反復中に直交空間を再構成するML(k)BiCGStab法

森屋　健太郎[†]　野寺　隆[††]

BiCGStab法は，非対称な大規模線形システムを解くための反復解法の1つである．近年，Yeungら[6]は，BiCGStab法にk個の直交ベクトルを組み合わせたML(k)BiCGStab法を提案した．しかし，kを大きくとり過ぎると，直交ベクトルを適用する計算量が増加し計算時間に対する残差ノルムの収束性が改善されないことがある．本稿ではkの初期値を1としておき，ピボットブレイクダウンが発生するときにkの値を増加させ，直交ベクトル空間を新たに計算し直すML(k)BiCGStab法の改良版について提案する．

ML(k)BiCGStab method with reconstracting the orthogonal subspace during the iterative process

Kentaro Moriya [†] and Takashi Nodera[††]

BiCGStab(ℓ) method is one of the iterative schemes for a large linear system of equations with nonsymmetric matrix. Recently, Yeung et. al.[6] has been proposed ML(k)BiCGStab method, which applies k orthogonal vectors to BiCGStab method. However, when k is too large, the computation cost increases, and the convergence of the residual norm does not always improve. In this paper, we proposed the new variant of ML(k)BiCGStab method. In our algorithm, we start with $k = 1$ as the inital value. When the pivot breakdown is likely to occur, we also increase the parameter k and recompute the orthogonal subspace.

1. はじめに

大型疎行列を係数とする連立1次方程式

$$A\boldsymbol{x} = \boldsymbol{b}, \quad A \in \boldsymbol{R}^{n\times n}, \quad \boldsymbol{x},\ \boldsymbol{b} \in \boldsymbol{R}^n \qquad (1)$$

の解$\boldsymbol{x}$を求めるには，反復解法がしばしば用いられる．本稿では，近年Yeungら[6]により提案されたBiCGStab法の改良阪であるML(k)BiCGStab法について考える．ML(k)BiCGStab法は，BiCGStab法にk個の直交ベクトルを適用し，それらの直交ベクトルによって，新たなk個のクリロフ部分空間

$$K(\boldsymbol{q}_0,\ A^T),\ K(\boldsymbol{q}_1,\ A^T),\ \ldots,\ K(\boldsymbol{q}_{k-1},\ A^T)$$

を生成する．しかし，直交ベクトルの個数kは元々ユーザが経験的に決定するしかなく，kを適切な値に設定しないと直交化の計算量が必要以上にかかるという問題がある．このようなML(k)BiCGStab法の難点を改善するために，五条方ら[7]は，残差ノルムの収束停滞を判定するいくつかのパラメータを導入し，kを動的に増加させる算法について提案した．それに対して本稿では，$k = 1$で反復をスタートし，ピボットブレイクダウンが起こりそうなときに直交ベクトルを再計算させると同時に，kを増加させることで，ML(k)BiCGStab法のピボットブレイクダウンを回避する算法について提案する．

最後に，数値実験において，本稿で提案した算法をBiCGStab(ℓ)法，ML(k)BiCGStab法と比較し，提案した算法が従来の算法の残差ノルムが収束しない問題で効果を発揮することを示す．

2. BiCGStab法

BiCGStab法[2]は積型反復解法の1つであり，1次のMR多項式をBiCG法[1]から求まった残差ベクトルに掛けることで，残差ノルムの収束を加速させている．ここでl回目の反復で求まったBiCG法の残差ベクトルを$\tilde{\boldsymbol{r}}_l$，l回目の反復における係数行列Aの1次のMR多項式を

$$P_l(A) = I - \alpha_l A \qquad (2)$$

とすると，l回目のBiCGStab法における残差ベクトル$\boldsymbol{r}_l$は

$$\boldsymbol{r}_l = P_l(A)P_{l-1}(A)P_{l-2}(A)\ldots P_0(A)\tilde{\boldsymbol{r}}_l$$

と表すことができる．ただし，係数α_lは残差ノルム$\|\boldsymbol{r}_l\|_2$を最小にするように決定する．BiCGStab法では，1次のMR多項式$P_l(A) = I - \alpha_l A$の係数α_lが0や0に近い値をとるとき，残差ノルムの収束停滞を起こしたり，ピボットブレイクダウン(0割算によ

† 青山学院大学理工学部
Faculty of Science and Technology, Aoyama Gakuin University
†† 慶應義塾大学理工学部
Faculty of Science and Technology, Keio University

る算法の破綻) が発生しやすくなる [2),4)]．それに対して Sleijpen ら [3)] は，一般の ℓ 次の多項式を利用することで，算法をより安定にした BiCGStab(ℓ) 法を提案した．ここで，BiCGStab 法は BiCGStab(ℓ) 法の $\ell = 1$ の場合といえる．

3. 直交ベクトルを利用する BiCGStab 法

本節では，ML(k)BiCG 法と ML(k)BiCGStab 法について述べる．

3.1 ML(k)BiCG 法

ML(k)BiCG 法では A^T を使ったクリロフ部分空間を k 個生成し，l 回目の反復における残差ベクトル $\tilde{\boldsymbol{r}}_l$ は，これらの部分空間すべてと直交するように計算される．従って

$$\boldsymbol{p}_{jk+i} = (A^T)^j \boldsymbol{q}_i, \quad i = 0, 1, \ldots, k-1, \quad j = 1, 2, \ldots$$

とすると，ML(k)BiCG 法の残差ベクトルは任意の $i,\ j$ について $\boldsymbol{p}_{jk+i}$ と直交するように生成されるので

$$\boldsymbol{r}_l \perp \mathrm{span}\{\boldsymbol{p}_1,\ \boldsymbol{p}_2,\ \ldots,\ \boldsymbol{p}_l\} \tag{3}$$

となる．また

$$\mathrm{Span}\{\boldsymbol{v}_0,\ \boldsymbol{v}_1,\ \ldots, \boldsymbol{v}_{l-1}\} \in K_l(\boldsymbol{v}_0,\ A) \tag{4}$$

とすると，近似解 $\boldsymbol{x}_l$ は

$$\boldsymbol{x}_l \in \boldsymbol{x}_0 + \mathrm{Span}\{\boldsymbol{v}_0,\ \boldsymbol{v}_1,\ \ldots, \boldsymbol{v}_{l-1}\} \tag{5}$$

と表すことができる．ただし，$K_l(\boldsymbol{r}_0,\ A)$ は

$$K_l(\boldsymbol{v}_0, A) = \mathrm{Span}\left\{\boldsymbol{v}_0,\ A\boldsymbol{v}_0,\ \ldots, A^{l-1}\boldsymbol{v}_0\right\} \tag{6}$$

となるクリロフ部分空間である．従って

$$\boldsymbol{v}_l \in A^l \boldsymbol{v}_0 + K_l(\boldsymbol{r}_0,\ A) \tag{7}$$

から

$$\boldsymbol{v}_l = A^l \boldsymbol{v}_0 + \gamma_0^{(l)} \boldsymbol{v}_0 + \gamma_1^{(l)} A\boldsymbol{v}_0 + \ldots + \gamma_{l-1}^{(l)} A^{l-1}\boldsymbol{v}_0 \tag{8}$$

と表すことができる．式 (4)，(6) の関係から，式 (8) は

$$\boldsymbol{v}_l = A\boldsymbol{v}_{l-1} + h_{l-1}^{(l-1)}\boldsymbol{v}_{l-1} + h_{l-2}^{(l-1)}\boldsymbol{v}_{l-2} + \ldots\ldots + h_0^{(l-1)}\boldsymbol{v}_0 \tag{9}$$

と変形できる．ここで，式 (9) の係数

$$h_{l-1}^{(l-1)},\ h_{l-2}^{(l-1)},\ \ldots, h_0^{(l-1)}$$

は，

$$\boldsymbol{v}_l \perp \mathrm{span}\{\boldsymbol{p}_1,\ \boldsymbol{p}_2,\ \ldots,\ \boldsymbol{p}_l\} \tag{10}$$

を満たすように計算を行うので

$$h_{l-1}^{(l-1)},\ h_{l-2}^{(l-1)},\ \ldots, h_{m_l}^{(l-1)}$$

だけが非ゼロの係数になる．従って，式 (9) は

$$\boldsymbol{v}_l = A\boldsymbol{v}_{l-1} + h_{l-1}^{(l-1)}\boldsymbol{v}_{l-1} + h_{l-2}^{(l-1)}\boldsymbol{v}_{l-2} + \ldots\ldots + h_{m_l}^{(l-1)}\boldsymbol{v}_{m_l} \tag{11}$$

となり，式 (11) のような $k+2$ 項の漸化式を得ることができる．ただし，$m_l = \max(l-k-1,\ 0)$ である．ここで，i 行 j 列に $h_{i-1}^{(j-1)}$ を持つヘッセンベルグ行列を H_l とすると

$$AV_l = V_l H_l \tag{12}$$

の関係が成立する．ただし，$V_l = (\boldsymbol{v}_0,\ \boldsymbol{v}_1,\ \ldots, \boldsymbol{v}_l)$ である．さらに，$P = [\boldsymbol{p}_1,\ \boldsymbol{p}_2,\ \ldots, \boldsymbol{p}_l]$ とすると，式 (10) の関係から $P_l^T V_l$ は下三角行列となる．従って，式 (5)，(12) を用いると近似解 $\boldsymbol{x}_l$ は

$$\boldsymbol{x}_l = \boldsymbol{x}_0 + V_l H_l^{-1} \boldsymbol{e}_1 \tag{13}$$

となる．ここで $\boldsymbol{r}_i = \xi_i \boldsymbol{v}_i,\ i = 0, 1, \ldots, l-1$ として，$\Theta_l = \mathrm{diag}\{\xi_0,\ \xi_1,\ \ldots, \xi_{l-1}\}$，$R_l = \{\boldsymbol{r}_0,\ \boldsymbol{r}_1,\ \ldots,\ \boldsymbol{r}_{l-1}\}$ と置くと，式 (13) は

$$\boldsymbol{x}_l = \boldsymbol{x}_0 + R_l \Theta_l^{-1} H_l^{-1} \boldsymbol{e}_1 \tag{14}$$

となる．さらに，行列 $H_l\Theta_l$ を LDU 分解したものを

$$H_l\Theta_l = L_l D_l U_l \tag{15}$$

とし，式 (15) を式 (14) に代入すると

$$\boldsymbol{x}_l = \boldsymbol{x}_0 + R_l U_l^{-1} D_l^{-1} L_l^{-1} \boldsymbol{e}_1 \tag{16}$$

となる．ここで

$$G_l = \left\{\boldsymbol{g}_0,\ \boldsymbol{g}_1,\ \ldots, \boldsymbol{g}_{l-1}\right\} = R_l U_l^{-1} \tag{17}$$

$$\boldsymbol{z}_l = D_l^{-1} L_l^{-1} \boldsymbol{e}_1 \tag{18}$$

と定義すると，$L_l D_l$ が下三角行列でなので，式 (18) を前進代入によりベクトル $\boldsymbol{z}_l$ を求めることを考える．$\boldsymbol{z}_l$ は上の行から順次求まるのでこれを漸化式によって表すと

$$\boldsymbol{z}_l = [\boldsymbol{z}_l,\ \alpha_l]^T \tag{19}$$

となる．従って，式 (16) に式 (17)，(18)，(19) を代入すると，近似解は

$$\begin{aligned}\boldsymbol{x}_l &= \boldsymbol{x}_0 + G_l \boldsymbol{z}_l \\ &= \boldsymbol{x}_0 + G_{l-1}\boldsymbol{z}_{l-1} + \alpha_l \boldsymbol{g}_{l-1} \\ &= \boldsymbol{x}_{l-1} + \alpha_l \boldsymbol{g}_{l-1}\end{aligned} \tag{20}$$

となる．また，残差ベクトルは

$$\boldsymbol{r}_l = \boldsymbol{r}_{l-1} - \alpha_l A\boldsymbol{g}_l \tag{21}$$

となる．ところで，式 (17) から $G_{l+1} = R_{l+1}U_{l+1}^{-1}$ かつ U_{l+1} が上三角行列なので，ベクトル $\boldsymbol{g}_l$ は後退代入を利用して求めることができる．従って，$\boldsymbol{g}_l$ を漸化式で表すと

$$\boldsymbol{g}_l = \boldsymbol{r}_l + \sum_{i=\bar{m}_l}^{l-1} \beta_i^{(l)} \boldsymbol{g}_i \tag{22}$$

である．ただし，$\bar{m}_l = \max(l-k, 0)$ である．ML(k)BiCG 法では，式 (21)，(22) におけるスカラ α_l と $\beta_i^{(l)},\ (i = \bar{m}_l,\ \bar{m}_l+1, \ldots, l-1)$ を求めることになる．α_l に関しては，$\boldsymbol{r}_l$ と $\boldsymbol{p}_l$ の直交性を用いて式 (21) の両辺に $\boldsymbol{p}_l$ を掛けると

$$\boldsymbol{p}_l^T \boldsymbol{r}_l = \boldsymbol{p}_l^T \boldsymbol{r}_{l-1} - \alpha_l \boldsymbol{p}_l^T A\boldsymbol{g}_{l-1} = 0 \tag{23}$$

であるから

$$\alpha_l = \boldsymbol{p}_l^T \boldsymbol{r}_{l-1} \,/\, \boldsymbol{p}_l^T A\boldsymbol{g}_{l-1} \tag{24}$$

となる．$\beta_i^{(l)}$ に関しては，$\boldsymbol{g}_i$ と $\boldsymbol{p}_i$ が A-直交するように取るので

$$\boldsymbol{p}_i^T A\boldsymbol{g}_j = 0, \quad i \le j \tag{25}$$

が成立する．式 (22) の両辺に $\boldsymbol{p}_i^T A$ を掛けたものを式 (25) に代入すると

$$\boldsymbol{p}_i^T A\boldsymbol{r}_l + \beta_{l-1}^{(l)} \boldsymbol{p}_i^T A\boldsymbol{g}_{l-1} + \Sigma_{j=\bar{m}_l}^{i-2} \beta_j^{(l)} A\boldsymbol{g}_j = 0 \qquad (26)$$

となり，次式が得られることになる．

$$\beta_{i-1}^{(l)} = -\frac{\boldsymbol{p}_i^T (A\boldsymbol{r}_l + \Sigma_{j=\bar{m}_l}^{i-2} \beta_j^{(l)} A\boldsymbol{g}_j)}{\boldsymbol{p}_i^T A\boldsymbol{g}_{i-1}}, \qquad (27)$$
$$(i = \bar{m}_l + 1,\ \bar{m}_l + 2,\ \ldots, l)$$

従って，式 (22)，(24)，(27) を式 (20)，(21) に代入すると，ML(k)BiCG 法の近似解と残差ベクトルを求めることができる．特に，$k = 1$ のときは 1 個のクリロフ部分空間 $K(\boldsymbol{q}_0,\ A^T)$ だけを生成するので，この場合の ML(k)BiCG 法は従来の BiCG 法と同じ算法となる．

3.2 ML(k)BiCGStab 法

前節で述べた ML(k)BiCG 法の残差ベクトルに 1 次の MR 多項式を掛けたものが ML(k)BiCGStab 法の残差ベクトルになる．本発表では **図 1** の"⟨⟨ beginning of new process ⟩⟩" と"⟨⟨ end of new process ⟩⟩" までを除いた部分を ML(k)BiCGStab 法の実装法として示すが，算法の詳細については文献 [6] を参照して欲しい．ML(k)BiCGStab 法には ML(k)BiCG 法のように A^T とベクトルの乗算をする必要がないという長所がある．

また，ML(k)BiCG 法と同様，とくに $k = 1$ のとき ML(k)BiCGStab 法は BiCGStab 法と同じ算法になる．

4. ピボットブレイクダウンを回避する算法

本節では，ML(k)BiCGStab 法において，ピボットブレイクダウンが発生する可能性がある時に直交ベクトルの個数を増加させて，直交ベクトルをすべて再計算させる算法について提案する．

4.1 ピボットブレイクダウンを判断する基準

BiCG 法では，l 回目における残差ベクトルと疑似残差ベクトルの内積が 0 や 0 に近い値となるときに，ピボットブレイクダウンが発生しやすくなる．これは，BiCGStab 法における初期疑似残差ベクトルと任意の反復回数における残差ベクトルの内積が 0 に近い値をとるときにピボットブレイクダウンが発生しやすくなることと同値である [2),4)]．そこで本稿では

$$\psi_j = \left(\boldsymbol{r}_l,\ \boldsymbol{q}_j\right) / (\|\boldsymbol{r}_l\|_2 \|\boldsymbol{q}_j\|_2) \qquad (28)$$

として，式 (28) によって表される $\psi_j (j = 0, 1, \ldots, k-1)$ のことをピボットと呼ぶことにする．式 (28) のピボットが 0 に近い値となるとき，ML(k)BiCGStab 法の残差ベクトル $\boldsymbol{r}_l$ と k 個のクリロフ部分空間 $K(\boldsymbol{q}_j, A^T),\ j = 0, 1, \ldots, k-1$ の直交性が崩れることになる．従って

$$|\psi_j| < \varepsilon \qquad (29)$$

なら ML(k)BiCGStab 法はピボットブレイクダウンの可能性があると判断する．ただし，ε は計算機の誤差限界に平方根をとった値となる [5)]．例えば，倍精度の浮動小数点演算を 64 ビットで行う計算機を利用する場合，誤差限界は $O(10^{-16})$ 程度になるので，$\varepsilon = 1.0 \times 10^{-8}$ と設定するのが妥当である．

4.2 直交ベクトルの再計算

本節では，ML(k)BiCGStab 法のピボットブレイクダウンを回避させるために，前節で述べた式 (29) が成立するときに直交ベクトルの個数を増加させ，直交ベクトルの計算をし直す算法について提案する．ピボットブレイクダウンを回避するための算法は以下のようになる．

(1) $|\psi_j| < \varepsilon$ なら，現時点で求まっていた残差ベクトルを正規化し，それを初期直交ベクトル $\boldsymbol{q}_0$ として設定する．

(2) $k < k_{max}$ なら k に 1 を加算する．すでに $k = k_{max}$ なら k を 1 にする．

(3) アーノルディ法を用いて，残りの $k-1$ 個の直交ベクトル $\boldsymbol{q}_1, \ldots,\ \boldsymbol{q}_{k-1}$ を計算する．

ただし，k_{max} は直交ベクトルの個数 k が取り得る最大値とする．ステップ (1) では，現時点で求まっていた残差ベクトルを正規化し，直交ベクトル $\boldsymbol{q}_0$ を新たに計算することで，ピボット ψ_j を最大値の 1.0 にすることができる．ステップ (2) では，直交ベクトルの個数 k を 1 だけ増やすことで算法を安定させ，ステップ (1) で求まった直交ベクトル $\boldsymbol{q}_0$ を元に k 個の直交ベクトルを再計算している．だたし，元々 $k = k_{max}$ であるなら，新たに k を 1 に戻して再度直交ベクトルを計算し直すことにする．なぜなら，このような場合は，元々計算して求まっていた k_{max} 個の直交ベクトルを利用していてもピボットブレイクダウンが起きることを意味しており，直交ベクトルの再計算が必要になるからである．つまり，不等式 (29) が成立する度に直交ベクトル $\boldsymbol{q}_0$ が再計算され，利用する直交ベクトルの個数が 1 つ増えるが，直交ベクトルの個数が最大値 k_{max} になったときでも，ピボットブレイクダウンを確実に回避させる措置として，直交ベクトルの個数を 1 に戻して再計算をさせることになる．このように直交ベクトルを反復の途中で再計算して，$k \leq k_{max}$ の範囲で直交ベクトルの個数を動的に増加させる手法を取り入れた ML(k)BiCGStab 法のことを本稿では特に ML($k \leq k_{max}$)BiCGStab 法と呼ぶことにする．図 1 に ML($k \leq k_{max}$)BiCGStab 法の算法を示す．従来の ML(k)BiCGStab 法と比較して新たに追加された部分は，"⟨⟨ beginning of new process ⟩⟩" と"⟨⟨ end of new process ⟩⟩" によってはさまれた部分である．ML(k)BiCGStab 法では利用する k 個の直交ベクトルは反復の開始前に求め，反復終了までまったく同じ直交ベクトルを用いるが，ML(k)BiCGStab 法で

ML($k \leq k_{max}$)BiCGStab method

choose x_0
$r_0 = b - Ax_0$, $g_0 = r_0$, $q_0 = r_0 / \|r_0\|_2$, frag=0, $j = 0$
while "$\|r_j\|_2$ is small enough" **do**
  $w_j = Ag_j$, $c_j = (q_0, w_j)$, $\alpha_{j+1} = (q_0, r_j)/c_j$
  $u_{j+1} = r_j - \alpha_{j+1} w_j$, $\rho_{j+1} = -(u_{j+1}, Au_{j+1}) / \|Au_{j+1}\|_2^2$
  $x_{j+1} = x_j - \rho_{j+1} u_{j+1} + \alpha_{j+1} g_j$, $r_{j+1} = \rho_{j+1} Au_{j+1} + u_{j+1}$
  for $i = 1$ **to** k **do**
    $z_d = u_{j+i}$, $z_g = r_{j+i}$, $z_w = 0$
    for $s = i$ **to** $k-1$ **and** frag=1 **do**
      $\beta_{j-k+s} = -(q_{s+1}, z_d)/c_{j-k+s}$
      $z_d = z_d + \beta_{j-k+s} d_{j-k+s}$, $z_g = z_g + \beta_{j-k+s} g_{j-k+s}$, $z_w = z_w + \beta_{j-k+s} w_{j-k+s}$
    enddo
    $\beta_j = -\frac{(q_0, r_{j+1}) + \rho_{j+1}(q_0, z_w)}{\rho_{j+1} c_j}$
    $z_g = z_g + \beta_j g_j$, $z_w = \rho_{j+1}(z_w + \beta_j w_j)$, $z_d = r_{j+1} + z_w$
    for $s = 1$ **to** $i-1$ **do**
      $\beta_{j+s} = -(q_{s+1}, z_d)/c_{j+s}$, $z_d = z_d + \beta_{j+s} d_{j+s}$, $z_g = z_g + \beta_{j+s} g_{j+s}$
    enddo
    $d_{j+i} = z_d - u_{j+i}$, $g_{j+i} = z_g + z_w$
    if $i < k$ **then**
      $c_{j+i} = (q_{i+1}, d_{j+i})$, $\alpha_{j+i+1} = (q_{i+1}, u_{j+i})/c_{j+i}$
      $u_{j+i+1} = u_{j+i} - \alpha_{j+i+1} d_{j+i}$, $x_{j+i+1} = x_{j+i} + \rho_{j+1} \alpha_{j+i+1} g_{j+i}$
      $w_{j+i} = Ag_{j+i}$, $r_{j+i+1} = r_{j+i} - \rho_{j+1} \alpha_{j+i+1} w_{j+i}$
    endif
  enddo
  $j = j + k$
  frag=1
  ⟨⟨ **beginning of new process** ⟩⟩
  for $s = 0$ **to** $k-1$ **do**
    $\psi_s = (r_{j-k+i+1}, q_s) / \|r_{j-k+i+1}\|_2$
    if $|\psi_s| < \varepsilon$ **then**
      $q_0 = r_{j-k+i+1} / \|r_{j-k+i+1}\|_2$
      if $k = k_{max}$ **then**
        $k = 1$
      else
        $k = k + 1$
        compute $q_1, \ldots, q_{k-1}$ by Arnoldi method
      endif
      frag=0
      break this loop
    endif
  enddo
  ⟨⟨ **end of new process** ⟩⟩
enddo

図 1 ML($k \leq k_{max}$)BiCGStab 法
Fig. 1 ML($k \leq k_{max}$)BiCGStab method

は直交ベクトルの計算を反復の途中でやり直している点が異なる．

5. 数 値 実 験

連立 1 次方程式 (1) を ML($k \leq k_{max}$)BiCGStab 法と BiCGStab 法，ML(k)BiCGStab 法によって解き，3 種類の算法について残差ノルムの収束性に関する性能比較を行った．ℓ, k, k_{max} の値としてはいずれも 1,2,4,8 とした．ここでは，2 つの数値例を示すことにするが，いずれの例にも共通して使用する条件は以下のとおりである．

初期近似解: 0 ベクトル
収束判定条件: $\|\boldsymbol{r}_l\|_2 / \|\boldsymbol{b}\|_2 < 1.0 \times 10^{-12}$
ε **の値:** 1.0×10^{-8}

数値実験の結果には，上記の収束判定条件を満たすまでにかかった反復回数と計算時間を掲載したが，各算法の性能比較は反復回数ではなく計算時間によって

表 1 数値例 1 の計算結果 (time: 計算時間 (秒), iter: 反復回数)
Table 1 The numerical results for example 1 (time: computation time (sec), iter: iterations)

算法	time	iter
BiCGStab(1)	...	...
BiCGStab(2)	...	...
BiCGStab(4)	...	...
BiCGStab(8)	273.0	68984
ML(1)BiCGStab	...	...
ML(2)BiCGStab	264.0	98430
ML(4)BiCGStab	...	...
ML(8)BiCGStab	...	...
ML($k \leq 1$)BiCGStab	199.0	71504
ML($k \leq 2$)BiCGStab	264.0	84146
ML($k \leq 4$)BiCGStab	...	...
ML($k \leq 8$)BiCGStab	264.0	74563

(...): 5 分間で残差ノルムが収束しなかった場合

行った．なぜなら，1 回にかかる反復回数は各算法によって異なるためである．

5.1 数値例 1: "Reservior simulation" の例

MatrixMarket[8] における"PORES2" という名の行列を係数とする連立 1 次方程式を考える．この係数行列の次元は 1224，非 0 要素は 9660 個である．右辺 $\boldsymbol{b}$ は，厳密解の全ての要素が 1 になるように設定した．以下に計算した環境を示す．

- AT 互換 PC
- CPU PentiumIII 700MHz
- Vine Linux 2.1.5

また，残差ノルムの収束にかかった時間と反復回数は **表 1** のようになる．BiCGStab(ℓ) 法と ML(k)BiCGStab 法に関しては，それぞれ，BiCGStab(8) 法と ML(2)BiCGStab 法でしか残差ノルムの収束を確認できなかった．それに対して ML(k)BiCGStab 法では，$k_{max} = 4$ の場合を除いて残差ノルムは収束し，その計算時間は $k_{max} = 2, 8$ で BiCGStab(8) 法，ML(2)BiCGStab 法と同じであり，$k_{max} = 1$ のときは 30%程度短くなっている．従って，ML($k \leq k_{max}$)BiCGStab 法の計算時間に関する性能は，BiCGStab(ℓ) 法と ML(k)BiCGStab 法と比べて，同じかもしくはそれ以上である．また，ML($k \leq k_{max}$)BiCGStab 法は BiCGStab(ℓ) 法と ML(k)BiCGStab 法よりもパラメータの設定範囲が広いということができる．

5.2 数値例 2: Toeplitz 行列を係数とする方程式

以下のような行列 A を係数とする連立 1 次方程式を考える [4]．

$$A = \begin{pmatrix} 2 & 1 & & & \\ 0 & 2 & 1 & & \\ \gamma & 0 & 2 & 1 & \\ & \gamma & 0 & 2 & \ddots \\ & & \ddots & \ddots & \ddots \end{pmatrix}$$

ただし，右辺 $\boldsymbol{b}$ の成分はすべて 1 とし，この方程式

表 3 数値例 2 の ML($k \leq k_{max}$)BiCGStab 法の式 (29) を満たした回数 ($\gamma = 1.9$)
Table 3 The number of satisfying formula (29) of ML($k \leq k_{max}$)BiCGStab method in example 2 ($\gamma = 1.9$)

k_{max}	j							
	0	1	2	3	4	5	6	7
1	53	–	–	–	–	–	–	–
2	49	4	–	–	–	–	–	–
4	19	4	0	0	–	–	–	–
8	19	2	5	0	0	0	0	0

の次元を 262144 と設定する．計算には Compaq の分散メモリ型並列計算機 Beowulf を使用し CPU には 600MHz の Alpha チップ 8 台を用いた．また，データ通信ライブラリには MPI を用いた．数値例 2 の実験結果は **表 2** に掲載した．BiCGStab(ℓ) 法では $\gamma = 1.7, 1.8$ のときに，BiCGStab(4) 法と BiCGStab(8) 法で ML($k \leq k_{max}$)BiCGStab 法より少ない計算時間で残差ノルムが収束したが，それ以外の場合は最大計算時間以内で残差ノルムは収束判定条件を満たさなかった．それに対して，ML($k \leq k_{max}$)BiCGStab 法では，数値例 2 のすべての場合で残差ノルムは収束条件を満たした．特に，$\gamma = 1.9, 2.0$ のときは，残差ノルムが収束したのは ML($k \leq k_{max}$)BiCGStab 法だけであった．そこで $\gamma = 1.9$ の場合について，**図 2** に残差ノルムの収束する様子とピボットの変化する様子を示した．BiCGStab(2) 法と ML(2)BiCGStab 法では，残差ノルムは発散しているかもしくは収束が停滞しているが，ML($k \leq 2$)BiCGStab 法では収束している．また，ピボットに関しては，BiCGStab(2) 法と ML(2)BiCGStab 法では 0 に近い値をとっているのに対して，ML($k \leq k_{max}$)BiCGStab 法については常に ε を上回る値がとられている．

次に，$\gamma = 1.9$ の場合について，ML($k \leq k_{max}$)BiCGStab 法のピボット ψ_j の式 (29) を満たした回数を **表 3** に示した．いずれの場合も j が小さいときにピボットブレイクダウンを起こす場合が多い．また，k_{max} が大きくなるとピボットブレイクダウンの判定式 (29) が成立する回数が減少している．従って，k_{max} が大きくなると算法がより安定する傾向にある．

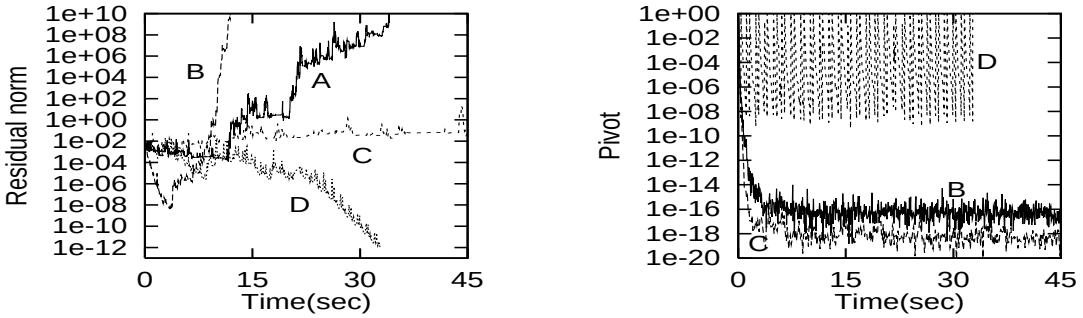
6. 結　　論

本稿では，反復の途中で直交ベクトルを計算し直し，直交ベクトルの個数 k を $k \leq k_{max}$ の範囲で動的に増加させる算法をとり入れた ML($k \leq k_{max}$)BiCGStab 法について提案した．従来の ML(k)BiCGStab 法では，反復を行う前に k 個の直交ベクトルを計算し，反復中は常にそれらを用いていた．しかし，それではピボットブレイクダウンを回避することができないという難点があった．それに対して ML($k \leq k_{max}$)BiCGStab 法では，反復の途中で直交ベクトルを計算し直すことによって，ピボットブレイクダウンを回避することができる．

表 2 数値例 2 の計算結果 (time: 計算時間 (秒), iter: 反復回数)
Table 2 Numerical results for the example 2 (time: computation time (sec), iter: iterations)

算法	γ							
	1.7		1.8		1.9		2.0	
	time	iter	time	iter	time	iter	time	iter
BiCGStab(1)	...	...	...	...	...	...	...	...
BiCGStab(2)	...	...	...	...	...	...	...	...
BiCGStab(4)	6.49	156	8.71	220	...	...	...	...
BiCGStab(8)	10.32	184	...	...	...	...	...	...
ML(2)BiCGStab	...	...	...	...	...	...	...	...
ML(4)BiCGStab	...	...	...	...	...	...	...	...
ML(8)BiCGStab	...	...	...	...	...	...	...	...
ML($k \leq 1$)BiCGStab	8.49	163	20.55	392	24.50	461	25.71	479
ML($k \leq 2$)BiCGStab	7.32	136	11.96	218	32.79	608	35.19	644
ML($k \leq 4$)BiCGStab	8.88	151	13.70	229	20.66	346	44.16	745
ML($k \leq 8$)BiCGStab	14.24	199	15.55	222	34.83	477	46.96	635

(...): 5 分の計算時間で残差ノルムの値が収束しなかった場合



(a) 残差ノルム vs 計算時間　　(b) ピボット (ψ_0 の場合) vs 計算時間

図 2 数値例 2 の計算時間に対する残差ノルムとピボットの履歴 ($\gamma = 1.9$), A: BiCGStab(1), B: BiCGStab(2), C: ML(2)BiCGStab, D: ML($k \leq 2$)BiCGStab
Fig. 2 The behavior of the residual norm and pivot vs computation time in example 2 ($\gamma = 1.9$), A: BiCGStab(1), B: BiCGStab(2), C: ML(2)BiCGStab, D: ML($k \leq 2$)BiCGStab

このことは，数値例 2 の 図 2(b) によって裏付けられている．さらに，この算法の利点は，パラメータ k を動的に決定する場合でも，経験的に求めなければならないパラメータを一切使用していないことにある．本稿の 2 つの数値実験の結果から，ML($k \leq k_{max}$)BiCGStab 法は，従来の ML(k)BiCGStab 法や BiCGStab(ℓ) 法では残差ノルムが収束しないような問題であっても，効果を発揮することが確認された．今後の課題は，偏微分方程式の境界値問題の離散化によって得られるような，より一般的な連立 1 次方程式について ML($k \leq k_{max}$)BiCGStab 法を適用し，その性能を確認することである．

参 考 文 献

1) Fletcher, R.: Conjugate Gradient Methods for Indefinite Systems, *Lecture Notes in Math.*, Vol. 506, pp. 73–89, (1976).
2) 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).
3) 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).
4) 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).
5) 森屋，野寺: 「適応的に ℓ を変化させる BiCGStab(ℓ) 法」情報処理学会論文誌，Vol. 40, No. 6, pp. 2669-2678, (1999).
6) Yeung. M. and Chan. T. F.:ML(k)BiCGStab: A BiCGSTAB Variant based on Multiple Lanczos Vectors, *SIAM J. Sci. Comput.*, Vol. 21, No, 4, pp. 1263–1290, (1999). *Int. J. of Appl. Sci. & Comput.*, Vol. 7, No. 2, pp. 87–95, (2000).
7) 五条方，野寺: ML(k)BiCGStab 法の改良版について，情報処理学会第 64 回全国大会, (2002).
8) MatrixMarket.: http://math.nist.gov/MatrixMarket/