Vol.2014-HPC-143 No.20
2014/3/4

# Moe 版一般化積型反復法の並列性能評価

藤野清次[†1]　岩里洸介[†2]　Moe Thu Thu [†3]

本論文では，分散並列環境下での反復解法の同期回数の削減という立場から従来の反復解法の効率について議論したい．ここで取り上げる反復法は，元の GPBiCG 法，GPBiCG_AR 法，そして，今回考案した GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\beta$ 法の合計 4 種類である．そして，これらの解法の反復 1 回当りの同期回数に着目し，それらの回数を減らすことに成功した．数値実験によって，これらの反復解法の性能を比較し，収束性の違いや並列効率の違いを明らかにする．

# Parallel performance evaluation of Moe's version of generalized product typed iterative methods

Seiji Fujino [†1] Kousuke Iwasato [†2]
and Moe Thu Thu [†3]

We propose GPBiCG_AR method with single synchronization per one iteration for solving a linear system of equations. The proposed GPBiCG_AR method has two variants, i.e., with and without transpose matrix $A^T$. Through numerical experiments, we will make reveal that the proposed GPBiCG_AR method outperforms among other iterative methods on parallel computers.

## 1. はじめに

非対称行列を係数に持つ線形方程式 $A\boldsymbol{x} = \boldsymbol{b}$ に対する有用な解法の一つに積型反復法と呼ばれる解法がある．積型反復法では，連立一次方程式の近似解が，残差の収束から求められる．准残差最小化に基づく積型反復法は，パラメータ $\alpha_k, \beta_k$ 式の准残差の計算式を見直すことによって同期点を削減できた．並列計算では，プロセッサー間の同期をとることが重要であるが，高速化という視点からは課題も多い．一般に，反復法ではその算法中に内積演算が含まれ，内積演算で発生する同期点を削減する手法が村上らによって提案された[1)2)3)5)]．

積型反復法の中には，GPBiCG 法 (Generalized Product type of BiCG: 一般化積型双共役勾配法) と GPBiCG_AR 法 (GPBiCG with Associate Residual: 准残差を求める一般化積型双共役勾配法) などの反復法がある．GPBiCG 法と GPBiCG_AR 法の同期点は，それぞれ 3 回と 2 回必要である．GPBiCG_AR 法[4)] は，パラメータ $\alpha_k, \beta_k$ 式の准残差の計算式を見直すことによって同期点を 1 回に削減する．$\beta_k$ 式の准残差を見直した解法を GPBiCG_AR_$\beta$ 法，$\alpha_k$ 式の准残差の計算式を見直した解法を GPBiCG_AR_$\alpha$ 法と呼ぶ．

本研究では，元の GPBiCG 法，GPBiCG_AR 法，GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\beta$ 法の 4 種類の解法を比較し，各解法の収束性の違いや並列効率の違いを明らかにする．

## 2. 従来の GPBiCG_AR 法の算法と同期

GPBiCG_AR 法の算法を以下に示す．この解法は，残差ベクトルを更新する手続きを見直すことによって生まれた反復法である．ただし，残差ベクトル $(=R_{k+1}(\lambda)H_{k+1}(\lambda)\boldsymbol{r}_0)$ の更新式の中に加速パラメータ $\zeta_k, \eta_k$ が含まれなくなったので，新たに准残差ベクトル $(=R_{k+1}(\lambda)H_k(\lambda)\boldsymbol{r}_0)$ を定義し，その 2 ノルムの最小化から $\zeta_k, \eta_k$ を求めた反復法である．ここで，$R_{k+1}(\lambda)$ は Lanczos 多項式，$H_{k+1}(\lambda)$ はいわゆる安定化多項式を各々表す．プロセッサー間の同期をとる個所を太字で示す．6-8 ラインで 1 回，17 ラインで 1 回の合計 2 回の内積演算の部分である．

1. Let $\boldsymbol{x}_0$ be an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
2. Set $\tilde{\boldsymbol{r}}_0 = \boldsymbol{r}_0$, $\beta_{-1} = \boldsymbol{p}_{-1} = \boldsymbol{u}_{-1} = \boldsymbol{t}_{-1} = \boldsymbol{z}_{-1} = 0$
3. **for** $k = 0, 1, \ldots,$ **do**
4. $\quad \boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1}(\boldsymbol{p}_{k-1} - \boldsymbol{u}_{k-1})$,
5. $\quad A\boldsymbol{p}_k = A\boldsymbol{r}_k + \beta_{k-1}(A\boldsymbol{p}_{k-1} - A\boldsymbol{u}_{k-1})$,
6. $\quad \alpha_k = \dfrac{(\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_k)}{(\tilde{\boldsymbol{r}}_0, A\boldsymbol{p}_k)}$,
7. $\quad \zeta_k = \dfrac{(A\boldsymbol{z}_k, A\boldsymbol{z}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k) - (A\boldsymbol{z}_k, \boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{z}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, A\boldsymbol{z}_k)}$,

†1 九州大学情報基盤研究開発センター
Research Institute for Information Technology, Kyushu University
†2 九州大学大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University
†3 Toshin Unso Corp.

**8.** $\eta_k = \dfrac{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, \boldsymbol{r}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{z}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, A\boldsymbol{z}_k)}$,

9. (if $k = 0$, then $\zeta_0 = \dfrac{(A\boldsymbol{r}_0, \boldsymbol{r}_0)}{(A\boldsymbol{r}_0, A\boldsymbol{r}_0)}, \eta_0 = 0$),

10. $\boldsymbol{u}_k = \zeta_k A\boldsymbol{p}_k + \eta_k(\boldsymbol{t}_{k-1} - \boldsymbol{r}_k + \beta_{k-1}\boldsymbol{u}_{k-1})$,

11. $\boldsymbol{t}_k = \boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k$,

12. $\boldsymbol{z}_k = \zeta_k \boldsymbol{r}_k + \eta_k \boldsymbol{z}_{k-1} - \alpha_k \boldsymbol{u}_k$,

13. $A\boldsymbol{z}_k = \zeta_k A\boldsymbol{r}_k + \eta_k A\boldsymbol{z}_{k-1} - \alpha_k A\boldsymbol{u}_k$,

14. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k + \boldsymbol{z}_k$,

15. $\boldsymbol{r}_{k+1} = \boldsymbol{t}_k - A\boldsymbol{z}_k$,

16. **if** $||\boldsymbol{r}_{k+1}||/||\boldsymbol{r}_0|| \leq \epsilon$ **stop**,

**17.** $\beta_k = \dfrac{\alpha_k}{\zeta_k}\dfrac{(\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_{k+1})}{(\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_k)}$,

18. **end do**

## 3. 同期 1 回版の GPBiCG_AR 法の算法と同期

同期点を削減した 2 種類の GPBiCG_AR_$\beta$ 法と GPBiCG_AR_$\alpha$ 法の算法を以下に示す．削減する基本原理は文献[5] に拠ったが，GPBiCG_AR 法への適用は始めてである．

### 3.1 GPBiCG_AR_$\beta$ 法

GPBiCG_AR_$\beta$ 法の算法を以下に示す．算法の中で同期をとる個所は 9-12 ラインの内積演算である．9 ライン目の $\beta_k$ の計算式で転置行列 $A^T$ が必要になるのが特徴である．ただし，$A^T\tilde{\boldsymbol{r}}_0$ は反復のループに入る前に 1 度だけ (3 ライン目で) 計算して保存しておく．

1. Let $\boldsymbol{x}_0$ be an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
2. Set $\tilde{\boldsymbol{r}}_0 = \boldsymbol{r}_0$, $\beta_{-1} = \boldsymbol{t}_{-1} = \boldsymbol{u}_{-1} = \boldsymbol{p}_{-1} = \boldsymbol{z}_{-1} = 0$
3. Compute $A^T\tilde{\boldsymbol{r}}_0$
4. **for** $k = 0, 1, \ldots,$ **do**
5. $\quad \boldsymbol{v}_k = \boldsymbol{t}_{k-1} - \boldsymbol{r}_k + \beta_{k-1}\boldsymbol{u}_{k-1}$
6. $\quad \boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1}(\boldsymbol{p}_{k-1} - \boldsymbol{u}_{k-1})$,
7. $\quad A\boldsymbol{p}_k = A\boldsymbol{r}_k + \beta_{k-1}(A\boldsymbol{p}_{k-1} - A\boldsymbol{u}_{k-1})$,
8. $\quad$ **if** $||\boldsymbol{r}_{k+1}||/||\boldsymbol{r}_0|| \leq \epsilon$ **stop**,
9. $\quad \beta_k = -\dfrac{(A^T\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_k) - \alpha_k(A^T\tilde{\boldsymbol{r}}_0, A\boldsymbol{p}_k)}{(\tilde{\boldsymbol{r}}_0, A\boldsymbol{p}_k)}$,
10. $\quad \alpha_k = \dfrac{(\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_k)}{(\tilde{\boldsymbol{r}}_0, A\boldsymbol{p}_k)}$,
11. $\quad \zeta_k = \dfrac{(A\boldsymbol{z}_k, A\boldsymbol{z}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k) - (A\boldsymbol{z}_k, \boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{z}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, A\boldsymbol{z}_k)}$,
12. $\quad \eta_k = \dfrac{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, \boldsymbol{r}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{z}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, A\boldsymbol{z}_k)}$,
13. $\quad$ (if $k = 0$, then $\zeta_0 = \dfrac{(A\boldsymbol{r}_0, \boldsymbol{r}_0)}{(A\boldsymbol{r}_0, A\boldsymbol{r}_0)}, \eta_0 = 0$),
14. $\quad \boldsymbol{u}_k = \zeta_k A\boldsymbol{p}_k + \eta_k \boldsymbol{v}_k$,
15. $\quad \boldsymbol{t}_k = \boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k$,
16. $\quad \boldsymbol{z}_k = \zeta_k \boldsymbol{r}_k + \eta_k \boldsymbol{z}_{k-1} - \alpha_k \boldsymbol{u}_k$,
17. $\quad A\boldsymbol{z}_k = \zeta_k A\boldsymbol{r}_k + \eta_k A\boldsymbol{z}_{k-1} - \alpha_k A\boldsymbol{u}_k$,
18. $\quad \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k + \boldsymbol{z}_k$,
19. $\quad \boldsymbol{r}_{k+1} = \boldsymbol{t}_k - A\boldsymbol{z}_k$,
20. **end do**

### 3.2 GPBiCG_AR_$\alpha$ 法

GPBiCG_AR_$\alpha$ 法の算法を以下に示す．算法の中で同期をとる個所は 5-8 ラインである．この算法では 5 ライン目の $\beta_k$ の計算式で転置行列 $A^T$ を不要にした．したがって，ループに入る前の準備も通常とおり初期シャドウ残差ベクトル $\tilde{\boldsymbol{r}}_0$ の設定だけである．

1. Let $\boldsymbol{x}_0$ be an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
2. Set $\tilde{\boldsymbol{r}}_0 = \boldsymbol{r}_0$, $\beta_{-1} = \alpha_{-1} = \boldsymbol{p}_{-1} = \boldsymbol{w}_{-1} = \boldsymbol{u}_{-1}$
   $= \boldsymbol{t}_{-1} = \boldsymbol{z}_{-1} = 0$
3. **for** $k = 0, 1, \ldots,$ **do**
4. $\quad$ **if** $||\boldsymbol{r}_{k+1}||/||\boldsymbol{r}_0|| \leq \epsilon$ **stop**,
5. $\quad \beta_k = \dfrac{\alpha_{k-1}}{\zeta_{k-1}}\dfrac{(\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_k)}{(\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_{k-1})}$,
6. $\quad \alpha_k = \dfrac{(\tilde{\boldsymbol{r}}_0, \boldsymbol{r}_k)}{(\tilde{\boldsymbol{r}}_0, A\boldsymbol{p}_k) + \beta_k(\tilde{\boldsymbol{r}}_0, \boldsymbol{w}_{k-1})}$,
7. $\quad \zeta_k = \dfrac{(A\boldsymbol{z}_k, A\boldsymbol{z}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k) - (A\boldsymbol{z}_k, \boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{z}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, A\boldsymbol{z}_k)}$,
8. $\quad \eta_k = \dfrac{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, \boldsymbol{r}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(A\boldsymbol{z}_k, A\boldsymbol{z}_k) - (A\boldsymbol{z}_k, A\boldsymbol{r}_k)(A\boldsymbol{r}_k, A\boldsymbol{z}_k)}$,

9. (if $k = 0$, then $\zeta_0 = \dfrac{(A\boldsymbol{r}_0, \boldsymbol{r}_0)}{(A\boldsymbol{r}_0, A\boldsymbol{r}_0)}, \eta_0 = 0$),
10. $\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1}(\boldsymbol{p}_{k-1} - \boldsymbol{u}_{k-1})$,
11. $A\boldsymbol{p}_k = A\boldsymbol{r}_k + \beta_{k-1}\boldsymbol{w}_{k-1}$,
12. $\boldsymbol{u}_k = \zeta_k A\boldsymbol{p}_k + \eta_k(\boldsymbol{t}_{k-1} - \boldsymbol{r}_k + \beta_{k-1}\boldsymbol{u}_{k-1})$,
13. $\boldsymbol{w}_k = A\boldsymbol{p}_k + A\boldsymbol{u}_k$,
14. $\boldsymbol{t}_k = \boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k$,
15. $\boldsymbol{z}_k = \zeta_k \boldsymbol{r}_k + \eta_k \boldsymbol{z}_{k-1} - \alpha_k \boldsymbol{u}_k$,
16. $A\boldsymbol{z}_k = \zeta_k A\boldsymbol{r}_k + \eta_k A\boldsymbol{z}_{k-1} - \alpha_k A\boldsymbol{u}_k$,
17. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k + \boldsymbol{z}_k$,
18. $\boldsymbol{r}_{k+1} = \boldsymbol{t}_k - A\boldsymbol{z}_k$,
19. **end do**

## 4. 数値実験

### 4.1 計算機環境と計算条件

計算機環境と計算条件を示す.

**計算機環境**

(1) CPU：Intel Xeon X5570 (2.93 GHz), メモリ：24GB, キャッシュメモリ: 8192KB, OS：Red Hat Enterprise Linux Server release 5.6 (Tikanga), ホスト名：dell-3
(2) プログラム：Fortran90，コンパイラ：Intel Fortran Intel 64 Compiler ver.11.1
(3) コンパイルオプションは，“-O3” を使用した.
(4) 計算は倍精度浮動小数点演算で行い，時間計測は Fortran95 の組み込み関数 cpu_time を用いた.

収束判定値は相対残差の 2 ノルム：$||\boldsymbol{r}_{k+1}||_2/||\boldsymbol{r}_0||_2 \leq 10^{-10}$ とした．初期近似解 $\boldsymbol{x}_0$ はすべて 0，最大反復回数は 50000 回とした．行列は予め対角スケーリングによって対角項を 1 に正規化した．シャドウ残差ベクトル $\tilde{\boldsymbol{r}}_0$ は，初期残差ベクトル $\boldsymbol{r}_0$ と同じベクトルを与えた．右辺項は厳密解を $\hat{x} = (1, 1, \ldots 1)^T$ とし $\boldsymbol{b} = A\hat{\boldsymbol{x}}$ で作成した．ここで，$T$ は転置 (Transpose) 行列を表す．GPBiCG 法，GPBiCG_AR 法，GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\alpha$ 法の 4 種類の解法を調べた．各行列の計算を 5 回行い，最大と最小を除いた 3 回の反復回数と計算時間の平均をとった.

### 4.2 テスト行列

表 1 に 10 個のテスト行列の主な特徴を示す.

行列はフロリダ大学のデータベース[7] より選出した．表中の “$nnz$” は行列の全体の非零要素数 (number of nonzero)，“ave.$nnz$” は平均 (average) 非零要素数を各々表す．表記の順番は概ね非零要素数の多い順とした

**表 1** 10 個のテスト行列の主な特徴

| matrix | dimension | $nnz$ | ave. $nnz$ |
|---|---|---|---|
| raefsky2 | 3,242 | 293,551 | 90.5 |
| raefsky3 | 21,200 | 1,488,768 | 70.2 |
| sme3Da | 12,504 | 874,887 | 70.0 |
| sme3Db | 29,067 | 2,081,063 | 71.6 |
| sme3Dc | 42,930 | 3,148,656 | 73.3 |
| water_tank | 60,740 | 2,035,281 | 33.5 |
| poisson3Da | 13,514 | 352,762 | 26.1 |
| poisson3Db | 85,623 | 2,374,949 | 27.7 |
| xenon1 | 48,600 | 1,181,120 | 24.3 |
| epb3 | 84,617 | 463,625 | 5.5 |
| tmt_unsym | 917,825 | 4,584,801 | 5.0 |

### 4.3 実験結果

表 2 は，itr. は反復回数 (iterations)，to-t は合計 (total) 計算時間，ratio1 は反復回数の比，ratio2 は計算時間の比，ave-t は平均 (average) 計算時間，ave-ratio は反復 1 回当りの平均計算時間の比を表す．反復回数比，計算時間比，平均計算時間比は，GPBiCG 法を 1.0 としたときの GPBiCG 法と GPBiCG_AR 法，GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\beta$ 法の該当項目の各々の比を表す．“TRR” は真の相対残差 (True Relative Residual) の略で近似解 $\boldsymbol{x}_{k+1}$ に対する $\log_{10}(||\boldsymbol{b} - A\boldsymbol{x}_{k+1}||_2/||\boldsymbol{b} - A\boldsymbol{x}_0||_2)$ の値である．表 2 では，4 種類の解法の中で，各行列の反復回数と計算時間が最も改善されたものを太字で表記した．表中では，紙面横幅の制約から，GPBiCG 法は GP，GPBiCG_AR 法は単に AR，GPBiCG_AR_$\beta$ 法は AR_$\beta$，GPBiCG_AR_$\alpha$ 法は AR_$\alpha$ で略記する.

表 3 に 4 種類の反復法の収束性のまとめを示す.

#### 4.3.1 観　　察

表 2 から，GPBiCG_AR 法について以下の観察が得られた.

**表 2** 4 種類の解法の収束性比較

(a)matrix: raefsky2 から matrix: sme3Dc まで

| | method | itr. | tot-t [s.] | ratio1 | ratio2 | ave-t [ms.] | ave-ratio | TRR |
|---|---|---|---|---|---|---|---|---|
| raef-sky2 | GP | 345 | 0.308 | 1.000 | 1.000 | 0.893 | 1.000 | -10.4 |
| | AR | 322 | 0.241 | 0.933 | 0.782 | 0.748 | 0.838 | -10.5 |
| | AR_$\beta$ | 318 | 0.235 | **0.922** | **0.763** | 0.739 | **0.828** | -10.3 |
| | AR_$\alpha$ | 321 | 0.241 | 0.930 | 0.782 | 0.751 | 0.841 | -10.9 |
| raef-sky3 | GP | 4,851 | 26.402 | **1.000** | **1.000** | 5.443 | 1.000 | -10.3 |
| | AR | 4,972 | 26.908 | 1.025 | 1.019 | 5.412 | 0.994 | -10.1 |
| | AR_$\beta$ | 4,990 | 26.729 | 1.029 | 1.012 | 5.357 | **0.984** | -10.0 |
| | AR_$\alpha$ | 5,079 | 27.352 | 1.047 | 1.036 | 5.385 | 0.989 | -10.0 |
| sme-3Da | GP | 2,749 | 9.321 | 1.000 | 1.000 | 3.391 | 1.000 | (-7.1) |
| | AR | 2,193 | 7.286 | **0.798** | **0.782** | 3.322 | 0.980 | -9.3 |
| | AR_$\beta$ | 2,233 | 7.353 | 0.812 | 0.789 | 3.293 | **0.971** | -9.3 |
| | AR_$\alpha$ | 2,233 | 7.569 | 0.812 | 0.812 | 3.390 | 1.000 | -9.3 |
| sme-3Db | GP | 4,160 | 40.712 | 1.000 | 1.000 | 9.787 | 1.000 | (-6.4) |
| | AR | 2,779 | 27.157 | 0.668 | 0.667 | 9.772 | 0.999 | -9.0 |
| | AR_$\beta$ | 2,618 | 25.547 | **0.629** | **0.628** | 9.758 | **0.997** | -9.0 |
| | AR_$\alpha$ | 2,736 | 27.046 | 0.658 | 0.664 | 9.885 | 1.010 | -9.0 |
| sme-3Dc | GP | 6,355 | 97.650 | 1.000 | 1.000 | 15.366 | 1.000 | (-5.4) |
| | AR | 4,735 | 72.729 | 0.745 | 0.745 | 15.360 | 1.000 | -9.5 |
| | AR_$\beta$ | 4,375 | 67.197 | **0.688** | **0.688** | 15.359 | **1.000** | -9.6 |
| | AR_$\alpha$ | 4,145 | 64.460 | 0.652 | 0.660 | 15.551 | 1.012 | -9.6 |

(b)matrix: water_tank から matrix: epb3 まで

| | method | itr. | tot-t [s.] | ratio1 | ratio2 | ave-t [ms.] | ave-ratio | TRR |
|---|---|---|---|---|---|---|---|---|
| water_tank | GP | 2,049 | 17.276 | 1.000 | 1.000 | 8.431 | 1.000 | -10.0 |
| | AR | 2,018 | 17.032 | **0.985** | **0.986** | 8.440 | 1.001 | -10.3 |
| | AR_$\beta$ | 2,038 | 17.034 | 0.995 | 0.986 | 8.358 | **0.991** | -10.0 |
| | AR_$\alpha$ | 2,083 | 17.599 | 1.017 | 1.019 | 8.449 | 1.002 | -10.0 |
| pois-3Da | GP | 118 | 0.201 | **1.000** | 1.000 | 1.703 | 1.000 | -10.0 |
| | AR | 124 | 0.182 | 1.051 | 0.905 | 1.468 | 0.862 | -10.0 |
| | AR_$\beta$ | 126 | 0.179 | 1.068 | **0.891** | 1.421 | **0.834** | -10.0 |
| | AR_$\alpha$ | 126 | 0.187 | 1.068 | 0.930 | 1.484 | 0.871 | -10.0 |
| pois-3Db | GP | 269 | 3.847 | **1.000** | **1.000** | 14.301 | 1.000 | -10.1 |
| | AR | 310 | 4.446 | 1.152 | 1.156 | 14.342 | 1.003 | -10.0 |
| | AR_$\beta$ | 290 | 4.083 | 1.078 | 1.061 | 14.079 | **0.984** | -10.0 |
| | AR_$\alpha$ | 300 | 4.370 | 1.115 | 1.136 | 14.567 | 1.019 | -10.0 |
| xen-on1 | GP | 1,048 | 5.587 | 1.000 | 1.000 | 5.331 | 1.000 | -10.0 |
| | AR | 797 | 4.195 | **0.760** | **0.751** | 5.263 | 0.987 | -10.0 |
| | AR_$\beta$ | 873 | 4.541 | 0.833 | 0.813 | 5.202 | **0.976** | -10.0 |
| | AR_$\alpha$ | 978 | 5.127 | 0.933 | 0.918 | 5.242 | 0.983 | -10.0 |
| epb3 | GP | 3,362 | 12.192 | 1.000 | 1.000 | 3.626 | 1.000 | -10.3 |
| | AR | 3,381 | 12.419 | 1.006 | 1.019 | 3.673 | 1.013 | -10.1 |
| | AR_$\beta$ | 3,163 | 11.273 | 0.941 | 0.925 | 3.564 | **0.983** | -10.0 |
| | AR_$\alpha$ | 3,052 | 11.212 | **0.908** | **0.920** | 3.674 | 1.013 | -10.2 |

**表 3** 4 種類の反復法の収束性のまとめ

| method | itr. ratio | time ratio | ave-time ratio |
|---|---|---|---|
| GP | 1.0 | 1.0 | 1.0 |
| AR | 0.912 | 0.881 | 0.968 |
| AR_$\beta$ | **0.899** | **0.856** | **0.955** |
| AR_$\alpha$ | 0.914 | 0.888 | 0.974 |

(1) GPBiCG_AR 法は，反復回数が行列 10 個中 6 個，計算時間が行列 10 個中 7 個改善した．
(2) GPBiCG_AR 法は，行列 sme3Db で反復回数と計算時間がともに 33%減少した．

同様に，GPBiCG_AR_$\beta$ 法について以下の観察が得られた．

(1) GPBiCG_AR_$\beta$ 法は，反復回数が行列 10 個中 7 個，計算時間が行列 10 個中 8 個改善した．同様に，行列 sme3Db で計算時間が 37%減少し，行列 sme3Dc でも計算時間が 31%短くなった．
(2) GPBiCG_AR_$\beta$ 法は最も収束性が高いことがわかった．

さらに，表 2 から，GPBiCG_AR_$\alpha$ 法についても以下の観察が得られた．

(1) GPBiCG_AR_$\alpha$ 法は，反復回数が行列 10 個中 6 個，計算時間が行列 10 個中 7 個も改善した．
(2) 同様に，行列 sme3Db で計算時間が 34%減少し，行列 sme3Dc でも計算時間が 35%短くなった．

表 3 に示した結果から，元の GPBiCG 法に比べて，GPBiCG_AR 法，GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\alpha$ 法は，反復回数と計算時間が大きく向上したことがわかった．3 種類の中では，GPBiCG_AR_$\beta$ 法が最も大きな効果があることがわかった．

また，行列 sme3Da, sme3Db, sme3Dc に対して，元の GPBiCG 法の TRR の値 (表中の括弧をつけた数字) が，反復法の要求精度 (=$10^{-10}$) と比較して極端に悪化していることがわかる．一方，GPBiCG_AR 法，GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\alpha$ 法の TRR は，満足できる精度の近似解がいずれも得られたことがわかる．

### 4.4 並列性能評価

#### 4.4.1 数 値 実 験

計算機環境と計算条件は以下の通りである．

- 計算機は九州大学情報基盤研究開発センターの計算機 CPU：Intel Xeon E5-2680 (2.7 GHz), メモリ：128GB, 8MB L3 Cache, 8cores×2, OS：Red Hat Enterprise Linux Server Release 6.1, (ホスト名：tatara) を使用した．
- 収束判定は相対残差の 2 ノルム：$||\boldsymbol{r}_{k+1}||_2/||\boldsymbol{r}_0||_2 \leq 10^{-8}$ とした．
- 初期近似解 $\boldsymbol{x}_0$ はすべて $\mathbf{0}$，最大反復回数は 50000 回とした．行列は予め対角スケーリングによって対角項を 1 に正規化した．
- 実験は，GPBiCG 法，GPBiCG_AR 法，GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\alpha$ 法の 4 解法で行った．
- シャドウ残差ベクトル $\boldsymbol{r}_0^*$ は，初期残差ベクトル $\boldsymbol{r}_0$ を与えた．右辺項は厳密解を $\hat{\boldsymbol{x}} = (1, 1, ...1)^T$ とし $\boldsymbol{b} = A\hat{\boldsymbol{x}}$ で作成した．
- 数値実験は，5 回計測し，最大最小の値を除いた 3 回の平均値を採用した．コンパイルオプションは，“-Kfast” を使用した．

表 4 に実験結果を示す．表中において，nPE はプロセッサー数，Mv は行列ベクトル積の計算回数，t1 は行列ベクトル積の経過時間，t2 はその他の経過時間，tot-t は合計経過時間，speedup は台数効果，ave.-t は反復 1 回当りの平均経過時間，そして “TRR” は得られた近似解に対する真の相対残差の値：$\log_{10}(||\boldsymbol{b} - A\boldsymbol{x}_{k+1}||_2/||\boldsymbol{b} - A\boldsymbol{x}_0||_2)$ を各々表す．

**表 4** 反復法の並列性能 (行列: tmt_unsym)

| | nPE | Mv | t1 [s] | t2 [s] | tot-t [s] | ave.-t [ms] | speed -up | TRR |
|---|---|---|---|---|---|---|---|---|
| GP | 1 | 9,474 | 140.51 | 73.94 | 214.40 | 34.5 | 1.00 | -4.5 |
| | 16 | 13,642 | 23.77 | 2.71 | 26.48 | 10.2 | 8.10 | -7.5 |
| | 32 | 9,452 | 7.51 | 4.96 | 12.47 | 39.7 | 17.19 | -7.1 |
| | 64 | 10,626 | 3.26 | 1.74 | 5.02 | 34.6 | 42.72 | -5.0 |
| | 128 | 10,746 | 1.73 | 1.14 | 2.86 | 39.7 | 74.91 | -5.3 |
| | 256 | 8,574 | 0.85 | 0.79 | 1.64 | 48.3 | 130.68 | -5.5 |
| AR | 1 | 11,252 | 168.84 | 67.12 | 235.97 | 28.4 | 1.00 | -7.2 |
| | 16 | 9,362 | 16.42 | 0.98 | 17.41 | 5.6 | 13.56 | -8.5 |
| | 32 | 9,134 | 7.18 | 3.19 | 10.42 | 30.6 | 22.65 | -7.6 |
| | 64 | 8,942 | 2.68 | 1.08 | 3.78 | 28.7 | 62.48 | -7.1 |
| | 128 | 8,940 | 1.43 | 0.71 | 2.14 | 33.3 | 110.06 | -7.6 |
| | 256 | 9,278 | 0.91 | 0.62 | 1.53 | 40.3 | 154.36 | -7.1 |
| AR_$\beta$ | 1 | 8,906 | 151.64 | 43.86 | 195.50 | 22.4 | 1.00 | -6.7 |
| | 16 | 10,560 | 19.28 | 0.99 | 20.25 | 4.9 | 9.65 | -8.3 |
| | 32 | 8,540 | 8.08 | 3.09 | 11.19 | 27.7 | 17.47 | -7.2 |
| | 64 | 8,604 | 2.88 | 1.09 | 3.98 | 27.3 | 49.13 | -7.2 |
| | 128 | 9,082 | 1.57 | 0.63 | 2.23 | 28.4 | 87.77 | -7.4 |
| | 256 | 8,854 | 0.93 | 0.45 | 1.36 | 32.9 | 143.36 | -7.0 |
| AR_$\alpha$ | 1 | 8,940 | 138.20 | 53.29 | 191.65 | 27.8 | 1.00 | -7.5 |
| | 16 | 9,732 | 17.33 | 1.17 | 18.46 | 6.3 | 10.38 | -8.0 |
| | 32 | 8,862 | 7.28 | 3.93 | 11.22 | 35.0 | 17.08 | -7.6 |
| | 64 | 9,318 | 2.97 | 1.38 | 4.33 | 31.8 | 44.26 | -7.5 |
| | 128 | 9,132 | 1.55 | 0.78 | 2.33 | 33.6 | 82.22 | -7.5 |
| | 256 | 10,204 | 1.05 | 0.72 | 1.76 | 40.8 | 109.12 | -7.2 |

## 5. まとめと今後の課題

本研究では，パラメータ $\alpha_k, \beta_k$ の計算式を変更することによって，反復 1 回当りの同期点を 2 回から 1 回に削減した．GPBiCG_AR 法，GPBiCG_AR_$\beta$ 法，GPBiCG_AR_$\alpha$ 法は，元の GPBiCG 法より計算時間が平均 10%以上短くなった．特に，GPBiCG_AR_$\beta$ 法は，反復回数，計算時間ともに解法の中で最も優れていた．並列効率については，もっと多くの数値実験をする必要があると思われる．

## 参 考 文 献

1) S. Fujino, K. Murakami, K. Iwasato: A proposal of Single-Synchronized Solver suited to large scale linear systems on parallel computers with distributed memory, Parco 2013 (Int. conference on Parallel computing), Munchen 10-13, September, 2013.
2) S. Fujino, K. Murakami, K. Iwasato: A Single-Synchronized BiCGSafe method suited to large scale linear systems on parallel computers with distributed memory, IRCITCS (Int. Research conference on Information Technology and computer sciences), Kuala Lumpur, Malaysia, 26-28, September, 2013.
3) S. Fujino, K. Murakami, K. Iwasato: A parallel variant of BiCGStar-plus method reduced to single global synchronization, AsiaSim 2013, Concorde Hotel, Singapore, 6-8, November, 2013.
4) Moe Thu Thu, S. Fujino: Stability of GPBiCG_AR method based on minimization of associate residual, Journal of ASCM, **5081**(2008), pp.108-120.
5) 村上啓一：同期点を削減した並列計算向き Krylov 部分空間法の提案，九州大学大学院システム情報科学府，修士論文，pp.23-26，2013.
6) 関本幹：IDR(s)-SOR 法を用いた可変的前処理付き GCR($m$) 法および GMRES($m$) 法の収束性，九州大学工学部電気情報工学科，卒業論文，pp.57-61，2010.
7) Univ. of Florida Sparse Matrix Collection: http://www.cise.ufl.edu/research/sparse/matrices/index.html