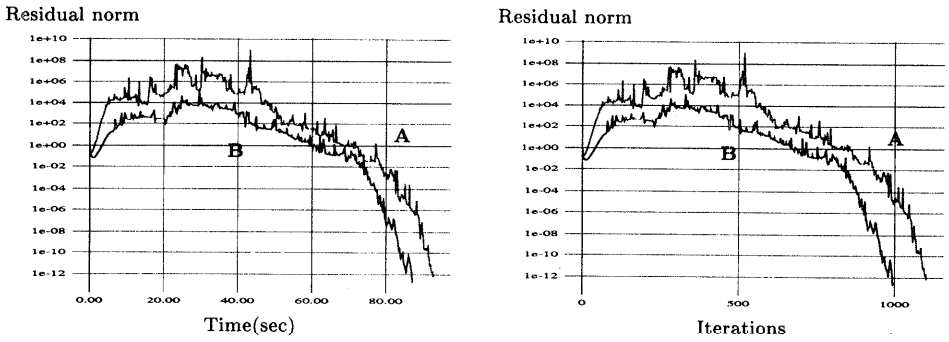


図 4 数値例 1 における計算時間と δ の関係 ($\ell_{max} = 4$, $stag = 15$), A: $Dh = 2^{-1}$, B: $Dh = 2^{-2}$
Fig. 4 The relation between computational time and δ for example 1 ($\ell_{max} = 4$, $stag = 15$), A: $Dh = 2^{-1}$, B: $Dh = 2^{-2}$.

表 3 数値例 1 の結果 (time：時間 (秒), iter：反復回数)
Table 3 Numerical results of the example 1 (time: computational time (second), iter: iteration count).

Dh	2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
算法	time	iter	time	iter	time	iter	time	iter	time	iter
BiCGStab	82.868	968	91.826	1108	99.094	1166	127.085	1600	175.935	2219
BiCGStab(2)	85.399	986	92.801	1102	100.709	1178	98.091	1152	107.110	1158
BiCGStab(4)	94.871	948	123.011	1240	131.605	1316	117.028	1164	111.256	1120
P-BiCGStab(1 : 2)	84.086	944	92.761	1063	110.902	1225	98.634	1106	107.151	1174
P-BiCGStab(1 : 4)	88.978	926	113.932	1172	112.597	1176	110.249	1122	109.982	1107
P-BiCGStab(1 : 6)	107.548	967	139.237	1231	140.437	1253	140.074	1241	135.977	1183
P-BiCGStab(1 : 8)	134.996	1048	174.494	1370	169.038	1290	170.504	1296	166.326	1256
P-BiCGStab(2 : 4)	92.888	977	114.016	1169	111.062	1157	114.863	1133	101.674	1093
P-BiCGStab(2 : 6)	109.477	978	143.670	1274	147.878	1308	140.479	1228	129.400	1148
P-BiCGStab(2 : 8)	133.701	1051	174.519	1343	173.184	1323	160.557	1231	160.092	1219
PSR-BiCGStab(2 : 4)	81.809	936	87.196	994	103.871	1156	113.902	1256	102.037	1122
PSR-BiCGStab(2 : 6)	89.197	962	107.010	1140	112.576	1208	124.126	1224	131.614	1250
PSR-BiCGStab(2 : 8)	100.134	1012	117.802	1198	129.935	1276	149.357	1330	147.711	1238
< PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法の ℓ の変化回数 >										
PSR-BiCGStab(2 : 4)	77		97		123		201		186	
PSR-BiCGStab(2 : 6)	31		88		68		75		150	
PSR-BiCGStab(2 : 8)	44		54		74		83		87	



(i) 計算時間 v.s. 残差ノルム **(ii)** 反復回数 v.s. 残差ノルム

図 5 数値例 1 に関する残差ノルムの収束する様子 ($Dh = 2^{-1}$), A: BiCGStab(2), B: PSR-BiCGStab(2 : 4)
Fig. 5 The convergence behavior of residual norm for example 1 ($Dh = 2^{-1}$), A: BiCGStab(2), B: PSR-BiCGStab(2 : 4).

表 4 数値例 1 の PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法における ℓ の変化回数 ($Dh = 2^{-1}$)
Table 4 The number of changing ℓ in PSR-BiCGStab(ℓ_{min} : ℓ_{max}) method for example 1 ($Dh = 2^{-1}$).

	ℓ の変化の要因	出現回数		
		PSR-BiCGStab(2:4)	PSR-BiCGStab(2:6)	PSR-BiCGStab(2:8)
(1)	ブレイクダウンによるもの	97	88	54
(2)	残差ノルムの収束停滞によるもの	1	0	0
(3)	(1) と (2) の両方によるもの	1	0	0

のときに BiCGStab(4) 法とそれぞれ同じ程度の性能であった．しかし，ℓ_{max} が 6 以上のときはいずれの BiCGStab(ℓ) 法と比べても，計算時間が 1 割から 5 割程度増大してしまった．また，P-BiCGStab(2 : ℓ_{max}) 法は，$\ell_{max} = 2$ のとき BiCGStab(4) 法に比べて計算時間が 1 割から 2 割程度短縮された．しかし，ℓ_{max} が 6 以上のときは BiCGStab(4) 法よりも計算時間が 1 割から 5 割程度増大してしまった．一方，PSR-BiCGStab(2 : 4) 法では，Dh が 2^{-2}, 2^{-1} および 2^{2} の場合に BiCGStab(2) 法と比べて 1 割程度の計算時間の短縮がみられた．その残差ノルムの収束傾向を**図 5** に示す．さらに，$Dh = 2^{-1}$ のときの PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法における ℓ の変化回数を**表 4** に示す．この例の場合には，ℓ の変化の大部分はブレイクダウンによるもので，残差ノルムの収束停滞による変化は，PSR-BiCGStab(2 : 4) 法の場合に 1 回だけしかみられなかった．また，PSR-BiCGStab(2 : 6) 法と PSR-BiCGStab(2 : 8) 法は，Dh が 2^{-2} か

表5 数値例2における適切な δ および停滞判定回数
Table 5 The appropriate judgement of iteration counts for the stagnation of residual norm and δ in example 2.

($\delta = 0.20$ の場合)

Dh		2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
ℓ_{max}	$stag$	time	iter	time	iter	time	iter	time	iter	time	iter
4	5	20.507	938	21.731	1012	26.695	1268	31.329	1484	42.326	1892
	10	19.918	930	22.347	1050	26.418	1244	30.700	1462	42.697	1908
	15	20.422	942	21.359	1002	26.877	1236	30.910	1492	42.715	1946
	20	20.260	940	21.418	1002	26.248	1236	29.957	1457	42.662	1906
6	5	22.357	940	23.064	1002	28.238	1250	34.012	1532	45.524	1960
	10	21.885	936	23.354	1000	28.148	1236	31.818	1484	44.669	1960
	15	21.491	932	22.732	1004	27.412	1232	31.157	1462	43.896	1918
	20	21.527	934	22.873	1022	28.917	1246	32.027	1490	46.741	1960
8	5	24.656	932	24.608	1020	31.896	1304	38.867	1510	47.574	1930
	10	24.344	932	25.726	1032	30.774	1296	38.768	1528	46.412	1934
	15	24.327	946	24.060	1024	29.876	1288	38.817	1518	46.197	1962
	20	23.764	928	24.329	1046	30.116	1278	38.035	1534	46.841	1932

($\delta = 0.10$ の場合)

Dh		2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
ℓ_{max}	$stag$	time	iter	time	iter	time	iter	time	iter	time	iter
4	5	20.142	908	22.318	982	27.408	1260	30.642	1448	41.888	1896
	10	19.146	912	21.736	1012	26.354	1214	32.017	1480	41.920	1924
	15	19.089	947	21.539	1084	24.860	1285	28.768	1475	41.697	1930
	20	21.208	928	22.430	1028	27.306	1292	30.179	1434	41.936	1920
6	5	22.634	928	23.498	1002	28.579	1264	34.249	1502	43.956	1916
	10	21.986	942	22.348	984	27.673	1246	31.967	1464	44.302	1922
	15	20.478	964	21.639	1022	25.899	1256	31.057	1456	43.976	1892
	20	21.970	928	24.309	1058	28.422	1276	32.629	1496	43.457	1912
8	5	24.215	908	24.749	1026	31.958	1278	39.542	1564	46.587	1920
	10	23.999	946	26.062	1064	29.149	1278	34.803	1340	46.007	1984
	15	23.603	940	25.501	1028	31.596	1220	38.214	1488	45.617	1960
	20	24.768	944	25.418	1046	29.842	1274	34.257	1462	46.017	1944

($\delta = 0.05$ の場合)

Dh		2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
ℓ_{max}	$stag$	time	iter	time	iter	time	iter	time	iter	time	iter
4	5	20.791	938	21.809	1012	27.734	1268	31.044	1484	42.217	1892
	10	20.977	930	22.458	1050	26.793	1244	30.446	1462	42.747	1908
	15	21.151	942	21.748	1002	26.688	1236	31.147	1492	43.604	1946
	20	21.142	940	21.379	1002	26.268	1236	30.337	1452	42.470	1906
6	5	21.834	924	23.659	1018	28.852	1270	33.317	1512	44.609	1944
	10	21.111	922	23.841	1032	28.558	1254	33.837	1528	43.948	1916
	15	24.591	946	23.158	1008	28.169	1254	32.323	1486	45.221	1926
	20	21.869	936	23.061	1008	28.142	1254	32.851	1486	43.910	1922
8	5	24.277	936	25.375	1034	31.056	1292	36.231	1530	46.910	1922
	10	23.806	944	25.860	1062	29.890	1272	34.317	1546	47.686	1990
	15	23.658	956	23.809	1024	31.168	1298	35.497	1532	46.322	1918
	20	24.098	952	24.788	1024	29.660	1262	34.360	1532	46.255	1918

ら 2^0 のときに，BiCGStab(4) 法に比べて計算時間を1割程度短縮させることができた．

[数値例2]

正方領域 $\Omega = [0,1] \times [0,1]$ における2階の偏微分方程式の境界値問題を考える[7]．

$$-u_{xx} - u_{yy} + D\{(y-1/2)u_x + (x-2/3)(x-1/3)u_y\} = f(x,y)$$

$$u(x,y)|_{\partial\Omega} = 1 + xy$$

ただし，数値例1と同様に，厳密解を $u(x,y) = 1+xy$ と定め，メッシュ幅を $h = 1/257$ として，5点中心差分を用いて離散化した．得られた連立1次方程式の次元は65536である．まず，数値例1と同様にして適切な δ と残差ノルムの収束に関する停滞判定回数を調べた．その結果を**表5**に示し，数値例1と同様にして，最も計算時間の短縮された箇所にアンダーラインを引いた．表5から，$\ell_{max} = 4$ のときでは Dh が 2^{-1} を除くすべての場合，$\ell_{max} = 6$ のときでは Dh が 2^2 をのぞくすべての場合で停滞判定回数が15回で δ が0.10のときに最も計算時間が短縮されると確認することができた．また，$\ell_{max} = 8$ のときも Dh が 2^{-2} と 2^2 の場合において停滞判定回数が15回で δ が0.10のときに最も計算時間が短縮されると分かった．さらに，数値例1のように $\ell_{max} = 4$ かつ停滞判定回数15回のときに，δ を0から0.20まで0.025ずつ変化させたときの計算時間を**図6**に示した．図6から，$\delta = 0.10$ のときに計算時間は最も短縮されるこ

表 6 数値例 2 の結果 (time：時間 (秒), iter：反復回数)
Table 6 Numerical results of the example 2 (time: computational time (second), iter: iteration count).

Dh	2^{-2}		2^{-1}		2^{0}		2^{1}		2^{2}	
算法	time	iter	time	iter	time	iter	time	iter	time	iter
BiCGStab	20.618	942	22.811	1030	26.558	1238	30.937	1476	39.057	1984
BiCGStab(2)	21.504	914	22.848	1014	28.368	1244	31.700	1478	39.666	1896
BiCGStab(4)	23.367	966	24.940	1062	30.628	1272	35.478	1530	48.191	1892
P-BiCGStab(1 : 2)	21.598	919	24.238	1079	28.117	1261	33.618	1469	43.916	1919
P-BiCGStab(1 : 4)	25.102	938	28.205	1028	32.395	1258	39.344	1502	48.156	1860
P-BiCGStab(1 : 6)	24.624	941	30.033	1021	37.238	1261	45.361	1545	55.573	1919
P-BiCGStab(1 : 8)	25.371	916	33.751	1034	44.122	1300	53.406	1586	65.449	1954
P-BiCGStab(2 : 4)	23.675	946	25.459	1014	31.062	1272	36.727	1475	46.965	1868
P-BiCGStab(2 : 6)	26.860	917	29.382	1019	35.440	1223	44.766	1526	57.238	1931
P-BiCGStab(2 : 8)	28.505	939	33.065	1016	43.926	1318	53.622	1601	69.489	2026
PSR-BiCGStab(2 : 4)	19.089	947	21.539	1084	24.860	1285	28.768	1475	41.697	1930
PSR-BiCGStab(2 : 6)	20.478	964	21.639	1022	25.899	1256	31.057	1456	43.976	1892
PSR-BiCGStab(2 : 8)	23.603	940	25.501	1028	31.596	1220	38.214	1488	45.617	1960
< PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法の ℓ の変化回数 >										
PSR-BiCGStab(2 : 4)	56		52		50		42		111	
PSR-BiCGStab(2 : 6)	37		34		41		27		71	
PSR-BiCGStab(2 : 8)	29		28		35		29		50	

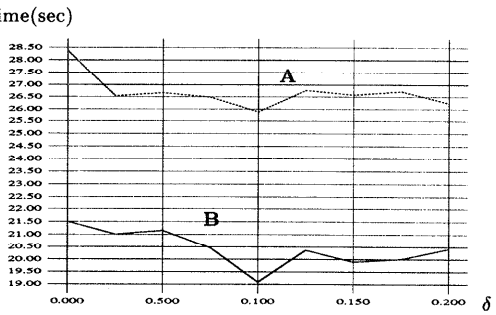


図 6 数値例 2 における計算時間と δ の関係 ($\ell_{max} = 4$, $stag = 15$), A: $Dh = 2^0$, B: $Dh = 2^{-2}$
Fig. 6 The relation between computational time and δ for example 2 ($\ell_{max} = 4$, $stag = 15$), A: $Dh = 2^0$, B: $Dh = 2^{-2}$.

とが確認できる．したがって，この数値例でも数値例 1 同様に PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法における残差ノルムの収束に関する停滞判定回数は 15 回，δ は 0.10 とする．

次に，残差ノルムが収束条件を満たすまでに要した計算時間と反復回数をそれぞれの算法について比較した．その結果を表 6 に示す．まず，P-BiCGStab(1 : ℓ_{max}) 法についてだが，係数 Dh と ℓ_{max} がいずれの場合でも BiCGStab 法および BiCGStab(2) 法と比べて計算時間が短縮されることはなく，逆に少なくとも 1 割，最大で 2 倍近く増大してしまった．P-BiCGStab(2 : ℓ_{max}) 法の場合でも，ℓ_{max} が 4 のときに計算時間は BiCGStab(4) 法と同じ程度であり，ℓ_{max} が 4 より大きいときには BiCGStab(4) 法よりも 2 割から 4 割程度増大してしまった．PSR-BiCGStab(2 : ℓ_{max}) 法に関しては，$\ell_{max} = 4, 6$ のとき，係数 Dh が 2^2 のときを除いたすべての場合において，BiCGStab 法および BiCGStab(2) 法よりも計算時間が 1 割から 2 割短縮された．また，$\ell_{max} = 8$ のときは Dh が 2^2 の場合に BiCGStab(4) 法よりも計算時間が 1 割短縮され，それ以外の場合でも BiCGStab(4) 法と同じ程度の性能を確認することができた．

6. おわりに

提案した算法に関して，数値実験を AP3000 で行った結果について比較検討を行った．P-BiCGStab(1 : ℓ_{max}) 法に関しては，従来の BiCGStab(ℓ) 法に比べて計算時間はほぼ同じか，もしくは ℓ_{max} を 6 以上に増大させると悪いときで約 5 割程度増大してしまうことが数値例 1，2 を通して確認された．一方，本稿で提案した PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法は，数値例 1，2 を通じて，多くの場合，停滞判定回数 15 回，$\delta = 0.10$ かつ $\ell_{min} = 2$，$\ell_{max} = 4$ のときに計算時間が最も短縮されるということが確認された．したがって，本稿における数値実験の結果をみる限り，2 次元の楕円型偏微分方程式の境界値問題を有限差分法で離散化することで得られる連立 1 次方程式の近似解を求めるとき，停滞判定回数は 15 回，δ の値は 0.10 と設定することで，提案した PSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法の性能を最も向上させることができる．また，PSR-BiCGStab(2 : 4) 法では BiCGStab(2) 法や BiCGStab 法よりも 1 割程度，PSR-BiCGStab(2 : 6) 法，PSR-BiCGStab(2 : 8) 法では BiCGStab(4) 法よりも 1 割から 2 割程度それぞれ計算時間を短縮できることが確認できた．さらに，この算法における ℓ を変化させるパラメータは，すべて BiCGStab(ℓ) 法の反復過程の中で生じた副産物のみで構成されているので，パラメータを計算するための余分なオーバヘッ

ドはかからない．それゆえ，2次元楕円型偏微分方程式の境界値問題を有限差分法の離散化によって得られる連立1次方程式の近似解を求める場合には，従来のBiCGStab(ℓ) 法よりも今回提案した算法のブレイクダウンや残差ノルムの停滞に対処した ℓ を適応的に変化させるPSR-BiCGStab(ℓ_{min} : ℓ_{max}) 法を利用する方が望ましいといえる．

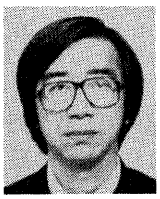
参考文献

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 minimize 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) Nodera, T.: New variant of BiCG method for solving non-symmetric systems, *Advances in Computer methods for Partial Differential Equations,* Vol.6, pp.130–135, IMACS (1987).
5) 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).
6) Nodera, T.: A note on BiCGStab algorithm, *SANUM,* Vol.18, pp.157–166 (1992).
7) Joubert, W.: Lanczos methods for the solution of nonsymmetric systems of linear equations, *SIAM J. Matrix. Anal. Appl.,* Vol.13, No.3, pp.928–943 (1992).
8) Gutknecht, M.H.: Variants of BiCGStab for matrices with complex spectrum, *SIAM J. Sci. Comput.,* Vol.14, pp.1020–1033 (1993).
9) 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).
10) Weiss, R.: Properties of Generalized Conjugate Gradient Methods, *Numer. Lin. Alg. Appl.,* Vol.1, No.1, pp.45–63 (1994).
11) Bruaset, A.M.: A survey of Preconditioned Iterative Methods, *Pitman Research Note in Mathematics,* No.32, Longman Scientific & Technical, UK (1995).
12) Sleijpen, G.L.G. and Van der Vorst, H.A.: An overview of approaches for the stable computation of hybrid BiCG methods, *Appl. Numer. Math.,* Vol.19, pp.235–254 (1995).
13) 野寺，野口：AP1000 における BiCGStab(ℓ) 法の有効性について，情報処理学会論文誌，Vol.38, No.11, pp.2089–2101 (1997).
14) Nodera, T. and Noguchi, Y.: A note on the BiCGStab(ℓ) method on AP1000, (*Iterative Method in Scientific Computation*), *IMACS Series in Computational and Applied Mathematics,* Vol.4, pp.53–58 (1998).

(平成10年4月20日受付)
(平成11年3月 5 日採録)

森屋健太郎（学生会員）

1997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に在学中．並列計算機による大規模計算に興味を持つ．



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，1986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）等がある．工学博士．エッセイスト．SIAM，日本応用数理学会各会員．