SIAM J. SCI. COMPUT. © 1997 Society for Industrial and Applied Mathematics
Vol. 18, No. 2, pp. 537–551, March 1997 011

GPBi-CG: GENERALIZED PRODUCT-TYPE METHODS BASED ON Bi-CG FOR SOLVING NONSYMMETRIC LINEAR SYSTEMS[*]

SHAO-LIANG ZHANG[†]

Abstract. Recently Bi-CGSTAB as a variant of Bi-CG has been proposed for solving nonsymmetric linear systems, and its attractive convergence behavior has been confirmed in many numerical experiments. Bi-CGSTAB can be characterized by its residual polynomial which consists of the product of the residual polynomial of Bi-CG with other polynomials generated from two-term recurrence relations. In this paper, we propose a unified way to generalize a class of product-type methods whose residual polynomials can be factored by the residual polynomial of Bi-CG and other polynomials with standard three-term recurrence relations. Such product-type methods which are based on Bi-CG can be regarded as generalizations of Bi-CGSTAB. From the unified way, the well-known variants of the product-type methods, like CGS, Bi-CGSTAB, Bi-CGSTAB2, are reacquired again.

Key words. Bi-CG, Bi-CGSTAB, Bi-CGSTAB2, CGS, nonsymmetric linear systems, product-type methods, restructuring Bi-CG, residual polynomial, three-term recurrence relations

AMS subject classification. 65F10

PII. S1064827592236313

1. Introduction.

For solving nonsymmetric linear systems, the biconjugate gradient (Bi-CG hereafter) [8, 4] was presented as a generalization of the conjugate gradient (CG hereafter) [7] for solving symmetric positive definite systems. However, operations of transpose matrix–vector multiplications are needed in Bi-CG because a Krylov subspace generated from the transpose matrix is used. In order to avoid calculating the transpose matrix–vector multiplications and improve the convergence rate in Bi-CG, recently many efforts have been devoted to deriving more efficient methods from restructuring Bi-CG. A common technique to design a new method by means of restructuring Bi-CG is to define its residual polynomial by a product of two polynomial factors where one factor is the residual polynomial of Bi-CG and the other one is an undetermined polynomial. For example, the conjugate gradient squared (CGS hereafter) [14] and Bi-CGSTAB [17] were derived from Bi-CG by this technique. In CGS, P. Sonneveld defined the undetermined polynomial by the same residual polynomial of Bi-CG, i.e., defined the residual polynomial of CGS by the square of that of Bi-CG. CGS was recognized as a powerful variant of Bi-CG in many numerical experiments [14, 11]. However, it often was observed that CGS has rather irregular and oscillatory convergence behaviors in many situations because of the presence of the round-off errors [17, 18, 6]. Therefore, in Bi-CGSTAB, H. A. van der Vorst selected a polynomial with two-term recurrence relations instead of one factor of CGS to design the residual polynomial of Bi-CGSTAB. Since undetermined parameters with respect to the residual polynomial of Bi-CGSTAB are chosen at least to minimize the residual 2-norm per iteration, Bi-CGSTAB is rather stable and more efficient. In fact, many numerical experiments also indicated that Bi-CGSTAB can often run faster and its convergence behavior is more smooth than CGS [17, 2]. Besides, M. H. Gutknecht replaced the two-term recurrent polynomial in Bi-CGSTAB with a three-term recurrent polyno-

[*]Received by the editors August 19, 1995; accepted for publication (in revised form) August 1, 1995. This research was supported in part by the Hori Information Science Promotion Foundation.
http://www.siam.org/journals/sisc/18-2/23631.html

[†]Institute of Information Sciences and Electronics, University of Tsukuba, Tennodai 1-1-1, Tsukuba, Ibaraki, 305, Japan (zhang@is.tsukuba.ac.jp).

mial only at even iterations to solve linear systems with complex spectrum, and implemented the algorithm of Bi-CGSTAB2 [5]. Bi-CGSTAB2 is explained as a generalization of Bi-CGSTAB, say, the combination Bi-CG with GMRES(2) [12], in the sense that Bi-CGSTAB is the combination of Bi-CG with GMRES(1). In [13], G. L. G. Sleijpen and D. R. Fokkema generated Bi-CGSTAB in this approach by combining GMRES(L) with Bi-CG, and implemented a new algorithm, Bi-CGSTAB(L) for modest L. In Bi-CGSTAB(L), however, computational work and storage costs are more expensive for larger L, and it is very hard to choose a better L for solving a realistic problem.

The purpose of this paper is to propose another unified way to derive a class of generalizations of Bi-CGSTAB with little computational work and low storage costs. By defining a new three-term recurrence relation modelled after the residual polynomial of Bi-CG, we will present a diverse collection of generalized product-type methods based on Bi-CG that includes the well-known CGS, Bi-CGSTAB, and Bi-CGSTAB2, without the disadvantage of storing extra iterates like that in GMRES [12] and GCR [3].

This paper is organized as follows: in the next section, we introduce the background concerned with restructuring Bi-CG, and characterize the product-type methods based on Bi-CG. In section 3, we propose a standard three-term recurrent polynomial to define the residual polynomial of the product-type methods, and derive the recurrence formulas among the related iterates. In section 4, an equivalent approach to reduce the recurrence formulas is presented. Several implementations of variants of the product-type methods are considered, and some well-known variants are recalled in section 5. In section 6, we report some numerical experiments. Finally, we make some concluding remarks in section 7.

Throughout this paper, superscript BCG is used to distinguish iterates generated in the algorithm of Bi-CG.

2. Restructuring Bi-CG.

To begin with, let us consider the solution of the linear system

$$Ax = b \tag{2.1}$$

where A is an $N \times N$ large and sparse nonsymmetric matrix, and is always assumed to be nonsingular.

The algorithm of Bi-CG for solving the system (2.1), described first by C. Lanczos [8], was given by R. Fletcher [4] as follows.

ALGORITHM 1. Bi-CG.

```
$x_0^{\mathrm{BCG}}$ is an initial guess, $r_0^{\mathrm{BCG}} = b - Ax_0^{\mathrm{BCG}}$;
set $p_0^{\mathrm{BCG}*} = r_0^{\mathrm{BCG}*} = p_0^{\mathrm{BCG}} = r_0^{\mathrm{BCG}}$;
For $n = 0, 1, \ldots$ until $\| r_n^{\mathrm{BCG}} \| \le \varepsilon \| b \|$ do :
$\alpha_n = \dfrac{(r_n^{\mathrm{BCG}*}, r_n^{\mathrm{BCG}})}{(p_n^{\mathrm{BCG}*}, Ap_n^{\mathrm{BCG}})}$,
$x_{n+1}^{\mathrm{BCG}} = x_n^{\mathrm{BCG}} + \alpha_n p_n^{\mathrm{BCG}}$,
$r_{n+1}^{\mathrm{BCG}} = r_n^{\mathrm{BCG}} - \alpha_n A p_n^{\mathrm{BCG}}, \quad r_{n+1}^{\mathrm{BCG}*} = r_n^{\mathrm{BCG}*} - \alpha_n A^T p_n^{\mathrm{BCG}*}$,
$\beta_n = \dfrac{(r_{n+1}^{\mathrm{BCG}*}, r_{n+1}^{\mathrm{BCG}})}{(r_n^{\mathrm{BCG}*}, r_n^{\mathrm{BCG}})}$,
$p_{n+1}^{\mathrm{BCG}} = r_{n+1}^{\mathrm{BCG}} + \beta_n p_n^{\mathrm{BCG}}, \quad p_{n+1}^{\mathrm{BCG}*} = r_{n+1}^{\mathrm{BCG}*} + \beta_n p_n^{\mathrm{BCG}*}$.
```

Let r_0^{BCG} and $r_0^{\mathrm{BCG}*}$ be abbreviated as r_0 and r_0^*. In the algorithm of Bi-CG, the two Krylov subspaces

$$K_n(A;r_0) := \mathrm{span}\{r_0, Ar_0, \dots, A^{n-1}r_0\} \text{ and}$$

$$K_n(A^T;r_0^*) := \mathrm{span}\{r_0^*, A^Tr_0^*, \dots, (A^T)^{n-1}r_0^*\}$$

are generated, and the approximate solution x_n^{BCG} ($\in x_0^{\mathrm{BCG}} + K_n(A;r_0)$) is given in such a way that the residual r_n^{BCG}($= b - Ax_n^{\mathrm{BCG}}$ theoretically; it is always assumed that the residual r_n^{BCG} is convergent toward zero throughout this paper) is made orthogonal with respect to $K_n(A^T;r_0^*)$. Therefore, we have the following orthogonalities of Bi-CG [4]:

$$r_n^{\mathrm{BCG}} \perp K_n(A^T;r_0^*) \quad \text{and} \quad Ap_n^{\mathrm{BCG}} \perp K_n(A^T;r_0^*). \tag{2.2}$$

Let R_n and P_n denote the polynomials corresponding to the residual r_n^{BCG} and the direction p_n^{BCG}; then the iterates r_n^{BCG}, p_n^{BCG}, $r_n^{\mathrm{BCG}*}$, and $p_n^{\mathrm{BCG}*}$ can been expressed as follows:

$$r_n^{\mathrm{BCG}} = R_n(A)r_0,\ p_n^{\mathrm{BCG}} = P_n(A)r_0,\ r_n^{\mathrm{BCG}*} = R_n(A^T)r_0^*,\ p_n^{\mathrm{BCG}*} = P_n(A^T)r_0^*.$$

R_n is the residual polynomial of Bi-CG and the so-called Lanczos polynomial. Notice that the basic recurrence relations between R_n and P_n hold as follows:

$$R_0(\lambda) = 1, \quad P_0(\lambda) = 1, \tag{2.3}$$

$$R_{n+1}(\lambda) = R_n(\lambda) - \alpha_n\lambda P_n(\lambda), \tag{2.4}$$

$$P_{n+1}(\lambda) = R_{n+1}(\lambda) + \beta_n P_n(\lambda), \quad \text{for } n = 1, 2, \dots. \tag{2.5}$$

In practical implementations, one is faced with two disadvantages first observed by P. Sonneveld in the algorithm of Bi-CG:

- to construct the sequence of the residuals r_0^{BCG}, $r_1^{\mathrm{BCG}}, \dots, r_n^{\mathrm{BCG}}$ in the Krylov subspace $K_n(A;r_0)$, one has to use the Krylov subspace $K_n(A^T;r_0^*)$ so that the operations of transpose matrix–vector multiplications appear per iteration;
- although $r_n^{\mathrm{BCG}*}$ also converges toward zero, this convergence property can not be exploited directly for improving the convergence rate of the residual r_n^{BCG}.

Next, a unified way to remedy the disadvantages above is proposed. Here, we attempt to use a polynomial H_n to accelerate r_n^{BCG}, say, make $H_n(A)r_n^{\mathrm{BCG}}$($= H_n(A)R_n(A)r_0$) as new residuals converge toward zero fast. By doing so, we can derive a diverse collection of methods which have a product-type residual polynomial. Here, we characterize the product-type methods based on Bi-CG by the following process:

- the residual polynomial of a product-type method is defined by the product of the two n degree polynomials

$$H_n(\lambda)R_n(\lambda) \tag{2.6}$$

  where R_n is the Lanczos polynomial and H_n is undetermined.

In the product-type methods, $r_n^{\mathrm{BCG}*}$ would never be formed, nor would the transpose matrix–vector multiplications be needed. According to the description above, we know that CGS, Bi-CGSTAB, and Bi-CGSTAB2 all belong to such kinds of product-type methods based on Bi-CG.

In general, when one attempts to derive a new product-type method from Bi-CG, one would encounter the following three problems:

(A) in computing the first kind of parameters α_n and β_n which are used to determine R_n;

(B) in designing the new polynomial H_n;

(C) in computing the second kind of parameters which are used to determine H_n.

In fact, problem (A) can be solved with a technique using the orthogonalities (2.2). This technique is due to P. Sonneveld [14], and developed later by H. A. van der Vorst [17]. Notice that $r_n^{\mathrm{BCG}*}$ and $p_n^{\mathrm{BCG}*}$ can be written as

$$r_n^{\mathrm{BCG}*} = R_n(A^T)r_0^* = \left((-1)^n \prod_{i=0}^{n-1} \alpha_i\right)(A^T)^n r_0^* + z_1,$$

$$p_n^{\mathrm{BCG}*} = P_n(A^T)r_0^* = \left((-1)^n \prod_{i=0}^{n-1} \alpha_i\right)(A^T)^n r_0^* + z_2$$

with z_1 and $z_2 \in K_n(A^T; r_0^*)$. And from the orthogonalities (2.2), auxiliary formulas for computing α_n and β_n can be recovered:

$$\alpha_n = \frac{(r_n^{\mathrm{BCG}*}, r_n^{\mathrm{BCG}})}{(p_n^{\mathrm{BCG}*}, Ap_n^{\mathrm{BCG}})} = \frac{((A^T)^n r_0^*, r_n^{\mathrm{BCG}})}{((A^T)^n r_0^*, Ap_n^{\mathrm{BCG}})}, \tag{2.7}$$

$$\beta_n = \frac{(r_{n+1}^{\mathrm{BCG}*}, r_{n+1}^{\mathrm{BCG}})}{(r_n^{\mathrm{BCG}*}, r_n^{\mathrm{BCG}})} = -\alpha_n \frac{((A^T)^{n+1} r_0^*, r_{n+1}^{\mathrm{BCG}})}{((A^T)^n r_0^*, r_n^{\mathrm{BCG}})}. \tag{2.8}$$

The formulas (2.7), (2.8) imply that we can directly use the basis vector $(A^T)^n r_0^*$ of the Krylov subspace $K_{n+1}(A^T; r_0^*)$ instead of vectors $r_n^{\mathrm{BCG}*}$ and $p_n^{\mathrm{BCG}*}$ to determine the first kind of parameters α_n and β_n in the algorithm of Bi-CG. The formulas are fundamental in establishing the product-type methods based on Bi-CG.

In order to deal with problem (B), i.e., designing the polynomial H_n in (2.6), three different polynomials had been presented [14, 17, 5]:

- in CGS, the polynomial H_n is chosen to be the Lanczos polynomial R_n;
- in Bi-CGSTAB, the polynomial H_n is defined by a factored polynomial Q_n with a second kind of parameter ω_{n-1}

$$Q_n = (1 - \omega_{n-1}\lambda)Q_{n-1};$$

- in Bi-CGSTAB2, the polynomial H_n is denoted by τ_n and chosen to satisfy the recurrence relations

$$\tau_{2n+1} = (1 - \chi_n\lambda)\tau_{2n}, \quad \tau_{2n+2} = (\mu_n + \nu_n\lambda)\tau_{2n+1} + (1 - \mu_n)\tau_{2n}.$$

The operations for the second kind of parameters can be omitted in CGS. For problem (C), the second kind of parameter ω_n (resp., χ_n, μ_n, and ν_n) in Bi-CGSTAB (resp., Bi-CGSTAB2) is determined in such a way that the residual 2-norm per iteration can be minimized locally.

The design of the polynomial H_n in practice is desired as follows:

(1) to make the polynomial H_n satisfy short-term recurrence relations so that little computational work and low storage costs are required per iteration;

(2) to choose the second kind of parameters of H_n reasonably to get rather fast and stable convergence behavior by making approximate solutions satisfy an optimality property.

3. Generalized product-type methods based on Bi-CG.

In this section, we will describe the basic idea to establish a standard polynomial H_n which leads to generalized product-type methods based on Bi-CG for solving the linear system (2.1).

3.1. Construction of polynomial.

Eliminating P_n from (2.4), (2.5), one would recover the three-term recurrence relations for R_n alone [15]:

$$R_0(\lambda) = 1, \quad R_1(\lambda) = (1 - \alpha_0\lambda)R_0(\lambda), \tag{3.1}$$

$$R_{n+1}(\lambda) = \left(1 + \frac{\beta_{n-1}}{\alpha_{n-1}}\alpha_n - \alpha_n\lambda\right) R_n(\lambda) - \frac{\beta_{n-1}}{\alpha_{n-1}}\alpha_n R_{n-1}(\lambda), \quad n = 1, 2, \ldots. \tag{3.2}$$

The three-term recurrence relations are fundamental for the efficiency of all iterative methods based on the Lanczos process. In fact, the three-term recurrence relations of Bi-CG provide a modest hint that motivates our search for developing the generalized product-type methods based on Bi-CG. Here, we introduce two independent parameters ζ_n and η_n and let the polynomial H_n model after the three-term recurrence relations (3.1), (3.2) of R_n as follows:

$$H_0(\lambda) := 1, \quad H_1(\lambda) := (1 - \zeta_0\lambda)H_0(\lambda), \tag{3.3}$$

$$H_{n+1}(\lambda) := (1 + \eta_n - \zeta_n\lambda)H_n(\lambda) - \eta_n H_{n-1}(\lambda), \quad n = 1, 2, \ldots \tag{3.4}$$

where the undetermined parameters ζ_n and η_n belong to the second kind.

3.2. Recurrence formulas for iterates.

Using the basic recurrence relations (2.4), (2.5), we have the following set of recurrence relations among the products of polynomials H_nR_n, H_nR_{n+1}, $(H_{n-1} - H_n)R_{n+1}$, H_nP_n, and $H_{n-1}P_n$:

$$H_{n+1}R_{n+1} = H_nR_{n+1} - \eta_n(H_{n-1} - H_n)R_{n+1} - \zeta_n\lambda H_nR_{n+1}, \tag{3.5}$$

$$H_nR_{n+1} = H_nR_n - \alpha_n\lambda H_nP_n, \tag{3.6}$$

$$(H_{n-1} - H_n)R_{n+1} = H_{n-1}R_n - H_nR_{n+1} - \alpha_n\lambda H_{n-1}P_n, \tag{3.7}$$

$$H_{n+1}P_{n+1} = H_{n+1}R_{n+1} - \beta_n\eta_nH_{n-1}P_n + \beta_n(1 + \eta_n)H_nP_n - \beta_n\zeta_n\lambda H_nP_n, \tag{3.8}$$

$$H_nP_{n+1} = H_nR_{n+1} + \beta_nH_nP_n. \tag{3.9}$$

From (2.6), we build up the product-type methods with residual

$$r_n := H_n(A)r_n^{\mathrm{BCG}} = H_n(A)R_n(A)r_0 = b - Ax_n \tag{3.10}$$

which can be obtained with the auxiliary iterates

$$t_n := H_n(A)r_{n+1}^{\mathrm{BCG}}, \; y_n := (H_{n-1}(A) - H_n(A))r_{n+1}^{\mathrm{BCG}}, \tag{3.11}$$

$$p_n := H_n(A)p_n^{\mathrm{BCG}}, \; s_n := H_{n-1}(A)p_n^{\mathrm{BCG}}. \tag{3.12}$$

According to the recurrence relations (3.5)–(3.9), we have the following recurrence formulas among the sequences of the iterates r_n, t_n, y_n, p_n, and s_n:

$$r_{n+1} = t_n - \eta_ny_n - \zeta_nAt_n, \tag{3.13}$$

$$t_n = r_n - \alpha_nAp_n, \tag{3.14}$$

$$y_n = t_{n-1} - t_n - \alpha_nAs_n, \tag{3.15}$$

$$p_{n+1} = r_{n+1} - \beta_n\eta_ns_n + \beta_n(1 + \eta_n)p_n - \beta_n\zeta_nAp_n, \tag{3.16}$$

$$s_{n+1} = t_n + \beta_np_n. \tag{3.17}$$

From the definition (3.10) of the residual r_n and the recurrence formula (3.13)–(3.17), we have the formula to update the approximate solution x_{n+1}:

$$x_{n+1} = -\eta_n(x_{n-1} + \alpha_{n-1}p_{n-1} + \alpha_n s_n) + (1+\eta_n)(x_n + \alpha_n p_n) + \zeta_n t_n. \tag{3.18}$$

3.3. Computations for $\boldsymbol{\alpha_n}$ and $\boldsymbol{\beta_n}$. Since the coefficient of the highest-order term of H_n is $(-1)^n \prod_{i=0}^{n-1} \zeta_i$, we have

$$(r_0^*, r_n) = (H_n(A^T)r_0^*, r_n^{\mathrm{BCG}}) = \left((-1)^n \prod_{i=0}^{n-1} \zeta_i\right)((A^T)^n r_0^*, r_n^{\mathrm{BCG}}),$$

$$(r_0^*, Ap_n) = (H_n(A^T)r_0^*, Ap_n^{\mathrm{BCG}}) = \left((-1)^n \prod_{i=0}^{n-1} \zeta_i\right)((A^T)^n r_0^*, Ap_n^{\mathrm{BCG}}).$$

Then, from the formulas (2.7), (2.8), α_n, and β_n can be recovered from the iterates r_{n+1}, r_n, and p_n:

$$\alpha_n = \frac{(r_0^*, r_n)}{(r_0^*, Ap_n)}, \quad \beta_n = \frac{\alpha_n}{\zeta_n} \cdot \frac{(r_0^*, r_{n+1})}{(r_0^*, r_n)}. \tag{3.19}$$

Now, we have accomplished a part of the purpose of this paper, i.e., proposed a unified way to derive generalized product-type methods from the restructuring of Bi-CG. Due to the lack of a criterion for choices of ζ_n and η_n, it is very hard in fact to determine the second kind of parameters ζ_n and η_n that are closely and indissolubly connected with convergence behavior in practice. Implementations of the second kind of parameters ζ_n and η_n will be discussed in section 5 in detail.

4. Equivalent approach.

Notice that the $(n-1)$th iterates x_{n-1} and p_{n-1} are used to update x_{n+1} in the formula (3.18). It is not appreciated in any ideal iterative method. In this section, we propose another mathematically equivalent approach to establish polynomial $H_n(\lambda)$ in (3.4). By doing so, we can present a new formula to update x_{n+1} without the $(n-1)$th iterates, like x_{n-1} and p_{n-1}.

4.1. Equivalent polynomial. Notice that $H_n(\lambda)$ generated by the three-term recurrence relations (3.4) satisfies $H_n(0) = 1$ for any n. Then, we have $H_{n+1}(0) - H_n(0) = 0$ for any n. We define an auxiliary polynomial $G_n(\lambda)$ with degree n as

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

Rewrite the three-term recurrence relations (3.4) as

$$H_{n+1}(\lambda) - H_n(\lambda) = -\zeta_n \lambda H_n(\lambda) + \eta_n(H_n(\lambda) - H_{n-1}(\lambda)),$$

then the double sets of polynomials $H_n(\lambda)$ and $G_n(\lambda)$ mutually interlocked by the following recurrence relations can be obtained:

$$H_0(\lambda) = 1, \quad G_0(\lambda) = 1, \tag{4.1}$$

$$H_{n+1}(\lambda) = H_n(\lambda) - \zeta_n \lambda G_n(\lambda), \tag{4.2}$$

$$G_{n+1}(\lambda) = H_{n+1}(\lambda) + \zeta_n \frac{\eta_{n+1}}{\zeta_{n+1}} G_n(\lambda), \quad n = 1, 2, \ldots. \tag{4.3}$$

Using the recurrence relations (4.2) and (4.3), we can reduce the formula (3.18).

4.2. Equivalent recurrence formulas for iterates. To compute the product of polynomials H_nR_n, we make use of a new set of recurrence relations among the products of polynomials H_nR_n, H_nR_{n+1}, $\zeta_{n-1}\lambda G_{n-1}R_{n+1}(= (H_{n-1}-H_n)R_{n+1})$, H_nP_n, λH_nP_{n+1}, $\zeta_n\lambda G_nP_n$, and $\zeta_nG_nR_{n+1}$ instead of the set of recurrence relations (3.5)–(3.9):

$$H_{n+1}R_{n+1} = H_nR_{n+1} - \eta_n\zeta_{n-1}\lambda G_{n-1}R_{n+1} - \zeta_n\lambda H_nR_{n+1} \tag{4.4}$$

$$= H_nR_n - \alpha_n\lambda H_nP_n - \zeta_n\lambda G_nR_{n+1}, \tag{4.5}$$

$$H_nR_{n+1} = H_nR_n - \alpha_n\lambda H_nP_n, \tag{4.6}$$

$$\zeta_n\lambda G_nR_{n+2} = H_nR_{n+1} - H_{n+1}R_{n+1} - \alpha_{n+1}\lambda H_nP_{n+1} + \alpha_{n+1}\lambda H_{n+1}P_{n+1}, \tag{4.7}$$

$$H_{n+1}P_{n+1} = H_{n+1}R_{n+1} + \beta_nH_nP_n - \beta_n\zeta_n\lambda G_nP_n, \tag{4.8}$$

$$\lambda H_nP_{n+1} = \lambda H_nR_{n+1} + \beta_n\lambda H_nP_n, \tag{4.9}$$

$$\zeta_n\lambda G_nP_n = \zeta_n\lambda H_nP_n + \eta_n(H_{n-1}R_n - H_nR_n + \beta_{n-1}\zeta_{n-1}\lambda G_{n-1}P_{n-1}), \tag{4.10}$$

$$\zeta_nG_nR_{n+1} = \zeta_nH_nR_n + \eta_n\zeta_{n-1}G_{n-1}R_n - \alpha_n\zeta_n\lambda G_nP_n. \tag{4.11}$$

To obtain residual $r_n := H_n(A)r_n^{\mathrm{BCG}}$, let us define new auxiliary iterates as

$$w_n := AH_n(A)p_{n+1}^{\mathrm{BCG}},\ u_n := \zeta_nAG_n(A)p_n^{\mathrm{BCG}},\ z_n := \zeta_nG_n(A)r_{n+1}^{\mathrm{BCG}}. \tag{4.12}$$

Then, we have new recurrence formulas among the iterates defined in (3.11), (3.12) and (4.12):

$$r_{n+1} = t_n - \eta_ny_n - \zeta_nAt_n \tag{4.13}$$

$$= r_n - \alpha_nAp_n - Az_n, \tag{4.14}$$

$$t_n = r_n - \alpha_nAp_n, \tag{4.15}$$

$$y_{n+1} = t_n - r_{n+1} - \alpha_{n+1}w_n + \alpha_{n+1}Ap_{n+1}, \tag{4.16}$$

$$p_{n+1} = r_{n+1} + \beta_n(p_n - u_n), \tag{4.17}$$

$$w_n = At_n + \beta_nAp_n, \tag{4.18}$$

$$u_n = \zeta_nAp_n + \eta_n(t_{n-1} - r_n + \beta_{n-1}u_{n-1}), \tag{4.19}$$

$$z_n = \zeta_nr_n + \eta_nz_{n-1} - \alpha_nu_n. \tag{4.20}$$

From (4.14) we have the following new formula to update the approximate solution x_{n+1}:

$$x_{n+1} = x_n + \alpha_np_n + z_n. \tag{4.21}$$

From the equivalence between the three-term recurrence relations (3.4) and (4.2), (4.3), it is easy to see that the approximate solutions x_{n+1} generated by formula (4.21) are mathematically equivalent to the approximate solutions x_{n+1} generated by formula (3.18). The formula (3.18) is reduced to the formula (4.21).

Clearly, the formulas of (3.19) for computing α_n and β_n are still valid here.

5. Implementation details.

In accordance with the requirement (2) described at the end of section 2, we summarize several possibilities to select ζ_n and η_n for the actual implementation of the product-type methods based on Bi-CG. As well-known variants, the algorithms of CGS, Bi-CGSTAB, and Bi-CGSTAB2 will be recalled in terms of special choices. A new variant named GPBi-CG will be implemented.

5.1. The choice for CGS. Suppose that $\zeta_n = \alpha_n$ and $\eta_n = \frac{\beta_{n-1}}{\alpha_{n-1}}\alpha_n$ in recurrence relations (3.3), (3.4); we obtain a significant variant of the product-type methods which only depends on information of Bi-CG. It is easy to see that this variant is mathematically equivalent to CGS.

Notice that $t_{n-1} - r_n = Az_{n-1}$. In this case, we have $H_n = R_n$ and $G_n = P_n$, and use the recurrence formula (4.14) to update r_{n+1}. This fact leads to relation $p_n - u_n = z_n/\alpha_n$ for any n, then the iterates t_n, y_n, and w_n are omissible in the recurrence formulas (4.13)–(4.20).

Noticing that $(r_0^*, Ap_n) = (r_0^*, u_n/\alpha_n)$, and setting new iterates $\tilde{u}_n := A^{-1}u_n/\alpha_n$ and $\tilde{z}_n := z_n/\alpha_n$, then the algorithm of CGS [14] is recalled as follows.

ALGORITHM 2. CGS.

```
$x_0$ is an initial guess, $r_0 = b - Ax_0$;
$r_0^*$ is an arbitrary vector, such that
$(r_0^*, r_0) \neq 0$, e.g., $r_0^* = r_0$; and set $\beta_{-1} = 0$;
For $n = 0, 1, \ldots$ until $\| r_n \| \leq \varepsilon \| b \|$ do :
$p_n = r_n + \beta_{n-1}\tilde{z}_{n-1}$,
$\tilde{u}_n = p_n + \beta_{n-1}(\tilde{z}_{n-1} + \beta_{n-1}\tilde{u}_{n-1})$,
$\alpha_n = \frac{(r_0^*, r_n)}{(r_0^*, A\tilde{u}_n)}$,
$\tilde{z}_n = p_n - \alpha_n A\tilde{u}_n$,
$x_{n+1} = x_n + \alpha_n(p_n + \tilde{z}_n)$,
$r_{n+1} = r_n - \alpha_n A(p_n + \tilde{z}_n)$,
$\beta_n = \frac{(r_0^*, r_{n+1})}{(r_0^*, r_n)}$.
```

5.2. The choice for Bi-CGSTAB. In the subsection, we attempt to get some variants of the product-type methods with little computational work. Here, suppose that parameter η_n in recurrence formulas (4.13)–(4.20) had been given in advance, the other parameter ζ_n is selected to minimize the residual 2-norm as a function of ζ:

$$f(\zeta) := \| r_{n+1} \| = \| t_n - \eta_n y_n - \zeta A t_n \|$$

which reaches its minimum value at

$$\zeta_n = \frac{(At_n, t_n - \eta_n y_n)}{(At_n, At_n)}.$$

Moreover, we consider the following special case that all η_n are defined by a quantity ω. ω is called the relaxation factor. This aspect of how to choose the relaxation factor for solving realistic problems needs further research.

1. According to the recurrence formulas (4.13)–(4.20), we have the following variant of the product-type methods, and name it GPBi-CG(ω).

   ALGORITHM 3. GPBi-CG(ω).

```
$x_0$ is an initial guess, $r_0 = b - Ax_0$;
$r_0^*$ is an arbitrary vector, such that
$(r_0^*, r_0) \neq 0$, e.g., $r_0^* = r_0$;
and set $t_{-1} = w_{-1} = 0$, $\beta_{-1} = 0$;
For $n = 0, 1, \ldots$ until $\| r_n \| \leq \varepsilon \| b \|$ do :
$p_n = r_n + \beta_{n-1}(p_{n-1} - u_{n-1})$,
$\alpha_n = \dfrac{(r_0^*, r_n)}{(r_0^*, Ap_n)}$,
$y_n = t_{n-1} - r_n - \alpha_n w_{n-1} + \alpha_n Ap_n$,
$t_n = r_n - \alpha_n Ap_n$,
$\eta_n = \omega$, (if $n = 0$, then $\eta_n = 0$)
$\zeta_n = \dfrac{(At_n, t_n - \eta_n y_n)}{(At_n, At_n)}$,
$u_n = \zeta_n Ap_n + \eta_n(t_{n-1} - r_n + \beta_{n-1}u_{n-1})$,
$z_n = \zeta_n r_n + \eta_n z_{n-1} - \alpha_n u_n$,
$x_{n+1} = x_n + \alpha_n p_n + z_n$,
$r_{n+1} = t_n - \eta_n y_n - \zeta_n At_n$,
$\beta_n = \dfrac{\alpha_n}{\zeta_n} \cdot \dfrac{(r_0^*, r_{n+1})}{(r_0^*, r_n)}$,
$w_n = At_n + \beta_n Ap_n$.
```

2. Here, we suppose that $\omega = 0$ for any n in GPBi-CG(ω). In this case, $u_n = \zeta_n Ap_n$, and then $z_n = \zeta_n t_n$. Notice that the iterates y_n, u_n, z_n, and w_n become worthless. In this way an important and economical variant will be obtained again, recalled Bi-CGSTAB [17].
   Algorithm 4. Bi-CGSTAB.

```
$x_0$ is an initial guess, $r_0 = b - Ax_0$;
$r_0^*$ is an arbitrary vector, such that
$(r_0^*, r_0) \neq 0$, e.g., $r_0^* = r_0$; and set $\beta_{-1} = 0$;
For $n = 0, 1, \ldots$ until $\| r_n \| \leq \varepsilon \| b \|$ do :
$p_n = r_n + \beta_{n-1}(p_{n-1} - \zeta_{n-1}Ap_{n-1})$,
$\alpha_n = \dfrac{(r_0^*, r_n)}{(r_0^*, Ap_n)}$,
$t_n = r_n - \alpha_n Ap_n$,
$\zeta_n = \dfrac{(At_n, t_n)}{(At_n, At_n)}$,
$x_{n+1} = x_n + \alpha_n p_n + \zeta_n t_n$,
$r_{n+1} = t_n - \zeta_n At_n$,
$\beta_n = \dfrac{\alpha_n}{\zeta_n} \cdot \dfrac{(r_0^*, r_{n+1})}{(r_0^*, r_n)}$.
```

5.3. The choice for GPBi-CG. It is convenient to determine parameters ζ_n and η_n in terms of minimizing the residual 2-norm as the function of ζ and η:

$$f(\zeta, \eta) := \| r_{n+1} \| = \| t_n - \eta y_n - \zeta At_n \| .$$

Thus, we have the following variant of the product-type methods and name it GPBi-CG.

Algorithm 5. GPBi-CG.

```
$x_0$ is an initial guess, $r_0 = b - Ax_0$;
$r_0^*$ is an arbitrary vector, such that
```

```
$(r_0^*, r_0) \neq 0, \text{e.g.}, r_0^* = r_0;$
and set  $t_{-1} = w_{-1} = 0,\ \beta_{-1} = 0;$
For $n = 0, 1, \dots$ until $\| r_n \| \leq \varepsilon \| b \|$ do :
$p_n = r_n + \beta_{n-1}(p_{n-1} - u_{n-1}),$
$\alpha_n = \frac{(r_0^*, r_n)}{(r_0^*, Ap_n)},$
$y_n = t_{n-1} - r_n - \alpha_n w_{n-1} + \alpha_n Ap_n,$
$t_n = r_n - \alpha_n Ap_n,$
$\zeta_n = \frac{(y_n, y_n)(At_n, t_n) - (y_n, t_n)(At_n, y_n)}{(At_n, At_n)(y_n, y_n) - (y_n, At_n)(At_n, y_n)},$
$\eta_n = \frac{(At_n, At_n)(y_n, t_n) - (y_n, At_n)(At_n, t_n)}{(At_n, At_n)(y_n, y_n) - (y_n, At_n)(At_n, y_n)},$
(if $n = 0$, then $\zeta_n = \frac{(At_n, t_n)}{(At_n, At_n)},\ \eta_n = 0$)
$u_n = \zeta_n Ap_n + \eta_n(t_{n-1} - r_n + \beta_{n-1} u_{n-1}),$
$z_n = \zeta_n r_n + \eta_n z_{n-1} - \alpha_n u_n,$
$x_{n+1} = x_n + \alpha_n p_n + z_n,$
$r_{n+1} = t_n - \eta_n y_n - \zeta_n At_n,$
$\beta_n = \frac{\alpha_n}{\zeta_n} \cdot \frac{(r_0^*, r_{n+1})}{(r_0^*, r_n)},$
$w_n = At_n + \beta_n Ap_n.$
```

5.4. Hybrid variants. Another possibility of implementation is to combine one variant of the product-type methods with another one of the product-type methods without restarting. For example, during the first half of the algorithm, we use Bi-CGSTAB, and during the latter half, we use CGS.

Actually, Bi-CGSTAB2 can be explained as a hybrid variant of Bi-CGSTAB and GPBi-CG. In other words, for a choice of the second kind of parameters, the minimization of residual 2-norm is defined as a function of two variables ζ and η at even iterations and as a function of one variable ζ under the assumption $\eta_n = 0$ at odd iterations.

Let $\eta_n = 0$ when $n = 2k$ in the GPBi-CG algorithm; Bi-CGSTAB2 reads as follows [5].

Algorithm 6. Bi-CGSTAB2.

```
$x_0$ is an initial guess, $r_0 = b - Ax_0;$
$r_0^*$ is an arbitrary vector, such that
$(r_0^*, r_0) \neq 0, \text{e.g.}, r_0^* = r_0;$
and set  $t_{-1} = w_{-1} = 0,\ \beta_{-1} = 0;$
For $n = 0, 1, \dots$ until $\| r_n \| \leq \varepsilon \| b \|$ do :
$p_n = r_n + \beta_{n-1}(p_{n-1} - u_{n-1}),$
$\alpha_n = \frac{(r_0^*, r_n)}{(r_0^*, Ap_n)},$
$y_n = t_{n-1} - r_n - \alpha_n w_{n-1} + \alpha_n Ap_n,$
$t_n = r_n - \alpha_n Ap_n,$
```

```
if $n = 2k$, then $\zeta_n = \dfrac{(At_n, t_n)}{(At_n, At_n)}$, $\eta_n = 0$,
else $\zeta_n = \dfrac{(y_n, y_n)(At_n, t_n) - (y_n, t_n)(At_n, y_n)}{(At_n, At_n)(y_n, y_n) - (y_n, At_n)(At_n, y_n)}$,
    $\eta_n = \dfrac{(At_n, At_n)(y_n, t_n) - (y_n, At_n)(At_n, t_n)}{(At_n, At_n)(y_n, y_n) - (y_n, At_n)(At_n, y_n)}$, end if
$u_n = \zeta_n Ap_n + \eta_n(t_{n-1} - r_n + \beta_{n-1}u_{n-1})$,
$z_n = \zeta_n r_n + \eta_n z_{n-1} - \alpha_n u_n$,
$x_{n+1} = x_n + \alpha_n p_n + z_n$,
$r_{n+1} = t_n - \eta_n y_n - \zeta_n At_n$,
$\beta_n = \dfrac{\alpha_n}{\zeta_n} \cdot \dfrac{(r_0^*, r_{n+1})}{(r_0^*, r_n)}$,
$w_n = At_n + \beta_n Ap_n$.
```

6. Numerical experiments. In this section we report some numerical experiments to show convergence behaviors of all the variants (except GPBi-CG(ω)) for several linear systems with complex spectrum. These experiments have been carried out with CGS, Bi-CGSTAB, Bi-CGSTAB2, and GPBi-CG in double precision floating point arithmetic on a SUN SPARCstation IPX computer. In all cases the iteration was started with $x_0 = 0$, and the iterations were stopped when $\| r_n \| / \| r_0 \| \leq 10^{-12}$. The convergence plots show the relative residual 2-norms $\| r_n \| / \| r_0 \|$ (on the vertical axis) versus the iteration number n (on the horizontal axis).

6.1. Example 1. In the first example, we consider the complex Toeplitz matrix of order 200 with a parameter γ [5].

$$A := \begin{bmatrix} 4 & 0 & 1 & .7 & & \\ \gamma\boldsymbol{i} & 4 & 0 & 1 & .7 & \\ & \gamma\boldsymbol{i} & 4 & 0 & 1 & \ddots \\ & & \gamma\boldsymbol{i} & 4 & 0 & \ddots \\ & & & \gamma\boldsymbol{i} & 4 & \ddots \\ & & & & \ddots & \ddots \end{bmatrix}.$$

Taking $b = (\boldsymbol{i}, \boldsymbol{i}, \ldots, \boldsymbol{i})^T$ as the right-hand side, we consider two cases of $\gamma = 3.5$ and $\gamma = 3.79$, and show its numerical results in Fig. 1 and Fig. 2, respectively.

From Fig. 1 and Fig. 2, we observe that CGS diverged, and Bi-CGSTAB, Bi-CGSTAB2, and GPBi-CG converged in the two cases. We observe also that Bi-CGSTAB2 and GPBi-CG converged slightly faster than Bi-CGSTAB.

6.2. Example 2. As the second example, we consider two linear systems with complex spectrum from a 30×30 and a 50×50 central difference discretization of the Helmholtz equation over $[0, \pi] \times [0, \pi]$ described in [1]:

$$u_{xx} + u_{yy} + k^2 u = 0$$

with Dirichlet condition $u = 0$ along $y = \pi$, Neumann conditions $u_x = \boldsymbol{i}\sqrt{k^2 - \frac{1}{4}}\cos(\frac{y}{2})$ along $x = 0$ and $u_y = 0$ along $y = 0$, and radiation condition $u_x - \boldsymbol{i}\sqrt{k^2 - \frac{1}{4}}u = 0$

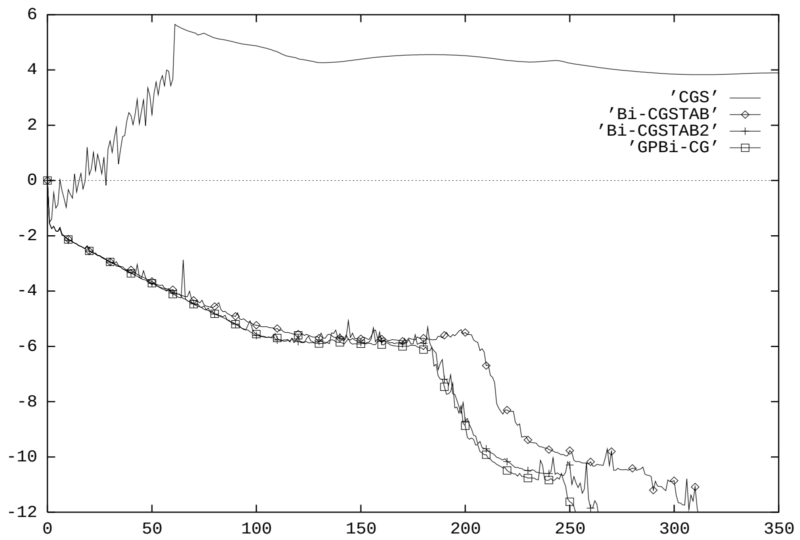

FIG. 1. *The residual* 2*-norm history for Example* 1 ($\gamma = 3.5$).

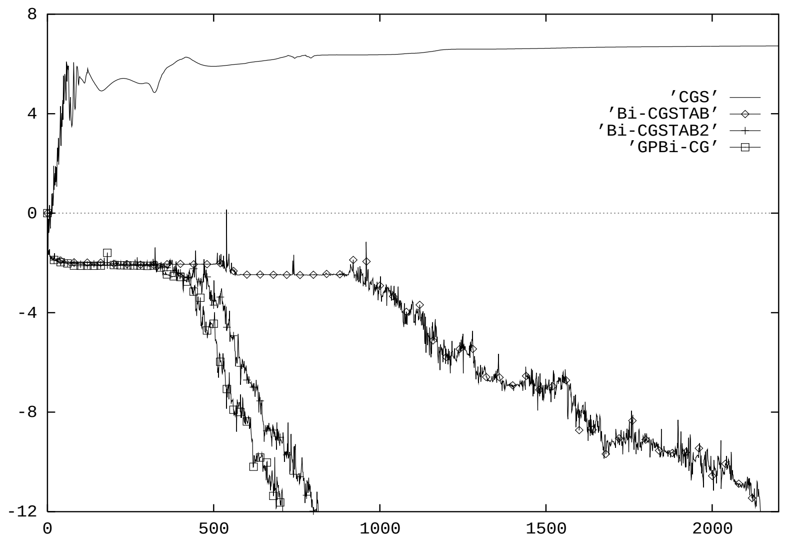

FIG. 2. *The residual* 2*-norm history for Example* 1 ($\gamma = 3.79$).

along $x = \pi$. This leads to two systems with unknowns 31×30, 51×50. Here we only consider the case $k = 2.27$.

In the two cases, although all the variants converged, CGS and GPBi-CG converged slightly faster than Bi-CGSTAB and Bi-CGSTAB2. CGS converged faster eventually, but we can see that the local effects were not smooth, and the final residual 2-norms were worse in this method (see Table 1).

Although GPBi-CG is more expensive with respect to the number of inner products and vector updates, from Fig. 3 we observe that GPBi-CG performs much better so that GPBi-CG required 42% of the iteration steps of Bi-CGSTAB and 52% of Bi-CGSTAB2 for this coarser grid. In Fig. 4, GPBi-CG required 579 iteration steps to get the residual 2-norm below 10^{-12}, Bi-CGSTAB required 1900 iteration steps, and

TABLE 1
Iteration step and $\log_{10}$ *of the final true residual* 2*-norm.*

	Example 1 (γ=3.5)	Example 1 (γ=3.79)	Example 2 ($N = 31$)	Example 2 ($N = 51$)
CGS	divergence	divergence	308 (−9.6)	597 (−7.4)
Bi-CGSTAB	312 (−12.2)	2145 (−12.0)	720 (−12.0)	1900 (−11.8)
Bi-CGSTAB2	264 (−12.2)	815 (−12.0)	576 (−9.1)	1255 (−10.8)
GPBi-CG	253 (−12.1)	708 (−12.1)	299 (−11.6)	579 (−10.1)

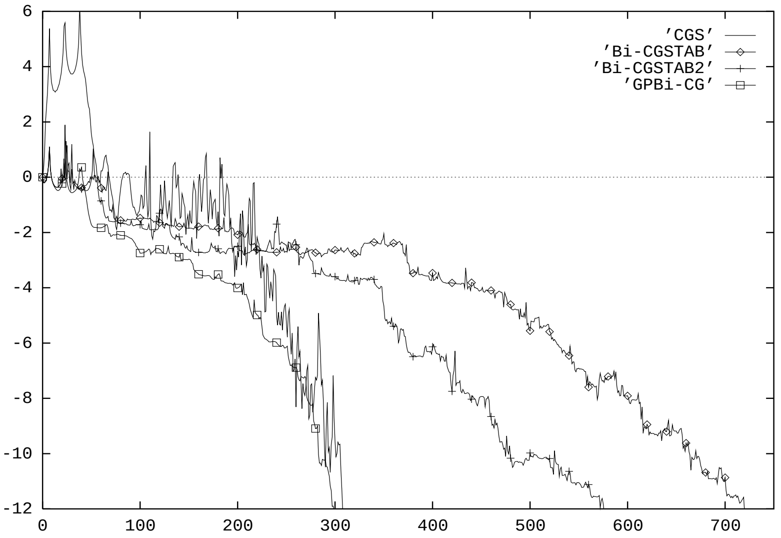

FIG. 3. *The residual* 2*-norm history for Example* 2 (31×30).

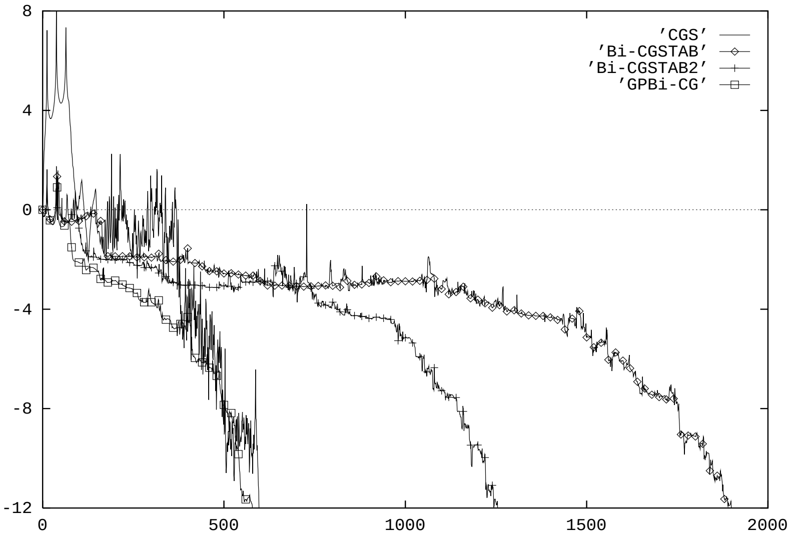

FIG. 4. *The residual* 2*-norm history for Example* 2 (51×50).

Bi-CGSTAB2 required 1255 iteration steps. For this finer grid, GPBi-CG needs only 30% of the iteration steps of Bi-CGSTAB and 46% of Bi-CGSTAB2.

7. Summary. In view of more stable convergence behavior and little computational work and low storage costs, we emphasize that the polynomial H_n generated by the three-term recurrence relation (3.4) is better than the others which come from such truncated iterative methods as GMRES(k)($k > 2$) for the requirement (1) described at the end of section 2.

In this paper, we have provided a way to see how CGS, Bi-CGSTAB, Bi-CGSTAB2, and GPBi-CG fit into a more general framework. From our experiments we have learned that GPBi-CG may be an attractive method in comparison with CGS, Bi-CGSTAB, and Bi-CGSTAB2 in many situations. Therefore, we conclude that GPBi-CG is a very competitive method for solving nonsymmetric linear systems with complex spectrum.

Although we omitted any detailed description of preconditioning techniques from this paper, all variants mentioned previously can be combined with the efficient preconditioning techniques, such as incomplete LU factorizations. For more detailed discussion on such studies see, e.g., [9, 10, 16].

Acknowledgments. The author is grateful to Dr. S. Fujino for his helpful discussions. The author also wishes to thank Dr. K. Kuwahara for the invitation to visit the Institute of Computational Fluid Dynamics, and for providing nice working conditions.

REFERENCES

[1] A. Bayliss, C. I. Glodstein, and E. Turkel, *An iterative method for the Helmholtz equation*, J. Comput. Phys., 49 (1983), pp. 443–457.

[2] M. Driessen and H. A. van der Vorst, *Bi-CGSTAB in semiconductor modeling*, in Simulation of Semiconductor Devices and Processes 4, W. Fichtner, ed., Hartung-Gorre, Zurich, 1991, pp. 45–54.

[3] S. C. Eisenstat, H. C. Elman, and M. H. Schultz, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983), pp. 345–357.

[4] R. Fletcher, *Conjugate gradient methods for indefinite systems*, Lecture Notes in Math. 506, Springer-Verlag, Berlin, Heidelberg, New York, 1976, pp. 73–89.

[5] M. H. Gutknecht, *Variants of Bi-CGSTAB for matrices with complex spectrum*, SIAM J. Sci. Comput., 14 (1993), pp. 1020–1033.

[6] S. Fujino and S. L. Zhang, *Analysis on convergence behavior of the CGS and Bi-CGSTAB method*, in Computer Arithmetic and Enclosure Methods (IMACS), L. Atanassova, ed., North–Holland, New York, 1992, pp. 381–390.

[7] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–435.

[8] C. Lanczos, *Solution of systems of linear equations by minimized iterations*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 33–53.

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

[10] J. A. Meijerink and H. A. van der Vorst, *Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems*, J. Comput. Phys., 44 (1981), pp. 134–155.

[11] N. M. Nachtigal, S. C. Reddy, and L. N. Trefethen, *How fast are nonsymmetric matrix iterations?*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 778–795.

[12] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[13] G. L. G. Sleijpen and D. R. Fokkema, *BICGSTAB(L) for linear equations involving unsymmetric matrices with complex spectrum*, Elec. Trans. Numer. Anal., 1 (1993), pp. 11–32.

[14] P. Sonneveld, *CGS, A fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[15] E. L. Stiefel, *Kernel polynomials in linear algebra and their numerical applications*, in Further Contributions to the Determination of Eigenvalues, NBS Applied Math. Ser. 49, U. S. Government Printing Office, Washington, D.C., 1958, pp. 1–22.

[16] H. A. van der Vorst, *Preconditioning by Incomplete Decompositions*, Ph.D. thesis, University of Utrecht, The Netherlands, December 16, 1982.

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

[18] S. L. Zhang and S. Fujino, *Observations on convergence behaviours of the Bi-CG, CGS, and Bi-CGSTAB from separating rounding errors*, Trans. of Japan SIAM, 3 (1993), pp. 135–146.