

漸化式に着目した IDRstab 法の偽収束改善 に関する数値的考察

Numerical study for IDRstab with reliable update strategies to remedy the residual gap

相原 研輔*, 阿部 邦美†, 石渡 恵美子**

* 東京理科大学大学院理学研究科, † 岐阜聖徳学園大学経済情報学部,

** 東京理科大学理学部

Kensuke Aihara*, Kuniyoshi Abe† and Emiko Ishiwata**

* Graduate School of Science, Tokyo University of Science

† Faculty of Economics and Information, Gifu Shotoku University

** Department of Mathematical Information Science, Tokyo University of Science

1 はじめに

大規模な n 次非対称疎行列 A を係数行列, n 次元ベクトル b を右辺項とする線形方程式

$$Ax = b$$

を帰納的次元縮小 (Induced Dimension Reduction, IDR) 定理に基づく IDRstab 法 [9] によって近似的に解くことを考える. IDRstab 法は, IDR(s) 法 [10] における 1 次の安定化多項式を ℓ 次に拡張した解法であり, $s = 1$ のとき, Bi-Conjugate Gradient stabilized (BiCGstab) (ℓ) 法 [6] に帰着される. IDRstab 法と数学的に等価であり, 実装法が異なる GBi-CGSTAB(s, L) 法 [11] も提案されているが, 本論文では IDRstab 法のみを扱う.

$s > 1, \ell > 1$ を用いる IDRstab 法は, しばしば IDR(s) 法や BiCGstab(ℓ) 法に比べて優れた収束性を示す. しかし, IDR(s) 法や BiCGstab(ℓ) 法と同様, 漸化式から求まる残差ノルム $\|r_k\|_2$ は収束するにも拘らず, 丸め誤差の影響により真の残差ノルム $\|b - Ax_k\|_2$ が反復の途中で停滞する現象 (以下, 偽収束と呼ぶ) が起こり, 十分な精度の近似解が得られない場合がある.

近年, 我々は偽収束を回避し近似解の精度を改善する変形アルゴリズムを提案した [2]. その基本的な着想は, IDRstab 法の近似解と残差を, 従来の BiCG 系統の解法でしばしば用いられる典型的な漸化式

$$x_k = x_{k-1} + p_k, \quad r_k = r_{k-1} - Ap_k \quad (1)$$

によって更新することである. ただし, ベクトル Ap_k は補助ベクトル p_k に A を陽に掛けることで得る. 従来の IDRstab 法では, ある写像を用いて Ap_k と数学的に等価なベクトルを計算するため, 計算過程の違いにより数値的には異なる収束性を示す. 実際に, 式 (1) の形式を用いた IDRstab 法では, 偽収束を回避できることを実験的に示している [2]. ただし, 文献 [2] で取り上げたテスト問題は 1 例だけであり, 客観的な傾向を示すためには十分とは言えない. また, なぜ偽収束を回避できるのか, その改善要因に関しても明確

ではなかった。そこで本論文では、まず複数のテスト問題に提案した IDRstab 法を適用し、その有効性をより詳しく検証する。次に、偽収束の改善要因を段階的な数値実験により考察し、ベクトル p_k に A を陽に掛けることが偽収束を回避するために重要であることを示す。

本論文の構成として、まず2節で、従来の IDRstab 法の概略を述べる。次に3節では、式(1)の形式を用いた IDRstab 法の変形アルゴリズムを導出する。4節で、提案した IDRstab 法では偽収束を回避できることを数値実験により示す。5節で、偽収束の改善要因に関する数値的考察を行う。最後に、6節でまとめを行う。

2 IDRstab 法

本節では、IDRstab 法の概略を述べる。初期近似解を x_0 、対応する初期残差を $r_0 \equiv b - Ax_0$ とおく。IDRstab 法の残差 r_k は、以下を満たすように生成される。

$$r_k \in S(P_k, A, \tilde{R}_0) \equiv \{P_k(A)v \mid v \perp \mathcal{K}_k(A^*, \tilde{R}_0)\}.$$

ただし、整数 k は ℓ の倍数であり、 $P_k(\lambda)$ は ℓ 次の安定化多項式の積で表される k 次多項式

$$P_k(\lambda) = (1 - \sum_{i=1}^{\ell} \gamma_{i,k} \lambda^i) P_{k-\ell}(\lambda)$$

である。 $\mathcal{K}_k(A^*, \tilde{R}_0)$ は、 k 次のブロック Krylov 部分空間であり、 A の共役転置行列 A^* 、および固定された $n \times s$ 行列 $\tilde{R}_0$ に対して、

$$\mathcal{K}_k(A^*, \tilde{R}_0) \equiv \left\{ \sum_{j < k} (A^*)^j \tilde{R}_0 \tilde{\gamma}_j \mid \tilde{\gamma}_j \in \mathbb{C}^s \right\}$$

で定義される。 $S(P_k, A, \tilde{R}_0)$ は Sonneveld 部分空間と呼ばれ、 $k = 0, 1, \dots$ に対して帰納的にその次元が縮小する [7, 9, 10]。 $S(P_k, A, \tilde{R}_0)$ に属する残差 r_k は、IDR step と polynomial step と呼ばれる2つの過程を経ることで、 $S(P_{k+\ell}, A, \tilde{R}_0)$ に属する残差 $r_{k+\ell}$ へと更新され、この計算過程を1サイクルと数える。

1サイクルの始まりにおいて、残差 r_k と対応する近似解 x_k 、および $n \times s$ 行列 AU_k と対応する行列 U_k が得られているとする。ただし、 r_k と AU_k の各列ベクトルは $S(P_k, A, \tilde{R}_0)$ に属するものとする。ここで、行列 $\Pi_i^{(j)}$ を次のように定める。

$$\Pi_i^{(j)} \equiv I - A^i U_k^{(j-1)} \sigma_j^{-1} \tilde{R}_0^* A^{j-i}, \quad \sigma_j \equiv \tilde{R}_0^* A^j U_k^{(j-1)}, \quad i = 0, 1, \dots, j.$$

まず、IDR step では、写像 $\Pi_i^{(j)}$ を用いて ℓ 回の繰り返し計算が行われる。ただし、 j 回目 ($j = 1, 2, \dots, \ell$) の繰り返しで生成される要素を上付き添え字“(j)”で表し、特に $r_k^{(0)} \equiv r_k$ 、 $x_k^{(0)} \equiv x_k$ 、 $U_k^{(0)} \equiv U_k$ とする。残差は $\Pi_1^{(j)}$ を用いて、

$$r_k^{(j)} \equiv \Pi_1^{(j)} r_k^{(j-1)} = r_k^{(j-1)} - AU_k^{(j-1)} \tilde{\alpha}^{(j)}, \quad \tilde{\alpha}^{(j)} \equiv \sigma_j^{-1} (\tilde{R}_0^* A^{j-1} r_k^{(j-1)}) \quad (2)$$

と更新される。近似解を更新する漸化式は、 $x_k^{(j)} = x_k^{(j-1)} + U_k^{(j-1)} \tilde{\alpha}^{(j)}$ となる。また、 j 回目の繰り返しでは、ベクトル $A^i r_k^{(j)} \equiv \Pi_{i+1}^{(j)} A^i r_k^{(j-1)}$ 、 $i = 1, 2, \dots, j-1$ を同様に写像によって計

算し, $A^{j-1}\mathbf{r}_k^{(j)}$ に A を陽に掛けることで, ベクトル $A^j\mathbf{r}_k^{(j)}$ を得る. そして, 行列 $A^iU_k^{(j)}$, $i = 0, 1, \dots, j+1$ は, $A^jU_k^{(j)}$ の各列ベクトルが Krylov 部分空間 $\mathcal{K}_s(\Pi_j^{(j)}A, \Pi_j^{(j)}A^j\mathbf{r}_k^{(j)})$ の基底となるように構築される. 特に, 行列 $A^iU_k^{(j)}$, $i = 0, 1, \dots, j$ の各列ベクトルは写像 $\Pi_i^{(j)}$ によって計算され, $A^jU_k^{(j)}$ の各列ベクトルに A を陽に掛けることで, 行列 $A^{j+1}U_k^{(j)}$ が得られる (Figures 1-3 参照).

次に, polynomial step では, 残差 $\mathbf{r}_k^{(\ell)}$ に安定化多項式を施すことで

$$\mathbf{r}_{k+\ell} = (I - \gamma_{1,k}A - \dots - \gamma_{\ell,k}A^\ell)\mathbf{r}_k^{(\ell)} = \mathbf{r}_k^{(\ell)} - \gamma_{1,k}A\mathbf{r}_k^{(\ell)} - \dots - \gamma_{\ell,k}A^\ell\mathbf{r}_k^{(\ell)} \quad (3)$$

が得られる. ただし, 係数 $\gamma_{1,k}, \gamma_{2,k}, \dots, \gamma_{\ell,k}$ は, 残差ノルム $\|\mathbf{r}_{k+\ell}\|_2$ の最小化により決定される. 近似解 $\mathbf{x}_k^{(\ell)}$ を更新する漸化式は, $\mathbf{x}_{k+\ell} = \mathbf{x}_k^{(\ell)} + \gamma_{1,k}\mathbf{r}_k^{(\ell)} + \dots + \gamma_{\ell,k}A^{\ell-1}\mathbf{r}_k^{(\ell)}$ となる. 同様に, 行列 $A^jU_k^{(\ell)}$, $j = 0, 1$ に安定化多項式を施すことで, 行列 $A^jU_{k+\ell}^{(\ell)} = (I - \sum_{i=1}^{\ell} \gamma_{i,k}A^i)A^jU_k^{(\ell)}$, $j = 0, 1$ が得られる.

IDR step の j 回目の繰り返しで得られる残差 $\mathbf{r}_k^{(j)}$ は, 直交条件 $\mathbf{v}_k^{(j)} \perp \mathcal{K}_{k+j}(A^*, \tilde{R}_0)$ を満たすベクトル $\mathbf{v}_k^{(j)}$ を用いて, $\mathbf{r}_k^{(j)} = P_k(A)\mathbf{v}_k^{(j)}$ と書ける. 従って, polynomial step で得られる残差 $\mathbf{r}_{k+\ell} = (I - \sum_{i=1}^{\ell} \gamma_{i,k}A^i)P_k(A)\mathbf{v}_k^{(\ell)}$ は $\mathcal{S}(P_{k+\ell}, A, \tilde{R}_0)$ に属する (詳細は文献 [9] 参照).

3 偽収束を回避する IDRstab 法

我々は, 近似解と残差を式 (1) の形式で更新することで, 偽収束の改善を図る. ただし, 式 (1) を用いる場合, 従来の方法に比べて行列ベクトル積が余分に必要となる. そこで文献 [1, 2] に倣い, ベクトル更新の計算回数を削減することで, アルゴリズム全体の計算量を従来の方法と同程度に抑える.

3.1 IDR step

1 サイクルの始まりにおいて, 残差 $\mathbf{r}_k$ と対応する近似解 $\mathbf{x}_k$, および $n \times s$ 行列 U_k が得られているとする. このとき, $(j-1)$ 回の繰り返しを行うことで, 残差 $\mathbf{r}_k^{(j-1)}$ と対応する近似解 $\mathbf{x}_k^{(j-1)}$, およびベクトル $A^i\mathbf{r}_k^{(j-1)}$, $i = 1, 2, \dots, j-2$ と行列 $A^iU_k^{(j-1)}$, $i = 0, 1, 2, \dots, j-1$ が得られ, j 回目の繰り返しは以下のように行う. ただし, 文献 [1, 2] に倣い, 反復を行う前に行列 $A^*\tilde{R}_0$ を一度だけ計算し保持しておき, 行列 σ_j を従来とは異なる式

$$\sigma_j = (A^*\tilde{R}_0)^*A^{j-1}U_k^{(j-1)}$$

によって計算する.

まず, ベクトル $\mathbf{p}_k^{(j)} \equiv U_k^{(j-1)}\tilde{\alpha}^{(j)}$, $\tilde{\alpha}^{(j)} \equiv \sigma_j^{-1}((A^*\tilde{R}_0)^*A^{j-2}\mathbf{r}_k^{(j-1)})$ を計算し, 近似解と残差を次の漸化式によって更新する.

$$\mathbf{x}_k^{(j)} = \mathbf{x}_k^{(j-1)} + \mathbf{p}_k^{(j)}, \quad \mathbf{r}_k^{(j)} = \mathbf{r}_k^{(j-1)} - A\mathbf{p}_k^{(j)}. \quad (4)$$

ただし, ベクトル $A\mathbf{p}_k^{(j)}$ は $\mathbf{p}_k^{(j)}$ に A を陽に掛けることで得る. ここで, 残差を更新する従来の漸化式 (2) では, $j = 1$ のとき, 行列 $AU_k^{(0)}$ が必要であるが, 式 (4) では不要である

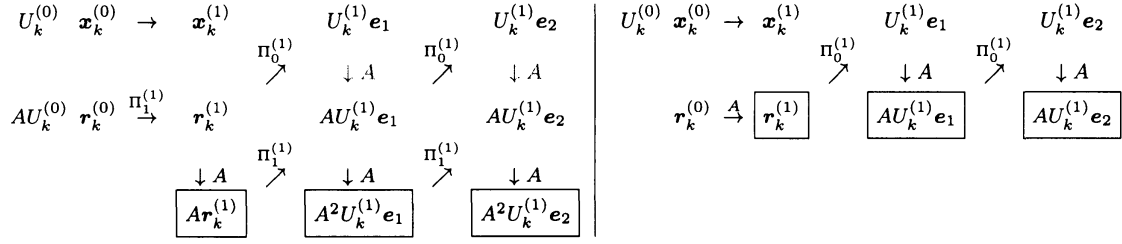


Figure 1: The schemes of the first repetition for the original IDRstab (on the left) and our variant (on the right).

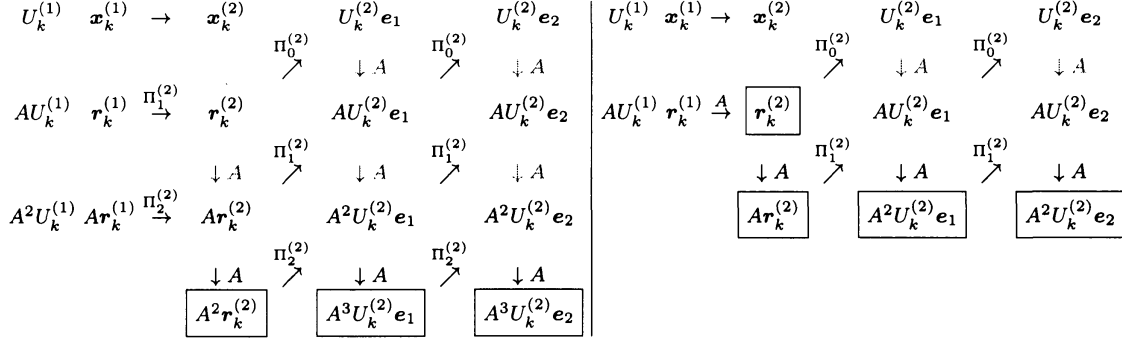


Figure 2: The schemes of the second repetition for the original IDRstab (on the left) and our variant (on the right).

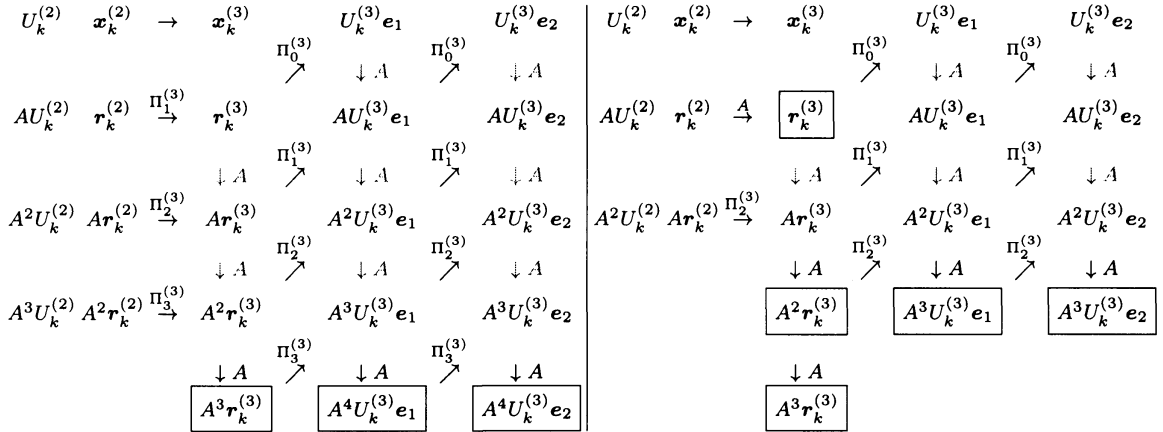


Figure 3: The schemes of the third repetition for the original IDRstab (on the left) and our variant (on the right).

(Figure 1 参照). また, ベクトル $\tilde{\alpha}^{(j)}$ の計算法も従来とは異なることに注意されたい. 次に, $j \geq 3$ のとき, ベクトル $A^i \mathbf{r}_k^{(j)}$, $i = 1, 2, \dots, j-2$ を, $\Pi_{i+1}^{(j)}$ を用いて

$$A^i \mathbf{r}_k^{(j)} \equiv \Pi_{i+1}^{(j)} A^i \mathbf{r}_k^{(j-1)} = A^i \mathbf{r}_k^{(j-1)} - A^{i+1} U_k^{(j-1)} \tilde{\alpha}^{(j)}$$

と計算する (Figure 3 参照). $j \geq 2$ のとき, $A^{j-2} \mathbf{r}_k^{(j)}$ に A を陽に掛けることで, ベクトル $A^{j-1} \mathbf{r}_k^{(j)}$ を得る (Figures 2, 3 参照). そして, 行列 $A^i U_k^{(j)}$, $i = 0, 1, \dots, j-1$ の 1 列目は, $\Pi_i^{(j)}$ を用いて

$$A^i U_k^{(j)} \mathbf{e}_1 \equiv \Pi_i^{(j)} A^i \mathbf{r}_k^{(j)} = A^i \mathbf{r}_k^{(j)} - A^i U_k^{(j-1)} \tilde{\beta}_1^{(j)}$$

とする。ただし、 $\tilde{\beta}_1^{(j)} \equiv \sigma_j^{-1} \rho_1^{(j)}$, $\rho_1^{(j)} \equiv (A^* \tilde{R}_0)^* A^{j-1} \mathbf{r}_k^{(j)}$ である。同様に、 $q < s$ に対して、 $A^{j-1} U_k^{(j)} \mathbf{e}_q$ に A を陽に掛けることで、ベクトル $\mathbf{c}_q^{(j)} \equiv A(A^{j-1} U_k^{(j)} \mathbf{e}_q)$ を計算し、 $A^i U_k^{(j)}$, $i = 0, 1, \dots, j-1$ の $(q+1)$ 列目は、 $\Pi_i^{(j)}$ を用いて

$$A^i U_k^{(j)} \mathbf{e}_{q+1} = \Pi_i^{(j)} A^{i+1} U_k^{(j)} \mathbf{e}_q = A^{i+1} U_k^{(j)} \mathbf{e}_q - A^i U_k^{(j-1)} \tilde{\beta}_{q+1}^{(j)}$$

と計算する。ただし、 $\tilde{\beta}_{q+1}^{(j)} \equiv \sigma_j^{-1} \rho_{q+1}^{(j)}$, $\rho_{q+1}^{(j)} \equiv (A^* \tilde{R}_0)^* \mathbf{c}_q^{(j)}$ である。最後に、 $j = \ell$ のとき、 $A^{\ell-1} \mathbf{r}_k^{(\ell)}$ に A を陽に掛けることで、ベクトル $A^\ell \mathbf{r}_k^{(\ell)}$ を得る (Figure 3 参照)。

Figures 1-3 に、従来の方法と提案する方法における 1-3 回目の繰り返しの計算過程を示す。ただし、 $(s, \ell) = (2, 3)$ とする¹。□ で囲まれたベクトルは、 A を陽に掛けることで得られる。また、文献 [9] に倣い、実装では数値的安定化のため、行列 $A^j U_k^{(j)}$ の各列ベクトルをアーノルドディ過程により正規直交化する。

3.2 polynomial step

IDR step の ℓ 回の繰り返しにより、残差 $\mathbf{r}_k^{(\ell)}$ と対応する近似解 $\mathbf{x}_k^{(\ell)}$ 、ベクトル $A^i \mathbf{r}_k^{(\ell)}$, $i = 1, 2, \dots, \ell$ 、および $\ell+1$ 個の $n \times s$ 行列 $A^i U_k^{(\ell)}$, $i = 0, 1, \dots, \ell$ が得られる。polynomial step では、係数 $\gamma_{1,k}, \gamma_{2,k}, \dots, \gamma_{\ell,k}$ を決定した後、ベクトル $\mathbf{p}_{k+\ell}^{(0)} \equiv \gamma_{1,k} \mathbf{r}_k^{(\ell)} + \gamma_{2,k} A \mathbf{r}_k^{(\ell)} + \dots + \gamma_{\ell,k} A^{\ell-1} \mathbf{r}_k^{(\ell)}$ を計算し、近似解と残差を次の漸化式によって更新する。

$$\mathbf{x}_{k+\ell} = \mathbf{x}_k^{(\ell)} + \mathbf{p}_{k+\ell}^{(0)}, \quad \mathbf{r}_{k+\ell} = \mathbf{r}_k^{(\ell)} - A \mathbf{p}_{k+\ell}^{(0)}. \quad (5)$$

ただし、ベクトル $A \mathbf{p}_{k+\ell}^{(0)}$ は $\mathbf{p}_{k+\ell}^{(0)}$ に A を陽に掛けることで得る。そして、行列 $U_k^{(\ell)}$ に安定化多項式を施すことで、行列 $U_{k+\ell}$ が得られ、1 サイクルを終了する。

以上の計算過程により、式 (1) の形式を用いると同時に、IDR step の j 回目の繰り返しでは行列 $A^{j+1} U_k^{(j)}$ を構築する必要がなく、polynomial step では行列 $A U_k^{(\ell)}$ から $A U_{k+\ell}$ への更新が不要となる。結果として、従来の方法に比べて 1 サイクルあたり $\ell(s^2 + 3s + \frac{1}{2}) - s - \frac{1}{2}$ 回のベクトル更新を削減することができる (計算量の詳細は文献 [2] 参照)。

提案する IDRstab 法の変形アルゴリズムを、Algorithm 1 に示す。ただし、アルゴリズムの表記は MATLAB コードに従う。 $q \leq s$ に対して、行列 $W = [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_s]$ の部分行列 $[\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_q]$ 、および列ベクトル $\mathbf{w}_q$ は、それぞれ $W_{(:,1:q)}$, $W_{(:,q)}$ と記述する。また、 $[W_0; W_1; \dots; W_j] \equiv [W_0^\top, W_1^\top, \dots, W_j^\top]^\top$ である。さらに、Algorithm 1 において、 $\mathbf{U}_i$, $\mathbf{V}_i$, $\mathbf{r}_i$, $\mathbf{u}_i$, $i = 0, 1, \dots, j$ は、それぞれ $\mathbf{U}$, $\mathbf{V}$, $\mathbf{r}$, $\mathbf{u}$ の成分を表し、 $\mathbf{U} = [\mathbf{U}_0; \mathbf{U}_1; \dots; \mathbf{U}_j]$, $\mathbf{V} = [\mathbf{V}_0; \mathbf{V}_1; \dots; \mathbf{V}_j]$, $\mathbf{r} = [\mathbf{r}_0; \mathbf{r}_1; \dots; \mathbf{r}_j]$, $\mathbf{u} = [\mathbf{u}_0; \mathbf{u}_1; \dots; \mathbf{u}_j]$ である。

4 数値実験

本節では、提案した IDRstab 法の有効性を数値実験により検証する。本節の数値実験は、HP の PC (Intel Core i7 2.67GHz CPU) で、Intel C++ 11.1.048 コンパイラの倍精度演算で行われた。初期近似解は $\mathbf{0}$ 、収束判定条件は $\|\mathbf{r}_k\|_2 < 10^{-12} \|\mathbf{b}\|_2$ とした。行列 $\tilde{R}_0$ は、各成分を区間 $(0, 1)$ 上の乱数で生成し、各列ベクトルを正規直交化して与えた。

¹一般に、 ℓ は数値的安定性の理由から偶数に選ばれるが [6]、ここでは便宜的に奇数の場合を示す。

Algorithm 1. Our proposed variant of IDRstab.

1. Select an initial guess $\mathbf{x}$ and an $(n \times s)$ matrix $\tilde{\mathbf{R}}_0$.
2. Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}$, $\mathbf{r} = [\mathbf{r}_0]$
3. *% Generate an initial $(n \times s)$ matrix $\mathbf{U} = [\mathbf{U}_0]$*
4. For $q = 1, 2, \dots, s$
5. if $q = 1$, $\mathbf{u}_0 = \mathbf{r}_0$, else, $\mathbf{u}_0 = \mathbf{A}\mathbf{u}_0$
6. $\tilde{\mu} = (\mathbf{U}_{0(:,1:q-1)})^* \mathbf{u}_0$, $\mathbf{u}_0 = \mathbf{u}_0 - \mathbf{U}_{0(:,1:q-1)} \tilde{\mu}$
7. $\mathbf{u}_0 = \mathbf{u}_0 / \|\mathbf{u}_0\|_2$, $\mathbf{U}_{0(:,q)} = \mathbf{u}_0$
8. End for
9. While $\|\mathbf{r}_0\|_2 > \text{tol}$
10. For $j = 1, 2, \dots, \ell$
11. *% The IDR step*
12. $\sigma = (\mathbf{A}^* \tilde{\mathbf{R}}_0)^* \mathbf{U}_{j-1}$
13. if $j = 1$, $\vec{\alpha} = \sigma^{-1}(\tilde{\mathbf{R}}_0^* \mathbf{r}_0)$, else, $\vec{\alpha} = \sigma^{-1}((\mathbf{A}^* \tilde{\mathbf{R}}_0)^* \mathbf{r}_{j-2})$
14. $\mathbf{x} = \mathbf{x} + \mathbf{U}_0 \vec{\alpha}$, $\mathbf{r}_0 = \mathbf{r}_0 - \mathbf{A}(\mathbf{U}_0 \vec{\alpha})$
15. For $i = 1, 2, \dots, j - 2$
16. $\mathbf{r}_i = \mathbf{r}_i - \mathbf{U}_{i+1} \vec{\alpha}$
17. End for
18. if $j > 1$, $\mathbf{r} = [\mathbf{r}; \mathbf{A}\mathbf{r}_{j-2}]$
19. For $q = 1, 2, \dots, s$
20. if $q = 1$, $\mathbf{u} = \mathbf{r}$, else, $\mathbf{u} = [\mathbf{u}_1; \mathbf{u}_2; \dots; \mathbf{u}_j]$
21. $\vec{\beta} = \sigma^{-1}((\mathbf{A}^* \tilde{\mathbf{R}}_0)^* \mathbf{u}_{j-1})$, $\mathbf{u} = \mathbf{u} - \mathbf{U} \vec{\beta}$, $\mathbf{u} = [\mathbf{u}; \mathbf{A}\mathbf{u}_{j-1}]$
22. $\tilde{\mu} = (\mathbf{V}_{j(:,1:q-1)})^* \mathbf{u}_j$, $\mathbf{u} = \mathbf{u} - \mathbf{V}_{(:,1:q-1)} \tilde{\mu}$
23. $\mathbf{u} = \mathbf{u} / \|\mathbf{u}_j\|_2$, $\mathbf{V}_{(:,q)} = \mathbf{u}$
24. End for
25. $\mathbf{U} = \mathbf{V}$
26. End for
27. $\mathbf{r} = [\mathbf{r}; \mathbf{A}\mathbf{r}_{\ell-1}]$
28. *% The polynomial step*
29. $\vec{\gamma} = [\gamma_1; \gamma_2; \dots; \gamma_\ell] = \arg \min_{\vec{\gamma}} \|\mathbf{r}_0 - [\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_\ell] \vec{\gamma}\|_2$
30. $\mathbf{x} = \mathbf{x} + [\mathbf{r}_0, \mathbf{r}_1, \dots, \mathbf{r}_{\ell-1}] \vec{\gamma}$, $\mathbf{r}_0 = \mathbf{r}_0 - \mathbf{A}([\mathbf{r}_0, \mathbf{r}_1, \dots, \mathbf{r}_{\ell-1}] \vec{\gamma})$
31. $\mathbf{U} = [\mathbf{U}_0 - \sum_{j=1}^{\ell} \gamma_j \mathbf{U}_j]$
32. End while

Table 1: Characteristic of coefficient matrices.

Matrices	n	nnz entries	Condition number	Application discipline
olm1000	1000	3996	1.5E+06	Computational fluid dynamics
sherman4	1104	3786	2.2E+03	Computational fluid dynamics
orsreg1	2395	17319	6.7E+04	Computational fluid dynamics
add20	2395	17319	1.2E+04	Circuit simulation
pde2961	2961	14585	6.4E+02	Partial differential equations
wang3	26064	177168	6.2E+03	Semiconductor device

nnz : The number of nonzero

4.1 数値例 1

まず, テスト行列として, Tim Davis: Sparse Matrix Collection [4] に収納されている非対称疎行列より olm1000, sherman4, orsreg1, add20, pde2961, wang3 を取り上げる. Table 1 に, 行列の次元数, 非零要素数, 条件数, 応用分野を示す. 右辺項 $\mathbf{b}$ は, 真の解 $\hat{\mathbf{x}} = (1, 1, \dots, 1)^T$ を与えて $\mathbf{b} = A\hat{\mathbf{x}}$ と設定した. パラメータは $(s, \ell) = (4, 4)$ とした.

Figure 4 に, olm1000 に対する各解法の相対残差 2 ノルム ($\|\mathbf{r}_k\|_2 / \|\mathbf{b}\|_2$), および真の相対残差 2 ノルム ($\|\mathbf{b} - A\mathbf{x}_k\|_2 / \|\mathbf{b}\|_2$) の振る舞いを示す. 一般に, グラフの横軸は行列ベクトル積数とする場合が多いが, 従来の IDRstab 法と提案した方法では, 1 サイクルあたりの行列ベクトル積数が異なる. そこで本節では, 各解法の収束速度を比較するため, 横軸をサイクル数とする. グラフの縦軸は相対残差 2 ノルムを表す. また Table 2 に, 各解法が収束までに要したサイクル数, 行列ベクトル積数, 計算時間 [秒], および反復終了時の真の相対残差 2 ノルムを, それぞれ “Cycles”, “MVs”, “Time[sec]”, “True res.” として示す.

Figure 4, Table 2 より, 次のことが言える. 従来の IDRstab 法では, 漸化式から求める残差ノルムと真の残差ノルムの振る舞いが異なり, 偽収束が発生している. 一方, 提案した IDRstab 法では偽収束を回避しており, 得られた近似解の精度は従来の方法に比べて向上している. 全ての行列に対して, 各解法が収束までに要したサイクル数, および計算時間は同程度である.

4.2 数値例 2

次に, 正方領域 $\Omega = [0, 1] \times [0, 1]$ における 2 次元楕円型偏微分方程式の境界値問題 [5] を取り上げる.

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} + D \left[\left(y - \frac{1}{2} \right) \frac{\partial u}{\partial x} + \left(x - \frac{1}{3} \right) \left(x - \frac{2}{3} \right) \frac{\partial u}{\partial y} \right] - 43\pi^2 u = G(x, y), \quad (6)$$

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

方程式 (6) を刻み幅 $h = 129^{-1}$ の 5 点中心差分近似により離散化し, 得られた線形方程式を解く. 係数行列の次元数は $128^2 = 16384$ であり, 非零要素数は 81408 となる. 右辺項

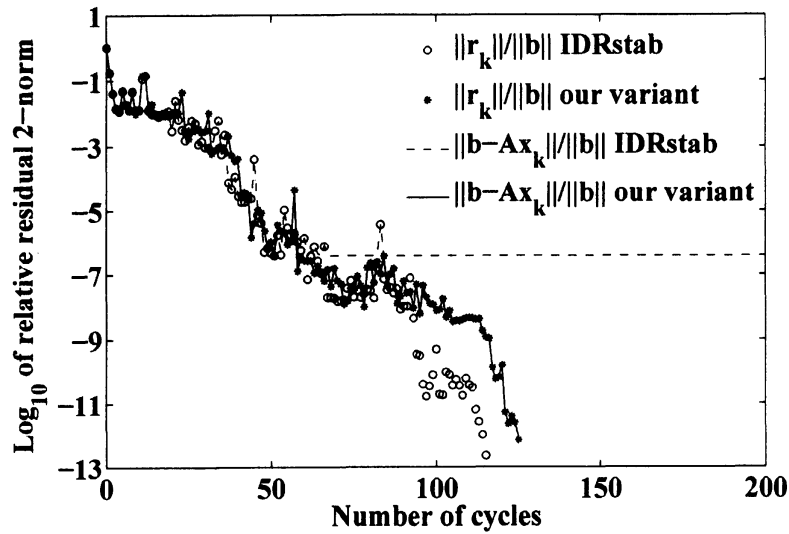


Figure 4: Convergence histories of the original IDRstab and our variant with $(s, \ell)=(4, 4)$ for olm1000.

Table 2: Numbers of cycles and MVs, computation times and true relative residual norms of the original IDRstab and our variant.

Matrix	Solver	Cycles	MVs	Time[sec]	True Res.
olm1000	IDRstab	115	2305	0.749	3.91E-07
	our variant	125	3129	0.749	8.78E-13
sherman4	IDRstab	10	205	0.104	1.61E-08
	our variant	11	279	0.099	2.44E-14
orsreg1	IDRstab	24	485	0.374	1.05E-09
	our variant	23	579	0.312	2.03E-12
add20	IDRstab	40	805	0.686	4.07E-10
	our variant	42	1054	0.671	9.08E-13
pde2961	IDRstab	17	345	0.344	1.68E-09
	our variant	18	454	0.328	1.41E-13
wang3	IDRstab	20	405	3.697	4.43E-10
	our variant	20	504	3.308	2.95E-13

は、方程式 (6) の解が $u(x, y) = 1 + xy$ となるように設定し、 $Dh = 2^{-1}$ とした。パラメータは $(s, \ell)=(4, 4), (6, 2), (2, 6)$ とした。

Figure 5 に、 $(s, \ell)=(2, 6)$ に対する各解法の相対残差 2 ノルム ($\|r_k\|_2/\|b\|_2$)、および真の相対残差 2 ノルム ($\|b - Ax_k\|_2/\|b\|_2$) の振る舞いを示す。グラフの横軸はサイクル数、縦軸は相対残差 2 ノルムを表す。また Table 3 に、各解法が収束までに要したサイクル数、行列ベクトル積数、計算時間 [秒]、および反復終了時の真の相対残差 2 ノルムを、それぞ

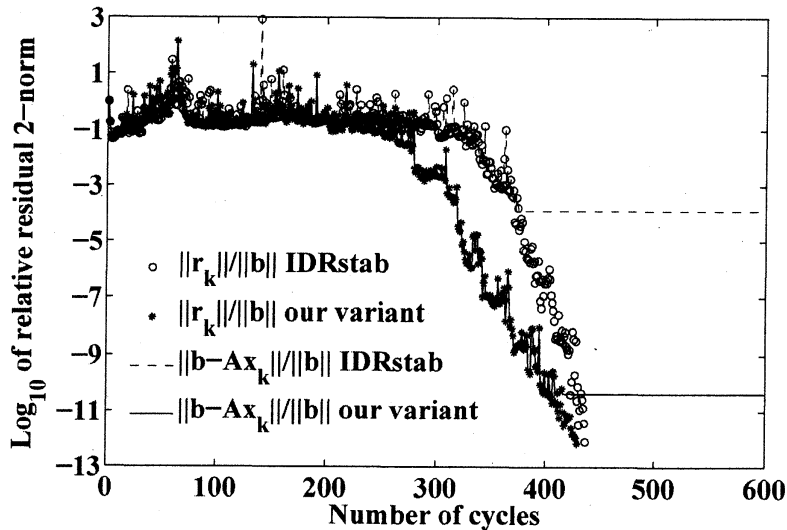


Figure 5: Convergence histories of the original IDRstab and our variant with $(s, \ell)=(2, 6)$.

Table 3: Numbers of cycles and MVs, computation times and true relative residual norms of the original IDRstab and our variant.

(s, ℓ)	Solver	Cycles	MVs	Time[sec]	True Res.
(4, 4)	IDRstab	226	4525	27.57	3.12E-05
	our variant	256	6404	27.13	1.86E-11
(6, 2)	IDRstab	315	4417	25.82	8.01E-07
	our variant	371	6313	25.83	4.67E-12
(2, 6)	IDRstab	436	7851	45.76	1.33E-04
	our variant	428	10702	37.72	4.27E-11

れ “Cycles”, “MVs”, “Time[sec]”, “True res.” として示す。

Figure 5, Table 3 より, 次のことが言える. 従来の IDRstab 法と提案した方法は, いずれも偽収束が発生している. ただし, 提案した IDRstab 法によって得られた近似解の精度は, 従来の方法に比べて大幅に向上しており, 偽収束が改善されていることが分かる. 提案した IDRstab 法において僅かに偽収束が発生した要因は, 残差ノルムの振動によるものと考えられる [8]. 各解法が収束までに要したサイクル数, および計算時間は同程度である.

5 改善要因の数値的考察

本節では, 提案した IDRstab 法ではなぜ偽収束を回避できるのか, その改善要因を2つの数値実験により考察し, ベクトル $p_k^{(j)}$ に A を陽に掛けることの重要性を示す.

以下、補助ベクトル $\mathbf{q}_k^{(j)}$, $j = 0, 1, \dots, \ell$ を次のようにおく.

$$\mathbf{q}_{k+\ell}^{(j)} \equiv \begin{cases} \sum_{i=1}^{\ell} \gamma_{i,k} A^i \mathbf{r}_k^{(\ell)} & (j = 0), \\ AU_{k+\ell}^{(j-1)} \bar{\alpha}^{(j)} & (j = 1, 2, \dots, \ell). \end{cases}$$

ただし, $\mathbf{q}_0^{(j)} \equiv AU_0^{(j-1)} \bar{\alpha}^{(j)}$, $j = 1, 2, \dots, \ell$ である. このとき, 従来の IDRstab 法の残差を更新する漸化式 (2), (3) は, それぞれ次のように書ける.

$$\mathbf{r}_k^{(j)} = \mathbf{r}_k^{(j-1)} - \mathbf{q}_k^{(j)} \quad (j = 1, 2, \dots, \ell), \quad \mathbf{r}_{k+\ell}^{(0)} = \mathbf{r}_k^{(\ell)} - \mathbf{q}_{k+\ell}^{(0)}. \quad (7)$$

一方, 提案した IDRstab 法では, 漸化式 (4), (5) より次のようになる.

$$\mathbf{r}_k^{(j)} = \mathbf{r}_k^{(j-1)} - A\mathbf{p}_k^{(j)} \quad (j = 1, 2, \dots, \ell), \quad \mathbf{r}_{k+\ell}^{(0)} = \mathbf{r}_k^{(\ell)} - A\mathbf{p}_{k+\ell}^{(0)}. \quad (8)$$

本節の数値実験は, GNU C++ 4.5.2 コンパイラの倍精度演算で行われた. テスト行列は, 対角行列 $A = \text{diag}(a_1, a_2, \dots, a_{1000})$, $a_i \equiv \sqrt{1 + 9.999(i-1)}$, $i = 1, 2, \dots, 1000$ を取り上げ, 収束判定条件は $\|\mathbf{r}_k\|_2 < 10^{-15} \|\mathbf{b}\|_2$ とした. パラメータは $(s, \ell) = (4, 4), (6, 2), (2, 6)$ とした. プロセッサ, 右辺項 $\mathbf{b}$, 初期近似解 $\mathbf{x}_0$, 行列 $\tilde{R}_0$ の設定については, 4.1 節と同様である.

5.1 丸め誤差の影響について

従来の IDRstab 法と提案した方法では, アルゴリズム中に現れるベクトル更新や内積の計算法が異なるため, そこから発生する丸め誤差の影響に違いが出る可能性がある. そのため, 両者の収束性を比較しただけでは, $\mathbf{p}_k^{(j)}$ に A を陽に掛けること自体の重要性を判断できない. そこでまず, 従来の IDRstab 法と提案した方法において, ベクトル更新や内積から発生する丸め誤差の影響に違いがないかを調べる. 具体的には, 従来の IDRstab 法の相対誤差ノルム

$$\frac{\|\mathbf{p}_k^{(j)} - \bar{\mathbf{p}}_k^{(j)}\|_2}{\|\bar{\mathbf{p}}_k^{(j)}\|_2}, \quad \frac{\|\mathbf{q}_k^{(j)} - \bar{\mathbf{q}}_k^{(j)}\|_2}{\|\bar{\mathbf{q}}_k^{(j)}\|_2} \quad (9)$$

の振る舞い, および提案した方法の相対誤差ノルム

$$\frac{\|\mathbf{p}_k^{(j)} - \bar{\mathbf{p}}_k^{(j)}\|_2}{\|\bar{\mathbf{p}}_k^{(j)}\|_2}, \quad \frac{\|A\mathbf{p}_k^{(j)} - \overline{A\mathbf{p}}_k^{(j)}\|_2}{\|\overline{A\mathbf{p}}_k^{(j)}\|_2} \quad (10)$$

の振る舞いをそれぞれ比較する. ただし, ベクトル $\bar{\mathbf{p}}_k^{(j)}$, $\bar{\mathbf{q}}_k^{(j)}$, $\overline{A\mathbf{p}}_k^{(j)}$ は, パッケージソフトウェア QD 2.3.12 [3] による擬似 8 倍精度演算で計算する. ここで, 擬似 8 倍精度演算では, 従来の IDRstab 法の $\bar{\mathbf{p}}_k^{(j)}$, $\bar{\mathbf{q}}_k^{(j)}$ と, 提案した方法の $\bar{\mathbf{p}}_k^{(j)}$, $\overline{A\mathbf{p}}_k^{(j)}$ は, それぞれ 16 桁一致することを付記しておく. また, 本実験では対角行列を用いるため, 行列ベクトル積から発生する丸め誤差の影響は十分に小さい.

計算結果: Figures 6, 7 に, 従来の IDRstab 法の相対誤差ノルム (9), および提案した方法の相対誤差ノルム (10) の履歴をそれぞれ示す. グラフの横軸は行列ベクトル積数, 縦軸は相対誤差 2 ノルムを表す. Figures 6, 7 より, 従来の IDRstab 法と提案した方法の相対誤差ノルムは, いずれも 10^{-7} を上回っており, 各パラメータ (s, ℓ) に対して, 両者は同様の振る舞いをしていることが分かる.

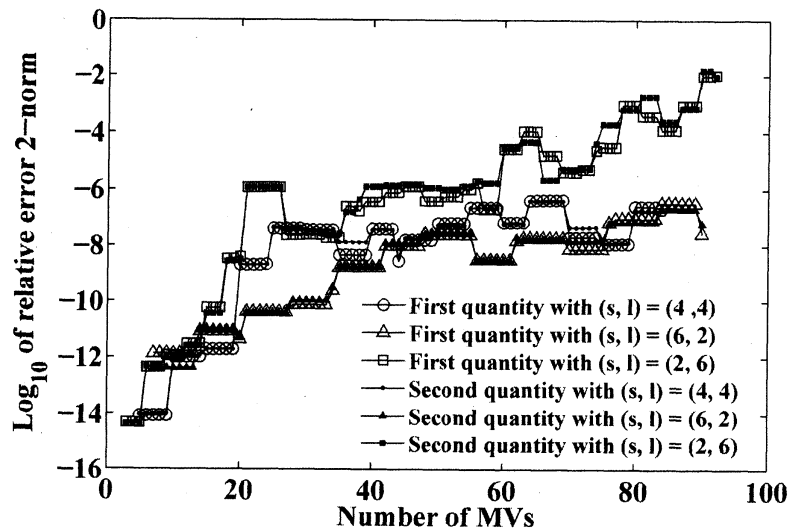


Figure 6: Histories of the first and second quantities in (9) for the original IDRstab.

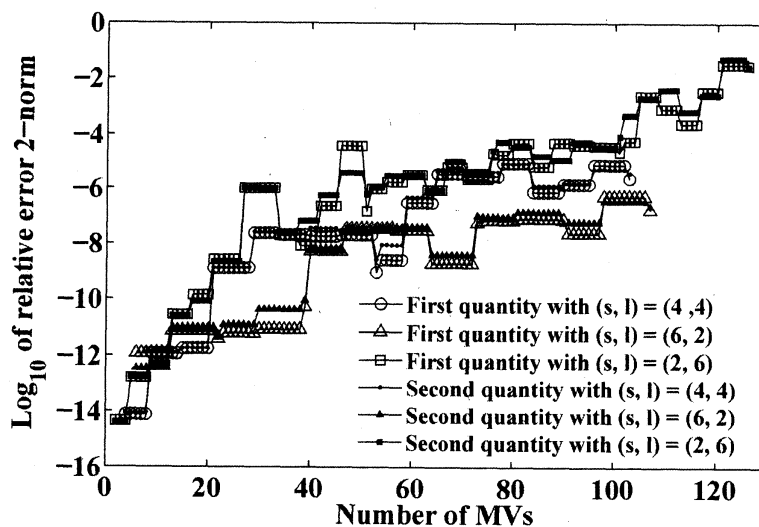


Figure 7: Histories of the first and second quantities in (10) for our variant.

5.2 行列 A を陽に掛けることの重要性

5.1節より，従来のIDRstab法の $p_k^{(j)}$, $q_k^{(j)}$ と，提案した方法の $p_k^{(j)}$, $Ap_k^{(j)}$ は，ベクトル更新や内積から発生する丸め誤差の影響を同程度に受けていると言える．従って，偽収束に関わる丸め誤差が発生する演算の違いは，写像により計算した $q_k^{(j)}$ を用いるのか， $p_k^{(j)}$ に A を陽に掛けるのか，という点だけであると判断できる．ここで，無限精度演算では，写像により計算した $q_k^{(j)}$ は $Ap_k^{(j)}$ に等しいが，倍精度演算では両者の間に差が生じる可能性がある．このとき，漸化式から求まる残差と真の残差の間にも差が生じ，偽収束が発生

すると推測される．そこで，写像により計算した $\mathbf{q}_k^{(j)}$ と $A\mathbf{p}_k^{(j)}$ の差を表す相対誤差ノルム

$$\frac{\|\mathbf{q}_k^{(j)} - A\mathbf{p}_k^{(j)}\|_2}{\|A\mathbf{p}_k^{(j)}\|_2} \quad (11)$$

の振る舞いを調べる．そして，ベクトル $\mathbf{p}_k^{(j)}$ に A を陽に掛けることで， $\mathbf{q}_k^{(j)}$ と $A\mathbf{p}_k^{(j)}$ の差が修正され，偽収束を回避できると予想される．本小節では， $\mathbf{q}_k^{(j)}$ と $A\mathbf{p}_k^{(j)}$ の差と，偽収束の相関を明らかにするため，アルゴリズム中で A を陽に掛ける箇所を段階的に変更し，相対誤差ノルム (11) と偽収束がどのように変化するかを考察する．従来の IDRstab 法と提案した方法の中間的な実装法としては，次の2つを考える．

phase 1: IDR step の1回目の繰り返しのみ漸化式 (8) によって残差を更新し，2回目以降の繰り返し，および polynomial step では，従来の漸化式 (7) によって残差を更新する．IDR step では，3節で述べたベクトル更新の削減を行い，1回目の繰り返しにおいて，行列 $AU_k^{(1)}$ は $U_k^{(1)}$ の各列ベクトルに A を陽に掛けることで得る (Figure 1 参照)．

phase 2: IDR step の各繰り返しで漸化式 (8) によって残差を更新し，polynomial step のみ従来の漸化式 (7) によって残差を更新する．phase 1 の場合と同様に，IDR step においてベクトル更新の削減を行い，行列 $AU_k^{(1)}$ は $U_k^{(1)}$ の各列ベクトルに A を陽に掛けることで得る (Figures 1-3 参照)．

Table 4 に，各解法で用いる漸化式の違いを示す．

計算結果: Figures 8-10 に，従来の IDRstab 法，phase 1，phase 2 に対する相対誤差ノルム (11) の履歴を示す．グラフの横軸は行列ベクトル積数，縦軸は相対誤差 2 ノルムを表す．ただし，対角行列 A とベクトル $\mathbf{p}_k^{(j)}$ の行列ベクトル積については，倍精度演算による結果と擬似 8 倍精度演算による結果が 16 桁一致することを確認している．よって，漸化式 (8) を用いる場合，相対誤差ノルム (11) は 0 であるとみなし，Figures 8-10 ではマシンイプシロン $2.22\text{E-}16$ を表示する．また，Table 5 に各解法に対する反復終了時の真の相対残差 2 ノルムを示す．

Figures 8-10 より，次のことが言える．従来の IDRstab 法では， $\mathbf{q}_k^{(j)}$ と $A\mathbf{p}_k^{(j)}$ の間に差が生じている．phase 1 の相対誤差ノルム (11) は，1 サイクルの始まりでは小さく，IDR step の繰り返しと共に増加している．1 サイクルの始まりで式 (11) の値が小さくなる要因

Table 4: The recursion formulas used in the original IDRstab, phase 1, 2 and our variant.

Solver	IDR step	polynomial step	Saving vector updates
IDRstab	(7)	(7)	no
phase 1	if $j = 1$, (8), else, (7)	(7)	yes
phase 2	(8)	(7)	yes
our variant	(8)	(8)	yes

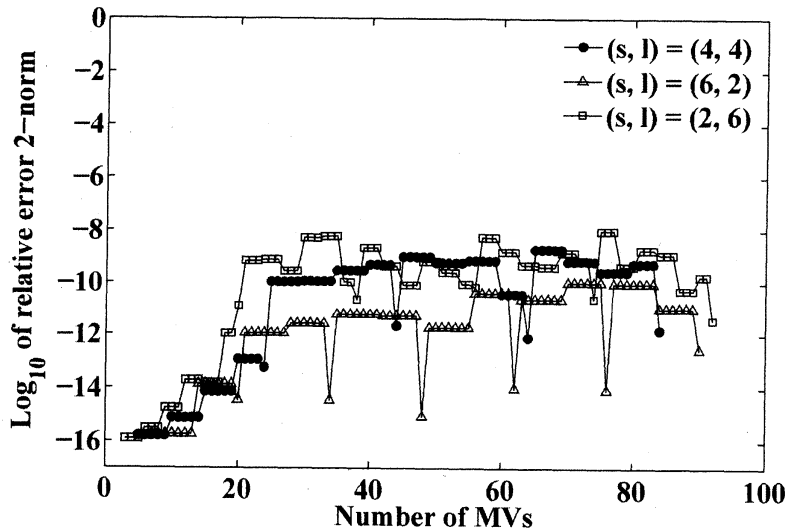


Figure 8: Histories of the quantity (11) for the original IDRstab.

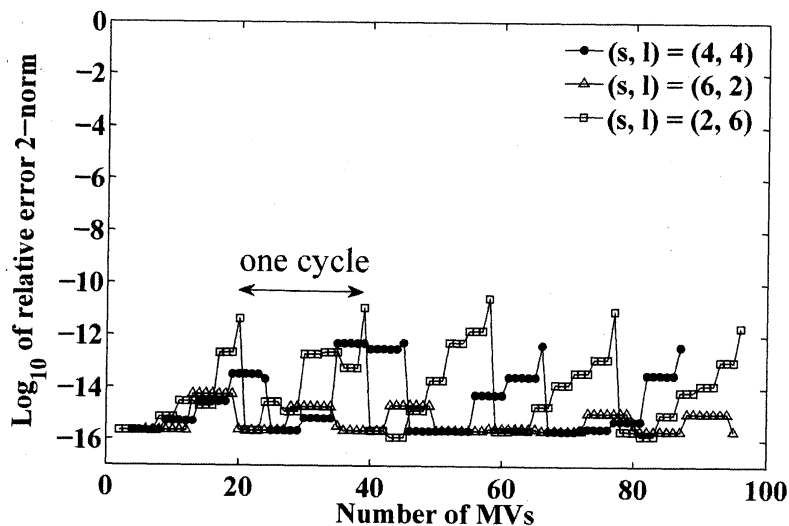


Figure 9: Histories of the quantity (11) for phase 1.

は、行列 $U_k^{(1)}$ に A を陽に掛けることで、サイクル毎に $q_k^{(j)}$ と $Ap_k^{(j)}$ の差が修正されるためであると考えられる。結果として、phase 1 の相対誤差ノルム (11) は、従来の IDRstab 法に比べて改善されている。phase 2 の相対誤差ノルム (11) は、 $(s, \ell) = (6, 2)$ のとき、十分に小さく、 $(s, \ell) = (4, 4)$, $(2, 6)$ のとき、従来の IDRstab 法と同程度である。提案した IDRstab 法では、すべての計算過程において、 $q_k^{(j)}$ と $Ap_k^{(j)}$ の差が 0 であるとみなせる。

以上の結果と Table 5 より、相対誤差ノルム (11) が小さいほど、漸化式から求まる残差ノルムと真の残差ノルムの差が小さいことが分かる。従って、ベクトル $q_k^{(j)}$ と $Ap_k^{(j)}$ が一致することが偽収束を回避するために重要であると結論づけられる。このことは、 $p_k^{(j)}$ に

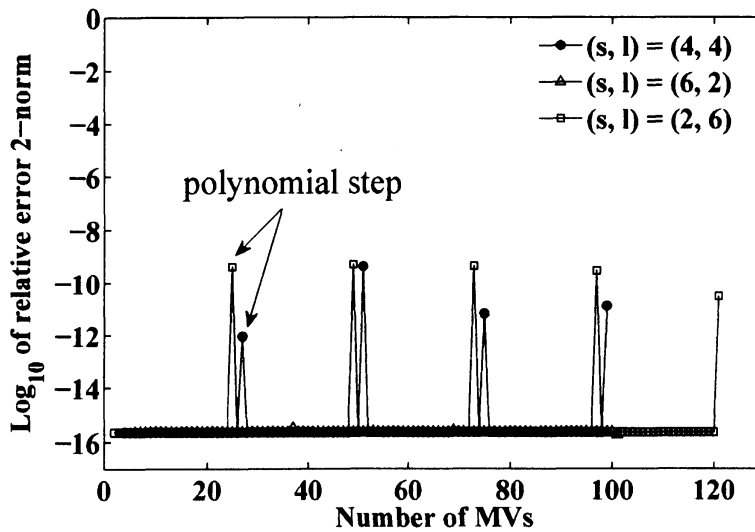


Figure 10: Histories of the quantity (11) for phase 2.

Table 5: True relative residual norms of the original IDRstab, phase 1, 2 and our variant.

Solver	(s, ℓ)		
	(4, 4)	(6, 2)	(2, 6)
IDRstab	4.62E-14	2.90E-15	3.11E-12
phase 1	1.24E-15	3.60E-16	1.89E-14
phase 2	1.89E-14	2.53E-16	1.90E-12
our variant	9.61E-16	2.18E-16	3.13E-16

A を陽に掛けることで、偽収束を改善できることを示唆している。

なお、従来のIDRstab法において、ベクトル $U_k^{(j-1)} \bar{\alpha}^{(j)}$ 、および $\sum_{i=1}^{\ell} \gamma_{i,k} A^{i-1} \mathbf{r}_k^{(\ell)}$ のそれぞれに A を陽に掛け、さらに各サイクルで行列 AU_k を U_k に A を陽に掛けて計算し直すことで、我々が提案した方法と同様に偽収束を回避できることも確認した。しかし、そのような実装法は計算量の観点から実用的ではないと言える。

6 まとめ

IDRstab法の偽収束を改善するため、残差を更新する漸化式を修正した変形アルゴリズムを提案した。従来のIDRstab法において偽収束が発生する場合に、提案した方法ではそれを回避し、得られる近似解の精度が向上することを数値実験により示した。そして、数値的考察により、ベクトル $\mathbf{p}_k$ に A を陽に掛けることが偽収束を回避するために重要であることを明らかにした。本論文では、偽収束の改善要因を調べるにあたり、実験的な手法によって考察を行ったが、丸め誤差解析により、偽収束を回避できることを理論的に示すことが今後の課題である。

謝辞

本研究の遂行にあたり, IDRstab 法の MATLAB コードのご提示, および有益なご助言を頂いた Gerard L. G. Sleijpen 博士 (Utrecht 大学) に深く御礼申し上げます.

参考文献

- [1] K. Aihara, K. Abe and E. Ishiwata, An alternative implementation of the IDRstab method saving vector updates, *JSIAM Letters*, **3** (2011), 69–72.
- [2] 相原研輔, 阿部邦美, 石渡恵美子, 近似解の精度を改善する IDRstab 法, 京都大学数理解析研究所講究録, **1791** (2012), 11–20.
- [3] D. H. Bailey, QD(C++/Fortran-90 double-double and quad-double package), <http://crd-legacy.lbl.gov/~dhbailey/>.
- [4] T. Davis, Sparse Matrix Collection, <http://www.cise.ufl.edu/research/sparse/matrices/>
- [5] W. Joubert, Lanczos methods for the solution of nonsymmetric systems of linear equations, *SIAM J. Matrix Anal. Appl.*, **13** (1992), 926–943.
- [6] G. L. G. Sleijpen and D. R. Fokkema, BiCGstab(ℓ) for linear equations involving unsymmetric matrices with complex spectrum, *Electron. Trans. Numer. Anal.*, **1** (1993), 11–32.
- [7] G. L. G. Sleijpen, P. Sonneveld and M. B. van Gijzen, Bi-CGSTAB as an induced dimension reduction method, *Appl. Numer. Math.*, **60** (2010), 1100–1114.
- [8] G. L. G. Sleijpen and H. A. van der Vorst, Reliable updated residuals in hybrid Bi-CG methods, *Comput.*, **2** (1996), 141–163.
- [9] G. L. G. Sleijpen and M. B. van Gijzen, Exploiting BiCGstab(ℓ) strategies to induce dimension reduction, *SIAM J. Sci. Comput.*, **32** (2010), 2687–2709.
- [10] P. Sonneveld and M. B. van Gijzen, IDR(s): a family of simple and fast algorithms for solving large nonsymmetric linear systems, *SIAM J. Sci. Comput.*, **31** (2008), 1035–1062.
- [11] M. Tanio and M. Sugihara, GBi-CGSTAB(s, L): IDR(s) with higher-order stabilization polynomials, *J. Comput. Appl. Math.*, **235** (2010), 765–784.