Applied Mathematics, 2015, 6, 1389-1406
Published Online July 2015 in SciRes. http://www.scirp.org/journal/am
http://dx.doi.org/10.4236/am.2015.68131

Scientific Research Publishing

Formulation of a Preconditioned Algorithm for the Conjugate Gradient Squared Method in Accordance with Its Logical Structure

Shoji Itoh[1], Masaaki Sugihara[2]

[1]Division of Science, School of Science and Engineering, Tokyo Denki University, Saitama, Japan
[2]Department of Physics and Mathematics, College of Science and Engineering, Aoyama Gakuin University, Kanagawa, Japan
Email: itosho@acm.org

Received 22 June 2015; accepted 27 July 2015; published 30 July 2015

Abstract

In this paper, we propose an improved preconditioned algorithm for the conjugate gradient squared method (improved PCGS) for the solution of linear equations. Further, the logical structures underlying the formation of this preconditioned algorithm are demonstrated via a number of theorems. This improved PCGS algorithm retains some mathematical properties that are associated with the CGS derivation from the bi-conjugate gradient method under a non-preconditioned system. A series of numerical comparisons with the conventional PCGS illustrate the enhanced effectiveness of our improved scheme with a variety of preconditioners. This logical structure underlying the formation of the improved PCGS brings a spillover effect from various bi-Lanczos-type algorithms with minimal residual operations, because these algorithms were constructed by adopting the idea behind the derivation of CGS. These bi-Lanczos-type algorithms are very important because they are often adopted to solve the systems of linear equations that arise from large-scale numerical simulations.

Keywords

Linear Systems, Krylov Subspace Method, Bi-Lanczos Algorithm, Preconditioned System, PCGS

1. Introduction

In scientific and technical computation, natural phenomena or engineering problems are described through numerical models. These models are often reduced to a system of linear equations:

How to cite this paper: Itoh, S. and Sugihara, M. (2015) Formulation of a Preconditioned Algorithm for the Conjugate Gradient Squared Method in Accordance with Its Logical Structure. *Applied Mathematics*, **6**, 1389-1406.
http://dx.doi.org/10.4236/am.2015.68131

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

where A is a large, sparse coefficient matrix of size $n \times n$, $\boldsymbol{x}$ is the solution vector, and $\boldsymbol{b}$ is the right-hand side (RHS) vector.

The conjugate gradient squared (CGS) method is a way to solve (1) [1]. The CGS method is a type of bi-Lanczos algorithm that belongs to the class of Krylov subspace methods.

Bi-Lanczos-type algorithms are derived from the bi-conjugate gradient (BiCG) method [2] [3], which assumes the existence of a dual system

$$A^{\mathrm{T}}\boldsymbol{x}^{\#} = \boldsymbol{b}^{\#}. \tag{2}$$

Characteristically, the coefficient matrix of (2) is the transpose of A. In this paper, we term (2) a "shadow system".

Bi-Lanczos-type algorithms have the advantage of requiring less memory than Arnoldi-type algorithms, another class of Krylov subspace methods.

The CGS method is derived from BiCG. Furthermore, various bi-Lanczos-type algorithms, such as BiCGStab [4], GPBiCG [5] and so on, have been constructed by adopting the idea behind the derivation of CGS. These bi-Lanczos-type algorithms are very important because they are often adopted to solve systems in the form of (1) that arise from large-scale numerical simulations.

Many iterative methods, including bi-Lanczos algorithms, are often applied together with some preconditioning operation. Such algorithms are called preconditioned algorithms; for example, preconditioned CGS (PCGS). The application of preconditioning operations to iterative methods effectively enhances their performance. Indeed, the effects attributable to different preconditioning operations are greater than those produced by different iterative methods [6]. However, if a preconditioned algorithm is poorly designed, there may be no beneficial effect from the preconditioning operation.

Consequently, PCGS holds an important position within the Krylov subspace methods. In this paper, we identify a mathematical issue with the conventional PCGS algorithm, and propose an improved PCGS[1]. This improved PCGS algorithm is derived rationally in accordance with its logical structure.

In this paper, *preconditioned algorithm* and *preconditioned system* refer to solving algorithms described with some preconditioning operator M (or preconditioner, preconditioning matrix) and the system converted by the operator based on M, respectively. These terms never indicate *the algorithm for the preconditioning operation itself*, such as "incomplete LU decomposition", "approximate inverse", and so on. For example, for a preconditioned system, the original linear system (1) becomes

$$\tilde{\boldsymbol{A}}\boldsymbol{x} = \boldsymbol{b}, \tag{3}$$

$$\tilde{A} = M_L^{-1}\boldsymbol{A}M_R^{-1}, \quad \tilde{\boldsymbol{x}} = M_R\boldsymbol{x}, \quad \tilde{\boldsymbol{b}} = M_L^{-1}\boldsymbol{b}, \tag{4}$$

under the preconditioner $M = M_L M_R \ (M \approx A)$. Here, the matrix and the vector in the preconditioned system are denoted by the tilde $\tilde{(\ )}$. However, the conversions in (3) and (4) are not implemented; rather, we construct the preconditioned algorithm that is equivalent to solving (3).

This paper is organized as follows. Section 2 provides an overview of the derivation of the CGS method and the properties of two scalar coefficients (α_k and β_k). This forms an important part of our argument for the preconditioned BiCG and PCGS algorithms in the next section. Section 3 introduces the PCGS algorithm. An issue with the conventional PCGS algorithm is identified, and an improved algorithm is proposed. In Section 4, we present some numerical results to demonstrate the effect of the improved PCGS algorithm with a variety of preconditioners. As a consequence, the effectiveness of the improved algorithm is clearly established. Finally, our conclusions are presented in Section 5.

2. Derivation of the CGS Method and Preconditioned Algorithm of the BiCG Method

In this section, we derive the CGS method from the BiCG method, and introduce the preconditioned BiCG algo-

[1]This improved PCGS was proposed in Ref. [7] from a viewpoint of deriving process's logic, but this manuscript demonstrates some mathematical theorems underlying the deriving process's logic. Further, numerical results using ILU(0) preconditioner only are shown in Ref. [7], but the results using a variety of typical preconditioners are shown in this manuscript.

rithm.

2.1. Structure of the BiCG Method

BiCG [2] [3] is an iterative method for linear systems in which the coefficient matrix A is nonsymmetric. The algorithm proceeds as follows:

Algorithm 1. BiCG method:

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, set $\beta_{-1} = 0$,
$\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_0\right) \neq 0$, e.g., $\boldsymbol{r}_0^{\#} = \boldsymbol{r}_0$,

For $k = 0, 1, 2, \cdots,$ until convergence, Do:

$$\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1}\boldsymbol{p}_{k-1}, \quad \boldsymbol{p}_k^{\#} = \boldsymbol{r}_k^{\#} + \beta_{k-1}\boldsymbol{p}_{k-1}^{\#},$$

$$\alpha_k = \frac{\left(\boldsymbol{r}_k^{\#}, \boldsymbol{r}_k\right)}{\left(\boldsymbol{p}_k^{\#}, A\boldsymbol{p}_k\right)}, \tag{5}$$

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

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k, \quad \boldsymbol{r}_{k+1}^{\#} = \boldsymbol{r}_k^{\#} - \alpha_k A^{\mathrm{T}}\boldsymbol{p}_k^{\#},$$

$$\beta_k = \frac{\left(\boldsymbol{r}_{k+1}^{\#}, \boldsymbol{r}_{k+1}\right)}{\left(\boldsymbol{r}_k^{\#}, \boldsymbol{r}_k\right)}, \tag{6}$$

End Do

BiCG implements the following Theorems.

Theorem 1 (Hestenes *et al*. [8], Sonneveld [1], and others.)[2] *For a non-preconditioned system, there are recurrence relations that define the degree k of the residual polynomial* $\mathsf{R}_k(\lambda)$ *and probing direction polynomial* $\mathsf{P}_k(\lambda)$. *These are*

$$\mathsf{R}_0(\lambda) = 1, \ \mathsf{P}_0(\lambda) = 1, \tag{7}$$

$$\mathsf{R}_k(\lambda) = \mathsf{R}_{k-1}(\lambda) - \alpha_{k-1}\lambda\mathsf{P}_{k-1}(\lambda), \tag{8}$$

$$\mathsf{P}_k(\lambda) = \mathsf{R}_k(\lambda) + \beta_{k-1}\mathsf{P}_{k-1}(\lambda), \tag{9}$$

where α_k *and* β_k *are based on* (5) *and* (6).

Using the polynomials of Theorem 1, the residual vector for the linear system (1) and the shadow residual vector for the shadow system (2) can be written as

$$\boldsymbol{r}_k = \mathsf{R}_k(A)\boldsymbol{r}_0, \tag{10}$$

$$\boldsymbol{r}_k^{\#} = \mathsf{R}_k\left(A^{\mathrm{T}}\right)\boldsymbol{r}_0^{\#}. \tag{11}$$

These probing direction vectors are represented by

$$\boldsymbol{p}_k = \mathsf{P}_k(A)\boldsymbol{r}_0, \tag{12}$$

$$\boldsymbol{p}_k^{\#} = \mathsf{P}_k\left(A^{\mathrm{T}}\right)\boldsymbol{r}_0^{\#}, \tag{13}$$

where $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$ is the initial residual vector and $\boldsymbol{r}_0^{\#} = \boldsymbol{b}^{\#} - A^{\mathrm{T}}\boldsymbol{x}_0^{\#}$ is the initial shadow residual vector. However, in practice, $\boldsymbol{r}_0^{\#}$ is set in such a way as to satisfy the conditions $\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_0\right) \neq 0$. $(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{v}$ denotes the inner product between vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. In this paper, we set $\boldsymbol{r}_0^{\#} = \boldsymbol{r}_0$ to satisfy the conditions $\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_0\right) \neq 0$

[2]Of course, [8] discuss the conjugate gradient method for a symmetric system.

exactly. This is a typical way of ensuring $\boldsymbol{r}_0^{\#}$; other settings are beyond the scope of this paper.

Theorem 2 (Fletcher [2]) *The BiCG method satisfies the following conditions*:

$$\left(\boldsymbol{r}_i^{\#}, \boldsymbol{r}_j\right) = 0 \quad (i \neq j) \quad \text{(bi-orthogonality conditions)}, \tag{14}$$

$$\left(\boldsymbol{p}_i^{\#}, A\boldsymbol{p}_j\right) = 0 \quad (i \neq j) \quad \text{(bi-conjugacy conditions)}. \tag{15}$$

2.2. Derivation of the CGS Method

The CGS method is derived by transforming the scalar coefficients in the BiCG method to avoid the A^{T} matrix [1]. The polynomial defined by (10)-(13) is substituted into (14) and (15), which construct the numerator and denominator of α_k and β_k in BiCG[3]. Then,

$$\left(\boldsymbol{r}_k^{\#}, \boldsymbol{r}_k\right) = \left(\mathrm{R}_k\left(A^{\mathrm{T}}\right)\boldsymbol{r}_0^{\#}, \mathrm{R}_k\left(A\right)\boldsymbol{r}_0\right) = \left(\boldsymbol{r}_0^{\#}, \mathrm{R}_k^2\left(A\right)\boldsymbol{r}_0\right), \tag{16}$$

$$\left(\boldsymbol{p}_k^{\#}, A\boldsymbol{p}_k\right) = \left(\mathrm{P}_k\left(A^{\mathrm{T}}\right)\boldsymbol{r}_0^{\#}, A\mathrm{P}_k\left(A\right)\boldsymbol{r}_0\right) = \left(\boldsymbol{r}_0^{\#}, A\mathrm{P}_k^2\left(A\right)\boldsymbol{r}_0\right), \tag{17}$$

and the following theorem can be applied.

Theorem 3 (Sonneveld [1]) *The CGS coefficients α_k and β_k are equivalent to these coefficients in BiCG under certain transformation and substitution operations based on the bi-orthogonality and bi-conjugacy conditions.*

Proof. We apply

$$\boldsymbol{r}_k^{\mathrm{CGS}} \equiv \mathrm{R}_k^2\left(A\right)\boldsymbol{r}_0, \quad \boldsymbol{p}_k^{\mathrm{CGS}} \equiv \mathrm{P}_k^2\left(A\right)\boldsymbol{r}_0 \tag{18}$$

to (16) and (17). Then,

$$\alpha_k^{\mathrm{BiCG}} = \frac{\left(\boldsymbol{r}_k^{\#}, \boldsymbol{r}_k\right)}{\left(\boldsymbol{p}_k^{\#}, A\boldsymbol{p}_k\right)} = \frac{\left(\boldsymbol{r}_0^{\#}, \mathrm{R}_k^2\left(A\right)\boldsymbol{r}_0\right)}{\left(\boldsymbol{p}_0^{\#}, A\mathrm{P}_k^2\left(A\right)\boldsymbol{r}_0\right)} = \frac{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_k^{\mathrm{CGS}}\right)}{\left(\boldsymbol{r}_0^{\#}, A\boldsymbol{p}_k^{\mathrm{CGS}}\right)} \equiv \alpha_k^{\mathrm{CGS}}, \tag{19}$$

$$\beta_k^{\mathrm{BiCG}} = \frac{\left(\boldsymbol{r}_{k+1}^{\#}, \boldsymbol{r}_{k+1}\right)}{\left(\boldsymbol{r}_k^{\#}, \boldsymbol{r}_k\right)} = \frac{\left(\boldsymbol{r}_0^{\#}, \mathrm{R}_{k+1}^2\left(A\right)\boldsymbol{r}_0\right)}{\left(\boldsymbol{r}_0^{\#}, \mathrm{R}_k^2\left(A\right)\boldsymbol{r}_0\right)} = \frac{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_{k+1}^{\mathrm{CGS}}\right)}{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_k^{\mathrm{CGS}}\right)} \equiv \beta_k^{\mathrm{CGS}}. \tag{20}$$

□

The CGS method is derived from BiCG by Theorem 4.

Theorem 4 (Sonneveld [1]) *The CGS method is derived from the linear system's recurrence relations in the BiCG method under the property of equivalence between the coefficients α_k, β_k in CGS and BiCG. However, the solution vector $\boldsymbol{x}_k^{\mathrm{CGS}}$ is derived from a recurrence relation based on $\boldsymbol{r}_k^{\mathrm{CGS}} = \boldsymbol{b} - A\boldsymbol{x}_k^{\mathrm{CGS}}$.*

Proof. The coefficients α_k and β_k in BiCG and CGS were derived in Theorem 3. The recurrence relations (8) and (9) for BiCG are squared to give:

$$\begin{aligned}\mathrm{R}_k^2\left(\lambda\right) &= \left(\mathrm{R}_{k-1}\left(\lambda\right) - \alpha_{k-1}\lambda\mathrm{P}_{k-1}\left(\lambda\right)\right)^2 \\ &= \mathrm{R}_{k-1}^2\left(\lambda\right) - 2\alpha_{k-1}\lambda\mathrm{P}_{k-1}\left(\lambda\right)\mathrm{R}_{k-1}\left(\lambda\right) + \alpha_{k-1}^2\lambda^2\mathrm{P}_{k-1}^2\left(\lambda\right),\end{aligned} \tag{21}$$

$$\begin{aligned}\mathrm{P}_k^2\left(\lambda\right) &= \left(\mathrm{R}_k\left(\lambda\right) + \beta_{k-1}\mathrm{P}_{k-1}\left(\lambda\right)\right)^2 \\ &= \mathrm{R}_k^2\left(\lambda\right) + 2\beta_{k-1}\mathrm{P}_{k-1}\left(\lambda\right)\mathrm{R}_k\left(\lambda\right) + \beta_{k-1}^2\mathrm{P}_{k-1}^2\left(\lambda\right).\end{aligned} \tag{22}$$

Further, we can apply $\boldsymbol{r}_k^{\mathrm{CGS}} \equiv \mathrm{R}_k^2\left(A\right)\boldsymbol{r}_0$, $\boldsymbol{p}_k^{\mathrm{CGS}} \equiv \mathrm{P}_k^2\left(A\right)\boldsymbol{r}_0$ from (18), and substitute $\boldsymbol{u}_k \equiv \mathrm{P}_k\left(A\right)\mathrm{R}_k\left(A\right)\boldsymbol{r}_0$, $\boldsymbol{q}_k \equiv \mathrm{P}_k\left(A\right)\mathrm{R}_{k+1}\left(A\right)\boldsymbol{r}_0$. □

Thus, we have derived the CGS method[4].

[3]In this paper, if we specifically distinguish this algorithm, we write α_k^{BiCG} and β_k^{BiCG}.

[4]In this paper, we use the superscript "CGS" alongside α_k, β_k, and relevant vectors to describe this algorithm.

Algorithm 2. CGS method:

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, set $\beta_{-1}^{\mathrm{CGS}} = 0$,
$\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_0\right) \neq 0$, e.g., $\boldsymbol{r}_0^{\#} = \boldsymbol{r}_0$,
For $k = 0,1,2,\cdots$, until convergence, Do:

$$\boldsymbol{u}_k = \boldsymbol{r}_k^{\mathrm{CGS}} + \beta_{k-1}^{\mathrm{CGS}} \boldsymbol{q}_{k-1},$$

$$\boldsymbol{p}_k^{\mathrm{CGS}} = \boldsymbol{u}_k + \beta_{k-1}^{\mathrm{CGS}} \left(\boldsymbol{q}_{k-1} + \beta_{k-1}^{\mathrm{CGS}} \boldsymbol{p}_{k-1}^{\mathrm{CGS}}\right),$$

$$\alpha_k^{\mathrm{CGS}} = \frac{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_k^{\mathrm{CGS}}\right)}{\left(\boldsymbol{r}_0^{\#}, A\boldsymbol{p}_k^{\mathrm{CGS}}\right)},$$

$$\boldsymbol{q}_k = \boldsymbol{u}_k - \alpha_k^{\mathrm{CGS}} A\boldsymbol{p}_k^{\mathrm{CGS}},$$

$$\boldsymbol{x}_{k+1}^{\mathrm{CGS}} = \boldsymbol{x}_k^{\mathrm{CGS}} + \alpha_k^{\mathrm{CGS}} \left(\boldsymbol{u}_k + \boldsymbol{q}_k\right),$$

$$\boldsymbol{r}_{k+1}^{\mathrm{CGS}} = \boldsymbol{r}_k^{\mathrm{CGS}} - \alpha_k^{\mathrm{CGS}} A\left(\boldsymbol{u}_k + \boldsymbol{q}_k\right),$$

$$\beta_k^{\mathrm{CGS}} = \frac{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_{k+1}^{\mathrm{CGS}}\right)}{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_k^{\mathrm{CGS}}\right)},$$

End Do

The following Proposition 5 and Corollary 1 are given as a supplementary explanation for Algorithm 2. These are almost trivial, but are comparatively important in the next section's discussion.

Proposition 5 *There exist the following relations*:

$$\boldsymbol{r}_0^{\mathrm{CGS}} = \boldsymbol{r}_0, \tag{23}$$

$$\boldsymbol{r}_0^{\mathrm{CGS}\#} = \boldsymbol{r}_0^{\#}, \tag{24}$$

where r_0^{CGS} *is the initial residual vector at* $k = 0$ *in the iterative part of CGS,* $r_0^{\mathrm{CGS}\#}$ *is the initial shadow residual vector in CGS,* $\boldsymbol{r}_0$ *is the initial residual vector, and* $\boldsymbol{r}_0^{\#}$ *is the initial shadow residual vector*.

Proof. Equation (23) follows because (18) for $k = 0$ gives $\boldsymbol{r}_0^{\mathrm{CGS}} = \mathsf{R}_0^2(A)\boldsymbol{r}_0 = \boldsymbol{r}_0$. Here, $\mathsf{R}_0(A) = I$ by (7), and I denotes the identity matrix.

Equation (24) is derived as follows. Applying (18) to (16), we obtain

$$\left(\boldsymbol{r}_k^{\#}, \boldsymbol{r}_k\right) = \left(\mathsf{R}_k\left(A^{\mathrm{T}}\right)\boldsymbol{r}_0^{\#}, \mathsf{R}_k(A)\boldsymbol{r}_0\right) = \left(\boldsymbol{r}_0^{\#}, \mathsf{R}_k^2(A)\boldsymbol{r}_0\right) \equiv \left(\boldsymbol{r}_0^{\mathrm{CGS}\#}, \boldsymbol{r}_k^{\mathrm{CGS}}\right).$$

This equation shows that the inner product of the CGS on the right is obtained from the inner product of the BiCG on the left. Therefore, $\boldsymbol{r}_0^{\#}$ that composes the polynomial $\mathsf{R}_k\left(A^{\mathrm{T}}\right)\boldsymbol{r}_0^{\#}$ to express the shadow residual vector of the BiCG is the same as $\boldsymbol{r}_0^{\mathrm{CGS}\#}$ in CGS.

Hereafter, $\boldsymbol{r}_0^{\mathrm{CGS}\#}$ can be represented by $\boldsymbol{r}_0^{\#}$, to the extent that neither can be distinguished. □

Corollary 1. *There exists the following relation*:

$$\boldsymbol{p}_0^{\mathrm{CGS}} = \boldsymbol{r}_0^{\mathrm{CGS}} = \boldsymbol{r}_0,$$

where $\boldsymbol{p}_0^{\mathrm{CGS}}$ *is the probing direction vector in CGS,* $\boldsymbol{r}_0^{\mathrm{CGS}}$ *is the initial residual vector at* $k = 0$ *in the iterative part of CGS, and* $\boldsymbol{r}_0$ *is the initial residual vector*.

2.3. Derivation of Preconditioned BiCG Algorithm

In this subsection, the preconditioned BiCG algorithm is derived from the non-preconditioned BiCG method (Algorithm 1). First, some basic aspects of the BiCG method under a preconditioned system are expressed, and a standard preconditioned BiCG algorithm is given.

When the BiCG method (Algorithm 1) is applied to linear equations under a preconditioned system:

$$\tilde{A}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}, \tag{25}$$

we obtain a "BiCG method under a preconditioned system" (Algorithm 3). We denote this as "PBiCG".

In this paper, matrices and vectors under the preconditioned system are denoted with " ˜ ", such as $\tilde{A}$, $\tilde{\boldsymbol{p}}_k$[5]. The coefficients α_k, β_k are specified by $\alpha_k^{\mathrm{PBiCG}}$, β_k^{PBiCG}.

Algorithm 3. BiCG method under the preconditioned system:

$\tilde{\boldsymbol{x}}_0$ is an initial guess, $\tilde{\boldsymbol{r}}_0 = \tilde{\boldsymbol{b}} - \tilde{A}\tilde{\boldsymbol{x}}_0$, set $\beta_{-1}^{\mathrm{PBiCG}} = 0$,
$\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right) \neq 0$, e.g., $\tilde{\boldsymbol{r}}_0^{\#} = \tilde{\boldsymbol{r}}_0$,
For $k = 0,1,2,\cdots,$ until convergence, Do:

$$\tilde{\boldsymbol{p}}_k = \tilde{\boldsymbol{r}}_k + \beta_{k-1}^{\mathrm{PBiCG}} \tilde{\boldsymbol{p}}_{k-1}, \quad \tilde{\boldsymbol{p}}_k^{\#} = \tilde{\boldsymbol{r}}_k^{\#} + \beta_{k-1}^{\mathrm{PBiCG}} \tilde{\boldsymbol{p}}_{k-1}^{\#},$$

$$\alpha_k^{\mathrm{PBiCG}} = \frac{\left(\tilde{\boldsymbol{r}}_k^{\#}, \tilde{\boldsymbol{r}}_k\right)}{\left(\tilde{\boldsymbol{p}}_k^{\#}, \tilde{A}\tilde{\boldsymbol{p}}_k\right)}, \tag{26}$$

$$\tilde{\boldsymbol{x}}_{k+1} = \tilde{\boldsymbol{x}}_k + \alpha_k^{\mathrm{PBiCG}} \tilde{\boldsymbol{p}}_k,$$

$$\tilde{\boldsymbol{r}}_{k+1} = \tilde{\boldsymbol{r}}_k - \alpha_k^{\mathrm{PBiCG}} \tilde{A}\tilde{\boldsymbol{p}}_k, \quad \tilde{\boldsymbol{r}}_{k+1}^{\#} = \tilde{\boldsymbol{r}}_k^{\#} - \alpha_k^{\mathrm{PBiCG}} \tilde{A}^{\mathrm{T}} \tilde{\boldsymbol{p}}_k^{\#},$$

$$\beta_k^{\mathrm{PBiCG}} = \frac{\left(\tilde{\boldsymbol{r}}_{k+1}^{\#}, \tilde{\boldsymbol{r}}_{k+1}\right)}{\left(\tilde{\boldsymbol{r}}_k^{\#}, \tilde{\boldsymbol{r}}_k\right)}, \tag{27}$$

End Do

We now state Theorem 6 and Theorem 7, which are clearly derived from Theorem 1 and Theorem 2, respectively.

Theorem 6. *Under the preconditioned system, there are recurrence relations that define the degree k of the residual polynomial* $R_k\left(\tilde{\lambda}\right)$ *and probing direction polynomial* $P_k\left(\tilde{\lambda}\right)$[6]. *These are*

$$R_0\left(\tilde{\lambda}\right) = 1, \ P_0\left(\tilde{\lambda}\right) = 1, \tag{28}$$

$$R_k\left(\tilde{\lambda}\right) = R_{k-1}\left(\tilde{\lambda}\right) - \alpha_{k-1}^{\mathrm{PBiCG}} \tilde{\lambda} P_{k-1}\left(\tilde{\lambda}\right), \tag{29}$$

$$P_k\left(\tilde{\lambda}\right) = R_k\left(\tilde{\lambda}\right) + \beta_{k-1}^{\mathrm{PBiCG}} P_{k-1}\left(\tilde{\lambda}\right), \tag{30}$$

where $\tilde{\lambda}$ *is the variation under the preconditioned system, and* $\alpha_k^{\mathrm{PBiCG}}$, β_k^{PBiCG} *in these relations are based on* (26) *and* (27).

Using the polynomials of Theorem 6, the residual vectors of the preconditioned linear system (25) and the shadow residual vectors of the following preconditioned shadow system:

$$\tilde{A}^{\mathrm{T}} \tilde{\boldsymbol{x}}^{\#} = \tilde{\boldsymbol{b}}^{\#} \tag{31}$$

can be represented as

$$\tilde{\boldsymbol{r}}_k = R_k\left(\tilde{A}\right)\tilde{\boldsymbol{r}}_0, \tag{32}$$

$$\tilde{\boldsymbol{r}}_k^{\#} = R_k\left(\tilde{A}^{\mathrm{T}}\right)\tilde{\boldsymbol{r}}_0^{\#}, \tag{33}$$

respectively. The probing direction vectors are given by

$$\tilde{\boldsymbol{p}}_k = P_k\left(\tilde{A}\right)\tilde{\boldsymbol{r}}_0, \tag{34}$$

[5]If we wish to emphasize different methods, a superscript is applied to the relevant vectors to denote the method, such as $\boldsymbol{p}_k^{\mathrm{PBiCG}}$, $\tilde{\boldsymbol{p}}_k^{\mathrm{BiCG}}$.

[6]We represent the polynomials *R* and *P* in *italic font* to denote a preconditioned system.

$$\tilde{\boldsymbol{p}}_k^{\#} = P_k\left(\tilde{A}^{\mathrm{T}}\right)\tilde{\boldsymbol{r}}_0^{\#}, \tag{35}$$

respectively. Under the preconditioned system, $\boldsymbol{r}_0^{\#}$ is set in such a way as to satisfy the conditions $\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right) \neq 0$. In this paper, we set $\boldsymbol{r}_0^{\#}$ based on the equation $\tilde{\boldsymbol{r}}_0^{\#} = \tilde{\boldsymbol{r}}_0$ to satisfy the conditions $\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right) \neq 0$ exactly. Some variations based on $\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right)$, such as Algorithm 4 below, are allowed. Other settings are beyond the scope of this paper.

Remark 1. *The shadow systems given by* (31) *do not exist, but it is very important to construct systems in which the transpose of the matrix* $\tilde{A}$ *exists.*

Theorem 7. *The BiCG method under the preconditioned system satisfies the following conditions*:

$$\left(\tilde{\boldsymbol{r}}_i^{\#}, \tilde{\boldsymbol{r}}_j\right) = 0 \quad (i \neq j) \quad \left(\text{bi-orthogonality conditions under the preconditioned system}\right), \tag{36}$$

$$\left(\tilde{\boldsymbol{p}}_i^{\#}, \tilde{A}\tilde{\boldsymbol{p}}_j\right) = 0 \quad (i \neq j) \quad \left(\text{bi-conjugacy conditions under the preconditioned system}\right). \tag{37}$$

Next, we derive the standard PBiCG algorithm. Here, the preconditioned linear system (25) and its shadow system (31) are formed as follows:

$$\left(M_L^{-1} A M_R^{-1}\right)\left(M_R \boldsymbol{x}\right) = M_L^{-1}\boldsymbol{b}, \tag{38}$$

$$\left(M_R^{-\mathrm{T}} A^{\mathrm{T}} M_L^{-\mathrm{T}}\right)\left(M_L^{\mathrm{T}} \boldsymbol{x}^{\#}\right) = M_R^{-\mathrm{T}}\boldsymbol{b}^{\#}. \tag{39}$$

Definition 1. *On the subject of the PBiCG algorithm, the solution vector is denoted as* $\boldsymbol{x}_k^{\mathrm{PBiCG}}$. *Furthermore, the residual vectors of the linear system and shadow system of the PBiCG algorithm are written as* $\boldsymbol{r}_k^{\mathrm{PBiCG}}$ *and* $\boldsymbol{r}_k^{\mathrm{PBiCG}\#}$, *respectively, and their probing direction vectors are* $\boldsymbol{p}_k^{\mathrm{PBiCG}}$ *and* $\boldsymbol{p}_k^{\mathrm{PBiCG}\#}$, *respectively.*

Using this notation, each vector of the BiCG under the preconditioned system given by Algorithm 3 is converted as below:

$$\tilde{\boldsymbol{r}}_k = M_L^{-1}\boldsymbol{r}_k^{\mathrm{PBiCG}}, \quad \tilde{\boldsymbol{r}}_k^{\#} = M_R^{-\mathrm{T}}\boldsymbol{r}_k^{\mathrm{PBiCG}\#}, \quad \tilde{\boldsymbol{p}}_k = M_R\boldsymbol{p}_k^{\mathrm{PBiCG}}, \quad \tilde{\boldsymbol{p}}_k^{\#} = M_L^{\mathrm{T}}\boldsymbol{p}_k^{\mathrm{PBiCG}\#}. \tag{40}$$

Substituting the elements of (40) into (36) and (37), we have

$$\left(\tilde{\boldsymbol{r}}_i^{\#}, \tilde{\boldsymbol{r}}_j\right) = \left(M_R^{-\mathrm{T}}\boldsymbol{r}_i^{\mathrm{PBiCG}\#}, M_L^{-1}\boldsymbol{r}_j^{\mathrm{PBiCG}}\right) = \left(\boldsymbol{r}_i^{\mathrm{PBiCG}\#}, M^{-1}\boldsymbol{r}_j^{\mathrm{PBiCG}}\right), \tag{41}$$

$$\left(\tilde{\boldsymbol{p}}_i^{\#}, \tilde{A}\tilde{\boldsymbol{p}}_j\right) = \left(M_L^{\mathrm{T}}\boldsymbol{p}_i^{\mathrm{PBiCG}\#}, \left(M_L^{-1}AM_R^{-1}\right)\left(M_R\boldsymbol{p}_j^{\mathrm{PBiCG}}\right)\right) = \left(\boldsymbol{p}_i^{\mathrm{PBiCG}\#}, A\boldsymbol{p}_j^{\mathrm{PBiCG}}\right). \tag{42}$$

Consequently, (26) and (27) become

$$\alpha_k^{\mathrm{PBiCG}} = \frac{\left(\tilde{\boldsymbol{r}}_k^{\#}, \tilde{\boldsymbol{r}}_k\right)}{\left(\tilde{\boldsymbol{p}}_k^{\#}, \tilde{A}\tilde{\boldsymbol{p}}_k\right)} = \frac{\left(\boldsymbol{r}_k^{\mathrm{PBiCG}\#}, M^{-1}\boldsymbol{r}_k^{\mathrm{PBiCG}}\right)}{\left(\boldsymbol{p}_k^{\mathrm{PBiCG}\#}, A\boldsymbol{p}_k^{\mathrm{PBiCG}}\right)}, \tag{43}$$

$$\beta_k^{\mathrm{PBiCG}} = \frac{\left(\tilde{\boldsymbol{r}}_{k+1}^{\#}, \tilde{\boldsymbol{r}}_{k+1}\right)}{\left(\tilde{\boldsymbol{r}}_k^{\#}, \tilde{\boldsymbol{r}}_k\right)} = \frac{\left(\boldsymbol{r}_{k+1}^{\mathrm{PBiCG}\#}, M^{-1}\boldsymbol{r}_{k+1}^{\mathrm{PBiCG}}\right)}{\left(\boldsymbol{r}_k^{\mathrm{PBiCG}\#}, M^{-1}\boldsymbol{r}_k^{\mathrm{PBiCG}}\right)}. \tag{44}$$

Before the iterative step, we give the following Definition 2.

Definition 2. *For some preconditioned algorithms, the initial residual vector of the linear system is written as* $\boldsymbol{r}_0^{\mathrm{P}}$ *and the initial shadow residual vector of the shadow system is written as* $\boldsymbol{r}_0^{\mathrm{P}\#}$ *before the iterative step.*

We adopt the following preconditioning conversion after (40).

$$\tilde{\boldsymbol{r}}_0 = M_L^{-1}\boldsymbol{r}_0^{\mathrm{P}}, \quad \tilde{\boldsymbol{r}}_0^{\#} = M_R^{-\mathrm{T}}\boldsymbol{r}_0^{\mathrm{P}\#}. \tag{45}$$

Consequently, we can derive the following standard PBiCG algorithm [9] [10].

Algorithm 4. Standard preconditioned BiCG algorithm:

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0^{\mathrm{P}} = \boldsymbol{b} - A\boldsymbol{x}_0$, set $\beta_{-1}^{\mathrm{PBiCG}} = 0$,

$\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right) = \left(\boldsymbol{r}_0^{\mathrm{P}\#}, M^{-1}\boldsymbol{r}_0^{\mathrm{P}}\right) \neq 0$, e.g., $\boldsymbol{r}_0^{\mathrm{P}\#} = M^{-1}\boldsymbol{r}_0^{\mathrm{P}}$,

For $k = 0,1,2,\cdots,$ until convergence, Do:

$$\boldsymbol{p}_k^{\mathrm{PBiCG}} = M^{-1}\boldsymbol{r}_k^{\mathrm{PBiCG}} + \beta_{k-1}^{\mathrm{PBiCG}}\boldsymbol{p}_{k-1}^{\mathrm{PBiCG}},$$

$$\boldsymbol{p}_k^{\mathrm{PBiCG}\#} = M^{-\mathrm{T}}\boldsymbol{r}_k^{\mathrm{PBiCG}\#} + \beta_{k-1}^{\mathrm{PBiCG}}\boldsymbol{p}_{k-1}^{\mathrm{PBiCG}\#},$$

$$\alpha_k^{\mathrm{PBiCG}} = \frac{\left(\boldsymbol{r}_k^{\mathrm{PBiCG}\#}, M^{-1}\boldsymbol{r}_k^{\mathrm{PBiCG}}\right)}{\left(\boldsymbol{p}_k^{\mathrm{PBiCG}\#}, A\boldsymbol{p}_k^{\mathrm{PBiCG}}\right)}, \tag{46}$$

$$\boldsymbol{x}_{k+1}^{\mathrm{PBiCG}} = \boldsymbol{x}_k^{\mathrm{PBiCG}} + \alpha_k^{\mathrm{PBiCG}}\boldsymbol{p}_k^{\mathrm{PBiCG}},$$

$$\boldsymbol{r}_{k+1}^{\mathrm{PBiCG}} = \boldsymbol{r}_k^{\mathrm{PBiCG}} - \alpha_k^{\mathrm{PBiCG}}A\boldsymbol{p}_k^{\mathrm{PBiCG}}, \tag{47}$$

$$\boldsymbol{r}_{k+1}^{\mathrm{PBiCG}\#} = \boldsymbol{r}_k^{\mathrm{PBiCG}\#} - \alpha_k^{\mathrm{PBiCG}}A^{\mathrm{T}}\boldsymbol{p}_k^{\mathrm{PBiCG}\#}, \tag{48}$$

$$\beta_k^{\mathrm{PBiCG}} = \frac{\left(\boldsymbol{r}_{k+1}^{\mathrm{PBiCG}\#}, M^{-1}\boldsymbol{r}_{k+1}^{\mathrm{PBiCG}}\right)}{\left(\boldsymbol{r}_k^{\mathrm{PBiCG}\#}, M^{-1}\boldsymbol{r}_k^{\mathrm{PBiCG}}\right)}, \tag{49}$$

End Do

Algorithm 4 satisfies $\boldsymbol{r}_0^{\mathrm{PBiCG}} = \boldsymbol{r}_0^{\mathrm{P}}$ and $\boldsymbol{r}_0^{\mathrm{PBiCG}\#} = \boldsymbol{r}_0^{\mathrm{P}\#}$ when $k = 0$ in the iterative part.

Remark 2. *Because we apply a preconditioning conversion such as* $\tilde{\boldsymbol{x}}_k = M_R\boldsymbol{x}_k^{\mathrm{PBiCG}}$ *in the iteration of the BiCG method under the preconditioned system* (Algorithm 3), $\boldsymbol{x}_0^{\mathrm{PBiCG}} = \boldsymbol{x}_0$ *when* $k = 0$ *in the iteration of* Algorithm 4. *Further, the initial solution to* Algorithm 3 *is technically* $\tilde{\boldsymbol{x}}_0$, *but this is actually calculated by multiplying* M_R *by the initial solution* $\boldsymbol{x}_0$.

In this section, we have shown that α_k in BiCG is equivalent to α_k in CGS using (19), and that β_k in BiCG is equivalent to β_k in CGS using (20). In the next section, we propose an improved PCGS algorithm by applying this result to the preconditioned system.

3. Improved PCGS Algorithm

In this section, we first explain the derivation of PCGS, and present the conventional PCGS algorithm. We identify an issue with this conventional PCGS algorithm, and propose an improved PCGS that overcomes this issue.

3.1. Derivation of PCGS Algorithm

Figure 1 illustrates the logical structure of the solving methods and preconditioned algorithms discussed in this paper.

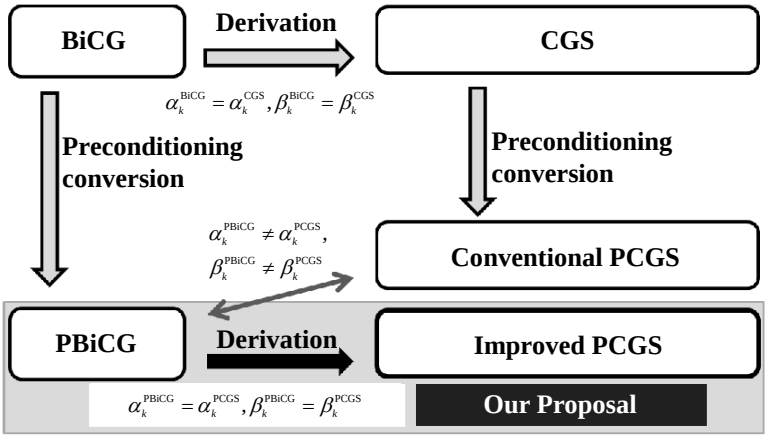


Figure 1. Relation between BiCG, CGS, and their preconditioned algorithms.

Typically, PCGS algorithms are derived via a "CGS method under a preconditioned system" (Algorithm 5).

Algorithm 5 is derived by applying the CGS method (Algorithm 2) to the preconditioned linear system (25). In this section, the vectors and α_k, β_k are PCGS elements, except for those before the iteration (Definition 2). If we wish to emphasize different methods, we apply a superscript to the relevant vectors, such as α_k^{PCGS}, β_k^{PCGS}, $\boldsymbol{p}_k^{\mathrm{PCGS}}$, $\tilde{\boldsymbol{p}}_k^{\mathrm{CGS}}$.

Algorithm 5. CGS method under the preconditioned system:

$\tilde{\boldsymbol{x}}_0$ is an initial guess, $\tilde{\boldsymbol{r}}_0 = \tilde{\boldsymbol{b}} - \tilde{A}\tilde{\boldsymbol{x}}_0$, set $\beta_{-1}^{\mathrm{PCGS}} = 0$,

$\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right) \neq 0$, e.g., $\tilde{\boldsymbol{r}}_0^{\#} = \tilde{\boldsymbol{r}}_0$,

For $k = 0,1,2,\cdots,$ until convergence, Do:

$$\tilde{\boldsymbol{u}}_k = \tilde{\boldsymbol{r}}_k + \beta_{k-1}^{\mathrm{PCGS}}\tilde{\boldsymbol{q}}_{k-1},$$

$$\tilde{\boldsymbol{p}}_k = \tilde{\boldsymbol{u}}_k + \beta_{k-1}^{\mathrm{PCGS}}\left(\tilde{\boldsymbol{q}}_{k-1} + \beta_{k-1}^{\mathrm{PCGS}}\tilde{\boldsymbol{p}}_{k-1}\right),$$

$$\alpha_k^{\mathrm{PCGS}} = \frac{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right)}{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{A}\tilde{\boldsymbol{p}}_k\right)},$$

$$\tilde{\boldsymbol{q}}_k = \tilde{\boldsymbol{u}}_k - \alpha_k^{\mathrm{PCGS}}\tilde{A}\tilde{\boldsymbol{p}}_k,$$

$$\tilde{\boldsymbol{x}}_{k+1} = \tilde{\boldsymbol{x}}_k + \alpha_k^{\mathrm{PCGS}}\left(\tilde{\boldsymbol{u}}_k + \tilde{\boldsymbol{q}}_k\right),$$

$$\tilde{\boldsymbol{r}}_{k+1} = \tilde{\boldsymbol{r}}_k - \alpha_k^{\mathrm{PCGS}}\tilde{A}\left(\tilde{\boldsymbol{u}}_k + \tilde{\boldsymbol{q}}_k\right),$$

$$\beta_k^{\mathrm{PCGS}} = \frac{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_{k+1}\right)}{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_k\right)},$$

End Do

The conventional PCGS algorithm (Algorithm 6) is derived via the CGS method, as shown in **Figure 1**, but this algorithm does not reflect the logic of subsection 2.2 in its preconditioning conversion. In contrast, our proposed improved PCGS algorithm (Algorithm 7) directly applies the derivation from BiCG to CGS to the PBiCG algorithm, thus maintaining the logic from subsection 2.2.

3.2. Conventional PCGS and Its Issue

The conventional PCGS algorithm is adopted in many documents and numerical libraries [4] [9] [11]. It is derived by applying the following preconditioning conversion to Algorithm 5:

$$\begin{aligned}&\tilde{A} = M_L^{-1}AM_R^{-1},\ \tilde{\boldsymbol{x}}_k = M_R\boldsymbol{x}_k,\ \tilde{\boldsymbol{b}} = M_L^{-1}\boldsymbol{b},\ \tilde{\boldsymbol{r}}_k = M_L^{-1}\boldsymbol{r}_k,\\ &\tilde{\boldsymbol{r}}_0^{\#} = M_L^{\mathrm{T}}\boldsymbol{r}_0^{\#},\ \tilde{\boldsymbol{p}}_k = M_L^{-1}\boldsymbol{p}_k,\ \tilde{\boldsymbol{u}}_k = M_L^{-1}\boldsymbol{u}_k,\ \tilde{\boldsymbol{q}}_k = M_L^{-1}\boldsymbol{q}_k.\end{aligned} \tag{50}$$

This gives the following Algorithm 6 ("Conventional PCGS" in **Figure 1**).

Algorithm 6. Conventional PCGS algorithm:

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,

$\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right) \neq 0$, e.g., $\boldsymbol{r}_0^{\#} = \boldsymbol{r}_0$, set $\beta_{-1} = 0$,

For $k = 0,1,2,\cdots,$ until convergence, Do:

$$\boldsymbol{u}_k = \boldsymbol{r}_k + \beta_{k-1}\boldsymbol{q}_{k-1},$$

$$\boldsymbol{p}_k = \boldsymbol{u}_k + \beta_{k-1}\left(\boldsymbol{q}_{k-1} + \beta_{k-1}\boldsymbol{p}_{k-1}\right),$$

$$\alpha_k = \frac{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_k\right)}{\left(\boldsymbol{r}_0^{\#}, AM^{-1}\boldsymbol{p}_k\right)}, \tag{51}$$

$$\begin{aligned}
&\boldsymbol{q}_k = \boldsymbol{u}_k - \alpha_k A M^{-1} \boldsymbol{p}_k, \\
&\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k M^{-1} \left(\boldsymbol{u}_k + \boldsymbol{q}_k \right), \\
&\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k A M^{-1} \left(\boldsymbol{u}_k + \boldsymbol{q}_k \right), \\
&\beta_k = \frac{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_{k+1} \right)}{\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_k \right)},
\end{aligned} \tag{52}$$

End Do

This PCGS algorithm was described in [4], which proposed the BiCGStab method, and has been employed as a standard approach as affairs stand now.

This version of PCGS seems to be a compliant algorithm on the surface, because the operation $\left(\boldsymbol{r}_0^{\#}, \boldsymbol{r}_k \right)$ in (51) and (52) does not include the preconditioning operator M^{-1} under the conversions $\tilde{\boldsymbol{r}}_k = M_L^{-1} \boldsymbol{r}_k$ and $\tilde{\boldsymbol{r}}_0^{\#} = M_L^{\mathrm{T}} \boldsymbol{r}_0^{\#}$ from (50). However, if we apply $\tilde{\boldsymbol{r}}_0^{\#} = M_L^{\mathrm{T}} \boldsymbol{r}_0^{\#}$ from (50) to the preconditioning conversion of the shadow residual vector in the BiCG method, we obtain

$$\tilde{\boldsymbol{r}}_k^{\#} = M_L^{\mathrm{T}} \boldsymbol{r}_k^{\#}. \tag{53}$$

This is different to the conversion given by (40), and we cannot obtain equivalent coefficients to $\alpha_k^{\mathrm{PBiCG}}$, β_k^{PBiCG} in (43) and (44) using (53).

3.3. Derivation of the CGS Method from PBiCG

In this subsection, we present an improved PCGS algorithm ("Improved PCGS" in **Figure 1**). We formulate this algorithm by applying the CGS derivation process to the BiCG method directly under the preconditioned system (PBiCG, Algorithm 3).

The polynomials (32)-(35) of the residual vectors and the probing direction vectors in PBiCG are substituted for the numerators and denominators of $\alpha_k^{\mathrm{PBiCG}}$ and β_k^{PBiCG}. We have

$$\left(\tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}\,\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}} \right) = \left(R_k\left(\tilde{A}^{\mathrm{T}} \right) \tilde{\boldsymbol{r}}_0^{\#}, R_k\left(\tilde{A} \right) \tilde{\boldsymbol{r}}_0 \right) = \left(\tilde{\boldsymbol{r}}_0^{\#}, R_k^2\left(\tilde{A} \right) \tilde{\boldsymbol{r}}_0 \right), \tag{54}$$

$$\left(\tilde{\boldsymbol{p}}_k^{\mathrm{BiCG}\,\#}, \tilde{A} \tilde{\boldsymbol{p}}_k^{\mathrm{BiCG}} \right) = \left(P_k\left(\tilde{A}^{\mathrm{T}} \right) \tilde{\boldsymbol{r}}_0^{\#}, \tilde{A} P_k\left(\tilde{A} \right) \tilde{\boldsymbol{r}}_0 \right) = \left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{A} P_k^2\left(\tilde{A} \right) \tilde{\boldsymbol{r}}_0 \right) \tag{55}$$

and apply the following Theorem 8.

Theorem 8. *The PCGS coefficients α_k and β_k are equivalent to these coefficients in PBiCG under certain transformation and substitution operations based on the bi-orthogonality and bi-conjugacy conditions under the preconditioned system.*

Proof. We apply

$$\tilde{\boldsymbol{r}}_k^{\mathrm{CGS}} \equiv R_k^2\left(\tilde{A} \right) \tilde{\boldsymbol{r}}_0, \quad \tilde{\boldsymbol{p}}_k^{\mathrm{CGS}} \equiv P_k^2\left(\tilde{A} \right) \tilde{\boldsymbol{r}}_0 \tag{56}$$

to (54) and (55). Then,

$$\alpha_k^{\mathrm{PBiCG}} = \frac{\left(\tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}\,\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}} \right)}{\left(\tilde{\boldsymbol{p}}_k^{\mathrm{BiCG}\,\#}, \tilde{A} \tilde{\boldsymbol{p}}_k^{\mathrm{BiCG}} \right)} = \frac{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{CGS}} \right)}{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{A} \tilde{\boldsymbol{p}}_k^{\mathrm{CGS}} \right)} \equiv \alpha_k^{\mathrm{PCGS}}, \tag{57}$$

$$\beta_k^{\mathrm{PBiCG}} = \frac{\left(\tilde{\boldsymbol{r}}_{k+1}^{\mathrm{BiCG}\,\#}, \tilde{\boldsymbol{r}}_{k+1}^{\mathrm{BiCG}} \right)}{\left(\tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}\,\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}} \right)} = \frac{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_{k+1}^{\mathrm{CGS}} \right)}{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{CGS}} \right)} \equiv \beta_k^{\mathrm{PCGS}}. \tag{58}$$

□

The PCGS method is derived from PBiCG using Theorem 9.

Theorem 9. *The PCGS method is derived from the linear system's recurrence relations in the PBiCG method under the property of equivalence between the coefficients α_k, β_k in PCGS and PBiCG. However, the solu-*

tion vector $\tilde{\boldsymbol{x}}_k^{\mathrm{CGS}}$ *is derived from a recurrence relation based on* $\tilde{\boldsymbol{r}}_k^{\mathrm{CGS}} = \tilde{\boldsymbol{b}} - \tilde{A}\tilde{\boldsymbol{x}}_k^{\mathrm{CGS}}$.

Proof. The coefficients α_k and β_k in PBiCG and PCGS were derived in Theorem 8. The recurrence relations in (29) and (30) for PBiCG are squared to give:

$$R_k^2(\tilde{\lambda}) = \left(R_{k-1}(\tilde{\lambda}) - \alpha_{k-1}\tilde{\lambda}P_{k-1}(\tilde{\lambda})\right)^2 = R_{k-1}^2(\tilde{\lambda}) - 2\alpha_{k-1}\tilde{\lambda}P_{k-1}(\tilde{\lambda})R_{k-1}(\tilde{\lambda}) + \alpha_{k-1}^2\tilde{\lambda}^2P_{k-1}^2(\tilde{\lambda}), \tag{59}$$

$$P_k^2(\tilde{\lambda}) = \left(R_k(\tilde{\lambda}) + \beta_{k-1}P_{k-1}(\tilde{\lambda})\right)^2 = R_k^2(\tilde{\lambda}) + 2\beta_{k-1}P_{k-1}(\tilde{\lambda})R_k(\tilde{\lambda}) + \beta_{k-1}^2P_{k-1}^2(\tilde{\lambda}). \tag{60}$$

Further, we can apply $\tilde{\boldsymbol{r}}_k^{\mathrm{CGS}} \equiv R_k^2(\tilde{A})\tilde{\boldsymbol{r}}_0$, $\tilde{\boldsymbol{p}}_k^{\mathrm{CGS}} \equiv P_k^2(\tilde{A})\tilde{\boldsymbol{r}}_0$ from (56), and substitute $\tilde{\boldsymbol{u}}_k^{\mathrm{CGS}} \equiv P_k(\tilde{A})R_k(\tilde{A})\tilde{\boldsymbol{r}}_0$, $\tilde{\boldsymbol{q}}_k^{\mathrm{CGS}} \equiv P_k(\tilde{A})R_{k+1}(\tilde{A})\tilde{\boldsymbol{r}}_0$. □

The following Proposition 10 and Corollary 2 are given as a supplementary explanation under the preconditioned system.

Proposition 10. *There exist the following relations*:

$$\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}} = \tilde{\boldsymbol{r}}_0, \tag{61}$$

$$\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}\#} = \tilde{\boldsymbol{r}}_0^{\#}, \tag{62}$$

where $\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}}$ *is the initial residual vector at* $k = 0$ *in the iterative part of PCGS,* $\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}\#}$ *is the initial shadow residual vector in PCGS,* $\tilde{\boldsymbol{r}}_0$ *is the initial residual vector, and* $\tilde{\boldsymbol{r}}_0^{\#}$ *is the initial shadow residual vector.*

Proof. Equation (61) follows because (56) for $k = 0$ gives $\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}} = R_0^2(\tilde{A})\tilde{\boldsymbol{r}}_0 = \tilde{\boldsymbol{r}}_0$. Here, $R_0(\tilde{A}) = I$ by (28).

Equation (62) is derived as follows. Applying (56) to (54), we obtain

$$\left(\tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{BiCG}}\right) = \left(R_k(\tilde{A}^{\mathrm{T}})\tilde{\boldsymbol{r}}_0^{\#}, R_k(\tilde{A})\tilde{\boldsymbol{r}}_0\right) = \left(\tilde{\boldsymbol{r}}_0^{\#}, R_k^2(\tilde{A})\tilde{\boldsymbol{r}}_0\right) \equiv \left(\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{CGS}}\right).$$

This equation shows that the inner product of the PCGS on the right is obtained from the inner product of the PBiCG on the left. Therefore, $\tilde{\boldsymbol{r}}_0^{\#}$ that composes the polynomial $R_k(\tilde{A}^{\mathrm{T}})\tilde{\boldsymbol{r}}_0^{\#}$ to express the shadow residual vector of the PBiCG is the same as $\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}\#}$ in PCGS.

Hereafter, $\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}\#}$ can be represented by $\tilde{\boldsymbol{r}}_0^{\#}$, to the extent that neither can be distinguished. □

Corollary 2. *There exists the following relation*:

$$\tilde{\boldsymbol{p}}_0^{\mathrm{CGS}} = \tilde{\boldsymbol{r}}_0^{\mathrm{CGS}} = \tilde{\boldsymbol{r}}_0,$$

where $\tilde{\boldsymbol{p}}_0^{\mathrm{CGS}}$ *is the probing direction vector in PCGS,* $\tilde{\boldsymbol{r}}_0^{\mathrm{CGS}}$ *is the initial residual vector at* $k = 0$ *in the iterative part of PCGS, and* $\tilde{\boldsymbol{r}}_0$ *is the initial residual vector.*

The CGS preconditioning conversion given by $\tilde{\boldsymbol{r}}_k^{\mathrm{CGS}}$, $\tilde{\boldsymbol{p}}_k^{\mathrm{CGS}}$, and $\tilde{\boldsymbol{r}}_0^{\#}$ is subjected to the same treatment as the BiCG preconditioning conversion of $\tilde{\boldsymbol{r}}_k$, $\tilde{\boldsymbol{p}}_k$, in (40) and (45). Further, $\boldsymbol{r}_0^{\mathrm{P}\#}$ is the same as $\boldsymbol{r}_0^{\#}$. Then,

$$\alpha_k^{\mathrm{PCGS}} = \frac{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_k^{\mathrm{CGS}}\right)}{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{A}\tilde{\boldsymbol{p}}_k^{\mathrm{CGS}}\right)} = \frac{\left(M_R^{-\mathrm{T}}\boldsymbol{r}_0^{\#}, M_L^{-1}\boldsymbol{r}_k^{\mathrm{PCGS}}\right)}{\left(M_R^{-\mathrm{T}}\boldsymbol{r}_0^{\#}, \left(M_L^{-1}AM_R^{-1}\right)\left(M_R\boldsymbol{p}_k^{\mathrm{PCGS}}\right)\right)} = \frac{\left(\boldsymbol{r}_0^{\#}, M^{-1}\boldsymbol{r}_k^{\mathrm{PCGS}}\right)}{\left(\boldsymbol{r}_0^{\#}, M^{-1}A\boldsymbol{p}_k^{\mathrm{PCGS}}\right)}, \tag{63}$$

$$\beta_k^{PCGS} = \frac{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_{k+1}^{CGS}\right)}{\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_k^{CGS}\right)} = \frac{\left(M_R^{-\mathrm{T}}\boldsymbol{r}_0^{\#}, M_L^{-1}\boldsymbol{r}_{k+1}^{\mathrm{PCGS}}\right)}{\left(M_R^{-\mathrm{T}}\boldsymbol{r}_0^{\#}, M_L^{-1}\boldsymbol{r}_k^{\mathrm{PCGS}}\right)} = \frac{\left(\boldsymbol{r}_0^{\#}, M^{-1}\boldsymbol{r}_{k+1}^{\mathrm{PCGS}}\right)}{\left(\boldsymbol{r}_0^{\#}, M^{-1}\boldsymbol{r}_k^{\mathrm{PCGS}}\right)}. \tag{64}$$

As a consequence, the following improved PCGS algorithm is derived[7].

Algorithm 7. Improved preconditioned CGS algorithm:

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
$\left(\tilde{\boldsymbol{r}}_0^{\#}, \tilde{\boldsymbol{r}}_0\right) \neq 0$, e.g., $\boldsymbol{r}_0^{\#} = M^{-1}\boldsymbol{r}_0$, set $\beta_{-1} = 0$,
For $k = 0, 1, 2, \cdots$, until convergence, Do:

$$\boldsymbol{u}_k = M^{-1}\boldsymbol{r}_k + \beta_{k-1}\boldsymbol{q}_{k-1},$$

[7]We apply the superscript "PCGS" to α_k, β_k, and relevant vectors to denote this algorithm.

$$\boldsymbol{p}_k = \boldsymbol{u}_k + \beta_{k-1}\left(\boldsymbol{q}_{k-1} + \beta_{k-1}\boldsymbol{p}_{k-1}\right),$$

$$\alpha_k = \frac{\left(\boldsymbol{r}_0^{\#}, M^{-1}\boldsymbol{r}_k\right)}{\left(\boldsymbol{r}_0^{\#}, M^{-1}A\boldsymbol{p}_k\right)},$$

$$\boldsymbol{q}_k = \boldsymbol{u}_k - \alpha_k M^{-1}A\boldsymbol{p}_k,$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\left(\boldsymbol{u}_k + \boldsymbol{q}_k\right),$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k A\left(\boldsymbol{u}_k + \boldsymbol{q}_k\right),$$

$$\beta_k = \frac{\left(\boldsymbol{r}_0^{\#}, M^{-1}\boldsymbol{r}_{k+1}\right)}{\left(\boldsymbol{r}_0^{\#}, M^{-1}\boldsymbol{r}_k\right)},$$

End Do

Algorithm 7 can also be derived by applying the following preconditioning conversion to Algorithm 5. Here, we treat the preconditioning conversions of $\tilde{\boldsymbol{u}}_k$ and $\tilde{\boldsymbol{q}}_k$ the same as the conversion of $\tilde{\boldsymbol{p}}_k$.

$$\tilde{A} = M_L^{-1}AM_R^{-1},\ \ \tilde{\boldsymbol{x}}_k = M_R\boldsymbol{x}_k,\ \ \tilde{\boldsymbol{b}} = M_L^{-1}\boldsymbol{b},\ \ \tilde{\boldsymbol{r}}_k = M_L^{-1}\boldsymbol{r}_k,$$

$$\tilde{\boldsymbol{r}}_0^{\#} = M_R^{-\mathrm{T}}\boldsymbol{r}_0^{\#},\ \ \tilde{\boldsymbol{p}}_k = M_R\boldsymbol{p}_k,\ \ \tilde{\boldsymbol{u}}_k = M_R\boldsymbol{u}_k,\ \ \tilde{\boldsymbol{q}}_k = M_R\boldsymbol{q}_k.$$

The number of preconditioning operations in the iterative part of Algorithm 7 is the same as that in Algorithm 6.

4. Numerical Experiments

In this section, we compare the conventional and improved PCGS algorithms numerically.

The test problems were generated by building real unsymmetric matrices corresponding to linear systems taken from the Tim Davis collection [12] and the Matrix Market [13]. The RHS vector $\boldsymbol{b}$ of (1) was generated by setting all elements of the exact solution vector $\boldsymbol{x}_{\text{exact}}$ to 1.0, and substituting this into (1). The solution algorithm was implemented using the sequential mode of the Lis numerical computation library (version 1.1.2 [14]) in double-precision, with the compiler options registered in the Lis "Makefile". Furthermore, we set the initial solution to $\boldsymbol{x}_0 = \boldsymbol{0}$, and considered the algorithm to have converged when $\|\boldsymbol{r}_k\|_2 / \|\boldsymbol{b}\|_2 \le 1.0\times10^{-12}$ (where $\boldsymbol{r}_k$ is the residual vector in the algorithm, and k is the iteration number). The maximum number of iterations was set to the size of the coefficient matrix.

The numerical experiments were executed on a DELL Precision T7400 (Intel Xeon E5420, 2.5 GHz CPU, 16 GB RAM) running the Cent OS (kernel 2.6.18) and the Intel icc 10.1 compiler.

The results using the non-preconditioned CGS are listed in **Table 1**.

The results given by the conventional PCGS and the improved PCGS are listed in **Tables 2-5**. Each table adopts a different preconditioner in Lis [14]: "(Point-)Jacobi", "ILU(0)"[8], "SAINV", and "Crout ILU". In these tables, significant advantages of one algorithm over the other are emphasized by **bold font**[9]. Additionally, matrix names given in italic font in **Table 1** encounter some difficulties. The evolution of the convergence for each preconditioner is shown in **Figures 2-5**. In this paper, we do not compare the computational speed of these preconditioners.

In many cases, the results given by the improved PCGS are better than those from the conventional algorithm. We should pay particular attention to the results from matrices "mcca", "mcfe" and "watt_1". In these cases, it appears that the conventional PCGS converges faster with any preconditioner, but the TRE values are worse than those from the improved algorithm. The iteration number for the conventional PCGS is not emphasized by bold font in these instances. The consequences of this anomaly are worth investigating further, possibly by analyzing them under PBiCG. This will be the subject of future work.

[8]Values of "zero" for ILU(0) indicates the fill-in level, that is, "no fill-in".

[9]Because the principal argument presented in this paper concerns the structure of the algorithm for PCGS based on an improved preconditioning transformation procedure, the time required to obtain numerical solutions and the convergence history graphs (**Figures 2-5**) are provided as a reference. As mentioned before, there is almost no difference between Algorithm 6 and Algorithm 7 as regards the number of operations in the iteration, which would exert a strong influence on the computation time.

Table 1. Numerical results for a veriety of test problems (CGS).

Matrix	N	NNZ	CGS (Algorithm 2)			
			Iter.	TRR	TRE	Time
*arc*130	130	1037	11	−12.20	*−7.05*	1.18e−4
bfwa782	782	7514	320	−11.29	−11.94	1.71e−2
*cryg*2500	2500	12349	*No convergence*			
*epb*1	14734	95053	770	*−7.45*	*−6.50*	5.93e−1
*jpwh_*991	991	6027	*Breakdown*			
mcca	180	2659	*No convergence*			
mcfe	765	24382	*No convergence*			
memplus	17758	99147	1334	*−9.16*	*−6.76*	1.33e+0
*olm*1000	1000	3996	*No convergence*			
*olm*5000	5000	19996	*No convergence*			
pde900	900	4380	113	−9.87	−10.49	4.13e−3
pde2961	2961	14585	256	−9.49	−10.19	3.06e−2
*sherman*2	1080	23094	*No convergence*			
*sherman*3	5005	20033	*No convergence*			
sherman5	3312	20793	1927	−10.36	−9.69	3.34e−1
*viscoplastic*2	32769	381326	801	−10.34	*−8.18*	2.20e+0
*watt_*1	1856	11360	306	−12.10	*−6.07*	2.72e−2

In this table, "N" is the problem size and "NNZ" is the number of nonzero elements. The items in each column are, from left to right, the number of iterations required to converge (denoted "Iter."), the true relative residual $\log_{10}$ 2-norm (denoted by "TRR", calculated as $\|\boldsymbol{b}-A\hat{\boldsymbol{x}}_k\|_2/\|\boldsymbol{b}\|_2$, where $\hat{\boldsymbol{x}}_k$ is the numerical solution), the true relative error $\log_{10}$ 2-norm (denoted by "TRE", calculated from the numerical solution and the exact solution, that is, $\|\hat{\boldsymbol{x}}_k-\boldsymbol{x}_{exact}\|_2/\|\boldsymbol{x}_{exact}\|_2$), and, as a reference, the CPU time (denoted "Time" [s]). Several matrix names are represented in *italic font*. These describe certain situations, such as "Breakdown", "No convergence", and insufficient accuracy on "TRR" or "TRE".

5. Conclusions

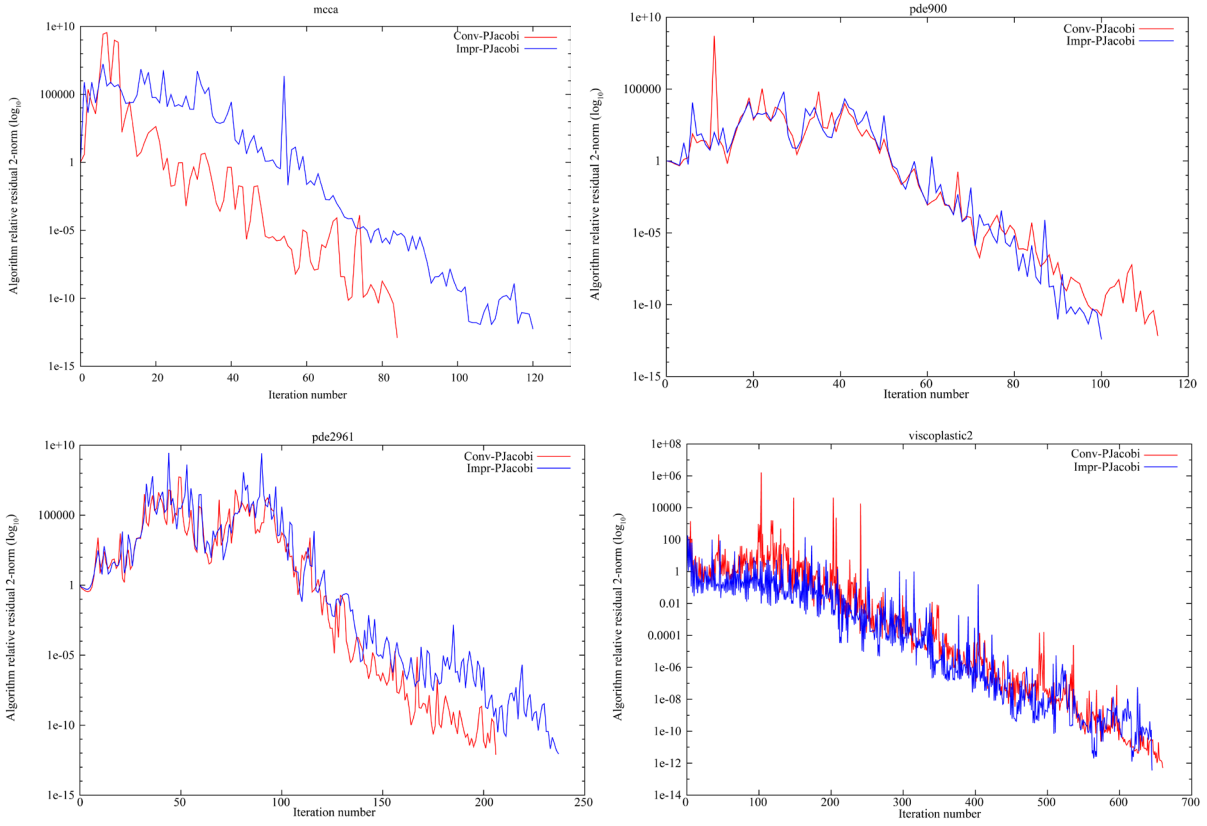
In this paper, we have developed an improved PCGS algorithm by applying the procedure for deriving CGS to the BiCG method under a preconditioned system, and we also have presented some mathematical theorems underlying the deriving process's logic. The improved PCGS does not increase the number of preconditioning operations in the iterative part of the algorithm. Our numerical results established that solutions obtained with the proposed algorithm are superior to those from the conventional algorithm for a variety of preconditioners.

However, the improved algorithm may still break down during the iterative procedure. This is an artefact of certain characteristics of the non-preconditioned BiCG and CGS methods, mainly the operations based on the bi-orthogonality and bi-conjugacy conditions. Nevertheless, this improved logic can be applied to other bi-Lanczos-based algorithms with minimal residual operations.

In future work, we will analyze the mechanism of the conventional and improved PCGS algorithms, and consider other variations of this algorithm. Furthermore, we will consider other settings of the initial shadow residual vector $\boldsymbol{r}_0^{\#}$, except for $\tilde{\boldsymbol{r}}_0^{\#}=\tilde{\boldsymbol{r}}_0$ to ensure that $\left(\tilde{\boldsymbol{r}}_0^{\#},\tilde{\boldsymbol{r}}_0\right)\neq 0$ holds.

Table 2. Numerical results for a veriety of test problems (Jacobi-CGS).

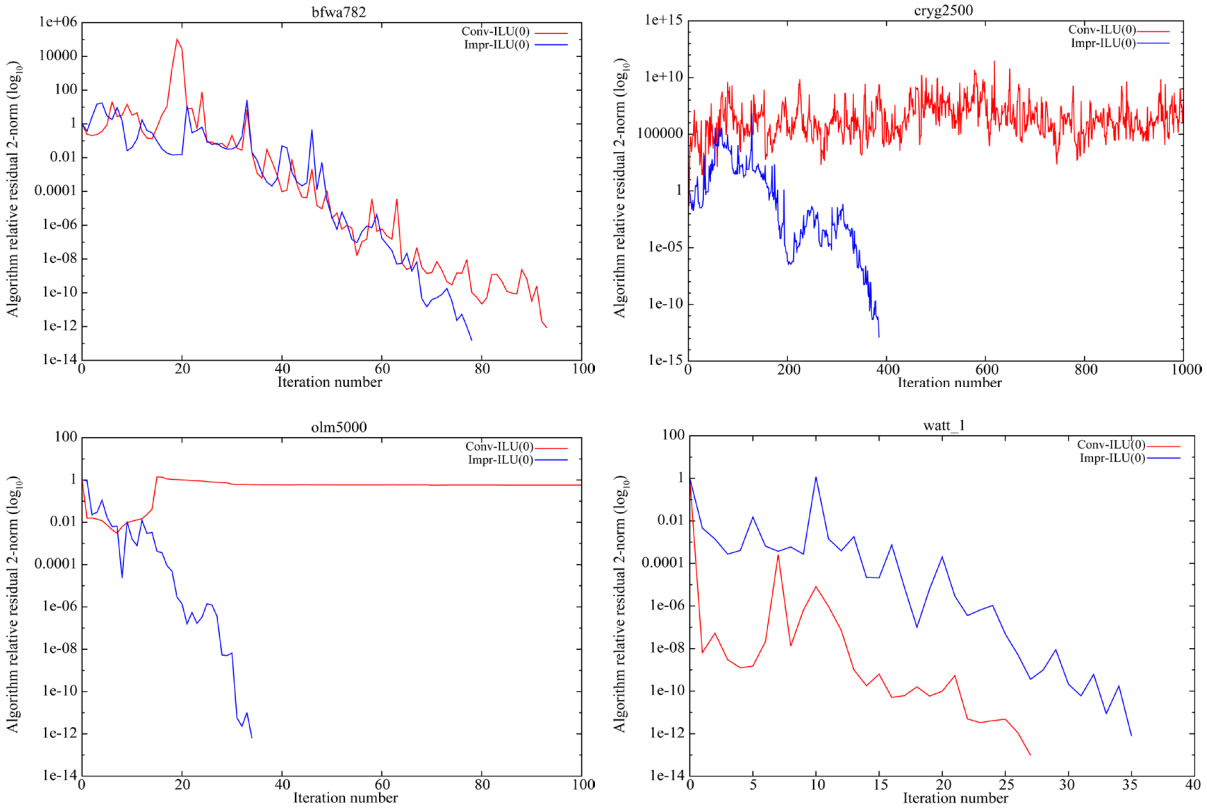
Matrix	Conv PCGS (Algorithm 6)				Impr PCGS (Algorithm 7)			
	Iter.	TRR	TRE	Time	Iter.	TRR	TRE	Time
*arc*130	5	−15.91	−10.61	6.88e−5	5	−15.87	−10.66	7.28e−5
bfwa782	**227**	−11.50	−12.30	1.25e−2	260	−11.98	−12.02	1.41e−2
*cryg*2500	No convergence				No convergence			
*epb*1	**578**	−7.66	−6.44	4.56e−1	591	−7.59	−6.86	4.68e−1
*jpwh*_991	Breakdown				Breakdown			
mcca	84	−6.53	−0.93	1.44e−3	120	**−8.93**	**−12.61**	2.06e−3
mcfe	764	−3.56	2.97	7.95e−2	908	**−6.26**	**−8.79**	9.54e−2
memplus	**213**	−12.10	−9.06	2.19e−1	230	−12.41	−9.38	2.37e−1
*olm*1000	No convergence				No convergence			
*olm*5000	No convergence				No convergence			
pde900	113	−7.44	−7.63	4.30e−3	**100**	**−11.22**	**−11.75**	3.81e−3
pde2961	**206**	**−8.27**	**−8.68**	2.55e−2	237	−6.44	−6.61	2.92e−2
*sherman*2	No convergence				No convergence			
*sherman*3	1012	−7.30	−8.40	2.15e−1	827	−6.74	−7.59	1.75e−1
sherman5	131	−12.29	−12.63	2.35e−2	128	−12.40	−12.03	2.28e−2
*viscoplastic*2	660	−9.99	−8.00	1.97e+0	**645**	**−12.45**	**−10.14**	1.88e+0
*watt*_1	80	−12.51	−5.75	7.53e−3	79	−12.61	−5.44	7.53e−3



Figue 2. Converging histories of relative residual 2-norm (Jacobi-CGS). Red line: Conventional PCGS, Blue line: Improved PCGS.

Table 3. Numerical results for a veriety of test problems (ILU(0)-CGS).

Matrix	Conv PCGS (Algorithm 6)				Impr PCGS (Algorithm 7)			
	Iter.	TRR	TRE	Time	Iter.	TRR	TRE	Time
*arc*130	2	−15.90	−6.35	2.94e−4	3	−16.26	**−11.07**	2.98e−4
bfwa782	93	−9.36	−10.29	1.18e−2	**78**	**−12.82**	−12.48	1.01e−2
*cryg*2500	No convergence				**385**	**−8.47**	−4.22	8.49e−2
*epb*1	**124**	**−11.38**	**−10.34**	2.14e−1	129	−9.23	−8.54	2.29e−1
*jpwh*_991	Breakdown				**16**	**−12.44**	**−12.53**	2.72e−3
mcca	7	−10.53	−11.10	5.99e−4	7	−9.98	−11.70	6.18e−4
mcfe	10	−12.83	−11.58	5.70e−3	9	−12.15	−10.61	5.52e−3
memplus	303	−12.13	−10.36	7.22e−1	305	−12.12	−10.61	7.18e−1
*olm*1000	No convergence				**34**	**−12.49**	**−9.19**	2.85e−3
*olm*5000	No convergence				**34**	**−12.20**	**−8.05**	1.41e−2
pde900	27	−13.61	−14.27	2.57e−3	27	−13.19	−13.92	2.59e−3
pde2961	53	−10.57	−11.34	1.49e−2	58	−11.78	−12.65	1.62e−2
*sherman*2	12	−13.60	−11.46	6.08e−3	11	−14.20	−11.55	5.91e−3
*sherman*3	103	−9.82	−11.57	4.39e−2	96	−10.82	−13.34	4.10e−2
sherman5	31	−13.68	−12.89	1.36e−2	30	−12.54	−12.42	1.31e−2
*viscoplastic*2	812	−7.55	−4.68	7.01e+0	844	**−11.80**	**−8.69**	7.18e+0
*watt*_1	27	−13.01	−5.96	6.17e−3	35	−12.11	**−9.77**	7.75e−3



Figue 3. Converging histories of relative residual 2-norm (ILU(0)-CGS). Red line: Conventional PCGS, Blue line: Improved PCGS.

Table 4. Numerical results for a veriety of test problems (SAINV-CGS).

Matrix	Conv PCGS (Algorithm 6)				Impr PCGS (Algorithm 7)			
	Iter.	TRR	TRE	Time	Iter.	TRR	TRE	Time
*arc*130	4	−17.54	−8.75	3.22e−4	5	−19.27	**−11.20**	3.24e−4
bfwa782	109	−9.49	−9.75	1.72e−2	**106**	**−12.34**	**−12.07**	1.73e−2
*cryg*2500	No convergence				No convergence			
*epb*1	69	−12.40	−11.91	2.90e+0	69	−12.31	−12.41	2.89e+0
*jpwh*_991	Breakdown				**41**	**−12.20**	**−13.06**	7.37e−3
mcca	81	−5.32	0.00	2.26e−3	111	**−8.60**	**−14.26**	3.04e−3
mcfe	764	−3.56	2.97	1.10e−1	908	**−6.26**	**−8.79**	1.29e−1
memplus	34	−12.30	−10.20	1.18e+0	34	−12.32	−10.24	1.20e+0
*olm*1000	No convergence				No convergence			
*olm*5000	No convergence				No convergence			
pde900	53	−10.37	−11.16	7.25e−3	51	−12.43	−13.03	7.28e−3
pde2961	110	−11.62	−12.02	6.91e−2	**96**	−12.08	−12.55	6.66e−2
*sherman*2	No convergence				No convergence			
*sherman*3	874	−7.49	−8.34	5.53e−1	**629**	**−9.49**	**−10.75**	4.24e−1
sherman5	157	−12.02	−11.19	9.20e−2	**127**	−12.57	−12.30	8.09e−2
*viscoplastic*2	No convergence				No convergence			
*watt*_1	1	−12.37	−4.05	1.49e+0	2	−14.81	**−11.31**	1.53e+0

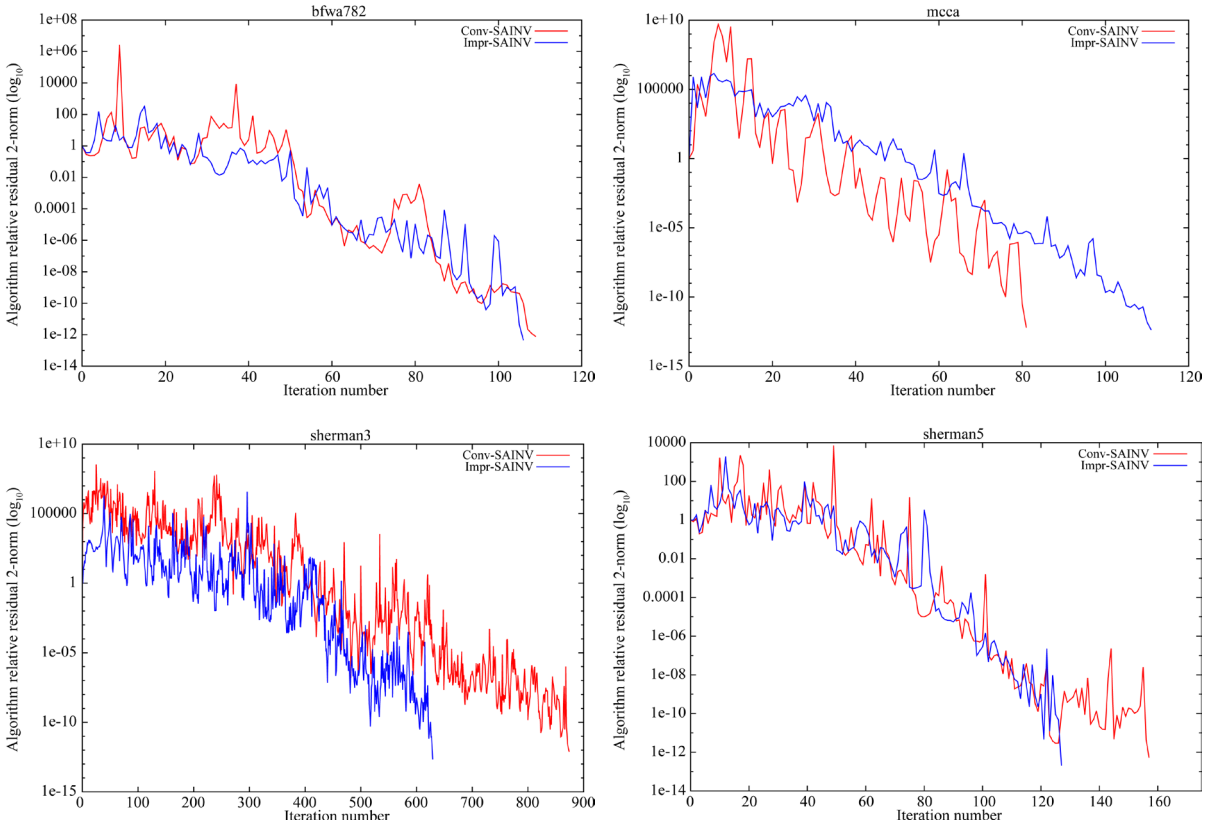


Figure 4. Converging histories of relative residual 2-norm (SAINV-CGS). Red line: Conventional PCGS, Blue line: Improved PCGS.

Table 5. Numerical results for a veriety of test problems (CroutILU-CGS).

Matrix	Conv PCGS (Algorithm 6)				Impr PCGS (Algorithm 7)			
	Iter.	TRR	TRE	Time	Iter.	TRR	TRE	Time
*arc*130	2	−12.42	−2.34	1.63e−4	4	−16.26	**−11.12**	1.82e−4
bfwa782	**122**	−12.09	−12.05	2.03e−2	149	−12.56	−13.15	2.42e−2
*cryg*2500	No convergence				**902**	−7.60	−2.67	2.09e−1
*epb*1	109	−11.91	−11.69	2.12e−1	106	−12.10	−11.79	2.06e−1
*jpwh*_991	Breakdown				**15**	**−12.53**	**−13.29**	3.46e−3
mcca	10	−9.75	−10.42	5.90e−4	12	−9.10	−15.41	6.40e−4
mcfe	23	−11.71	−10.36	6.24e−3	26	−11.73	−13.99	6.82e−3
memplus	92	−12.60	−10.00	1.59e−1	88	−12.52	−10.57	1.53e−1
*olm*1000	No convergence				**38**	**−12.24**	**−8.04**	3.37e−3
*olm*5000	67	−9.70	−7.93	2.75e−2	75	−11.64	**−9.12**	3.06e−2
pde900	13	−12.56	−12.66	1.96e−3	12	−12.42	−12.43	1.88e−3
pde2961	24	−13.13	−13.16	9.18e−3	25	−12.48	−12.77	9.50e−3
*sherman*2	901	−9.72	−6.24	2.19e−1	**841**	−10.61	**−8.71**	2.05e−1
*sherman*3	79	−10.65	−12.01	3.94e−2	74	−11.68	−13.90	3.73e−2
sherman5	40	−13.04	−11.95	1.53e−2	40	−13.30	−12.93	1.53e−2
*viscoplastic*2	No convergence				No convergence			
*watt*_1	19	−13.05	−5.75	6.63e−3	25	−15.25	**−9.43**	8.12e−3

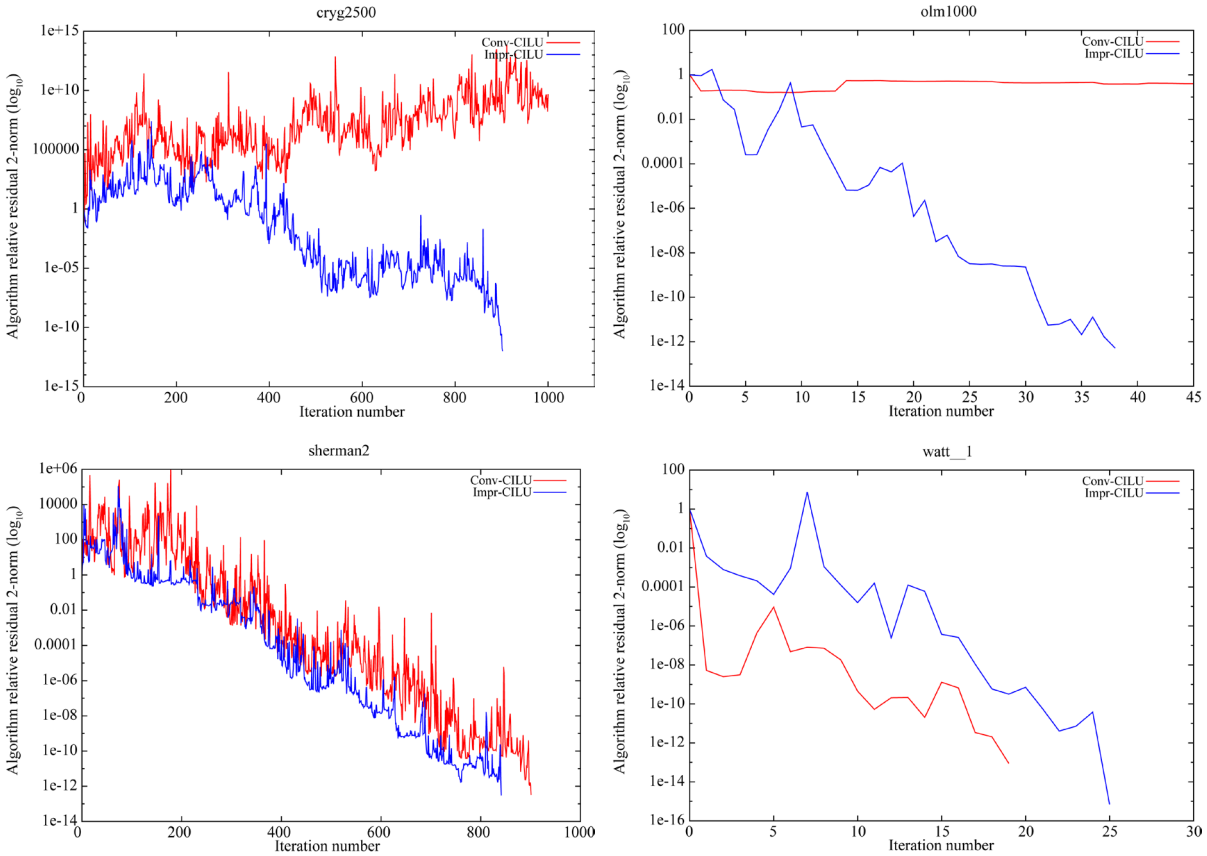


Figure 5. Converging histories of relative residual 2-norm (CroutILU-CGS). Red line: Conventional PCGS, Blue line: Improved PCGS.

Acknowledgements

This work is partially supported by a Grant-in-Aid for Scientific Research (C) No. 25390145 from MEXT, Japan.

References

[1] Sonneveld, P. (1989) CGS, A Fast Lanczos-Type Solver for Nonsymmetric Linear Systems. *SIAM Journal on Scientific and Statistical Computing*, **10**, 36-52. http://dx.doi.org/10.1137/0910004

[2] Fletcher, R. (1976) Conjugate Gradient Methods for Indefinite Systems. In: Watson, G., Ed., *Numerical Analysis Dundee* 1975, Lecture Notes in Mathematics, Vol. 506, Springer-Verlag, Berlin, New York, 73-89.

[3] Lanczos, C. (1952) Solution of Systems of Linear Equations by Minimized Iterations. *Journal of Research of the National Bureau of Standards*, **49**, 33-53. http://dx.doi.org/10.6028/jres.049.006

[4] Van der Vorst, H.A. (1992) Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems. *SIAM Journal on Scientific and Statistical Computing*, **13**, 631-644. http://dx.doi.org/10.1137/0913035

[5] Zhang, S.-L. (1997) GPBi-CG: Generalized Product-Type Methods Based on Bi-CG for Solving Nonsymmetric Linear Systems. *SIAM Journal on Scientific Computing*, **18**, 537-551. http://dx.doi.org/10.1137/s1064827592236313

[6] Itoh, S. and Sugihara, M. (2010) Systematic Performance Evaluation of Linear Solvers Using Quality Control Techniques. In: Naono, K., Teranishi, K., Cavazos, J. and Suda, R., Eds., *Software Automatic Tuning*: *From Concepts to State-of-the-Art Results*, Springer, 135-152.

[7] Itoh, S. and Sugihara, M. (2013) Preconditioned Algorithm of the CGS Method Focusing on Its Deriving Process. *Transactions of the Japan SIAM*, **23**, 253-286. (in Japanese)

[8] Hestenes, M.R. and Stiefel, E. (1952) Methods of Conjugate Gradients for Solving Linear Systems. *Journal of Research of the National Bureau of Standards*, **49**, 409-435. http://dx.doi.org/10.6028/jres.049.044

[9] Barrett, R., Berry, M., Chan, T.F., Demmel, J., Donato, J., Dongarra, J., *et al*. (1994) Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. *SIAM*. http://dx.doi.org/10.1137/1.9781611971538

[10] Van der Vorst, H.A. (2003) Iterative Krylov Methods for Large Linear Systems. Cambridge University Press, Cambridge. http://dx.doi.org/10.1017/CBO9780511615115

[11] Meurant, G. (2005) Computer Solution of Large Linear Systems. Elsevier, Amsterdam.

[12] Davis, T.A. The University of Florida Sparse Matrix Collection. http://www.cise.ufl.edu/research/sparse/matrices/

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

[14] SSI Project, Lis. http://www.ssisc.org/lis/