AN IMPROVEMENT OF SAINV AND RIF PRECONDITIONINGS OF CG METHOD BY DOUBLE DROPPING STRATEGY

Seiji FUJINO* and Yusuke IKEDA**

*Computing and Communications Center, Kyushu University
**Graduate School of Information Science and Electrical Engineering, Kyushu University
(Current affiliation: Toshiba Digital Media Network Company)
*E-mail: fujino@cc.kyushu-u.ac.jp

ABSTRACT

Preconditioning based on incomplete factorization of the matrix A is among the best known and most popular methods for solving a linear system of equations with symmetric positive definite coefficient matrix. However, the existence of an incomplete factorization is a delicate issue which must be overcame if one has a desire to design reliable preconditioning. Stabilized AINV (Approximate INVerse) and RIF (Robust Incomplete Factorization) preconditionings with single dropping have been proposed. Dropping procedure is a key to improvement of efficiency of computation. In this paper new dropping strategy for improvement of both SAINV and RIF preconditionings will be proposed. Moreover comparisons with other incomplete factorization and original SAINV and RIF preconditionings using challenging linear systems from realistic structural analysis are presented. We discuss double dropping strategy in the context of computation time of CG method with preconditioning for successful convergence and memory requirement for factorization.

1. INTRODUCTION

We consider preconditioning of Conjugate Gradient (CG) method for solving the linear system of equations

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

where $A \in R^{n\times n}$ is a given large, sparse symmetric positive definite (SPD) matrix, $\boldsymbol{b}$ is a given right-hand side vector and $\boldsymbol{x}$ is a solution vector. When A is M-matrix, the incomplete Cholesky (IC) factorization preconditioning [14] is an effective technique for solving equation (1). However, the case when A is not M-matrix, e.g., the linear systems that arise from the finite element method (FEM) applied to realistic problems in structural and solid mechanics analysis, is substantially more difficult to solve efficiently by means of CG method with IC factorization.

This difficulty has led to the development of a wide variety of preconditioning having varying degrees of success. Among a number of preconditioning techniques, algebraic preconditioning based on sparse approximate representation of inverse matrices (refered to as AINV) in factored form proposed by M. Benzi and M. Tuma can be regarded as a powerful and promising preconditioning [1]. Moreover they devised a safe and reliable method of computing an approximate factorization $A \approx LDL^T$ with L unit lower triangular and D diagonal. Unlike the standard Cholesky factorization, their technique is based on an inherently stable A-orthogonalization process. Their method was quite effective for avoiding breakdown.

Dropping procedure in an A-orthogonalization process is a significant key to improvement of efficiency of computation [8] [10]. Performance dependency of SAINV and Improved SAINV preconditioning on various types of computers was also revealed in the previous paper [9]. Moreover how to choose the threshold values was discussed in the same reference. Therefore we focus our attention on dropping procedure in the two types of preconditioning based upon A- orthogonalization process. In this paper, we propose a double dropping strategy for both SAINV and RIF preconditionings, and examine the effectiveness as preconditioning for the CG method using realistic problems which stem from the design of a concrete bridge.

This paper is organized as follows: We discuss stabilized AINV preconditioning in Section 2. We describe the robust incomplete factorization as a safe and reliable way of computing an approximate factorization in Section 3. We present algorithm of the double dropping strategy in the course of factorization based on A-orthogonalization process. We refer to the proposed double dropping strategy as improved SAINV (abbreviated to ISAINV) and improved RIF (Robust Incomplete Factorization) (abbreviated to **IRIF**). The performance of ISAINV and **IRIF** is discussed in Section 5. Section 6 concludes the paper with a summary of our findings and indications for future work.

Proceedings of the Seventh International Conference on High Performance Computing and Grid in Asia Pacific Region (HPCAsia'04)
0-7695-2138-X/04 $ 20.00 IEEE

2. STABILIZED AINV PRECONDITIONING

The factorization of A is obtained by an A-orthogonalization process applied to the unit basis vectors $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$. This is simply the Gram-Schmidt process with respect to the inner product generated by the SPD matrix A. A-orthogonalization produces also the inverse factorization $A^{-1} = ZD^{-1}Z^t$ with Z unit upper triangular and D diagonal. This fact leads to construction of factored sparse approximate inverse preconditioners. The algorithm was first shown in reference [6].

Moreover stabilized variants of AINV were proposed (refered to as SAINV), and have been shown to be reliable in solving highly ill-conditioned linear systems [3] [11]. The algorithm of stabilized AINV preconditioning is shown as below. The differences between AINV and SAINV preconditionings lie in underlines.

Algorithm of SAINV :

$$
\begin{array}{l}
\textbf{for } i = 1, \cdots, n \\
\quad \boldsymbol{z}_i^{(0)} = \boldsymbol{e}_i \\
\textbf{end for} \\
\textbf{for } i = 1, \cdots, n \\
\quad \underline{\boldsymbol{v} = \mathrm{A}\boldsymbol{z}_i^{(i-1)}} \\
\quad \textbf{for } j = i, \cdots, n \\
\quad\quad \underline{d_j = \boldsymbol{v}^t \boldsymbol{z}_j^{(i-1)}} \\
\quad \textbf{end for} \\
\quad \textbf{for } j = i+1, \cdots, n \\
\quad\quad \boldsymbol{z}_j^{(i)} = \boldsymbol{z}_j^{(i-1)} - \dfrac{d_j}{d_i}\boldsymbol{z}_i^{(i-1)} \\
\quad \textbf{end for} \\
\textbf{end for}
\end{array}
$$

Let $Z = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_n)$ and D = diag $(d_1, d_2, \ldots, d_n)$. $\boldsymbol{z}_j$ denotes jth column of the matrix Z, and $\boldsymbol{z}_j^{(i)}$ denotes the jth vector at the ith iteration step. To obtain a sparse preconditioner, a dropping rule is applied to the $\boldsymbol{z}_i$ vectors after each update step. In particular, in order to gain an efficient algorithm, the update loop has to be carefully implemented taking account of sparsity in matrix A and in the $\boldsymbol{z}_i$ vectors.

The algorithm of CG method with AINV preconditioning is shown as follows:

$$
\begin{array}{l}
\text{An initial guess } \boldsymbol{x}_0 \text{ is given,} \\
\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0,\ \boldsymbol{p}_0 = ZD^{-1}Z^t\boldsymbol{r}_0, \\
\textbf{for } m = 1, 2, \ldots \\
\quad \alpha_m = \dfrac{(\boldsymbol{r}_{m-1}, ZD^{-1}Z^t\boldsymbol{r}_{m-1})}{(\boldsymbol{p}_{m-1}, A\boldsymbol{p}_{m-1})}, \\
\quad \boldsymbol{x}_m = \boldsymbol{x}_{m-1} + \alpha_m \boldsymbol{p}_{m-1}, \\
\quad \boldsymbol{r}_m = \boldsymbol{r}_{m-1} - \alpha_m A\boldsymbol{p}_{m-1}, \\
\quad \textbf{if } ||\boldsymbol{r}_m||_2/||\boldsymbol{r}_0||_2 \le \varepsilon \textbf{ stop} \\
\quad \beta_m = \dfrac{(\boldsymbol{r}_m, ZD^{-1}Z^t\boldsymbol{r}_m)}{(\boldsymbol{r}_{m-1}, ZD^{-1}Z^t\boldsymbol{r}_{m-1})}, \\
\quad \boldsymbol{p}_m = ZD^{-1}Z^t\boldsymbol{r}_m + \beta_m \boldsymbol{p}_{m-1}, \\
\textbf{end for.}
\end{array}
$$

3. RIF(ROBUST INCOMPLETE FACTORIZATION)

We consider the complete factorization (without dropping) with $\bar{L}$ unit lower triangular and $\bar{D}_{ic}$ diagonal as follows:

$$A \quad = \quad \bar{L}\bar{D}_{ic}\bar{L}^t. \tag{2}$$

Moreover we can also define the inverse A^{-1} with $\bar{Z}$ unit upper triangular matrix and $\bar{D}_{ainv}^{-1}$ diagonal which are obtained from the A-orthogonalization process as follows:

$$A^{-1} \quad = \quad \bar{Z}\bar{D}_{ainv}^{-1}\bar{Z}^t. \tag{3}$$

The factorization (2) corresponds to the complete Cholesky factorization, and we can represent its inverse as

$$A^{-1} \quad = \quad \bar{L}^{-t}\bar{D}_{ic}^{-1}\bar{L}^{-1}. \tag{4}$$

From uniqueness of factorization and the above relationships (3) and (4), the following relationships follows easily.

$$\bar{Z}^t \quad = \quad \bar{L}^{-1}, \tag{5}$$

$$\bar{D}_{ainv} \quad = \quad \bar{D}_{ic}. \tag{6}$$

Therefore we can observe the relationship:

$$\bar{L} = A\bar{Z}\bar{D}_{ainv}^{-1} \quad (\text{or} \quad \bar{L}\bar{D}_{ainv} = A\bar{Z}). \tag{7}$$

This equation (7) implies that we can identify mathematically $\bar{L}$ with $A\bar{Z}\bar{D}_{ainv}^{-1}$. Accordingly the incomplete factorized L of A can be obtained as a by-product of the A-orthogonalization process, at no extra cost.

$$L \leftarrow AZD_{ainv}^{-1} \quad (\text{or} \quad LD_{ainv} \leftarrow AZ). \tag{8}$$

The arrow above denotes a computing procedure for gaining the L factor of A. In this way, the L factor of A was obtained as a by-product of A-orthogonalization process. That is, we refer to a factorization method which calculates L from Z based upon the process of (8) as **RIF** (Robust Incomplete Factorization).

4. DROPPING PROCEDURE

We give an outline standard dropping procedure, followed by double dropping strategy for SAINV and RIF preconditionings, respectively.

4.1. Standard dropping procedure in SAINV

The incomplete factorization based on A-orthogonalization needs two dropping tolerances: one for the SAINV process, to be applied to the $\boldsymbol{z}$ vectors, and a second one to be applied to the entries of the lower triangular matrix L [4] [11]. The latter is also simply post-filtration. That is, once a column of L has been computed, it does not enter the computation of the remaining ones. The post-filtration strategy, however, is still impractical and inefficient [7]. The post-filtration often degrades the convergence rate of preconditioned CG method. That is, one important competitive issue of dropping procedure to other preconditionings is reduction of small entries which appear in the course of factorization. Standard dropping is represented as follows:

- Standard dropping:

$$
\begin{array}{l}
\textbf{for } k = 1, \cdots, i \\
\quad \textbf{if } |z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)}| > \text{tol-1} \\
\qquad z_{kj}^{(i)} = z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)} \\
\quad \textbf{else} \\
\qquad z_{kj}^{(i)} = 0 \\
\quad \textbf{end if} \\
\textbf{end for}
\end{array} \tag{9}
$$

However, from the viewpoint of efficiency of preconditioning for CG method, standard dropping procedure only lacks ability for improvement of convergence property of CG method. Hence, new dropping strategy for CG method is greatly necessary for enhancement of preconditioning based upon A-orthogonalization process.

4.2. Double dropping strategy in SAINV

Our dropping strategy (refered to as double dropping) has non-degradation of the convergence rate. The basic idea for double dropping strategy is derived from judgement on execution of dropping itself within the filtration. The double dropping technique using two tolerance values of tol-1 and tol-2 is shown as below. $z_{kj}^{(i)}$ represents the kth element of jth of column at the ith iteration step.

- Double dropping:

$$
\begin{array}{l}
\underline{\textbf{if } |\frac{d_j}{d_i}| > \text{tol-2}} \\
\quad \textbf{for } k = 1, \cdots, i \\
\qquad \textbf{if } |z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)}| > \text{tol-1} \\
\qquad\quad z_{kj}^{(i)} = z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)} \\
\qquad \textbf{else} \\
\qquad\quad z_{kj}^{(i)} = 0 \\
\qquad \textbf{end if} \\
\quad \textbf{end for} \\
\underline{\textbf{end if}}
\end{array} \tag{10}
$$

4.3. Standard dropping procedure in RIF

Double dropping strategy as described above, it allows the application in RIF process. Namely in the dropping procedure of SAINV, removing small entries from the columns of the incomplete L factor leads to great degradation of the convergence rate. On the contrary, this is often compensated by the savings in computational work obtained from a sparser L factor. The drawback is the need to have a second dropping tolerance. As a result, we can circumvent this problem by using the same value of the dropping tolerance for the SAINV process and the entries of the incomplete factor L. We show the implementation of dropping for the absolute value of $l_{ji} = \frac{d_j}{d_i}$ in **RIF** process as follows:

- Standard dropping:

$$
\begin{array}{l}
\textbf{if } |\frac{d_j}{d_i}| > \text{tol} \\
\quad \underline{l_{ji} = \frac{d_j}{d_i}} \\
\textbf{end if} \\
\textbf{for } k = 1, \cdots, i \\
\quad \textbf{if } |z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)}| > \text{tol} \\
\qquad z_{kj}^{(i)} = z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)} \\
\quad \textbf{else} \\
\qquad z_{kj}^{(i)} = 0 \\
\quad \textbf{end if} \\
\textbf{end for}
\end{array} \tag{11}
$$

l_{ji} represents the element of jth row and ith column of incomplete factored L. We denote $z_{kj}^{(i)}$ for the kth element of jth of column at the ith iteration step. We note that the index j varies from $i+1$ to n in the preservation process of l_{ji}.

4.4. Double Dropping strategy in RIF process

Double dropping strategy with dropping tolerance of tol_dd is underlined for the absolute value of $\frac{d_j}{d_i}$ as shown in **RIF** process as well as the SAINV process as below.

- Double dropping:

$$
\begin{array}{l}
\textbf{if } |\frac{d_j}{d_i}| > \text{tol} \\
\quad l_{ji} = \frac{d_j}{d_i} \\
\textbf{end if} \\
\underline{\textbf{if } |\frac{d_j}{d_i}| > \text{tol\_dd}} \\
\quad \textbf{for } k = 1, \cdots, i \\
\quad\quad \textbf{if } |z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)}| > \text{tol} \\
\quad\quad\quad z_{kj}^{(i)} = z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)} \\
\quad\quad \textbf{else} \\
\quad\quad\quad z_{kj}^{(i)} = 0 \\
\quad\quad \textbf{end if} \\
\quad \textbf{end for} \\
\underline{\textbf{end if}}
\end{array}
\tag{12}
$$

There are some possibilities in the combination of dropping tolerances of tol and tol_dd for $|\frac{d_j}{d_i}|$ and $|z_{kj}^{(i-1)} - \frac{d_j}{d_i} z_{ki}^{(i-1)}|$. However, we cound not find out significant difference within our numerical experiments. The lower factored L and diagonal D_{ic} obtained in **RIF** process are used in the iteration of the CG method.

Table 1. Specification of test matrices for problem 1.

name	n	no. entries	comments, source
BCSSTK24	3562	81736	Eigenvalue problem [13]
BCSSTK35	30237	740200	Automobile seat frame and body [15]
NASASRB	54870	1366097	Shuttle rocket booster [15]
TUBE1-2	21498	459277	Thin shell [12]

Table 2. Specification of tested matrices for problem 2.

	name	
description	BEAM	CABLE
n	10626	59002
number of entries	233268	1986094
Average band width	576	1741
Total number of nodes	1977	20194
Total number of elements	2832	16084

5. NUMERICAL EXPERIMENTS

5.1. Test matrices and computation conditions

Specifications of test matrices for problem 1 and 2 are listed in Tables 1 and 2, respectively. The table lists the name, problem size, number of entries and discipline and additional comments with source. Test matrices shown in Table 1 are linear systems arising in the FEM analysis [12] [13] [15]. For each matrix we provide the problem size "n", the number of non-zero entries "no. entries" in the lower triangular part of coefficient matrix A. On the other hand, test matrices shown in Table 2 were used actually in the stress analysis for a beam and joint cable division of a concrete bridge with plural trucks, respectively. We refer to the matrices as "BEAM" and "CABLE", respectively. The matrix "BEAM" is obtained from the discretization with shell elements, and "CABLE" is obtained from the discretization with solid elements.

All experiments reported in this paper are done on Fujitsu SPARC64 processor, with 1.5 Giga bytes of main memory per single processor and a single 1.3GHz processor. Version 5.3 of the Fujitsu C compiler was used, with identical optimization option -Kfast. The right-hand side $\boldsymbol{b}$ of a linear system of equations is set as the exact solutions are all 1.0 in problem 1. On the other hand, in problem 2, for the right-hand side $\boldsymbol{b}$, the physically realistic load vector was used. The stopping criterion for successful convergence of the CG method is less than 10^{-9} of the relative residual l_2-norm $||\boldsymbol{r_m}||_2/||\boldsymbol{b} - A\boldsymbol{x_0}||_2$. In all cases the iteration was started with the initial solution vector $\boldsymbol{x}_0 = 0$. The maximum iterations of CG method is set as same as the problem size n, respectively. All matrices are normalized with diagonal scaling.

In original SAINV and RIF preconditionings, dropping tolerance of tol ranges from 0.01 to 0.16 in increments of 0.01. On the other hand, in the improved SAINV and **IRIF** preconditionings, dropping tolerance of tol_dd which implies the ratio to tolerance of tol ranges from 1.0 to 5.0 in increments of 0.5. In total, 144 cases were tested for some preconditionings above, respectively.

In Fig. 1 distribution of stress of x-y component gained from analysis of the matrix BEAM is depicted. Similarly in Fig. 2 we show distribution of stress of x-y component gained from analysis of the matrix CABLE. From consideration with a wider range of physical phenomenon, we have found that this analysis is reasonable for designing the concrete bridge. Total stress was estimated to be within expected limitation[5].

5.2. Numerical Results

5.2.1. Results for problem 1

Numerical results in case of the least computation time among tested cases are indicated in Table 3. IC denotes the standard incomplete Cholesky decomposition. ISAINV and **IRIF** denote the improved of stabilized AINV and improved RIF preconditioned CG methods, respectively. On the other hand SAINV and RIF denote the standard SAINV and RIF pre-

Table 3. Numerical results of several kinds of preconditioned CG methods for problem 1.

Matrix	precond.	tol-1	tol-2 (ratio)	mem. (MB)	ratio no. entries	its.	pre-t (sec.)	its-t (sec.)	total-t (sec.)	ratio total-t
BCSSTK24	none	–	–	–	–	∞	–	–	–	–
	IC	–	–	0.03	1.00	∞	0.07	–	–	–
	SAINV	0.10	–	7.00	0.45	1061	0.84	2.29	3.13	1.59
	ISAINV	0.13	0.455	2.26	0.12	1044	0.16	1.64	1.80	0.91
	RIF	0.10	–	6.90	0.22	666	0.85	1.12	1.97	1.00
	IRIF	0.04	0.100	3.40	0.42	289	0.38	0.56	0.94	**0.48**
BCSSTK35	none	–	–	–	–	∞	–	–	–	–
	IC	–	–	0.23	1.00	15176	1.66	727	729	4.73
	SAINV	0.13	–	27.9	0.43	13848	12.0	426	438	2.84
	ISAINV	0.14	0.420	21.3	0.24	11199	5.46	272	277	1.80
	RIF	0.02	–	53.5	1.07	1991	59.2	94.4	154	1.00
	IRIF	0.01	0.030	34.9	1.25	1098	31.2	57.4	88.5	0.57
NASASRB	none	–	–	–	–	14605	–	468	468	1.80
	IC	–	–	0.42	1.00	5477	1.93	491	493	1.93
	SAINV	0.12	–	43.8	0.34	7761	16.6	392	408	1.56
	ISAINV	0.15	0.750	36.6	0.07	7923	13.1	285	298	1.15
	RIF	0.08	–	46.8	0.40	4155	18.4	241	259	1.00
	IRIF	0.02	0.100	50.9	0.86	1405	20.5	111	132	**0.51**
TUBE1-2	none	–	–	–	–	13832	–	146	146	2.24
	IC	–	–	0.16	1.00	5746	0.77	167	168	2.58
	SAINV	0.13	–	35.3	0.32	4134	25.9	68.0	93.8	1.44
	ISAINV	0.05	0.250	15.0	0.51	2050	4.48	42.0	46.5	0.71
	RIF	0.15	–	35.7	0.15	2558	27.0	38.0	65.1	1.00
	IRIF	0.04	0.200	15.5	0.53	1094	5.26	22.7	27.9	**0.42**

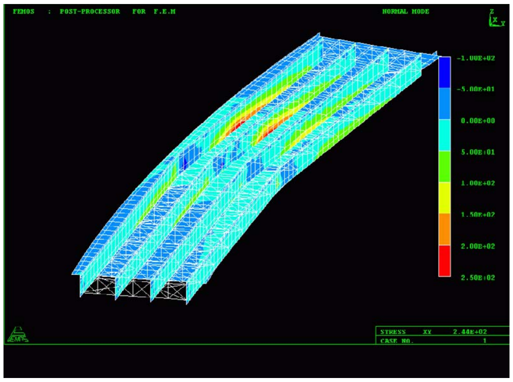

Fig. 1. Distribution of stress of x-y component gained from analysis for matrix BEAM.

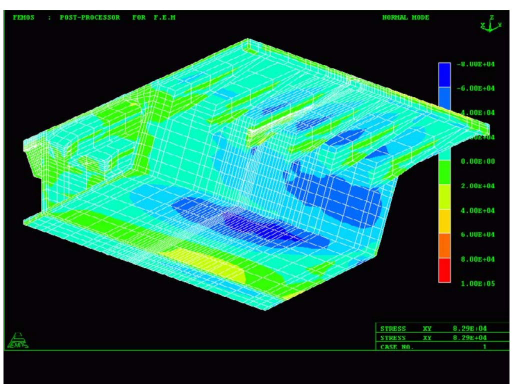

Fig. 2. Distribution of stress of x-y component gained from analysis for matrix CABLE.

conditioned CG methods, respectively.

In Tables “precond.” means the preconditioning. Similarly, “ratio total-time” means ratio of total time of preconditioning to the original RIF preconditioning, respectively. Moreover, “tol-1”, “tol-2” and “tol”, “tol_dd” represent the dropping tolerance values, and “its.” means iteration counts for convergence of the CG method. “mem. (MB)” means needed amount of memory in Mega bytes for lower triangular matrix L, and “ratio no. entries” means the ratio of number of nonzero elements of each preconditioner matrix to that of the original coefficient matrix. In the same way, “pre-t” and “its-t” mean computation time in seconds of preconditioning and iteration for convergence of the CG method, respectively. Also, “total-t” and “ratio total-t” mean total computation time in seconds and the ratio of computation time of each preconditioning to that of RIF preconditioning, respectively. The symbol “∞” denotes nonconvergence until the maximum iterations. In two cases (matrices BCSSTK24 and BCSSTK35), CG method without preconditioning did not converge, and in one case (matrix BCSSTK24) CG method with IC factorization did not converge until iterations reached at maximum iteration counts. Furthermore the bold figures of **IRIF** preconditioning perform significantly well among other preconditionings.

The results, shown in Table 3, include the following statistics for improved SAINV and RIF preconditionings.

1. Over all preconditionings, CG method with **IRIF** preconditioning has the robust and fastest convergence property for tested four matrices. Typically this requires nearly half as much time as original RIF preconditioning which is the least among other preconditionings.

2. Iteration counts of **IRIF** preconditioning decrease drastically over other preconditionings. In particular, iterations needed for **IRIF** preconditioning for matrix BCSSTK35 is within 10% of that of conventional IC factorization preconditioning.

3. The feature of memory requirement of **IRIF** preconditioning results in a modest decrease in memory use except for only one case (matrix NASASRB) in case of optimized cases as shown in Table 3, respectively.

We show convergence history of CG methods with IC, SAINV, ISAINV, RIF and **IRIF** preconditionings for matrices TUBE1-2 in Fig. 3. The horizontal axis of Figure means the count of iterations of CG method, and the vertical axis means relative residual l_2 norm $||r_m||_2/||r_0||_2$ in logarithm $\log_{10}$. The relative residual $||r_m||_2/||r_0||_2$ of **IRIF**, ISAINV, RIF and SAINV preconditionings are plotted as dashed lines in sky blue, orange, pink and dark blue color, respectively. Also that of IC preconditioning is plotted as solid line in green color.

Proceedings of the Seventh International Conference on High Performance Computing and Grid in Asia Pacific Region (HPCAsia'04)
0-7695-2138-X/04 $ 20.00 IEEE

Table 4. Numerical results of several kinds of preconditioned CG methods for problem 2.

Matrix	precond.	tol	tol_dd (ratio)	mem. (MB)	ratio no. entries	its.	pre-t (sec.)	its-t (sec.)	total-t (sec.)	ratio total-t
BEAM	none	–	–	–	–	∞	–	–	–	–
	IC	–	–	0.08	1.00	7620	0.3	102	102	6.46
	SAINV	0.11	–	11.8	0.47	3166	2.68	28.6	31.2	1.97
	ISAINV	0.08	0.280	6.83	0.23	1468	0.85	10.4	11.2	0.71
	RIF	0.11	–	11.5	0.25	1735	2.70	13.1	15.8	1.00
	IRIF	0.02	0.070	9.33	0.56	**447**	1.67	4.49	6.16	**0.39**
CABLE	none	–	–	–	–	7330	–	314	314	2.92
	IC	–	–	0.45	1.00	3024	2.24	376	378	3.41
	SAINV	0.07	–	75.3	0.27	1470	36.0	96.6	133	1.20
	ISAINV	0.06	0.060	58.7	0.32	1388	23.9	96.7	121	1.10
	RIF	0.04	–	88.1	0.38	786	49.2	61.9	111	1.00
	IRIF	0.04	0.080	61.2	0.37	785	28.9	61.3	90.2	0.81

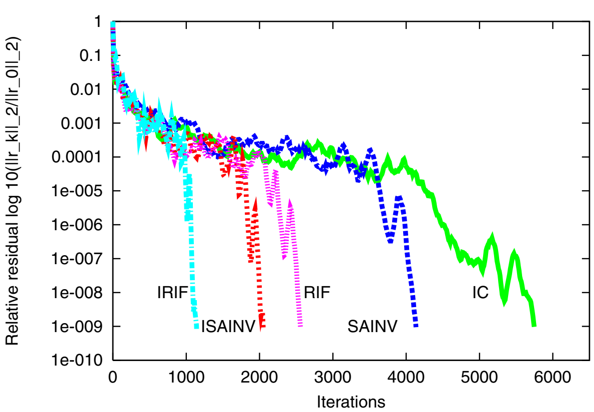

Fig. 3. Convergence history of CG methods with IC, SAINV, ISAINV, RIF and **IRIF** preconditionings for matrix TUBE1-2.

5.2.2. Results for problem 2

Numerical results in case of the least computation time among tested cases are indicated in Table 4.

We show convergence history of CG methods with IC, SAINV, ISAINV, RIF and **IRIF** preconditionings for matrices BEAM and CABLE in Figs. 4 and 5, respectively. As well as Fig. 3 , the relative residual $||r_m||_2/||r_0||_2$ of **IRIF**, ISAINV, RIF and SAINV preconditionings are plotted in dashed lines, and that of IC is plotted in solid line. From Table 4, we can observe that improvement of both ISAINV and **IRIF** preconditionings for matrix BEAM is highly noticeable. On the other hand, improvement of convergence of preconditioning for matrix CABLE is slightly comparable with original preconditionings. This is caused by difference between shell elements and solid elements used in each FEM analysis. See reference [2] for more detail.

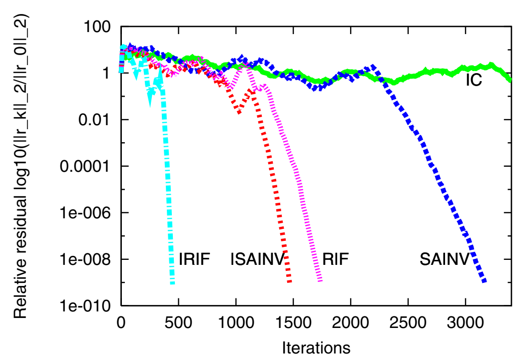

Fig. 4. Convergence history of CG methods with IC, SAINV, ISAINV, RIF and **IRIF** preconditionings for matrix BEAM.

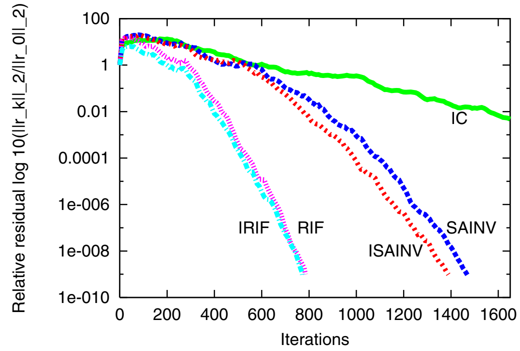

Fig. 5. Convergence history of CG methods with IC, SAINV, ISAINV, RIF and **IRIF** preconditionings for matrix CABLE.

6. CONCLUDING REMARKS

We have demonstrated the advantage of the improved SAINV and RIF preconditionings. The resulting double dropping strategy performs well for matrices from a wide range of disciplines, and is an improvement over the previous SAINV and RIF preconditionings. We also mention that our current discussion compares performance on only one machine and comparative behavior can be strongly influenced by the computing platform being used. However, what we would like to highlight is the improvement that double dropping strategy is at least comparable in performance with other dropping procedures on our test set. Since effectiveness of **IRIF** preconditioning with double dropping is not so noticeable in case of a matrix which stems from FEM analysis with solid elements, this issue remains as a future work.

Acknowledgments.
We would like to thank sincerely Prof. M. Benzi and Dr. M. Tuma for useful suggestions and discussions.

7. REFERENCES

[1] M. Benzi, C.D. Meyer, M. Tuma: A sparse approximate inverse preconditioner for the conjugate gradient method, *SIAM J. on Scientific Computing*, 17(1996), pp.1135-1149.

[2] M. Benzi, R. Kouhia, M. Tuma: An assessment of some preconditioning techniques in shell problems, *Commun. in Numer. method in Engineering*, 14(1998), pp.897-906.

[3] M. Benzi, J.K. Cullum, M. Tuma: Robust approximate inverse preconditioning for the conjugate gradient method, *SIAM J. on Scientific Computing*, 22(2000), pp.1318-1332.

[4] M. Benzi, M. Tuma: A robust incomplete factorization preconditioner for positive definite matrices, *Numer. Lin. Alg. Appl.*, 10(2003), pp.385-400.

[5] Y. Harada: Structural analysis with FEMLEEG, Hoct system (2003).

[6] L. Fox, H.D. Huskey, J.H. Wilkinson: Notes on the solution of algebraic linear simultaneous equations, *Quarterly J. of Mech. and Appl. Math.*, 1(1948), pp.149-173.

[7] Y. Ikeda, S. Fujino: An improvement of stabilized AINV preconditioning by double dropping, Proc. of the International Symposium ISEE2003, pp.393-396, Fukuoka, Nov. 13-14, 2003.

[8] Y. Ikeda, S. Fujino: An Improvement of Stabilized Approximate INVerse preconditioning by double dropping, *Trans. of IPSJ*, Vol.45 No.SIG1(ACS4) (2004), pp.10-17. (in Japanese)

[9] Y. Ikeda, S. Fujino: An effective use of Improved Stabilized Approximate INVerse preconditioning according to characteristics of computers, *Trans. of INFORMATION*, 7(2004). (in press) (in Japanese)

[10] Y. Ikeda, S. Fujino, M. Kakihara, A. Inoue: An enhancement of efficiency for Robust Incomplete Factorization preconditioning based upon A-orthogonalization process, *Trans. of IPSJ*, Vol.45 No.SIG1(ACS6) (2004). (in press) (in Japanese)

[11] S.A. Kharchenko, L.Y. Kolotilina, A.A. Nikishin, A.Yu. Yeremin: A robust AINV-type method for constructing sparse approximate inverse preconditioners in factored form, *Numer. Lin. Alg. Appl.*, 8(2001), pp.165-179.

[12] R. Kouhia, Sparse Matrices web page: http://www.hut.fi/~kouhia/sparse.html

[13] Matrix Market web page: http://math.nist.gov/MatrixMarket/

[14] J.A. Meijerink, H.A. van der Vorst: An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Mathematics of Computation*, 31(1977), pp. 148–162.

[15] University of Florida Sparse Matrix web page: http://www.cise.ufl.edu/research/sparse/matrices/