Applied Numerical Mathematics 60 (2010) 1100–1114

Contents lists available at ScienceDirect

Applied Numerical Mathematics

www.elsevier.com/locate/apnum

Bi-CGSTAB as an induced dimension reduction method

Gerard L.G. Sleijpen [a,*], Peter Sonneveld [b], Martin B. van Gijzen [b]

[a] Mathematical Institute, Utrecht University, PO Box 80010, 3508 TA Utrecht, The Netherlands
[b] Delft Institute of Applied Mathematics, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands

ARTICLE INFO

Article history:
Received 30 June 2008
Received in revised form 25 June 2009
Accepted 4 July 2009
Available online 9 July 2009

MSC:
65F15
65N25

Keywords:
Bi-CGSTAB
Bi-CG
Iterative linear solvers
Krylov subspace methods
IDR

ABSTRACT

The Induced Dimension Reduction method [P. Wesseling, P. Sonneveld, Numerical experiments with a multiple grid- and a preconditioned Lanczos type method, in: Lecture Notes in Mathematics, vol. 771, Springer-Verlag, Berlin, 1980, pp. 543–562] was proposed in 1980 as an iterative method for solving large nonsymmetric linear systems of equations. IDR can be considered as the predecessor of methods like CGS (Conjugate Gradient Squared) [P. Sonneveld, CGS, a fast Lanczos-type solver for nonsymmetric linear systems, SIAM J. Sci. Statist. Comput. 10 (1989) 36–52] and Bi-CGSTAB (Bi-Conjugate Gradients STABilized [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 (2) (1992) 631–644]). All three methods are based on efficient short recurrences. An important similarity between the methods is that they use orthogonalizations with respect to a fixed 'shadow residual'. Of the three methods, Bi-CGSTAB has gained the most popularity, and is probably still the most widely used short recurrence method for solving nonsymmetric systems.

Recently, Sonneveld and van Gijzen revived the interest for IDR. In [P. Sonneveld, M. van Gijzen, IDR(s): a family of simple and fast algorithms for solving large nonsymmetric systems of linear equations, Preprint, Delft Institute of Applied Mathematics, Delft University of Technology, Delft, The Netherlands, March 2007], they demonstrate that a higher dimensional shadow space, defined by the $n \times s$ matrix $\tilde{\mathbf{R}}_0$, can easily be incorporated into IDR, yielding a highly effective method.

The original IDR method is closely related to Bi-CGSTAB. It is therefore natural to ask whether Bi-CGSTAB can be extended in a way similar to IDR. To answer this question we explore the relation between IDR and Bi-CGSTAB and use our findings to derive a variant of Bi-CGSTAB that uses a higher dimensional shadow space.

1. Introduction

Transpose free Bi-CG (bi-conjugate gradients) methods, as CGS [9] and Bi-CGSTAB [11], also referred to as hybrid Bi-CG methods, are among the most popular iterative methods for solving sparse high dimension linear systems of equations

$$\mathbf{Ax} = \mathbf{b}.$$

Here, $\mathbf{A}$ is a given $n \times n$ matrix and $\mathbf{b}$ is a given vector. As GMRES, these methods try to find appropriate approximate solutions in Krylov subspaces $\mathbf{x}_0 + \mathcal{K}_k(\mathbf{A}, \mathbf{r}_0)$ generated by $\mathbf{A}$ and initial residual $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$. Unlike GMRES, they do not find the approximate solutions with smallest residual norm. But, in contrast to GMRES, these methods use short recurrences, and

* Corresponding author.
E-mail addresses: sleijpen@math.uu.nl (G.L.G. Sleijpen), P.Sonneveld@tudelft.nl (P. Sonneveld), M.B.vanGijzen@tudelft.nl (M.B. van Gijzen).

0168-9274/$30.00 © 2009 IMACS. Published by Elsevier B.V. All rights reserved.
doi:10.1016/j.apnum.2009.07.001

as a result are often much more efficient, both with respect to memory and to computations, for problems where GMRES needs many iterations to find a sufficiently accurate solution.

Krylov subspace methods grow in each iteration. This makes it possible to construct increasingly better approximate solutions using a suitable selection criterion. Efforts of finding appropriate approximate solutions have mainly focused on constructing residuals with small or smallest norm. For instance, for symmetric systems (i.e. $\mathbf{A}^* = \mathbf{A}$), CR (conjugate residuals) constructs residuals with smallest Euclidean norm, while the residuals for CG (conjugate gradients) have smallest $\mathbf{A}^{-1}$-norm. Bi-CG has been viewed as a CG process with respect to a quadratic form rather than an inner-product, and residuals were considered to be 'minimized' with respect to this quadratic form.

Bi-CGSTAB is a combination of two methods, of Bi-CG and restarted GMRES (or GCR). In the restarted GMRES part, a residual is minimized in each step, in the Bi-CG part, the Bi-CG process is incorporated. The focus in Bi-CGSTAB is on preserving the Bi-CG coefficients, hoping that the nice features of Bi-CG (read 'residual minimization') are transferred to Bi-CGSTAB.

An alternative approach has been taken in [12] in the construction of the IDR (induced dimension reduction) method, 'squeezing' the residuals to zero. The IDR method finds residuals in a sequence of *shrinking subspaces*. Bi-CG can also be viewed in such a perspective. Bi-CG residuals belong to a sequence of growing Krylov subspaces, but they also belong to a sequence of shrinking subspaces: Bi-CG uses so-called 'shadow' Krylov subspaces $\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{r}}_0)$ for testing, that is, the residuals are constructed to be *orthogonal to* this sequence of growing shadow spaces. The starting point for the IDR construction is more abstract, more elegant than for Bi-CG. Residuals are constructed in spaces $\mathcal{G}_k$ that are defined recursively by $\mathcal{G}_{k+1} = (\mathbf{I} - \omega_k \mathbf{A})(\mathcal{G}_k \cap \mathcal{S})$ (see Section 2). Here, $\mathcal{S}$ is a fixed proper subspace of $\mathbb{C}^n$. If $\mathbf{A}$ has no eigenvector that is orthogonal to the subspace $\mathcal{S}$, then $\mathcal{G}_{k+1} \subset \mathcal{G}_k$ and the dimension of $\mathcal{G}_k$ reduces with increasing k (see also Theorem 7 below).

The subspace $\mathcal{S}$ in the original IDR paper [12] from 1980 consists of all vectors orthogonal to some (random) vector $\tilde{\mathbf{r}}_0$. The method turned out to be very effective. In fact, as effective as the more recent Bi-CGSTAB method (from 1992). Unfortunately, the original formulation of the IDR algorithm contained some instabilities, and IDR did not gain the popularity of Bi-CGSTAB. In a recent paper [10], it was shown that the IDR concept allows elegant incorporation of a smaller fixed subspace $\mathcal{S}$, or, equivalently, of a space of vectors orthogonal to all columns of an $n \times s$ matrix $\tilde{\mathbf{R}}_0$ with $s > 1$. The idea to use a subspace $\mathcal{S}$ with co-dimension s combined with a more stable computation of the residuals and approximate solutions led to the IDR(s) algorithm. For $s = 1$, this algorithm is mathematically equivalent to the original IDR method, but has superior numerical stability. Moreover, the numerical experiments in [10] show that, for modest values of s, as $s = 4$, even for 'tough' linear equations, IDR(s) often achieves almost the same convergence as GMRES, that is, comparable accuracy for the same number of MVs. However, the additional costs per MV in IDR are modest and limited (as with Bi-CGSTAB), whereas these costs in GMRES grow proportional to the iteration step.

In Section 4 of this paper, the IDR principle is formulated in terms of Krylov subspaces. Although this formulation is less elegant, it may provide more insight, since it facilitates comparison of IDR and Bi-CGSTAB type of methods, and it may lead to even more effective variants. In Section 5, we give an alternative proof from the one [10, §5.2] of the mathematical equivalence of IDR(s) with $s = 1$ and Bi-CGSTAB. The proof here gives a more detailed relation and also serves as an introduction to the subsequential sections. Via a block variant of Bi-CG in Section 6, we explain in Section 7 how the case $s > 1$ can be incorporated in Bi-CGSTAB as well, yielding a variant that is equivalent to IDR(s) (see Section 8).

Our block variant of Bi-CG uses blocks of size $p = 1$ (that is, n-vectors) in the search Krylov subspace, and blocks of size s (that is, $n \times s$ matrices) for testing. The focus in literature on block methods is mainly on the case where the size p of the blocks for searching and the size s of the blocks for testing are equal ($p = s$, cf., e.g., [5,4,6]). A general block Lanczos version (with $p \neq s$) has been discussed in [1]. There is no Bi-CGSTAB version. The case where $p > 1$ and $s = 1$ for Bi-CGSTAB has been discussed in [3]. The purpose of all these papers is to develop methods for solving block systems, also called *multiple* right-hand side system, i.e., systems where $\mathbf{b}$ is an $n \times p$ matrix with $p > 1$. IDR(s), however, aims for fast convergence for the *single* right-hand side case (i.e., $p = 1$) by using higher dimensional blocks ($s > 1$) for testing. One paper [13] works in precisely our setting ($p = 1, s > 1$). This interesting paper generalizes the Bi-CG polynomial recursions to derive the block Bi-CGSTAB variant ML(k)BiCGSTAB. Unfortunately, this derivation leads to complicated formula's, which makes the exposition rather intransparant. Also, their experiments concentrate on high values of s (e.g. $s = 100$), where, in our experience, only low values of s are attractive. We feel that the subspace formulation of IDR is more elegant and more flexible. In this paper we bridge the subspace approach and the polynomial one.

The purpose of this paper is to provide insight. Therefore, we focus on mathematical concepts rather than on technical details. The new algorithms that we describe in this paper should therefore not be considered as matured, but as bridging steps between the Bi-CGSTAB approach and the IDR approach, with the ultimate goals to improve both types of algorithms.

1.1. Some remarks on the notation

Notation 1. As a general rule, we will use notations and terminology that are common in the literature about Bi-CG methods. Our notation is therefore different from the one used in [10]. For example, we will use the term 'shadow space' for the space spanned by the columns of a matrix $\tilde{\mathbf{R}}$. This terminology and notation is linked to the 'shadow residuals' in Bi-CG. In IDR, $\mathcal{S}$ is the left-Null space of $\tilde{\mathbf{R}}$, and is unrelated to the concept of a shadow residual.

Notation 2. If $\tilde{\mathbf{R}}$ is an $n \times s$ matrix, and $\mathbf{v}$ is a vector, then we put $\mathbf{v} \perp \tilde{\mathbf{R}}$ if $\mathbf{v}$ is orthogonal to all column vectors of $\tilde{\mathbf{R}}$ and we say that $\mathbf{v}$ is *orthogonal to* $\tilde{\mathbf{R}}$. The linear subspace of all vectors $\mathbf{v}$ that are orthogonal to $\tilde{\mathbf{R}}$ is denoted by $\tilde{\mathbf{R}}^{\perp}$.

Notation 3. When we mention the number of iterations, or the number of steps, we always refer to the number of matrix–vector multiplications (MVs), where each step or iteration always corresponds to one (one-dimensional) MV. A multiplication $\mathbf{AV}$, with $\mathbf{V}$ an $n \times s$ matrix is counted as s MVs.

Notation 4. In the paper we use sometimes IDR and sometimes IDR(s). When we use the latter, we always refer to the specific prototype-IDR method that is described in [10]. The more general IDR refers to any IDR-type method, i.e., a method that constructs residuals in subspaces $\mathcal{G}_k$ of shrinking dimension.

Notation 5. Updates of the form $\mathbf{v} - \mathbf{C}\vec{\beta}$ will play a crucial role in this paper. Here, $\mathbf{v}$ is an n-vector and $\mathbf{C}$ is an $n \times s$ matrix. When considering such updates, both $\mathbf{v}$ and $\mathbf{C}$ are available. Often, the s-vector $\vec{\beta}$ is determined by an orthogonality requirement $\mathbf{v} - \mathbf{C}\vec{\beta} \perp \tilde{\mathbf{R}}$ where $\tilde{\mathbf{R}}$ is some given $n \times s$ matrix. For ease of notation, we will simply put

"$\mathbf{v} - \mathbf{C}\vec{\beta} \perp \tilde{\mathbf{R}}$" if we mean "Let $\vec{\beta}$ be such that $\mathbf{v} - \mathbf{C}\vec{\beta} \perp \tilde{\mathbf{R}}$".

Note that, with $\sigma \equiv \tilde{\mathbf{R}}^*\mathbf{C}$, $\vec{\beta}$ can be computed as $\vec{\beta} = \sigma^{-1}(\tilde{\mathbf{R}}^*\mathbf{v})$ (or with a more stable variant as repeated or modified). The operator $\mathbf{I} - \mathbf{C}\sigma^{-1}\tilde{\mathbf{R}}^*$ is a skew projection onto the orthogonal complement of $\tilde{\mathbf{R}}$.

2. The IDR principle

The IDR (induced dimension reduction) method finds residual vectors in a sequence $(\mathcal{G}_k)$ of shrinking subspaces, i.e., $\mathcal{G}_{k+1} \subset \mathcal{G}_k$ and $\dim(\mathcal{G}_{k+1}) < \dim(\mathcal{G}_k)$ for all $k \in \mathbb{N}_0$ unless $\mathcal{G}_k = \{\mathbf{0}\}$. The existence of such a sequence is guaranteed by the fundamental IDR theorem that we repeat here (Theorem 7) for ease of explanation (see also [10, Theorem 1]). The theorem is an easy consequence of the following lemma.

Lemma 6. *Let $\mathcal{G}_0$ and $\mathcal{S}$ be subspaces and $\mu_0 \in \mathbb{C}$.*
Put $\mathcal{G}_1 \equiv (\mu_0\mathbf{I} - \mathbf{A})(\mathcal{G}_0 \cap \mathcal{S})$.[1]

(1) *If $\mathcal{G}_1 \subset \mathcal{G}_0$ then $(\mu_1\mathbf{I} - \mathbf{A})(\mathcal{G}_1 \cap \mathcal{S}) \subset \mathcal{G}_1$.*
(2) *If $\mathcal{G}_1 = \mathcal{G}_0 \neq \{\mathbf{0}\}$, then $\mathcal{G}_0 \cap \mathcal{S}$ contains an eigenvector of $\mathbf{A}$.*

Proof. Note that $(\mu_1\mathbf{I} - \mathbf{A})(\mathcal{G}_1 \cap \mathcal{S}) \subset \mathcal{G}_1$ iff $(\mu_0\mathbf{I} - \mathbf{A})(\mathcal{G}_1 \cap \mathcal{S}) \subset \mathcal{G}_1$. Therefore, (1) follows from the inclusion $(\mu_0\mathbf{I} - \mathbf{A})(\mathcal{G}_1 \cap \mathcal{S}) \subset (\mu_0\mathbf{I} - \mathbf{A})(\mathcal{G}_0 \cap \mathcal{S}) = \mathcal{G}_1$.

If $\mathcal{G}_0 = \mathcal{G}_1$ then $\dim(\mathcal{G}_0) = \dim((\mu_0\mathbf{I} - \mathbf{A})(\mathcal{G}_0 \cap \mathcal{S})) \leqslant \dim(\mathcal{G}_0 \cap \mathcal{S}) \leqslant \dim(\mathcal{G}_0)$. In particular, $\dim(\mathcal{G}_0 \cap \mathcal{S}) = \dim(\mathcal{G}_0)$, whence $\mathcal{G}_0 \subset \mathcal{S}$ and $\mathcal{G}_0 = (\mu_0\mathbf{I} - \mathbf{A})(\mathcal{G}_0 \cap \mathcal{S}) = (\mu_0\mathbf{I} - \mathbf{A})\mathcal{G}_0$, which implies that $\mu_0\mathbf{I} - \mathbf{A}$, and therefore $\mathbf{A}$, has an eigenvector in $\mathcal{G}_0$ unless $\mathcal{G}_0 = \{\mathbf{0}\}$. □

The IDR theorem in [10] has been formulated in terms of subspaces of a linear subspace $\mathcal{S}$. As we will explain in Note 3 below, it is convenient to formulate the theorem in terms of a complement space $\mathcal{S} = \tilde{\mathbf{R}}_0^{\perp}$.

Theorem 7. *Let $\tilde{\mathbf{R}}_0 = [\tilde{\mathbf{r}}_1, \ldots, \tilde{\mathbf{r}}_s]$ be an $n \times s$ matrix and let (μ_j) be a sequence in $\mathbb{C}$. With $\mathcal{G}_0 \equiv \mathbb{C}^n$, define,*

$$\mathcal{G}_{k+1} \equiv (\mu_k\mathbf{I} - \mathbf{A})\big(\mathcal{G}_k \cap \tilde{\mathbf{R}}_0^{\perp}\big) \quad (k = 0, 1, \ldots). \tag{1}$$

If $\tilde{\mathbf{R}}_0^{\perp}$ does not contain an eigenvector of $\mathbf{A}$, then, for all $k = 0, 1, \ldots$, we have that

(1) *$\mathcal{G}_{k+1} \subset \mathcal{G}_k$, and*
(2) *$\dim \mathcal{G}_{k+1} < \dim \mathcal{G}_k$ unless $\mathcal{G}_k = \{\mathbf{0}\}$.*

Proof. Take $\mathcal{S} = \tilde{\mathbf{R}}_0^{\perp}$ and apply the lemma inductively. □

Note 1. The theorem is correct for any linear subspace $\mathcal{G}_0$ that is invariant under multiplication by $\mathbf{A}$: $\mathbf{A}\mathcal{G}_0 \subset \mathcal{G}_0$. In particular, the theorem is correct if $\mathcal{G}_0$ is a *full* Krylov subspace $\mathcal{K}(\mathbf{A}, \mathbf{v}) \equiv \mathrm{Span}\{\mathbf{A}^k\mathbf{v} \mid k = 0, 1, \ldots\}$.

[1] We use this slightly more elegant formulation instead of the equivalent $\mathcal{G}_1 \equiv (\mathbf{I} - \omega_0\mathbf{A})(\mathcal{G}_0 \cap \mathcal{S})$ to avoid the condition $\omega_0 \neq 0$. In the description of the algorithms we use the latter.

3. The IDR(*s*) algorithm

The IDR(*s*) algorithm (cf. [10]), that we discuss in this section, constructs residuals in the spaces $\mathcal{G}_k$. The algorithm updates the residuals in each step using short recurrences. The following three ideas, in all of which residuals play a central role, are exploited.

In this section, $\tilde{\mathbf{R}}_0$ is an $n \times s$ matrix of full rank.

1. *Providing a cheap ride for approximate solutions*. As in Bi-CG, and in hybrid Bi-CG methods as Bi-CGSTAB, the updates for an IDR residual $\mathbf{r}$ are of the form $\mathbf{r}_+ = \mathbf{r} - \mathbf{c}\alpha$, with $\mathbf{c} = \mathbf{A}\mathbf{u}$ and the vector $\mathbf{u}$ explicitly available. This allows to update the associated approximate solution $\tilde{\mathbf{x}}$, $\mathbf{r} = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}$, at an additional cost of one AXPY (vector update) only: with $\tilde{\mathbf{x}}_+ \equiv \tilde{\mathbf{x}} + \mathbf{u}\alpha$ we have that $\mathbf{r}_+ = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}_+$. The MV (matrix–vector multiplication) $\mathbf{c} = \mathbf{A}\mathbf{u}$ and possibly one (or a few) DOT product(s) to compute the scalar α where needed to find the update for $\mathbf{r}$. The approximate solutions get a (n almost) free ride. Of course, a number of vector updates can be combined: if $\mathbf{r}_+ = \mathbf{r} - \mathbf{C}\vec{\alpha}$ with $\mathbf{C} = \mathbf{A}\mathbf{U}$ and the $n \times s$ matrix $\mathbf{U}$ explicitly available, then $\tilde{\mathbf{x}}_+ = \tilde{\mathbf{x}} + \mathbf{U}\vec{\alpha}$. Note that a column of $\mathbf{U}$ can be updated as $\mathbf{U}\vec{\gamma} + \omega\mathbf{v}$ if the corresponding column of $\mathbf{C}$ is updated as $\mathbf{C}\vec{\gamma} + \omega\mathbf{A}\mathbf{v}$. Keeping these observations in mind, we will concentrate on the residuals and updates of the form $\mathbf{A}\mathbf{u}$, in the discussion and the derivation of the algorithms (for IDR as well as for Bi-CG and Bi-CGSTAB) in the sequel of this paper.
2. *Bi-orthogonalization*. To move from $\mathbf{r} \in \mathcal{G}_k$ to $\mathbf{r}_+ \in \mathcal{G}_{k+1}$, we first construct a residual vector $\mathbf{v}$ in $\mathcal{G}_k \cap \tilde{\mathbf{R}}_0^{\perp}$ by subtracting a vector of the form $\mathbf{S}\vec{\gamma}$ from $\mathbf{r}$: with $\vec{\gamma} = (\tilde{\mathbf{R}}_0^*\mathbf{S})^{-1}\tilde{\mathbf{R}}_0^*\mathbf{r}$, we take $\mathbf{v} = \mathbf{r} - \mathbf{S}\vec{\gamma}$ (and $\mathbf{S}$ of the form $\mathbf{A}\mathbf{U}$). Then, for some appropriate scalar ω, we multiply $\mathbf{v}$ by $\mathbf{I} - \omega\mathbf{A}$: $\mathbf{r}_+ = \mathbf{v} - \omega\mathbf{A}\mathbf{v}$.
3. *Residual differences*. The skew projection in idea 2 assumes a non-singular matrix $\tilde{\mathbf{R}}_0^*\mathbf{S}$. In particular, the matrix $\mathbf{S}$ has to be $n \times s$. We construct our matrix $\mathbf{S}$ as a difference of $s+1$ residuals. Note that $\mathbf{r}_+ - \mathbf{r}$ is of the form $\mathbf{A}\mathbf{u}$ with $\mathbf{u} = \tilde{\mathbf{x}} - \tilde{\mathbf{x}}_+$ if $\mathbf{r} = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}$ and $\mathbf{r}_+ = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}_+$. (Note that $\mathbf{v} - \mathbf{r} = \mathbf{S}\vec{\gamma}$ is in the span of the columns of $\mathbf{S}$ and does not introduce new information, whereas $\mathbf{r}_+ - \mathbf{r}$ introduces the 'new' vector $\mathbf{A}\mathbf{v}$.) If $\tilde{\mathbf{R}}_0^*\mathbf{S}$ happens to be singular or ill-conditioned, we have to reduce s. For details, see [10, §4]. See also Note 4. We will not give details here: the purpose of this paper is to give insight on the relation between IDR and Bi-CGSTAB methods.

The following two propositions express the effectiveness of these ideas.

Proposition 8. *Suppose* $\mathbf{s}_i = \mathbf{A}\mathbf{u}_i$ *for* $i < j$, $\mathbf{r}_j = \mathbf{b} - \mathbf{A}\mathbf{x}_j$.
Put $\mathbf{U}_j \equiv [\mathbf{u}_{j-s}, \ldots, \mathbf{u}_{j-1}]$ *and* $\mathbf{S}_j \equiv [\mathbf{s}_{j-s}, \ldots, \mathbf{s}_{j-1}]$. *Select an* $\omega \in \mathbb{C}$. *If*

$$\begin{aligned} &\mathbf{v} \equiv \mathbf{r}_j - \mathbf{S}_j\vec{\gamma}, \qquad \mathbf{r}_{j+1} = \mathbf{v} - \omega\mathbf{A}\mathbf{v}, \qquad \mathbf{s}_j \equiv \mathbf{r}_j - \mathbf{r}_{j+1} \\ &\mathbf{u}_k \equiv \mathbf{U}_k\vec{\gamma} + \omega\mathbf{v}, \qquad \mathbf{x}_{j+1} \equiv \mathbf{x}_j + \mathbf{u}_j, \end{aligned} \tag{2}$$

then $\mathbf{s}_j = \mathbf{A}\mathbf{u}_j$ *and* $\mathbf{r}_{j+1} = \mathbf{b} - \mathbf{A}\mathbf{x}_{j+1}$.

Note that the proposition offers four alternatives for computing $\mathbf{s}_j$. Computing $\mathbf{s}_j \equiv \mathbf{S}_j\vec{\gamma} + \omega\mathbf{A}\mathbf{v}$ and $\mathbf{u}_j \equiv \mathbf{U}_j\vec{\gamma} + \omega\mathbf{v}$ requires $2s+2$ vector updates, whereas the alternative $\mathbf{u}_j \equiv \mathbf{U}_j\vec{\gamma}_j + \omega\mathbf{v}$ and $\mathbf{s}_j = \mathbf{A}\mathbf{u}_j$ requires only $s+1$ vector updates. Moreover, in our experience this way of updating is numerically more stable than the first alternative. In both cases $\mathbf{r}_{j+1}$ can be computed as $\mathbf{r}_{j+1} = \mathbf{r}_j - \mathbf{s}_j$. If the computation of ω relies on $\mathbf{A}\mathbf{v}$, e.g., by minimizing the norm of $\mathbf{r}_{j+1} = \mathbf{v} - \omega\mathbf{A}\mathbf{v}$, then the second alternative is less attractive since it would require 2MVs. For maintaining accuracy, it can be useful to compute $\mathbf{r}_{j+1}$ at strategically selected values for j (and if ω is available) as $\mathbf{r}_{j+1} = \mathbf{b} - \mathbf{A}\mathbf{x}_{j+1}$ (see [8]), and $\mathbf{s}_j$ can be obtained as $\mathbf{s}_j = \mathbf{r}_j - \mathbf{r}_{j+1}$.

Proposition 9. *Let* $\mathcal{G}_{k+1} = (\mathbf{I} - \omega\mathbf{A})(\mathcal{G}_k \cap \tilde{\mathbf{R}}_0^{\perp})$.
Suppose $\mathrm{Span}(\mathbf{S}_j) \subset \mathcal{G}_k$ *and* $\mathbf{r}_j \in \mathcal{G}_{k+i}$ *for* i *either* 0 *or* 1.
If $\mathbf{v}, \mathbf{s}_j$ *and* $\mathbf{r}_{j+1}$ *are as defined in* (2) *with* $\vec{\gamma}$ *such that* $\mathbf{v} \perp \tilde{\mathbf{R}}_0$, *then*

$$\mathbf{v} \in \mathcal{G}_k \cap \tilde{\mathbf{R}}_0^{\perp}, \quad \mathbf{r}_{j+1} \in \mathcal{G}_{k+1}, \quad \mathbf{s}_k \in \mathcal{G}_{k+i}.$$

The proposition shows that in $s+1$ steps of (2) $s+1$ residuals in $\mathcal{G}_k$ (and s differences in $\mathcal{G}_k$) can be 'lifted' to $s+1$ residuals in $\mathcal{G}_{k+1}$. At step 1 of a cycle of $s+1$ steps we are free to select an ω. In the other s steps of the cycle, we use the same ω. We follow the Bi-CGSTAB strategy for selecting the 'free' ω; we minimize the norm of $\mathbf{v} - \omega\mathbf{A}\mathbf{v}$.

Algorithm 1 displays the resulting algorithm.

The initialization requires an $n \times s$ matrix $\mathbf{U}$. For ease of discussion, we will assume in the following sections that $\mathbf{U} = [\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^{s-1}\mathbf{r}_0]$. However, the matrix $\mathbf{U}$ (and $\mathbf{S}$) can also be constructed by s steps of a method as GCR [2]. The approximate solution $\mathbf{x}$ and associated residual $\mathbf{r}$ can also be updated in these s steps. Then, the initial $\mathbf{x}$ and initial $\mathbf{r}$ mentioned in the initialization of Algorithm 1 can be selected as the results of these s steps of GCR. Note that, in the GCR approach, the ith column of $\mathbf{S} = \mathbf{A}\mathbf{U}$ is a multiple of the difference of the ith and the $(i-1)$th GCR residual: the columns of $\mathbf{S}$ are differences of residuals according to idea 3.

```
Select an x_0.
Select n × s matrices R̃_0 and U
Compute S = AU
x = x_0,  r = b − Ax
i = 1,  j = 0
while ‖r‖ > tol
  Solve R̃_0^* S γ⃗ = R̃_0^* r for γ⃗
  v = r − S γ⃗,  c = Av
  If j = 0,  ω ← c^* v / c^* c,
  U e_i ← U γ⃗ + ω v,  x ← x + U e_i
  r_1 ← v − ω c,  S e_i ← r − r_1,  r ← r_1
  i ← i + 1,  if i > s,  i = 1
  j ← j + 1,  if j > s,  j = 0
end while
```

Algorithm 1. (IDR(s).) $\mathbf{S} = [\mathbf{s}_{k-i+1}, \ldots, \mathbf{s}_{k-1}, \mathbf{s}_{k-s}, \mathbf{s}_{k-s-1}, \ldots, \mathbf{s}_{k-i}]$ at the start of the kth loop. Here, the $\mathbf{s}_k$s are as in (2): the ith column $\mathbf{S}e_i$ of $\mathbf{S}$ is equal to $\mathbf{s}_{k-s}$, which is replaced in the loop by the newly computed $\mathbf{s}_k$. The matrix $\mathbf{U}$ has a similar relation with the $\mathbf{u}_k$ of (2) and follows a similar update strategy.

The initial matrices $\mathbf{U}$ and $\tilde{\mathbf{R}}_0$ have to be such that $\tilde{\mathbf{R}}_0^*\mathbf{S}$ is non-singular.

In Algorithm 1, we suggest to replace the 'oldest' column of $\mathbf{U}$ and $\mathbf{S}$ by the 'newest', rather than deleting the first column and adding the new column as last column as suggested Proposition 8.

4. IDR and Krylov subspaces

The subspace $\mathcal{G}_k$ in Theorem 7 can also be formulated in terms of Krylov subspaces as we will see in the following theorem. The Krylov subspaces in this theorem are generated by an $n \times s$ matrix rather than by an n-vector (that is, an $n \times 1$ matrix):

Definition 10. The Krylov subspace $\mathcal{K}_k(\mathbf{B}, \tilde{\mathbf{R}})$ of order k generated by an $n \times n$ matrix $\mathbf{B}$ and an $n \times s$ matrix $\tilde{\mathbf{R}}$ is given by

$$\mathcal{K}_k(\mathbf{B}, \tilde{\mathbf{R}}) \equiv \left\{ \sum_{j=0}^{k-1} \mathbf{B}^j \tilde{\mathbf{R}} \vec{\gamma}_j \;\middle|\; \vec{\gamma}_j \in \mathbb{C}^s \right\}. \tag{3}$$

In case $s = 1$, we have the usual Krylov subspaces. The Krylov subspace as defined here, are also called "block Krylov subspaces" (see, e.g., [1]).

Note that the $\vec{\gamma}_j$ are s vectors that generally do not commute with the $n \times s$ matrices $\mathbf{B}^j\tilde{\mathbf{R}}$. In particular, $\sum_{j=0}^{k-1} \mathbf{B}^j \tilde{\mathbf{R}} \vec{\gamma}_j$ is not of the form $q(\mathbf{B})\tilde{\mathbf{R}}$ with q a polynomial of degree $< k$. Nevertheless, $\bar{q}(\mathbf{B}^*)\mathbf{v} \perp \tilde{\mathbf{R}}$ if $\mathbf{v} \perp \mathcal{K}_k(\mathbf{B}, \tilde{\mathbf{R}})$.

Theorem 11. *Let $\tilde{\mathbf{R}}_0$, (μ_j) and $\mathcal{G}_k$ be as in Theorem 7. Consider the polynomial p_k of degree k given by $p_k(\lambda) \equiv \prod_{j=0}^{k-1}(\mu_j - \lambda)$ $(\lambda \in \mathbb{C})$. Then*

$$\mathcal{G}_k = \left\{ p_k(\mathbf{A})\mathbf{v} \;\middle|\; \mathbf{v} \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0) \right\}. \tag{4}$$

Proof. The claim for $k = 0$ is trivial. We use an induction argument to prove the theorem. Assume that (4) is correct.

Then,

$$\mathcal{G}_k \cap \tilde{\mathbf{R}}_0^{\perp} = \left\{ p_k(\mathbf{A})\mathbf{v} \;\middle|\; p_k(\mathbf{A})\mathbf{v} \perp \tilde{\mathbf{R}}_0, \mathbf{v} \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0) \right\}.$$

Since $\mathbf{v} \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$, we have that $q(\mathbf{A})\mathbf{v} \perp \tilde{\mathbf{R}}_0$ for all polynomials q of degree $< k$. Hence, $p_k(\mathbf{A})\mathbf{v} \perp \tilde{\mathbf{R}}_0$ if and only if $\mathbf{A}^k\mathbf{v} \perp \tilde{\mathbf{R}}_0$, which is equivalent to $\mathbf{v} \perp (\mathbf{A}^*)^k\tilde{\mathbf{R}}_0$, or, equivalently, $\mathbf{v} \perp (\mathbf{A}^*)^k\tilde{\mathbf{R}}_0\vec{\gamma}_k$ for all s vectors $\vec{\gamma}_k$. Apparently,

$$\mathcal{G}_k \cap \tilde{\mathbf{R}}_0^{\perp} = \left\{ p_k(\mathbf{A})\mathbf{v} \;\middle|\; \mathbf{v} \perp \mathcal{K}_{k+1}(\mathbf{A}^*, \tilde{\mathbf{R}}_0) \right\},$$

whence

$$\mathcal{G}_{k+1} = (\mu_k\mathbf{I} - \mathbf{A})\left(\mathcal{G}_k \cap \tilde{\mathbf{R}}_0^{\perp}\right) = \left\{ (\mu_k\mathbf{I} - \mathbf{A})p_k(\mathbf{A})\mathbf{v} \;\middle|\; \mathbf{v} \perp \mathcal{K}_{k+1}(\mathbf{A}^*, \tilde{\mathbf{R}}_0) \right\}.$$

Since $(\mu_k\mathbf{I} - \mathbf{A})p_k(\mathbf{A}) = p_{k+1}(\mathbf{A})$ this proves the theorem. □

The theorem suggests that IDR and Bi-CGSTAB are related. Before we go into details on this in Section 5, we first discuss some obvious consequences.

Note 2. If $\mathcal{G}_0$ is a linear subspace that is invariant under multiplication by $\mathbf{A}$, then $\mathcal{G}_k = \{p_k(\mathbf{A})\mathbf{v} \mid \mathbf{v} \in \mathcal{G}_0, \mathbf{v} \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)\}$.

Note 3. In the proof of the theorem, we used that fact that if $\mathbf{Av} \perp \tilde{\mathbf{R}}_0$ and $\mathbf{v} \perp \mathcal{K}$ for some linear subspace $\mathcal{K}$ then $\mathbf{v}$ is orthogonal to the subspace $\mathrm{Span}(\mathbf{A}^*\tilde{\mathbf{R}}_0) \oplus \mathcal{K}$. The analogue expression in case $\tilde{\mathbf{R}}_0^{\perp}$ is given as a linear subspace $\mathcal{S}$ (i.e., if $\mathbf{Av} \in \mathcal{S}$ and $\mathbf{v} \in \tilde{\mathcal{K}}$ then $\mathbf{v} \in \ldots$) is less elegant.

The existence of an eigenvector orthogonal to $\tilde{\mathbf{R}}_0$ in Theorem 7 can also be expressed in terms of the "shadow" Krylov subspace $\mathcal{K}(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$, where

$$\mathcal{K}(\mathbf{A}^*, \tilde{\mathbf{R}}_0) \equiv \bigcup_{k=0}^{\infty} \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0).$$

Proposition 12. *The following three statements are equivalent*:

(a) *There exists an eigenvector of* $\mathbf{A}$ *that is orthogonal to* $\tilde{\mathbf{R}}_0$.
(b) $\mathcal{K}(\mathbf{A}^*, \tilde{\mathbf{R}}_0) \neq \mathbb{C}^n$.
(c) *There is a non-trivial vector* $\mathbf{x}$ *such that* $\mathcal{K}(\mathbf{A}, \mathbf{x}) \perp \tilde{\mathbf{R}}_0$.

The characterization of $\mathcal{G}_k$ in Theorem 11 leads to an alternative proof of (1) of Theorem 7. The proof that

$$\mathcal{G}_{k+1} = \left\{(\mu_k \mathbf{I} - \mathbf{A})p_k(\mathbf{A})\mathbf{v} \;\middle|\; \mathbf{v} \perp \mathcal{K}_{k+1}(\mathbf{A}^*, \tilde{\mathbf{R}}_0)\right\} \subset \left\{p_k(\mathbf{A})\tilde{\mathbf{v}} \;\middle|\; \tilde{\mathbf{v}} \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)\right\} = \mathcal{G}_k$$

follows from the fact that $\mathbf{v} \perp \mathcal{K}_{k+1}(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$ implies that $\tilde{\mathbf{v}} \equiv (\mu_k \mathbf{I} - \mathbf{A})\mathbf{v} \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$.

The following corollary provides some insight on the decrease of the dimension of $\mathcal{G}_k$ with increasing k. The result is an immediate consequence of Theorem 11 (we leave the proof to the reader).

Corollary 13. *If* $p_k(\mathbf{A})$ *is non-singular, i.e., if none of the* μ_j *is an eigenvalue of* $\mathbf{A}$ $(j = 0, \ldots, k-1)$, *then*

$$\dim(\mathcal{G}_k) = n - \dim\big(\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)\big).$$

Note that

$$d_{k+1} - d_k \leqslant d_k - d_{k-1} \leqslant s, \quad \text{where } d_k \equiv \dim\big(\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)\big). \tag{5}$$

In general (generic case), when $ks \leqslant n$ we will have that $d_k = ks$: if the column vectors of $\tilde{\mathbf{R}}_0$ have been randomly selected, then, with probability 1, we will have that $d_k = ks$ whenever $ks < n$ and $\mathbf{A}^*$ has n linearly independent eigenvectors. However, $d_{k+1} - d_k$ can be $< s$ as the following example shows.

Example 14. If $\tilde{\mathbf{R}}_0 = [\mathbf{v}, \mathbf{A}^*\mathbf{v}, (\mathbf{A}^*)^3\mathbf{v}]$, then $d_1 = s = 3$, $d_{2+i} = 5 + i$.

5. Bi-CGSTAB and IDR in case $s = 1$

Theorem 11 suggests a relation between IDR and Bi-CGSTAB. In this section, we concentrate on the case $s = 1$. We put $\tilde{\mathbf{r}}_0$ instead of $\tilde{\mathbf{R}}_0$ (that is, we assume that $\tilde{\mathbf{R}}_0 = [\tilde{\mathbf{r}}_0]$).

Bi-CGSTAB has been introduced as a transpose free variant of Bi-CG. The kth Bi-CG residual $\mathbf{r}_k^{\text{Bi-CG}}$ is of the form

$$\mathbf{r}_k^{\text{Bi-CG}} = q_k(\mathbf{A})\mathbf{r}_0, \tag{6}$$

with q_k a polynomial of degree k such that

$$q_k(0) = 1 \quad \text{and} \quad q_k(\mathbf{A})\mathbf{r}_0 \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0). \tag{7}$$

The first property makes q_k to a 'residual polynomial', i.e., $q_k(\mathbf{A})\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_k$ for some $\mathbf{x}_k$ ($\mathbf{x}_k = \mathbf{x}_0 + \tilde{q}(\mathbf{A})\mathbf{r}_0$, where $\tilde{q}$ is such that $q(\lambda) = 1 - \lambda\tilde{q}(\lambda)$). Note that the two properties in (7) determine q_k uniquely.[2]

An auxiliary polynomial p_k of degree k is used in Bi-CGSTAB for further reduction of the Bi-CG residual:

$$\mathbf{r}_k^{\text{Bi-CGSTAB}} = p_k(\mathbf{A})\mathbf{r}_k^{\text{Bi-CG}}. \tag{8}$$

[2] For ease of explanation, we implicitly assume that k is small enough to have Krylov subspaces of full dimension: $k = \dim(\mathcal{K}_k(\mathbf{A}, \mathbf{r}_0)) = \dim(\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{r}}_0))$. The purpose of this section is to provide inside on the relation between IDR and Bi-CGSTAB: we will not discuss the consequences here of 'degenerated' Krylov subspaces.

The polynomial p_k is of the form $p_k(\lambda) = (1-\omega_k\lambda)\cdots(1-\omega_1\lambda)$, with

$$\omega_k = \text{minarg}_\omega \left\| (1-\omega\mathbf{A})\, p_{k-1}(\mathbf{A})\mathbf{r}_k^{\text{Bi-CG}} \right\|_2.$$

If the Bi-CGSTAB process does not stagnate, then the ω_j will be non-zero and, with $\mu_{j-1} \equiv 1/\omega_j$, we have that $p_k(\lambda) = \frac{1}{\mu_{k-1}\cdots\mu_0}(\mu_{k-1}-\lambda)\cdots(\mu_0-\lambda)$. Therefore, in view of (7) and Theorem 11, we can conclude that $\mathbf{r}_k^{\text{Bi-CGSTAB}}$ belongs to $\mathcal{G}_k$.

In view of the uniqueness remarks on the form of the Bi-CG residual and the selection of the ω_j in IDR and Bi-CGSTAB (to minimize residual norms), we can conclude that IDR for $s=1$ and Bi-CGSTAB are equivalent: assuming exact arithmetic, then, with the same initialization (the same $\mathbf{r}_0$ and $\tilde{\mathbf{r}}_0$), they produce the same residuals every second step.

Also as an introduction to the case $s>1$, we give some more details (still assuming $s=1$ and k to be small to enough to have Krylov subspaces of full dimension; see footnote 2).

Bi-CG relies on coupled two-term recurrences:

Scalars α_k and β_{k+1} are computed such that

$$\begin{cases} \mathbf{r}_{k+1}^{\text{Bi-CG}} = \mathbf{r}_k^{\text{Bi-CG}} - \mathbf{c}_k^{\text{Bi-CG}}\alpha_k \perp \tilde{\mathbf{r}}_k, \\ \mathbf{u}_{k+1}^{\text{Bi-CG}} = \mathbf{r}_{k+1}^{\text{Bi-CG}} - \mathbf{u}_k^{\text{Bi-CG}}\beta_{k+1} \quad \text{such that} \quad \mathbf{c}_{k+1}^{\text{Bi-CG}} \equiv \mathbf{A}\mathbf{u}_{k+1}^{\text{Bi-CG}} \perp \tilde{\mathbf{r}}_k. \end{cases} \tag{9}$$

Here, $\tilde{\mathbf{r}}_0, \ldots, \tilde{\mathbf{r}}_j$ is a basis of $\mathcal{K}_{j+1}(\mathbf{A}^*, \tilde{\mathbf{r}}_0)$ for all $j \leqslant k$. Multiplying the second recurrence relation by $\mathbf{A}$ allows the following reformulation of (9):

$$\begin{cases} \mathbf{r}_{k+1}^{\text{Bi-CG}} = \mathbf{r}_k^{\text{Bi-CG}} - \mathbf{c}_k^{\text{Bi-CG}}\alpha_k \perp \tilde{\mathbf{r}}_k, \\ \mathbf{c}_{k+1}^{\text{Bi-CG}} = \mathbf{A}\mathbf{r}_{k+1}^{\text{Bi-CG}} - \mathbf{c}_k^{\text{Bi-CG}}\beta_{k+1} \perp \tilde{\mathbf{r}}_k. \end{cases} \tag{10}$$

This formulation is slightly more compact than the one in (9), but can also be used as an alternative for computing $\mathbf{c}_{k+1}^{\text{Bi-CG}}$ and $\mathbf{u}_{k+1}^{\text{Bi-CG}}$. Because, once β_{k+1} and $\mathbf{c}_{k+1}^{\text{Bi-CG}}$ have been determined such that $\mathbf{c}_{k+1}^{\text{Bi-CG}} \perp \tilde{\mathbf{r}}_k$, then $\mathbf{u}_{k+1}^{\text{Bi-CG}}$ can be obtained as an update at the cost of one additional AXPY (per step): $\mathbf{u}_{k+1}^{\text{Bi-CG}} = \mathbf{r}_{k+1}^{\text{Bi-CG}} - \mathbf{u}_k^{\text{Bi-CG}}\beta_{k+1}$. This formulation is closer to the IDR approach. Therefore, we will used it to derive Bi-CGSTAB below.

An induction argument shows that (10) implies that

$$\mathbf{r}_{k+1}^{\text{Bi-CG}}, \mathbf{c}_{k+1}^{\text{Bi-CG}} \perp \tilde{\mathbf{r}}_k, \tilde{\mathbf{r}}_{k-1}, \ldots, \tilde{\mathbf{r}}_0:$$

local bi-orthogonality implies global bi-orthogonality.

Bi-CG leads to Bi-CGSTAB: put $\mathbf{P}_k \equiv p_k(\mathbf{A})$ with p_k a polynomial of degree k with $p_k(0)=1$. The first relation in (10) leads to (take $\tilde{\mathbf{r}}_k = \bar{p}_k(\mathbf{A}^*)\tilde{\mathbf{r}}_0$): (select α_k such that)

$$\mathbf{v}_k \equiv \mathbf{P}_k\mathbf{r}_{k+1}^{\text{Bi-CG}} = \mathbf{P}_k\mathbf{r}_k^{\text{Bi-CG}} - \mathbf{P}_k\mathbf{c}_k^{\text{Bi-CG}}\alpha_k \perp \tilde{\mathbf{r}}_0.$$

With $\mathbf{r}_k \equiv \mathbf{P}_k\mathbf{r}_k^{\text{Bi-CG}}$ and $\mathbf{c}_k = \mathbf{P}_k\mathbf{c}_k^{\text{Bi-CG}}$, this reads as

$$\mathbf{v}_k = \mathbf{r}_k - \mathbf{c}_k\alpha_k \perp \tilde{\mathbf{r}}_0.$$

With $p_{k+1}(\lambda) \equiv (1-\omega_{k+1}\lambda)p_k(\lambda)$, we have

$$\mathbf{r}_{k+1} \equiv \mathbf{P}_{k+1}\mathbf{r}_{k+1}^{\text{Bi-CG}} = (\mathbf{I}-\omega_{k+1}\mathbf{A})\mathbf{P}_k\mathbf{r}_{k+1}^{\text{Bi-CG}} = (\mathbf{I}-\omega_{k+1}\mathbf{A})\mathbf{v}_k.$$

The second relation in (10) allows us to compute $\mathbf{P}_k\mathbf{c}_{k+1}^{\text{Bi-CG}}$:

$$\mathbf{P}_k\mathbf{c}_{k+1}^{\text{Bi-CG}} = \mathbf{A}\mathbf{P}_k\mathbf{r}_{k+1}^{\text{Bi-CG}} - \mathbf{P}_k\mathbf{c}_k^{\text{Bi-CG}}\beta_{k+1} = \mathbf{A}\mathbf{v}_k - \mathbf{c}_k\beta_{k+1} \perp \tilde{\mathbf{r}}_0.$$

With $\mathbf{u}_k \equiv \mathbf{P}_k\mathbf{u}_k^{\text{Bi-CG}}$ and $\mathbf{w} \equiv \mathbf{v}_k - \mathbf{u}_k\beta_{k+1}$, we have that $\mathbf{A}\mathbf{w} = \mathbf{P}_k\mathbf{c}_{k+1}^{\text{Bi-CG}}$ and $\mathbf{u}_{k+1} = (\mathbf{I}-\omega_{k+1}\mathbf{A})\mathbf{w} = \mathbf{w} - \omega_{k+1}\mathbf{A}\mathbf{w}$.

The algorithm derived here is given in the left panel of Algorithm 2. Note that this version of Bi-CGSTAB is slightly different from the standard one in the computation of the vector $\mathbf{u}_k$. As explained above, the formulation here relies on two explicit orthogonalization on $\tilde{\mathbf{r}}_0$. In the standard formulation, one of the orthogonalization is implicit. We could make the orthogonalization explicit at the cost of one additional AXPY (as compared to the standard approach) (the last three lines in the 'repeat loop' replace the lines $\mathbf{u}' = \mathbf{u} - \omega\mathbf{A}\mathbf{u}$, compute β, $\mathbf{u} = \mathbf{r} + \mathbf{u}'\beta$, of the standard algorithm).

The IDR algorithm for $s=1$ is given in the right panel of Algorithm 2. The algorithm has been slightly simplified: two steps

$$\mathbf{r}' = \mathbf{r} - \mathbf{A}\mathbf{u}, \quad \alpha \text{ such that } \mathbf{v} = \mathbf{r}' - \mathbf{A}\mathbf{u}\alpha \perp \tilde{\mathbf{r}}_0,$$

have been combined into one single step

$$\alpha' \text{ such that } \mathbf{v} = \mathbf{r} - \mathbf{A}\mathbf{u}\alpha' \perp \tilde{\mathbf{r}}_0.$$

Here, we used the fact that $\mathbf{s} = \mathbf{r} - \mathbf{r}' = \mathbf{A}\mathbf{u}$ and

$$\mathbf{r}' - \mathbf{s}\alpha = \mathbf{r}' - (\mathbf{r}-\mathbf{r}')\alpha = \mathbf{r} - (\mathbf{r}-\mathbf{r}')\alpha' \quad \text{for } \alpha' = 1+\alpha.$$

```
Select an x.
Select an r̃0.
Compute r = b − Ax.
Set u = r.
repeat
  Compute [Au]
  α such that
      v = r − Auα ⊥ r̃0
  Compute [Av]
  Select ω, r = v − ωAv
  β such that
      Aw = Av − Auβ ⊥ r̃0
  w = v − uβ
  u = w − ωAw
end repeat
```

```
Select an x.
Select an r̃0.
Compute r = b − Ax.
Set u = r.
repeat
  Compute [Au]
  α such that
      v = r − Auα ⊥ r̃0
  Compute [Av]. r′ = r.
  Select ω, r = v − ωAv
  s = r′ − r, u′ = uα + ωv
  β such that
      v′ = r − sβ ⊥ r̃0
  u = u′β + ωv′
end repeat
```

Algorithm 2. Bi-CGSTAB (left) and IDR (right). The boxed expressions as [**Av**] indicate a multiplication by **A**. The unboxed expressions **Av** represent vectors that are already available and do not require a multiplication by **A**. The vectors **r** and the vectors **v** of Bi-CGSTAB and IDR coincide, the **u** vectors are co-linear. Indices have been suppressed to emphasize that newly computed vectors can replace old ones (that are represented by the same letter). The vectors $\mathbf{v}'$ and $\mathbf{u}'$ in IDR can be stored in the location for **v** and **u**, respectively (the prime has been added to facilitate the description of the relation with Bi-CGSTAB). The recursions to update the approximate solutions have not been included: if **r** (and **v**) is updated by a vector of the form $-\mathbf{A}\mathbf{u}\alpha$ then update **x** by $\mathbf{u}\alpha$.

To see that the algorithms are equivalent, add indices. Then the IDR loop reads as:

$$\alpha_k \text{ such that } \mathbf{v}_k = \mathbf{r}_k - \mathbf{A}\mathbf{u}_k\alpha_k \perp \tilde{\mathbf{r}}_0$$
$$\text{Select } \omega_{k+1},\ \mathbf{r}_{k+1} = \mathbf{v}_k - \omega_{k+1}\mathbf{A}\mathbf{v}_k$$
$$\mathbf{s}_{k+1} = \mathbf{r}_k - \mathbf{r}_{k+1},\ \mathbf{u}'_{k+1} = \mathbf{u}_k\alpha + \omega_{k+1}\mathbf{v}_k$$
$$\beta \text{ such that } \mathbf{v}'_{k+1} = \mathbf{r}_{k+1} - \mathbf{s}_{k+1}\beta \perp \tilde{\mathbf{r}}_0$$
$$\mathbf{u}_{k+1} = \mathbf{u}'_{k+1}\beta + \omega_{k+1}\mathbf{v}'_{k+1}$$

Note that $\mathbf{s}_j = \mathbf{A}\mathbf{u}'_j$. Hence

$$\begin{aligned}\mathbf{A}\mathbf{u}_{k+1} &= \mathbf{s}_{k+1}\beta + \omega_{k+1}\mathbf{A}\mathbf{v}'_{k+1} = \mathbf{r}_{k+1} - \mathbf{v}'_{k+1} + \omega_{k+1}\mathbf{A}\mathbf{v}'_{k+1}\\ &= (\mathbf{I} - \omega_{k+1}\mathbf{A})(\mathbf{v}_k - \mathbf{v}'_{k+1}).\end{aligned}$$

Note that $\mathbf{v}'_{k+1} - \mathbf{v}_k \perp \tilde{\mathbf{r}}_0$. Since $\mathbf{v}'_{k+1} - \mathbf{v}_k = \mathbf{r}_{k+1} - \mathbf{v}_k - \mathbf{s}_{k+1}\beta$, we have that

$$\mathbf{v}'_{k+1} - \mathbf{v}_k = -\omega_{k+1}\mathbf{A}\mathbf{v}_k - \mathbf{A}(\mathbf{u}_k\alpha + \omega_{k+1}\mathbf{v}_k)\beta \in \text{Span}(\mathbf{A}\mathbf{v}_k, \mathbf{A}\mathbf{u}_k) \cap \tilde{\mathbf{r}}_0^{\perp}.$$

Hence, assuming that the $\mathbf{v}_k$ and $\mathbf{u}_k$ of IDR and Bi-CGSTAB coincide (except for a scalar multiple), we see that $\mathbf{v}'_{k+1} - \mathbf{v}_k$ is a multiple of **Aw** of Bi-CGSTAB. Therefore, $\mathbf{A}\mathbf{u}_{k+1}$ is a multiple of $(\mathbf{I} - \omega_{k+1}\mathbf{A})\mathbf{A}\mathbf{w}$ of Bi-CGSTAB, which shows that the $\mathbf{u}_{k+1}$ of IDR and Bi-CGSTAB are co-linear and that the $\mathbf{v}_{k+1}$ and $\mathbf{r}_{k+1}$ coincide. Summarizing we have

Proposition 15. *If* $\mathbf{x}_0$ *and* $\tilde{\mathbf{r}}_0$ *coincide, if we use exact arithmetic, and if we use the same selection strategy for the* ω_k (*minimizing residual norms*), *then the vectors* $\mathbf{v}_k$ *and* $\mathbf{r}_k$ *in* IDR *and* Bi-CGSTAB *coincide and the vectors* $\mathbf{u}_k$ *in* IDR *and* Bi-CGSTAB *are co-linear.*

Or in a less mathematical formulation: except for the last three lines in the 'repeat loops' in Algorithm 2, the IDR and the Bi-CGSTAB algorithm are the same and essentially produce the same quantities.

6. Bi-CG

As recalled in the previous section, where we discussed the case $s = 1$, Bi-CGSTAB has been introduced as a transpose free variant of Bi-CG. To explain how an s-dimensional initial shadow residual $\tilde{\mathbf{R}}_0$ can be incorporated in Bi-CGSTAB, we first have to explain how Bi-CG can be formulated for an s-dimensional initial shadow residual. This is the subject of this section.

Suppose $\tilde{\mathbf{R}}_0$ is an $n \times s$ matrix of full rank.

Recall that $\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$ is the space of all vectors $\sum_{j<k}(\mathbf{A}^*)^j\tilde{\mathbf{R}}_0\vec{\beta}_j$ with $\vec{\beta}_j$ in $\mathbb{C}^s$. $\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$ is the (block) Krylov subspace of order k generated by $\mathbf{A}^*$ and $\tilde{\mathbf{R}}_0$. Let $\tilde{\mathbf{R}}_i$ be $n \times s$ matrices such that the columns of the matrices $\tilde{\mathbf{R}}_0, \ldots, \tilde{\mathbf{R}}_{k-1}$ span the space $\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$ (for all k sufficiently small, cf. footnote 2).

```
Select an x_0
Select an R̃_0
Compute r_0 = b − Ax_0
k = −1 ,  σ_k = I ,
Set U_k = 0 ,  C_k = 0 ,  R̃_k = 0 .
repeat
  s = r_{k+1}
  for j = 1, ..., s
    u = s ,  Compute s = Au
    β_k = σ_k^{-1}(R̃_k^* s)
    u = u − U_k β_k ,  U_{k+1} e_j = u
    s = s − C_k β_k ,  C_{k+1} e_j = s
  end for
  k ← k + 1
  σ_k = R̃_k^* C_k ,  α_k = σ_k^{-1}(R̃_k^* r_k)
  x_{k+1} = x_k + U_k α_k ,  r_{k+1} = r_k − C_k α_k
end repeat
```

Algorithm 3. Bi-CG. $\mathbf{0}$ is the $n \times s$ zero matrix, I is the $s \times s$ identity matrix. $\tilde{\mathbf{R}}_k$ is an $n \times s$ matrix such that $\mathrm{Span}(\tilde{\mathbf{R}}_k) + \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$ equals $\mathcal{K}_{k+1}(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$.

Example 16. $\tilde{\mathbf{R}}_k = \bar{p}_k(\mathbf{A}^*)\tilde{\mathbf{R}}_0$ with $p_k(\lambda) = (1-\omega_k\lambda)\cdots(1-\omega_1\lambda)$.

We will now explain how to construct a residual vector $\mathbf{r}_k$ in $\mathcal{K}_K(\mathbf{A}, \mathbf{r}_0)$ with $K = ks+1$ that is orthogonal to $\mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$.

Let $\mathbf{C}_0$ be the $n \times s$ matrix with columns $\mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^s\mathbf{r}_0$. The matrix $\mathbf{C}_0$ is of the form $\mathbf{A}\mathbf{U}_0$ with $\mathbf{U}_0$ explicitly available: $\mathbf{U}_0 = [\mathbf{r}_0, \ldots, \mathbf{A}^{s-1}\mathbf{r}_0]$.

Now, find a vector $\vec{\alpha}_0 \in \mathbb{C}^s$ such that $\mathbf{r}_1 \equiv \mathbf{r}_0 - \mathbf{C}_0\vec{\alpha}_0 \perp \tilde{\mathbf{R}}_0$, $\mathbf{r}_1 \in \mathcal{K}_{s+1}(\mathbf{A}, \mathbf{r}_0)$, and update $\mathbf{x}_0$ as $\mathbf{x}_1 = \mathbf{x}_0 + \mathbf{U}_0\vec{\alpha}_0$.

Note that, with $\sigma_0 \equiv \tilde{\mathbf{R}}_0^*\mathbf{C}_0$ any vector of the form $\mathbf{w} - \mathbf{C}_0\sigma_0^{-1}\tilde{\mathbf{R}}_0^*\mathbf{w}$ is orthogonal to $\tilde{\mathbf{R}}_0$: $\mathbf{I} - \mathbf{C}_0\sigma_0^{-1}\tilde{\mathbf{R}}_0$ is a skew projection onto the orthogonal complement of $\tilde{\mathbf{R}}_0$. Here, for ease of explanation, we assume that σ_0 is $s \times s$ non-singular. If σ_0 is singular (or ill conditioned), then s can be reduced to overcome breakdown or loss of accuracy (see Note 4 and [10] for more details).

Let $\mathbf{v} = \mathbf{r}_1$. We construct an $n \times s$ matrix $\mathbf{C}_1$ orthogonal to $\tilde{\mathbf{R}}_0$ as follows:

$$\mathbf{s} = \mathbf{A}\mathbf{v}, \quad \mathbf{s} = \mathbf{s} - \mathbf{C}_0(\sigma_0^{-1}\tilde{\mathbf{R}}_0^*\mathbf{s}), \quad \mathbf{C}_1 e_j = \mathbf{s}, \quad \mathbf{v} = \mathbf{s} \quad \text{for } j = 1, \ldots, s.$$

Here, $\mathbf{C}_1 e_j = \mathbf{s}$ indicates that the j-column of $\mathbf{C}_1$ is set to the vector $\mathbf{s}$: e_j is the jth (s-dimensional) standard basis vector. Then $\mathbf{C}_1$ is orthogonal to $\tilde{\mathbf{R}}_0$ and its columns form a basis for the Krylov subspace of order s generated by $\mathbf{A}_1 \equiv (\mathbf{I} - \mathbf{C}_0\sigma_0^{-1}\tilde{\mathbf{R}}_0^*)\mathbf{A}$ and $\mathbf{A}_1\mathbf{r}_1$. Note that there is a matrix $\mathbf{U}_1$ such that $\mathbf{C}_1 = \mathbf{A}\mathbf{U}_1$. The columns of $\mathbf{U}_1$ can be computed simultaneously with the columns of $\mathbf{C}_1$: $\mathbf{U}_1 e_j = \mathbf{v} - \mathbf{U}_0(\sigma_0^{-1}\tilde{\mathbf{R}}_0^*\mathbf{s})$. Note that more stable approaches as GCR (or Arnoldi) for computing a basis of this Krylov subspace could have been used as well. Now, with $\sigma_1 \equiv \tilde{\mathbf{R}}_1^*\mathbf{C}_1$, the vector $\mathbf{r}_2 \equiv \mathbf{r}_1 - \mathbf{C}_1(\sigma_1^{-1}\tilde{\mathbf{R}}_1^*\mathbf{r}_1)$ is orthogonal to $\tilde{\mathbf{R}}_0$ as well as to $\tilde{\mathbf{R}}_1$, it belongs to $\mathcal{K}_{2s+1}(\mathbf{A}, \mathbf{r}_0)$, and $\mathbf{x}_2 = \mathbf{x}_1 + \mathbf{U}_1(\sigma_1^{-1}\tilde{\mathbf{R}}_1^*\mathbf{r}_1)$ is the associated approximate solution. Repeating the procedure

$$\begin{aligned}
&\mathbf{r}_{k+1} = \mathbf{r}_k - \mathbf{C}_k\vec{\alpha}_k \perp \tilde{\mathbf{R}}_k\\
&\mathbf{v} = \mathbf{r}_{k+1}\\
&\text{for } j = 1, \ldots, s\\
&\quad \mathbf{s} = \mathbf{A}\mathbf{v}\\
&\quad \mathbf{C}_{k+1}e_j = \mathbf{s} - \mathbf{C}_k\vec{\beta}_j \perp \tilde{\mathbf{R}}_k\\
&\quad \mathbf{v} = \mathbf{C}_{k+1}e_j\\
&\text{end for}
\end{aligned} \tag{11}$$

leads to the residuals as announced: $\mathbf{r}_k \in \mathcal{K}_{ks+1}(\mathbf{A}, \mathbf{r}_0)$, $\mathbf{r}_k \perp \mathcal{K}_k(\mathbf{A}^*, \tilde{\mathbf{R}}_0)$. With $\sigma_k \equiv \tilde{\mathbf{R}}_k^*\mathbf{C}_k$, the $\vec{\alpha}_k$ and $\vec{\beta}_j = \vec{\beta}_j^{(k)}$ can be computed as $\vec{\alpha}_k = \sigma_k^{-1}(\tilde{\mathbf{R}}_k^*\mathbf{r}_k)$, $\vec{\beta}_j = \sigma_k^{-1}(\tilde{\mathbf{R}}_k^*\mathbf{s})$. Simultaneously with the computation of the columns of $\mathbf{C}_{k+1}$, the columns of a matrix $\mathbf{U}_{k+1}$ can be computed. A similar remark applies to the update of $\mathbf{x}_k$. The resulting algorithm can be found in Algorithm 3.

The columns of $\mathbf{C}_{k+1}$ form a Krylov basis of the Krylov subspace generated by $\mathbf{A}_{k+1} \equiv (\mathbf{I} - \mathbf{C}_k\sigma_k^{-1}\tilde{\mathbf{R}}_k^*)\mathbf{A}$ and $\mathbf{A}_{k+1}\mathbf{r}_k$.

The generalization of Bi-CG that is presented above is related to the method ML(k)BiCG [13]. The orthogonalization conditions on the ML(k)BiCG residuals, however, are more strict.

Note 4. If $\sigma_k = \tilde{\mathbf{R}}_k^*\mathbf{C}_k$ is (close to) singular, then reduce s. We can take the following approach (which has not been included in Algorithm 3).

Determine the singular value decomposition of σ_k, $\sigma_k = Q\,\Sigma V^*$ with Q and V $s \times s$ unitary matrices, $\Sigma = \mathrm{diag}(\nu_1, \ldots, \nu_s)$, with singular values ν_j ordered such that $\nu_1 \geqslant \cdots \geqslant \nu_s \geqslant 0$. Find $p < s$ such that $\nu_p > \delta > \nu_{p+1}$ for some appropriate (small) $\delta > 0$. Now, replace $\tilde{\mathbf{R}}_k$ by $\tilde{\mathbf{R}}_k Q(:, 1{:}p)$ (here we use MATLAB's notation) and $\mathbf{C}_k$ by $\mathbf{C}_k(:, 1{:}p)$. Then the 'new' σ_k has singular values $\nu_1, \ldots, \nu_p$.

7. Bi-CGSTAB

We are now ready to formulate the Bi-CGSTAB version of IDR for $s > 1$.

As before, for each k, select a polynomial p_k of exact degree k, and put $\mathbf{P}_k \equiv p_k(\mathbf{A})$. For ease of notation, replace $\mathbf{C}_k = \mathbf{C}_k^{\text{Bi-CG}}$, $\mathbf{U}_k = \mathbf{U}_k^{\text{Bi-CG}}$, and $\mathbf{r}_k = \mathbf{r}_k^{\text{Bi-CG}}$ in (11) by C_k, U_k, and r_k, respectively. Now (11) can be reformulated as (take $\tilde{\mathbf{R}}_k = \bar{p}_k(\mathbf{A}^*)\tilde{\mathbf{R}}_0$)

$$
\begin{aligned}
&\mathbf{P}_k r_{k+1} = \mathbf{P}_k r_k - \mathbf{P}_k C_k \vec{\alpha}_k \perp \tilde{\mathbf{R}}_0\\
&\mathbf{v} = \mathbf{P}_k r_{k+1}\\
&\text{for } j = 1, \ldots, s\\
&\quad \mathbf{s} = \mathbf{A}\mathbf{v}\\
&\quad \mathbf{P}_k C_{k+1} e_j = \mathbf{s} - \mathbf{P}_k C_k \vec{\beta}_j \perp \tilde{\mathbf{R}}_0\\
&\quad \mathbf{v} = \mathbf{P}_k C_{k+1} e_j\\
&\text{end for}
\end{aligned}
\tag{12}
$$

Note that, for obtaining a formula for computing $\vec{\alpha}_k$ and $\vec{\beta}_j = \vec{\beta}_j^{(k)}$ there is no need to refer to (11): it suffices to refer to the orthogonality conditions in (12).

To repeat the k-loop, represented in (12), k has to be increased, that is, $\mathbf{P}_{k+1} r_{k+1}$ and $\mathbf{P}_{k+1} C_{k+1}$ have to be computed from $\mathbf{P}_k r_{k+1}$ and $\mathbf{P}_k C_{k+1}$, respectively. If $\mathbf{P}_{k+1} = (\mathbf{I} - \omega_k \mathbf{A})\mathbf{P}_k$, then these quantities can be obtained simply by multiplication by $\mathbf{I} - \omega_k \mathbf{A}$. This naive approach would require $s + 1$ additional multiplications by $\mathbf{A}$. However, in the process of updating $\mathbf{P}_k C_{k+1}$, the vectors $\mathbf{A}\mathbf{P}_k r_{k+1}$ and $\mathbf{A}\mathbf{P}_k C_{k+1} e_j$ $(j = 1, \ldots, s-1)$ are computed. The following loop exploits this information.

$$
\begin{aligned}
&\mathbf{P}_k r_{k+1} = \mathbf{P}_k r_k - \mathbf{A}\mathbf{P}_k U_k \alpha_k \perp \tilde{\mathbf{R}}_0\\
&\mathbf{v} = \mathbf{P}_k r_{k+1},\ \mathbf{s} = \mathbf{A}\mathbf{v},\ \mathbf{A}\mathbf{P}_k r_{k+1} = \mathbf{s}\\
&\text{for } j = 1, \ldots, s\\
&\quad \mathbf{A}\mathbf{P}_k U_{k+1} e_j = \mathbf{s} - \mathbf{A}\mathbf{P}_k U_k \beta_j \perp \tilde{\mathbf{R}}_0\\
&\quad \mathbf{P}_k U_{k+1} e_j = \mathbf{v} - \mathbf{P}_k U_k \beta_j\\
&\quad \mathbf{v} = \mathbf{A}\mathbf{P}_k U_{k+1} e_j,\ \mathbf{s} = \mathbf{A}\mathbf{v},\ \mathbf{A}^2 \mathbf{P}_k U_{k+1} e_j = \mathbf{s}\\
&\text{end for}\\
&\text{Select an } \omega_k,\ \mathbf{P}_{k+1} r_{k+1} = \mathbf{P}_k r_{k+1} - \omega_k \mathbf{A}\mathbf{P}_k r_{k+1}\\
&\text{for } i = 0, 1,\ \mathbf{A}^i \mathbf{P}_{k+1} U_{k+1} = \mathbf{A}^i \mathbf{P}_k U_{k+1} - \omega_k \mathbf{A}^{i+1} \mathbf{P}_k U_{k+1},\ \text{end for}
\end{aligned}
\tag{13}
$$

Here, $\mathbf{A}U_k$ equals C_k. We used U_k in our formulation here since we also need $\mathbf{A}^2 U_k$ and we want to limit the number of different symbols.

Computing $\mathbf{P}_{k+1} r_{k+1}$ as soon as $\mathbf{P}_k r_{k+1}$ is available, and computing $\mathbf{P}_{k+1} C_{k+1} e_j$ as soon as $\mathbf{P}_k C_{k+1} e_j$, is available leads to the following variant that requires less memory.

$$
\begin{aligned}
&\mathbf{P}_k r_{k+1} = \mathbf{P}_k r_k - \mathbf{P}_k C_k \alpha_k \perp \tilde{\mathbf{R}}_0\\
&\mathbf{v} = \mathbf{P}_k r_{k+1},\ \mathbf{s} = \mathbf{A}\mathbf{v}\\
&\text{Select an } \omega_k,\ \mathbf{P}_{k+1} r_{k+1} = \mathbf{v} - \omega_k \mathbf{s}\\
&\text{for } j = 1, \ldots, s\\
&\quad \mathbf{v} = \mathbf{s} - \mathbf{P}_k C_k \beta_j \perp \tilde{\mathbf{R}}_0\\
&\quad \mathbf{s} = \mathbf{A}\mathbf{v},\ \mathbf{P}_{k+1} C_{k+1} e_j = \mathbf{v} - \omega_k \mathbf{s}\\
&\text{end for}
\end{aligned}
\tag{14}
$$

The additional matrix $\mathbf{P}_k C_{k+1}$ need not to be stored. The formation of $\mathbf{P}_k C_{k+1}$ and $\mathbf{A}\mathbf{P}_k C_{k+1}$ in the last step of (13) may be superfluous if $\mathbf{P}_{k+1} r_{k+1} = \mathbf{P}_k r_{k+1} - \omega_k \mathbf{A}\mathbf{P}_k r_{k+1}$ turns out to be small enough. Unfortunately, this step cannot be avoided in (13), whereas (14) allows a 'break' after the computation of $\mathbf{P}_{k+1} r_{k+1}$. Nevertheless, the approach in (13) may be useful, since it may allow easier generalization to other hybrid Bi-CG variants. Note that in coding (13) the matrix update of $\mathbf{P}_{k+1} C_{k+1}$ can exploit BLAS2 subroutines with better parallelization properties then the sequence of vector updates in (14).

With $\mathbf{r}_k \equiv \mathbf{P}_k r_k$, $\mathbf{S}_k \equiv \mathbf{P}_k C_k$ and $\mathbf{v}_k \equiv \mathbf{P}_k r_{k+1}$, we obtain the algorithm as displayed in the left panel of Algorithm 4.

Select an $\mathbf{x}_0$
Select an $\tilde{\mathbf{R}}_0$
Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
$k = 0$, $\mathbf{s} = \mathbf{r}_0$
for $j = 1, \ldots, s$
  $\mathbf{U}_0 e_j = \mathbf{s}$, $\mathbf{s} = \mathbf{A}\mathbf{s}$, $\mathbf{S}_0 e_j = \mathbf{s}$
end for
repeat
  $\sigma_k = \tilde{\mathbf{R}}_0^* \mathbf{S}_k$, $\vec{\alpha}_k = \sigma_k^{-1}(\tilde{\mathbf{R}}_0^* \mathbf{r}_k)$
  $\mathbf{v} = \mathbf{r}_k - \mathbf{S}_k \vec{\alpha}_k$
  $\mathbf{s} = \mathbf{A}\mathbf{v}$
  Select an ω_k
  $\mathbf{r}_{k+1} = \mathbf{v} - \omega_k \mathbf{s}$
  for $j = 1, \ldots, s$
    $\vec{\beta}_j = \sigma_k^{-1}(\tilde{\mathbf{R}}_0^* \mathbf{s})$
    $\mathbf{v} = \mathbf{s} - \mathbf{S}_k \vec{\beta}_j$
    $\mathbf{s} = \mathbf{A}\mathbf{v}$
    $\mathbf{S}_{k+1} e_j = \mathbf{v} - \omega_k \mathbf{s}$
  end for
  $k \leftarrow k + 1$
end repeat

Select an $\mathbf{x}$
Select an $\tilde{\mathbf{R}}_0$
Compute $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$
$\mathbf{s} = \mathbf{r}$
for $j = 1, \ldots, s$
  $\mathbf{U} e_j = \mathbf{s}$, $\mathbf{s} = \mathbf{A}\mathbf{s}$, $\mathbf{S} e_j = \mathbf{s}$
end for
repeat
  $\sigma = \tilde{\mathbf{R}}_0^* \mathbf{S}$, $\vec{\alpha} = \sigma^{-1}(\tilde{\mathbf{R}}_0^* \mathbf{r})$
  $\mathbf{x} = \mathbf{x} + \mathbf{U}\vec{\alpha}$, $\mathbf{v} = \mathbf{r} - \mathbf{S}\vec{\alpha}$
  $\mathbf{s} = \mathbf{A}\mathbf{v}$
  Select an ω
  $\mathbf{x} = \mathbf{x} + \omega \mathbf{v}$, $\mathbf{r} = \mathbf{v} - \omega \mathbf{s}$
  for $j = 1, \ldots, s$
    $\vec{\beta} = \sigma^{-1}(\tilde{\mathbf{R}}_0^* \mathbf{s})$
    $\mathbf{u} = \mathbf{v} - \mathbf{U}\vec{\beta}$, $\mathbf{v} = \mathbf{s} - \mathbf{S}\vec{\beta}$
    $\mathbf{s} = \mathbf{A}\mathbf{v}$
    $\mathbf{U}' e_j = \mathbf{u} - \omega \mathbf{v}$, $\mathbf{S}' e_j = \mathbf{v} - \omega \mathbf{s}$
  end for
  $\mathbf{U} = \mathbf{U}'$, $\mathbf{S} = \mathbf{S}'$
end repeat

Algorithm 4. Bi-CGSTAB. For analysis purposes, we present the algorithm with indices (and no update of the approximate solution), see the left panel. The right panel includes the update of the approximate solution, but the algorithm here has been displayed without indices: new values are allowed to replace old values.

8. Bi-CGSTAB and IDR(s)

We put $\mathbf{s}_{k+1} \equiv \mathbf{r}_{k+1} - \mathbf{r}_k$ and $\mathbf{S}_k \equiv [\mathbf{s}_k, \mathbf{s}_{k-1}, \ldots, \mathbf{s}_{k+1-s}]$. The IDR($s$) loop runs as follows (below the horizontal line)

$$\mathbf{v}_k = \mathbf{r}_k - \mathbf{S}_k \vec{\alpha}_k \perp \tilde{\mathbf{R}}_0$$

Select ω, $\mathbf{r}_{k+1} = (\mathbf{I} - \omega \mathbf{A})\mathbf{v}_k$
for $j = 1, \ldots, s$
  $\mathbf{v}_{k+j} = \mathbf{r}_{k+j} + \mathbf{S}_{k+j} \vec{\beta}_{k+j} \perp \tilde{\mathbf{R}}_0$
  $\mathbf{r}_{k+j+1} = (\mathbf{I} - \omega \mathbf{A})\mathbf{v}_{k+j}$
end for
$\mathbf{v}_{k+1+s} = \mathbf{r}_{k+1+s} - \mathbf{S}_{k+1+s} \vec{\alpha}_{k+1+s} \perp \tilde{\mathbf{R}}_0$

As before, $\ldots \perp \tilde{\mathbf{R}}_0$ indicates that the s-vectors ($\vec{\alpha}$ and $\vec{\beta}$) in the expression in front of the symbol $\perp$ has been selected such that the expression is orthogonal to $\tilde{\mathbf{R}}_0$.

The following lemma implies that IDR(s) and Bi-CGSTAB are equivalent: if they produce the same $\mathbf{v}_k$ and $\mathbf{r}_k$, then, in exact arithmetic, they produce the same $\mathbf{r}_{k+1}$, $\mathbf{v}_{k+s+1}$ and $\mathbf{r}_{k+1+s}$. With the same initialization, assuming exact arithmetic, they produce the same residuals and approximations every $s + 1$ step.

Lemma 17. *Let* $\mathbf{w}_{k+j} \equiv \mathbf{A}\mathbf{v}_{k+j} + \mathbf{S}_k \vec{\gamma}_{k+j} \perp \tilde{\mathbf{R}}_0$.

(a) $\mathbf{v}_{k+1+s} = \mathbf{r}_{k+1} - \mathbf{S}_{k+1+s} \vec{\tilde{\alpha}} \perp \tilde{\mathbf{R}}_0$.
(b) $\mathrm{Span}(\mathbf{S}_{k+1+s}) = (\mathbf{I} - \omega \mathbf{A})\,\mathrm{Span}([\mathbf{w}_{k+s-1}, \mathbf{w}_{k+s-2}, \ldots, \mathbf{w}_k])$.
(c) *With*

$$\tilde{\mathbf{A}} \equiv \left(\mathbf{I} - \mathbf{S}_k \left(\tilde{\mathbf{R}}_0^* \mathbf{S}_k\right)^{-1} \tilde{\mathbf{R}}_0^*\right) \mathbf{A}, \tag{15}$$

$\mathrm{Span}([\mathbf{w}_{k+s-1}, \mathbf{w}_{k+s-2}, \ldots, \mathbf{w}_k])$ *is the Krylov subspace of order s generated by* $\tilde{\mathbf{A}}$ *and* $\tilde{\mathbf{A}}\mathbf{v}_k$.

Proof. We first prove by induction that

$$\mathrm{Span}\left([\mathbf{s}_{k+j+1}, \ldots, \mathbf{s}_{k+1}]\right) + \mathrm{Span}(\mathbf{S}_k) = \mathrm{Span}\left([\mathbf{w}_{k+j}, \ldots, \mathbf{w}_k]\right) + \mathrm{Span}(\mathbf{S}_k).$$

Note that

$$\mathbf{s}_{k+j+1} = \mathbf{r}_{k+j+1} - \mathbf{r}_{k+j} = -\omega \mathbf{A}\mathbf{v}_{k+j} + \mathbf{v}_{k+j} - \mathbf{r}_{k+j} \in -\omega \mathbf{A}\mathbf{v}_{k+j} + \mathrm{Span}(\mathbf{S}_{k+j}) \subset \mathrm{Span}\left([\mathbf{w}_{k+j}, \ldots, \mathbf{w}_k]\right) + \mathrm{Span}(\mathbf{S}_k).$$

The last inclusion follows by induction.

For $j > 0$, we have

$$\mathbf{s}_{k+j+1} = \mathbf{r}_{k+j+1} - \mathbf{r}_{k+j} = (\mathbf{I} - \omega\mathbf{A})(\mathbf{v}_{k+j} - \mathbf{v}_{k+j-1}). \tag{16}$$

Since $\mathbf{v}_{k+1+j} - \mathbf{v}_{k+j} = \mathbf{r}_{k+j+1} - \mathbf{v}_{k+j} + \mathbf{S}_{k+j+1}\vec{\beta} = -\omega\mathbf{A}\mathbf{v}_{k+j} + \mathbf{S}_{k+j+1}\vec{\beta}$, we have that $\mathbf{v}_{k+1+j} - \mathbf{v}_{k+j} \in \mathrm{Span}([\mathbf{w}_{k+j}, \ldots, \mathbf{w}_k]) + \mathrm{Span}(\mathbf{S}_k)$. In addition, $\vec{\beta}$ is such that $\mathbf{v}_{k+1+j} - \mathbf{v}_{k+j} \perp \tilde{\mathbf{R}}_0$. We also have that $\mathrm{Span}([\mathbf{w}_{k+j}, \ldots, \mathbf{w}_k]) \perp \tilde{\mathbf{R}}_0$, which implies that $\mathbf{v}_{k+1+j} - \mathbf{v}_{k+j} \in \mathrm{Span}([\mathbf{w}_{k+j}, \ldots, \mathbf{w}_k])$. Therefore,

$$\mathrm{Span}\big([\mathbf{v}_{k+1+j} - \mathbf{v}_{k+j}, \ldots, \mathbf{v}_{k+1} - \mathbf{v}_k]\big) = \mathrm{Span}\big([\mathbf{w}_{k+j}, \ldots, \mathbf{w}_k]\big). \tag{17}$$

In particular, $\mathrm{Span}([\mathbf{v}_{k+s} - \mathbf{v}_{k+s-1}, \ldots, \mathbf{v}_{k+1} - \mathbf{v}_k]) = \mathrm{Span}([\mathbf{w}_{k+s-1}, \ldots, \mathbf{w}_k])$, which, in combination with (16) proves (b).

Clearly, $\mathrm{Span}([\mathbf{w}_{k+j-1}, \mathbf{w}_{k+j-2}, \ldots, \mathbf{w}_k]) = \tilde{\mathbf{A}}(\mathrm{Span}([\mathbf{v}_{k+j-1}, \mathbf{v}_{k+j-2}, \ldots, \mathbf{v}_k]))$ $(j = 2, \ldots, s)$. Note that

$$\mathrm{Span}\big([\mathbf{v}_{k+j-1}, \mathbf{v}_{k+j-2}, \ldots, \mathbf{v}_k]\big) = \mathrm{Span}\big([\mathbf{v}_{k+j-1} - \mathbf{v}_{k+j-2}, \ldots, \mathbf{v}_{k+1} - \mathbf{v}_k, \mathbf{v}_k]\big).$$

Therefore, (17) implies that

$$\mathrm{Span}\big([\mathbf{v}_{k+j-1}, \mathbf{v}_{k+j-2}, \ldots, \mathbf{v}_k]\big) = \tilde{\mathbf{A}}\big(\mathrm{Span}\big([\mathbf{v}_{k+j-1}, \mathbf{v}_{k+j-2}, \ldots, \mathbf{v}_k]\big)\big) + \mathrm{Span}(\mathbf{v}_k).$$

Repeating this argument proves (c) of the lemma. □

Proposition 18. *If $\mathbf{x}_0$ and $\tilde{\mathbf{R}}_0$ coincide, if we use exact arithmetic and if we use the same selection strategy for the ω_k (minimizing residual norms), then every $s+1$-st step the vectors $\mathbf{v}_k$ and $\mathbf{r}_k$ in* IDR(s) *coincide with $\mathbf{v}$ and $\mathbf{r}$ vectors in* Bi-CGSTAB.

In a less mathematical formulation, the differences between Bi-CGSTAB and IDR(s) are in the main 'for $j = 1, \ldots, s$' loop of Algorithm 4. Bi-CGSTAB keeps the matrices $\mathbf{S}_k$ and $\mathbf{U}_k$ fixed in this loop (the updates take place out of this loop, where they are replaced by $\mathbf{S}_{k+1}$ and $\mathbf{U}_{k+1}$, respectively), whereas IDR(s) updates these matrices in each step (that is, after each MV), replacing the 'oldest' column by a 'new' vector (of residual differences). At the beginning of the loop, the **S**-matrices of Bi-CGSTAB and IDR(s) span the same space, a Krylov subspace generated by a projected matrix (see (15)).

9. Numerical experiments

We present two numerical examples. The first example serves to demonstrate that the generalized Bi-CGSTAB method and IDR(s) produce the same residuals every $s+1$-st step (as long as numerical effects do not play a role). The second example exposes a difference in numerical stability between the two methods. The experiments have been performed using MATLAB 6.5. We have used the MATLAB implementation of IDR(s) that is described in [10]. For Bi-CGSTAB we use an implementation of Algorithm 4.

To illustrate the equivalence of Bi-CGSTAB and IDR we consider the Sherman4 matrix from the MATRIX MARKET collection. We have taken a right-hand side vector corresponding to a solution vector that consists of ones. Fig. 1 shows the convergence of IDR(s) and Bi-CGSTAB for a shadow space of dimension four.[3] For IDR(s) we have only plotted the residual norms at every $s+1$-st step, although the method produces a residual in every iteration (= MV-multiplications). Clearly, the Bi-CGSTAB-residual norms basically coincide with the IDR-residual norms at these crucial steps, until numerical effects start to play a role.

Table 1 presents the number of IDR- and Bi-CGSTAB MVs for increasing values of the dimension of the shadow space that is needed to make the scaled residual norm smaller than 10^{-8}. Both methods use the recursively updated residual to check for convergence. To validate that this desired accuracy has actually been achieved, we have tabulated the scaled norm of the true residual $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|/\|\mathbf{b}\|$. The table also includes the (optimal) number of GMRES MVs. These results are only included to indicate how close IDR(s) (or Bi-CGSTAB) is to this optimal number. Of course IDR (and Bi-CGSTAB) iterations are much cheaper than GMRES iterations. The results show that increasing s reduces the number of IDR and Bi-CGSTAB iterations to a value that is only 25% above the optimal value for GMRES. The overall gain of increasing s is for this example significant, but rather limited. This is simply because the results for $s = 1$ are already good, only 50% above the optimal value of 120 MVs. Except Bi-CGSTAB for $s = 4$, all methods compute the solution to the desired accuracy. Probably the most noteworthy observation for this example is that IDR(s) and Bi-CGSTAB require basically the same number of MVs for the same dimension of the shadow space, which confirms the mathematical equivalence between the methods.

As a second example we consider the ADD20 matrix, also from the MARIX-MARKET collection. We have again taken the right-hand side vector corresponding to a solution vector that consists of ones. Fig. 2 shows the convergence of IDR(s) and Bi-GCSTAB for a shadow space of dimension four. Also for this example, initially the residual norms of IDR(s) and Bi-CGSTAB coincide. However, numerical effects are much stronger, and after about 100 iterations the convergence curves of the two methods start to differ significantly, resulting in a significantly different number of MVs to achieve the desired accuracy.

Table 2 presents the number of IDR- and Bi-CGSTAB MVs for increasing values of the dimension of the shadow space that is needed to make the scaled residual norm smaller than 10^{-8}, and also the scaled norms of the true residuals $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|/\|\mathbf{b}\|$.

[3] The notation Bi-CGSTAB(4) in figures and tables indicates the use of a shadow space of dimension four. Note the difference in notation with BiCGstab(ℓ) in Section 10, where the ℓ refers to polynomial factors of degree ℓ.

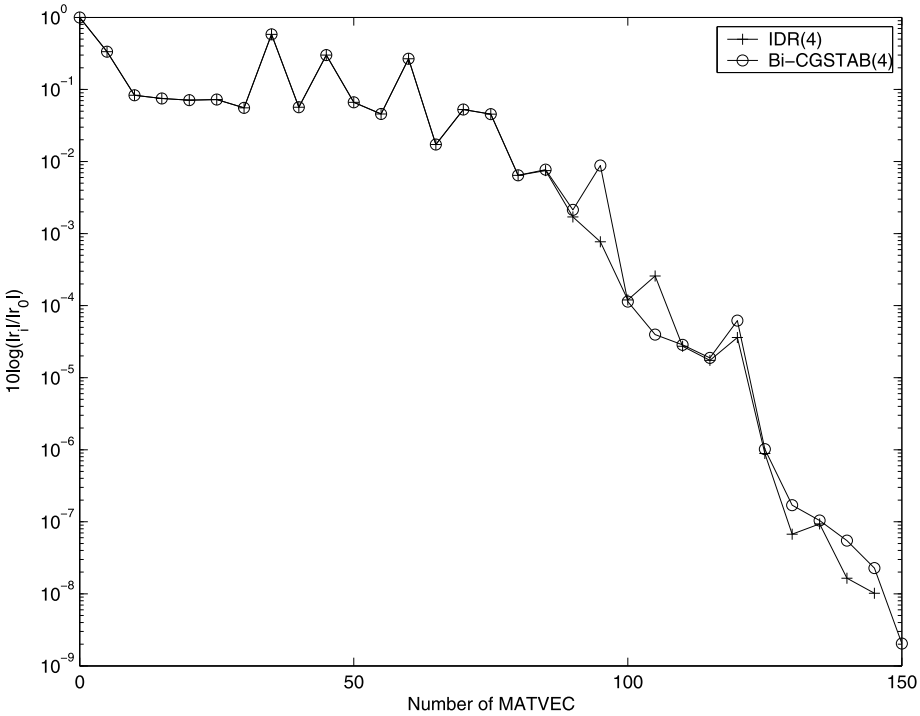

Fig. 1. Convergence of Bi-CGSTAB(4) and IDR(4) for SHERMAN4.

Table 1
Number of matrix–vector multiplications to solve the SHERMAN4 system such that the scaled norm of the updated residual is less than 10^{-8}.

Method	Number of MVs	$\|\mathbf{b}-\mathbf{Ax}\|/\|\mathbf{b}\|$
GMRES	120	9.8×10^{-9}
IDR(1)	179	6.6×10^{-9}
IDR(2)	161	8.3×10^{-9}
IDR(3)	153	8.7×10^{-9}
IDR(4)	146	3.0×10^{-9}
IDR(5)	150	1.2×10^{-9}
Bi-CGSTAB(1)	180	9.0×10^{-9}
Bi-CGSTAB(2)	162	8.6×10^{-9}
Bi-CGSTAB(3)	156	4.6×10^{-9}
Bi-CGSTAB(4)	150	1.8×10^{-7}
Bi-CGSTAB(5)	144	7.1×10^{-9}

Table 2
Number of matrix–vector multiplications to solve the ADD20 system such that the scaled norm of the updated residual is less than 10^{-8}.

Method	Number of MVs	$\|\mathbf{b}-\mathbf{Ax}\|/\|\mathbf{b}\|$
GMRES	295	1.0×10^{-8}
IDR(1)	672	9.0×10^{-9}
IDR(2)	581	9.9×10^{-9}
IDR(3)	588	4.8×10^{-9}
IDR(4)	480	5.3×10^{-9}
IDR(5)	444	9.4×10^{-9}
Bi-CGSTAB(1)	728	9.0×10^{-9}
Bi-CGSTAB(2)	648	9.8×10^{-9}
Bi-CGSTAB(3)	528	2.0×10^{-7}
Bi-CGSTAB(4)	545	7.1×10^{-6}
Bi-CGSTAB(5)	702	0.026

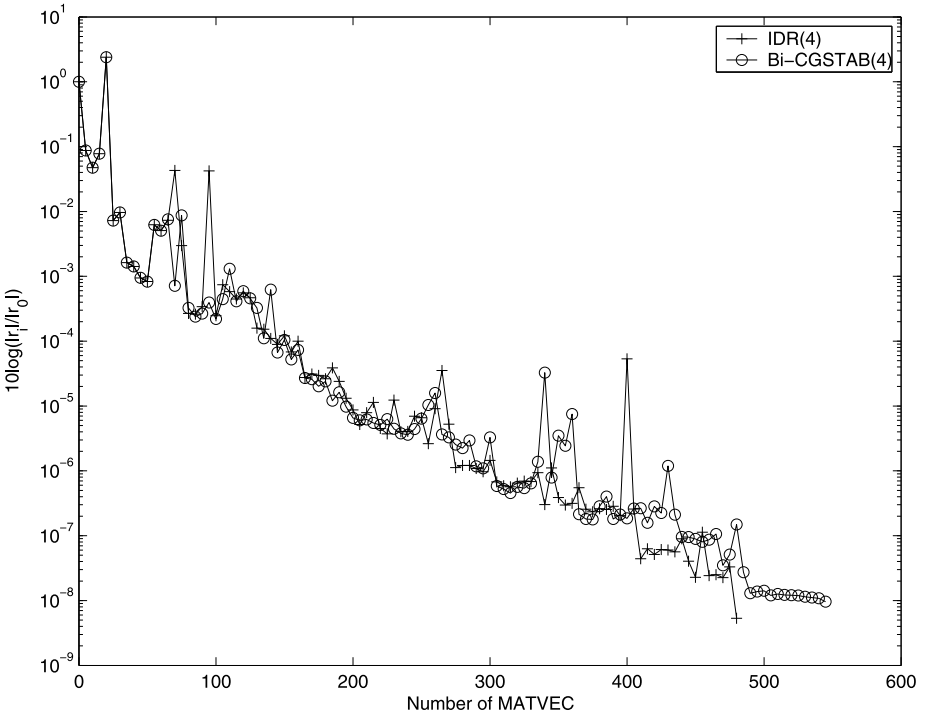

Fig. 2. Convergence of Bi-CGSTAB and IDR(s) for ADD20 with a four-dimensional shadow space.

IDR(s) performs well for this example. A significant reduction of the number of MVs is achieved if s is increased, and the accuracy that is reached remains satisfactory. Bi-CGSTAB, on the other hand, shows to be numerically less stable than IDR(s), and fails to compute an accurate solution for $s = 5$.

The above example shows that the generalized Bi-CGSTAB algorithm Algorithm 4 should not be considered as an improvement over IDR(s). However, we did not formulate the algorithm for this reason, but to reflect the insights we have gained about the relation between IDR and Bi-CGSTAB.

10. Conclusions and future work

The IDR approach offers an elegant view on transpose free Bi-CG methods with initial shadow residual of dimension s, with $s > 1$. The inclusion of such a higher dimensional shadow space in IDR proved to be remarkable effective and can be included in Bi-CG and Bi-CGSTAB as well. Inclusion in Bi-CGSTAB is less elegant than in IDR, and in its present formulation also numerically less stable. Nevertheless, with this inclusion, IDR and Bi-CGSTAB are equivalent.

Extensive experiments in [10], show that for many problems IDR with modest s, as $s = 4$, requires a similar number of MVs as GMRES for obtaining approximate solutions of comparable accuracy. But in contrast to GMRES, the additional costs per MV are fixed and modest for IDR.

However, experiments in [10] with BiCGstab(ℓ) (see [7]) for $\ell = 2$ show that there is still room for improvement: BiCGstab(2) is more robust. Quadratic polynomial factors with real coefficients appear to be more suitable then linear factor with real coefficients for strongly nonsymmetric problems. Inclusion of these higher degree factors in IDR is our next goal, and we believe that the insight into the relation between IDR and Bi-CGSTAB that we have obtained in this paper is an essential step towards achieving this.

Acknowledgements

Part of this research has been funded by the Dutch BSIK/BRICKS project.

References

[1] J.I. Aliaga, D.L. Boley, R.W. Freund, V. Hernández, A Lanczos-type method for multiple starting vectors, Math. Comp. 69 (2000) 1577–1601.
[2] S.C. Eisenstat, H.C. Elman, M.H. Schultz, Variational iterative methods for nonsymmetric systems of linear equations, SIAM J. Numer. Anal. 20 (2) (1983) 345–357.
[3] A. El Guennouni, K. Jbilou, H. Sadok, A block version of BiCGSTAB for linear systems with multiple right-hand sides, Electron. Trans. Numer. Anal. 16 (2003) 129–142 (electronic only).

[4] R.W. Freund, M. Malhotra, A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides, in: Proceedings of the Fifth Conference of the International Linear Algebra Society, Atlanta, GA, 1995, Linear Algebra Appl. 254 (1997) 119–157.
[5] D. O'Leary, The block conjugate gradient algorithm, Linear Algebra Appl. 99 (1980) 293–322.
[6] V. Simoncini, A stabilized qmr version of block bicg, SIAM J. Matrix Anal. Appl. 18 (2) (1997) 419–434, http://link.aip.org/link/?SML/18/419/1.
[7] G.L.G. Sleijpen, D.R. Fokkema, BiCGstab(l) for linear equations involving unsymmetric matrices with complex spectrum, Electron. Trans. Numer. Anal. 1 (1993) 11–32 (electronic only).
[8] G.L.G. Sleijpen, H.A. van der Vorst, Reliable updated residuals in hybrid Bi-CG methods, Computing 56 (2) (1996) 141–163.
[9] P. Sonneveld, CGS, a fast Lanczos-type solver for nonsymmetric linear systems, SIAM J. Sci. Statist. Comput. 10 (1989) 36–52.
[10] P. Sonneveld, M. van Gijzen, IDR(s): a family of simple and fast algorithms for solving large nonsymmetric systems of linear equations, Preprint, Delft Institute of Applied Mathematics, Delft University of Technology, Delft, The Netherlands, March 2007.
[11] 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 (2) (1992) 631–644.
[12] P. Wesseling, P. Sonneveld, Numerical experiments with a multiple grid- and a preconditioned Lanczos type method, in: Lecture Notes in Mathematics, vol. 771, Springer-Verlag, Berlin, 1980, pp. 543–562.
[13] M.-C. Yeung, T.F. Chan, ML(k)BiCGSTAB: A BiCGSTAB variant based on multiple Lanczos starting vectors, SIAM J. Sci. Comput. 21 (4) (1999) 1263–1290, http://link.aip.org/link/?SCE/21/1263/1.