Proceedings of the International MultiConference of Engineers and Computer Scientists 2012 Vol II,
IMECS 2012, March 14 - 16, 2012, Hong Kong

Performance Evaluation of GPBiCGSafe Method without Reverse-Ordered Recurrence for Realistic Problems

Seiji Fujino*, Takashi Sekimoto †

Abstract— **GPBiCG method is an attractive iterative method for the solution of a linear system of equations with nonsymmetric coefficient matrix. However, we meet often with instability of convergence when GPBiCG method is adopted in the solution of realistic problems. In this paper, we consider a new algorithm with minimization of the associate residual and without reverse-ordered recurrence in the algorithm. We refer to a new iterative method as GPBiCG with safety convergence (abbreviated as GPBiCGSafe) method. Moreover we will support that GPBiCGSafe method yields safety convergence through numerical experiments.**

Keywords: GPBiCG method, GPBiCG_AR method, nonsymmetric coefficient matrix, Associate residual, Reverse-ordered recurrence, GPBiCGSafe method

1 Introduction

Generalized Product Bi-Conjugate Gradient (abbreviated as GPBiCG) method [7] is an attractive iterative method for the solution of linear systems with a nonsymmetric coefficient matrix. However, the popularity of GPBiCG method has diminished over time except for the context of limited field of analysis because of instability of convergence. Therefore, some variants of GPBiCG method which have stability of convergence and robustness compared with the original GPBiCG method have been proposed.

We proposed a safety variant (abbreviated as BiCGSafe) of Generalized Product type Bi-CG method from the viewpoint of reconstruction of the residual polynomial and determination of two acceleration parameters ζ_n and η_n [3]. It embodied that a particular strategy for remedying instability of convergence, acceleration parameters are decided from minimization of the associate residual of 2-norm [5]. However, we could not reveal the reason of instability of GPBiCG method because of reconstruction of the algorithm in a different way. Though both convergence rate and stability of BiCGSafe method were fairly improved, instability itself of GPBiCG method could not corresponds directly to its algorithm.

*Research Institute for Information Technology, Kyushu University, Email:fujino@kyushu-u.ac.jp

†Graduate School of Information Science and Electrical Engineering, Kyushu University

In this paper, we consider a new algorithm based on minimization of the associate residual and without reverse-ordered recurrence. We refer to as GPBiCG with safety convergence (hereafter abbreviated as GPBiCGSafe) method. Moreover we will make verification of stability of GPBiCGSafe method, and make it clear that the reason of instability of GPBiCG method corresponds to reverse-ordered recurrence adopted in the algorithm of GPBiCG method. We will verify that GPBiCGSafe method yields safety and robustness of convergence through many numerical experiments.

2 Brief description of product-type methods

We consider iterative methods for solving a linear system of equations

$$A\boldsymbol{x} = \boldsymbol{b} \tag{1}$$

where $A \in R^{N\times N}$ is a given unsymmetric matrix, and $\boldsymbol{x}$, $\boldsymbol{b}$ is a solution vector and right-hand side vector, respectively. When A is a large, sparse matrix which arises from realistic problems, efficient solution of (1) is substantially very difficult. This difficulty has led to the development of a rich variety of generalized CG type methods having varying degrees of success (see, e.g., [6]).

The biconjugate gradient (so-called BiCG) method based of the Lanczos algorithm is a crucial example of a generalized CG method. In many cases, the Lanczos algorithm give some of the fastest solution times and stability of convergence among all generalized CG methods. The Lanczos algorithm, however, is known to break down during iteration process. In practice, the occurrence of breakdown can cause failure to irregularly converge to the solution of (1). The fact that the Lanczos algorithms perform well in some cases but fail in others heightens the need for further insight and development of the Lanczos type iterative methods.

We note that the basic recurrence relations between Lanczos polynomials $R_n(\lambda)$ and $P_n(\lambda)$ hold as follows:

$$\begin{aligned} R_0(\lambda) &= 1, \quad P_0(\lambda) = 1, && (2)\\ R_{n+1}(\lambda) &= R_n(\lambda) - \alpha_n \lambda P_n(\lambda), && (3)\\ P_{n+1}(\lambda) &= R_{n+1}(\lambda) + \beta_n P_n(\lambda), && (4)\\ & n = 0, 1, 2, \ldots. \end{aligned}$$

ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Then we can introduce the three-term recurrence relations for Lanczos polynomials $R_n(\lambda)$ only by eliminating $P_n(\lambda)$ from (2) and (4) as follows:

$$\begin{aligned} R_0(\lambda) &= 1,\ R_1(\lambda) = (1-\alpha_0\lambda)R_0(\lambda) && (5)\\ R_{n+1}(\lambda) &= (1+\frac{\beta_{n-1}}{\alpha_{n-1}}\alpha_n - \alpha_n\lambda)R_n(\lambda) \\ &\quad -\frac{\beta_{n-1}}{\alpha_{n-1}}\alpha_n R_{n-1}(\lambda),\quad n=1,2,\ldots && (6)\end{aligned}$$

Zhang [7] discovered that moderate convergence property can be gained by choosing for acceleration polynomials $H_n(\lambda)$ that are built up in the three-term recurrence form as polynomial $R_n(\lambda)$ in (5) and (6) by adding suitable undetermined parameters ζ_n and η_n as follows:

$$\begin{aligned} H_0(\lambda) &= 1, && (7)\\ H_1(\lambda) &= (1-\zeta_0\lambda)H_0(\lambda), && (8)\\ H_{n+1}(\lambda) &= (1+\eta_n-\zeta_n\lambda)H_n(\lambda) - \eta_n H_{n-1}(\lambda), && (9)\\ &\quad n=1,2,\ldots.\end{aligned}$$

The polynomials $H_n(\lambda)$ satisfies $H_n(0)=1$ and relation as $H_{n+1}(0)-H_n(0)=0$ for all n. Here we introduce an auxiliary polynomials $G_n(\lambda)$ as

$$G_n(\lambda) := (H_n(\lambda) - H_{n+1}(\lambda))/(\zeta_n\lambda). \quad (10)$$

By reconstruction of (6) using the acceleration polynomials $H_n(\lambda)$ and $G_n(\lambda)$, we have the following coupled two-term recursion of the form as

$$\begin{aligned} H_0(\lambda) &= 1,\ G_0(\lambda)=\zeta_0, && (11)\\ H_n(\lambda) &= H_{n-1}(\lambda) - \lambda G_{n-1}(\lambda), && (12)\\ G_n(\lambda) &= \zeta_n H_n(\lambda) + \eta_n G_{n-1}(\lambda), && (13)\\ &\quad n=1,2,\ldots.\end{aligned}$$

Using these acceleration polynomials $H_n(\lambda)$ and $G_n(\lambda)$, his discover led to the generalized product-type methods based on Bi-CG method for solving the linear system with unsymmetric coefficient matrix. He refered as GPBiCG method [7]. However, the original Lanczos algorithm is also known to break down or nearly break down in some cases. In practice, the occurrence of a break down cause failure to converge to the solution of linear equations, and the increase of the iterations introduce numerical error into the approximate solution. Therefore, convergence of the generalized product-type methods is affected. Comparatively little is known about the theoretical properties of the generalized product-type methods. The fact that the generalized product-type methods perform very well sometimes but fail in others motivates the need for further insight into the construction of polynomials for the product-type residual $H_{n+1}(\lambda)R_{n+1}(\lambda)$.

In a usual approach, acceleration parameters are decided from local minimization of the residual vector of 2-norm $||\boldsymbol{r}_{n+1}(:= H_{n+1}(\lambda)R_{n+1}(\lambda))||_2$, where $R_{n+1}(\lambda)$ denotes the residual polynomial of the Lanczos algorithm and $H_{n+1}(\lambda)$ denotes the acceleration polynomial for convergence. Instead, it embodies that a particular strategy for remedying instability of convergence. That is, algorithm of GPBiCG_AR[4] method based on local minimization of associate residual $\boldsymbol{a\_r}_n(:= H_{n+1}(\lambda)R_n(\lambda))$ can be written as follows:

$$\boldsymbol{a\_r}_n = \boldsymbol{r}_n - \eta_n A\boldsymbol{z}_{n-1} - \zeta_n A\boldsymbol{r}_n. \quad (14)$$

Here $\boldsymbol{r}_n$ is the residual vector of the algorithm. Matrix-vector multiplication of $A\boldsymbol{u}_n$ and $A\boldsymbol{r}_{n+1}$ are directly computed according to definition of multiplication of matrix A and vector. On the other hand, $A\boldsymbol{p}_n$ and $A\boldsymbol{z}_n$ are computed using its recurrence. In the algorithm of GPBiCG_AR method, modification parts which differ from the original GPBiCG method are indicated with underlines.

2.1 Reverse-ordered recurrence

In the algorithm of GPBiCG_AR method, an auxiliary vector of $\boldsymbol{u}_n$ is developed using underlined reverse-ordered recurrence as below.

The reverse-ordered recurrence can be disappeared owing to the underlined modification.

$$\begin{aligned} \boldsymbol{u}_n &:= \lambda G_n P_n \\ &= \lambda P_n(\zeta_n H_n + \eta_n G_{n-1}) \\ &= \zeta_n\lambda H_n P_n + \eta_n \lambda G_{n-1}(R_n + \beta_{n-1}P_{n-1}) \\ &= \zeta_n\lambda H_n P_n + \eta_n(\underline{\lambda G_{n-1}R_n} + \beta_{n-1}\lambda G_{n-1}P_{n-1}) \\ &= \zeta_n A\boldsymbol{p}_n + \eta_n(\underline{A\boldsymbol{z}_{n-1}} + \beta_{n-1}A\boldsymbol{u}_{n-1}) && (15)\end{aligned}$$

We replace computation of $\lambda G_{n-1}R_n$ with $A\boldsymbol{z}_{n-1}$. Computation of $A\boldsymbol{z}_{n-1}$ is already used in determination of parameters. Then computational cost doesn't increase.

As a result, we could devise GPBiCGSafe method without reverse-ordered recurrence. Next we exhibit an algorithm of preconditioned GPBiCGSafe method. In GPBiCGSafe method, parameters ζ_n, η_n are decided from minimization of 2-norm of associate residual.

Algorithm of preconditioned GPBiCGSafe method

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
Choose $\boldsymbol{r}_0^*$ such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$, $\beta_{-1}=0$,
set $\boldsymbol{p}_{-1} = \boldsymbol{u}_{-1} = \boldsymbol{z}_{-1} = 0$,
for $n = 0, 1, \cdots$ until $||\boldsymbol{r}_{n+1}|| \leq \varepsilon\ ||\boldsymbol{r}_0||$ do :
begin

$$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_{n-1}(\boldsymbol{p}_{n-1} - \boldsymbol{u}_{n-1}), \quad (16)$$

$$AK^{-1}\boldsymbol{p}_n = AK^{-1}\boldsymbol{r}_n + \beta_{n-1}(AK^{-1}\boldsymbol{p}_{n-1} - AK^{-1}\boldsymbol{u}_{n-1}), \quad (17)$$

ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

$$\alpha_n = (\boldsymbol{r}_0^*, \boldsymbol{r}_n)/(\boldsymbol{r}_0^*, AK^{-1}\boldsymbol{p}_n), \tag{18}$$
$$\boldsymbol{a}_n = \boldsymbol{r}_n, \boldsymbol{b}_n = AK^{-1}\boldsymbol{z}_{n-1}, \boldsymbol{c}_n = AK^{-1}\boldsymbol{r}_n, \tag{19}$$
$$\zeta_n = \frac{(\boldsymbol{b}_n, \boldsymbol{b}_n)(\boldsymbol{c}_n, \boldsymbol{a}_n) - (\boldsymbol{b}_n, \boldsymbol{a}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{b}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)}, \tag{20}$$
$$\eta_n = \frac{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{a}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{a}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{b}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)}, \tag{21}$$
$$(\text{if } n = 0, \text{ then } \zeta_n = (\boldsymbol{c}_n, \boldsymbol{a}_n)/(\boldsymbol{c}_n, \boldsymbol{c}_n),\ \eta_n = 0) \tag{22}$$
$$\boldsymbol{u}_n = \zeta_n AK^{-1}\boldsymbol{p}_n + \eta_n(AK^{-1}\boldsymbol{z}_{n-1} + \beta_{n-1}\boldsymbol{u}_{n-1}), \tag{23}$$
$$\boldsymbol{t}_n = \boldsymbol{r}_n - \alpha_n AK^{-1}\boldsymbol{p}_n, \tag{24}$$
$$\boldsymbol{z}_n = \zeta_n \boldsymbol{r}_n + \eta_n \boldsymbol{z}_{n-1} - \alpha_n \boldsymbol{u}_n, \tag{25}$$
$$AK^{-1}\boldsymbol{z}_n = \zeta_n AK^{-1}\boldsymbol{r}_n + \eta_n AK^{-1}\boldsymbol{z}_{n-1} - \alpha_n AK^{-1}\boldsymbol{u}_n, \tag{26}$$
$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n K^{-1}\boldsymbol{p}_n + K^{-1}\boldsymbol{z}_n, \tag{27}$$
$$\boldsymbol{r}_{n+1} = \boldsymbol{t}_n - AK^{-1}\boldsymbol{z}_n, \tag{28}$$
$$\beta_n = \frac{\alpha_n}{\zeta_n} \cdot \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_{n+1})}{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)}, \tag{29}$$

end

We show computational cost of six kinds of iterative methods with ILU(0) preconditioning per one iteration in Table 1. In Table 1, "N" means dimension of matrix, and "nnz" means the number of total nonzero entries of matrix. "K^{-1}", "$\boldsymbol{u}, \boldsymbol{v}$" and "$\alpha$" means preconditioner, vectors and scalar value, respectively. From Table 1, we see that computational cost per one iteration of GPBiCG_AR, GPBiCGSafe and BiCGSafe methods are small amount compared with that of other three iterative methods.

3 Numerical experiments

3.1 Computational environment and conditions

All computations were done in double precision floating point arithmetics, and performed on Nehalem (CPU: Intel Xenon X5570, Clock: 2.93GHz, memory: 24Gbytes, OS: RedHat Enterprise Linux 5.2).

All codes were compiled with the "-O3" optimization option. The right-hand side $\boldsymbol{b}$ was imposed from the physical load conditions. The stopping criterion for successful convergence of the iterative methods is less than 10^{-10} of the relative residual 2-norm $||\boldsymbol{r}_{n+1}||_2/||\boldsymbol{r}_0||_2$. In all cases the iteration was started with the initial guess solutions $\boldsymbol{x}_0 = \boldsymbol{0}$. The maximum number of iterations is fixed as 10^4. Matrices are normalized with diagonal scaling. The initial shadow residual $\boldsymbol{r}_0^* = \boldsymbol{b} - A\boldsymbol{x}_0$ of GPBiCG and GPBiCG_AR methods is equal to the initial residual $\boldsymbol{r}_0$. As test matrices as shown in Table 2, 17 matrices in total are taken from Florida sparse matrix collection[2]. We examined performance of GPBiCG, AS_GPBiCG_v1 (Abe-Sleijpen GPBiCG variant-1), AS_GPBiCG_v2 (Abe-Sleijpen GPBiCG variant-2), GPBiCG_AR, GPBiCGSafe and BiCGSafe methods preconditioned using ILU(0) without extra fill-ins. In Table 2 we show specifications of test matrices.

Table 3-4 show the numerical results of GPBiCGSafe method and other iterative methods. "itr." means number of iterations, "time" means computation time in seconds and "ratio" means the ratio of computation time of GPBiCG_AR method to that of GPBiCG method. "max" denotes non-convergence until iterations reach at the maximum iteration counts.

Table 1: Computational costs of six kinds of iterative methods with ILU(0) preconditioning per one iteration.

method	$K^{-1}\boldsymbol{v}$ ($\times 2nnz$)	$A\boldsymbol{v}$ ($\times 2nnz$)	$(\boldsymbol{u}, \boldsymbol{v})$ ($\times 2N$)	$\boldsymbol{u} \pm \boldsymbol{v}$ ($\times N$)	$\alpha\boldsymbol{v}$ ($\times N$)
GPBiCG	2	2	7	17	14
AS_GPBiCG_v1	2	2	8	15	16
AS_GPBiCG_v2	2	2	8	15	15
GPBiCG_AR	2	2	7	15	13
GPBiCGSafe	2	2	7	14	13
BiCGSafe	2	2	7	14	13

Table 2: Specifications of test matrices.

matrix	N	nnz	ave. nnz	analytical field
air-cfl5	1,536,000	19,435,428	12.65	hydro
atmosmodd	1,270,432	8,814,880	6.94	dynamic
poisson3Db	85,623	2,374,949	27.74	
raefsky3	21,200	1,488,768	70.22	
water_tank	60,740	2,035,281	33.51	
bcircuit	68,902	375,558	5.45	electrical
Freescale1	3,428,755	17,052,626	4.97	circuit
memplus	17,758	126,150	7.1	
sme3Da	12,504	874,887	69.97	structural
sme3Db	29,067	2,081,063	71.6	
sme3Dc	42,930	3,148,656	73.34	
appu	14,000	1,853,104	132.36	directed
big	13,209	91,465	6.92	weighted graph
ecl32	51,993	380,415	7.32	semiconductor
epb3	84,617	463,625	5.48	thermal
k3plates	11,107	378,927	34.12	acoustics
waseda	19,060	24,377,548	1279.	electromagnetics

In Table 3-4, "TRR" means true relative residual of the approximate solution $\boldsymbol{x}_{n+1}$. For matrices air-cfl5 and atmosmodd, when iteration count for convergence is small, GPBiCGSafe method works well because of small amount of computational cost per one iteration. For matrices raefsky3 and water_tank, when iteration count is moderate, performance of GPBiCGSafe method is competitive. Sometimes GPBiCGSafe method is the most efficient. However, the difference of performance is small.

On the contrary, when iteration count is many, performance of GPBiCG_AR, GPBiCGSafe and BiCGSafe methods are competitive. For example, for matrices bcircuit and Freescale1, we can see the above same tendency. Sometimes GPBiCGSafe method is the most efficient. TRR of AS_GPBiCG_v1 is larger than the demanded accuracy of 10^{-10} for matrix bcircuit. As the most

ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

important fact, when property of matrix becomes ill-conditioned as matrix k3plates, robustness of GPBiCG, AS_GPBiCG_v1, _v2 are not sufficient. On the contrary, GPBiCG_AR, GPBiCGSafe and BiCGSafe methods are significantly robust.

Table 3: Performance of six kinds of iterative methods.

matrix	method	itr.	time [sec.]	ave. time [ms]	$\log_{10}$ (TRR)
air-cfl5	GPBiCG	21	7.032	258.238	-10.11
	AS_GPBiCG_v1	21	7.205	266.143	-10.11
	AS_GPBiCG_v2	21	7.190	264.952	-10.11
	GPBiCGAR	21	6.776	245.952	-10.06
	GPBiCGSafe	21	**6.749**	244.190	-10.06
	BiCGSafe	21	6.787	246.429	-10.06
atmosmodd	GPBiCG	89	14.157	153.955	-10.08
	AS_GPBiCG_v1	90	14.821	159.667	-10.03
	AS_GPBiCG_v2	90	14.716	158.533	-10.10
	GPBiCGAR	90	13.341	143.078	-10.09
	GPBiCGSafe	90	**13.247**	142.122	-10.24
	BiCGSafe	93	13.685	142.215	-10.15
poisson3Db	GPBiCG	86	3.481	33.884	-10.12
	AS_GPBiCG_v1	84	3.440	34.250	-10.07
	AS_GPBiCG_v2	84	3.424	34.060	-10.08
	GPBiCGAR	84	3.383	33.548	-10.06
	GPBiCGSafe	84	3.369	33.405	-10.23
	BiCGSafe	83	**3.354**	33.506	-10.05
raefsky3	GPBiCG	129	1.759	12.047	-10.04
	AS_GPBiCG_v1	133	1.803	12.023	-10.80
	AS_GPBiCG_v2	125	**1.711**	12.032	-10.23
	GPBiCGAR	134	1.803	11.948	-10.65
	GPBiCGSafe	134	1.806	11.940	-10.36
	BiCGSafe	140	1.882	11.964	-10.28
water_tank	GPBiCG	270	5.260	18.641	-10.02
	AS_GPBiCG_v1	285	5.537	18.628	-10.27
	AS_GPBiCG_v2	276	5.344	18.551	-10.02
	GPBiCGAR	279	5.338	18.351	-10.29
	GPBiCGSafe	269	**5.154**	18.305	-10.01
	BiCGSafe	275	5.260	18.305	-10.29
bcircuit	GPBiCG	6995	55.321	7.905	-10.01
	AS_GPBiCG_v1	4765	38.645	8.104	(-9.41)
	AS_GPBiCG_v2	4553	36.275	7.961	-10.18
	GPBiCGAR	3594	27.715	7.702	-10.52
	GPBiCGSafe	3239	**24.851**	7.663	-10.14
	BiCGSafe	4114	32.091	7.793	-10.21
Freescale1	GPBiCG	1681	675.306	400.659	-10.09
	AS_GPBiCG_v1	1567	659.221	419.536	-10.05
	AS_GPBiCG_v2	2135	890.542	416.282	-9.94
	GPBiCGAR	1347	**500.602**	370.317	-10.02
	GPBiCGSafe	1384	512.137	368.749	-10.11
	BiCGSafe	1409	531.936	376.251	-10.11
memplus	GPBiCG	251	0.390	1.510	-10.04
	AS_GPBiCG_v1	257	0.416	1.572	-10.04
	AS_GPBiCG_v2	243	0.398	1.588	-10.03
	GPBiCGAR	251	0.379	1.458	-10.08
	GPBiCGSafe	244	**0.363**	1.443	-10.13
	BiCGSafe	248	0.377	1.476	-10.09

We exhibit summary of performance of six kinds of preconditioned iterative methods in Table 5. From Table 5, it can be seen that GPBiCG_AR, GPBiCGSafe and BiCGSafe methods converged for all test matrices. In particular, GPBiCGSafe method outperforms among the product-typed iterative methods.

We demonstrate the ranking of convergence time and overall

Table 4: Performance of six kinds of iterative methods. (cont'd)

matrix	method	itr.	time [sec.]	ave. time [ms]	$\log_{10}$ (TRR)
sme3Da	GPBiCG	1498	12.038	7.836	-10.22
	AS_GPBiCG_v1	1090	8.860	7.850	-10.23
	AS_GPBiCG_v2	1295	10.446	7.836	-9.68
	GPBiCGAR	1039	8.387	7.782	-10.04
	GPBiCGSafe	1072	8.604	7.745	-10.07
	BiCGSafe	949	**7.664**	7.759	-10.16
sme3Db	GPBiCG	1192	27.262	22.193	(-9.59)
	AS_GPBiCG_v1	1096	25.162	22.220	-10.06
	AS_GPBiCG_v2	1389	31.587	22.162	(-9.12)
	GPBiCGAR	831	**19.137**	22.060	-10.07
	GPBiCGSafe	1005	22.955	22.040	-10.01
	BiCGSafe	924	21.149	22.018	-10.04
sme3Dc	GPBiCG	2599	94.239	35.755	-10.80
	AS_GPBiCG_v1	2207	80.437	35.854	(-8.64)
	AS_GPBiCG_v2	1756	64.182	35.812	-9.96
	GPBiCGAR	1628	59.232	35.582	-10.17
	GPBiCGSafe	1603	**58.482**	35.669	-10.04
	BiCGSafe	1614	58.713	35.575	-10.01
appu	GPBiCG	27	1.765	18.296	-10.32
	AS_GPBiCG_v1	26	1.744	18.346	-10.05
	AS_GPBiCG_v2	26	**1.739**	18.346	-10.05
	GPBiCGAR	29	1.787	18.207	-10.22
	GPBiCGSafe	29	1.789	18.172	-10.20
	BiCGSafe	29	1.796	18.379	-10.31
big	GPBiCG	936	1.267	1.346	-10.00
	AS_GPBiCG_v1	831	1.161	1.390	-10.30
	AS_GPBiCG_v2	948	1.304	1.368	-10.49
	GPBiCGAR	769	**1.012**	1.308	-10.14
	GPBiCGSafe	803	1.056	1.305	-10.34
	BiCGSafe	812	1.082	1.324	-10.36
ecl32	GPBiCG	125	0.775	5.952	-10.11
	AS_GPBiCG_v1	126	0.805	6.167	-10.21
	AS_GPBiCG_v2	121	0.752	5.992	-10.15
	GPBiCGAR	124	0.755	5.831	-10.13
	GPBiCGSafe	123	**0.741**	5.797	-10.20
	BiCGSafe	124	0.754	5.847	-10.06
epb3	GPBiCG	89	0.784	8.528	-10.12
	AS_GPBiCG_v1	89	0.790	8.629	-10.73
	AS_GPBiCG_v2	89	0.900	9.820	-10.20
	GPBiCGAR	94	0.807	8.362	-10.02
	GPBiCGSafe	90	0.767	8.267	-10.06
	BiCGSafe	89	**0.722**	7.899	-10.15
k3plates	GPBiCG	max	-	-	-9.69
	AS_GPBiCG_v1	7675	24.443	3.180	-10.30
	AS_GPBiCG_v2	max	-	-	-9.54
	GPBiCGAR	5580	**17.357**	3.104	-10.33
	GPBiCGSafe	6391	19.845	3.100	-10.08
	BiCGSafe	5680	17.672	3.105	-10.94

Table 5: Summary of performance of six kinds of preconditioned iterative methods.

method	converged	non-conv.	spurious conv.	fastest cases
GPBiCG	15/17	2/17	1/17	0/17
AS_GPBiCG_v1	15	2	2	0
AS_GPBiCG_v2	14	3	1	2
GPBiCG_AR	**17**	0	0	4
GPBiCGSafe	**17**	0	0	**8**
BiCGSafe	**17**	0	0	3

ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

ranking for six kinds of preconditioned iterative methods in Table 6. From total score and ranking, it is concluded that convergence rate of GPBiCGSafe method is excellent from the viewpoint of robustness and efficiency. Convergence rate of GPBiCG_AR and BiCGSafe methods are competitive. On the other hand, performance of GPBiCG, AS_GPBiCG_v1, _v2 are poor.

Table 6: The ranking of convergence time and overall ranking for six kinds of preconditioned iterative methods.

method	ranking						total	overall
	1	2	3	4	5	6	score	ranking
GPBiCG	0	2	3	3	6	3	73	4
AS_GPBiCG_v1	0	1	1	8	1	6	78	5
AS_GPBiCG_v2	2	0	0	4	4	7	80	6
GPBiCG_AR	4	5	5	1	2	0	43	2
GPBiCGSafe	8	4	3	0	2	0	**35**	**1**
BiCGSafe	3	6	5	1	0	2	46	3
total	17	18	17	17	15	18	-	-

4 Conclusions

In this paper we proposed GPBiCGSafe method without reverse-ordered recurrence. We made clear that reverse-ordered recurrence affected convergence rate of the original GPBiCG method. GPBiCGSafe method outperforms among tested product-type iterative methods. GPBiCG_AR and BiCGSafe methods work well compared with the conventional GPBiCG method.

References

[1] K. Abe, G. Sleijpen: Solving linear equations with a stabilized GPBiCG method, Applied Numerical Math., 2012. (in press)

[2] Tim Davis' sparse matrix collection of Florida University: http://www.cise.ufl.edu/research/sparse/matrices/

[3] S. Fujino, M. Fujiwara and M. Yoshida: BiCGSafe method based on minimization of associate residual, JSCES, No.20050028, 2005.

[4] MoeThuthu, Y. Onoue, S. Fujino, Experimental comparison of preconditioned GPBi-CG and GPBiCG_AR methods in view of stability of convergence, JSCES, No.20090008, 2009.

[5] H.A. van der Vorst: Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems, SIAM J. Sci. Stat. Comput., **13**(1992), 631-644.

[6] H.A. van der Vorst: Iterative Krylov preconditionings for large linear systems, Cambridge University Press, Cambridge, 2003.

[7] S.-L. Zhang: GPBi-CG: Generalized product-type preconditionings based on Bi-CG for solving nonsymmetric linear systems, SIAM J. Sci. Comput., **18**(1997), 537-551.

ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)