Journal of Computational and Applied Mathematics 272 (2014) 57–69

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

ML(n)BiCGStabt: A ML(n)BiCGStab variant with **A**-transpose

Man-Chung Yeung

Department 3036, 1000 East University Avenue, Laramie, WY 82071, United States

ARTICLE INFO

Article history:
Received 18 January 2013
Received in revised form 27 January 2014

MSC:
primary 65F10
65F15
secondary 65F25
65F30

Keywords:
IDR
CGS
BiCGStab
ML(n)BiCGStab
Multiple starting Lanczos
Krylov subspace

ABSTRACT

The 1980 IDR method (Wesseling and Sonneveld, 1980 [12]) plays an important role in the history of Krylov subspace methods. It started the research of transpose-free Krylov subspace methods. The ML(n)BiCGStab method (Yeung, 2012) is one of such methods. In this paper, we present a new ML(n)BiCGStab variant that involves **A**-transpose in its implementation. Comparison of this new algorithm with the existing ML(n)BiCGStab algorithms and some other Krylov subspace algorithms will be presented.

1. Introduction

ML(n)BiCGStab is a transpose-free Krylov subspace method for the solution of linear systems

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{1.1}$$

where $\mathbf{A} \in \mathbb{C}^{N\times N}$ and $\mathbf{b} \in \mathbb{C}^N$. It was introduced by Yeung and Chan [1] in 1999 and its algorithms were recently reformulated by Yeung [2]. ML(n)BiCGStab is a natural generalization of BiCGStab [3], built from a multiple starting BiCG-like algorithm called ML(n)BiCG, through the Sonneveld–van der Vorst–Lanczos procedure (SVLP), namely, the procedure introduced by Sonneveld [4] and van der Vorst [3] to construct CGS and BiCGStab from BiCG [5]. In theory, ML(n)BiCGStab is a method that lies between the Lanczos-based BiCGStab and the Arnoldi-based GMRES/FOM [6]. In fact, it is a BiCGStab when $n = 1$ and becomes a GMRES/FOM when $n = N$ (see [2,7]). In computation, ML(n)BiCGStab can be much more stable and converge much faster than BiCGStab. We once tested it on the standard oil reservoir simulation test data called SPE9 which contains a sequence of linear systems and found that it reduced the total computational time by 60% when compared to BiCGStab. Tests made on the data from matrix markets also supported the superiority of ML(n)BiCGStab over BiCGStab. For details, one is referred to [2,1].

The author once constructed a new version of ML(n)BiCG where the left residuals are not just given by the monomial basis, but are orthogonalized against previous right-hand side residuals. In structure, this new ML(n)BiCG is closer to the classical BiCG than the one in [1] is. Numerical experiments, however, showed that this new ML(n)BiCG was unstable and weaker than the standard BiCG. Moreover, in [8], Yeung and Boley derived a SVLP from a one-sided multiple starting band Lanczos procedure (MSLP) with n left-starting and m right-starting vectors. From their experiments about the multi-input

E-mail address: myeung@uwyo.edu.

http://dx.doi.org/10.1016/j.cam.2014.04.011
0377-0427/

multi-output time-invariant linear dynamical systems, they observed that SVLP is more stable than the both-sided MSLP when $m \neq n$. These two comparing examples hint that, when $m \neq n$, a stable multiple starting procedure with **A**-transpose may come from a modification of a SVLP. In this paper, we make a first attempt in this direction by introducing **A**-transpose into ML(n)BiCGStab. We call the resulting algorithm ML(n)BiCGStabt [7,9], standing for ML(n)BiCGStab with transpose. We remark that $\mathbf{A}^H$ has been used in [10] to improve the parallelism of GPBiCG(m, l) [11]. Here we want to use $\mathbf{A}^H$ to enhance the numerical stability of ML(n)BiCGStab.

There exist two ML(n)BiCGStab algorithms, labeled as Algorithms 4.1 and 5.1 respectively in [2], derived from different definitions of the residual vectors $\mathbf{r}_k$. While both algorithms are numerically stable in general, one is relatively more stable than the other. ML(n)BiCGStabt is a modified version of Algorithm 5.1 so that it enjoys the same level of stability with Algorithm 4.1.

Other extensions of IDR [12], CGS and BiCGStab exist. Among them are BiCGStab2 [13], BiCGStab(l) [14], GPBi-CG [15], IDR(s) [16,17], IDRstab [18], GPBiCG(m, l) [11], and GBi-CGSTAB(s, l) [19]. Related articles include [20–23].

The outline of the paper is as follows. In Section 2, index functions in [8] are introduced. They are helpful in the construction of a ML(n)BiCGStab algorithm. In Section 3, we present the ML(n)BiCG algorithm from [1]. The derivation of every ML(n)BiCGStab algorithm is based on it. In Section 4, we introduce the ML(n)BiCGStabt algorithm and its properties. In Section 5, numerical experiments are presented, and in Section 6, concluding remarks are given.

2. Index functions

Let be given a $n \in \mathbb{N}$, the set of positive integers. For all $k \in \mathbb{Z}$, the set of all integers, we define

$$g_n(k) = \lfloor (k-1)/n \rfloor \quad \text{and} \quad r_n(k) = k - n g_n(k)$$

where $\lfloor \cdot \rfloor$ rounds its argument to the nearest integer towards minus infinity. We call g_n and r_n index functions; they are defined on $\mathbb{Z}$ with ranges $\mathbb{Z}$ and $\{1, 2, \ldots, n\}$, respectively.

If we write

$$k = jn + i \tag{2.1}$$

with $1 \leq i \leq n$ and $j \in \mathbb{Z}$, then

$$g_n(jn+i) = j \quad \text{and} \quad r_n(jn+i) = i.$$

3. ML(n)BiCG

Analogously to the derivation of BiCGStab from BiCG, the ML(n)BiCGStab algorithms [2] were derived from a BiCG-like algorithm named ML(n)BiCG, which was built upon a one-sided band Lanczos process with n left starting vectors and a single right starting vector. In this section, we present the ML(n)BiCG algorithm from [1].

Consider the solution of (1.1). Throughout the paper we do not assume the coefficient matrix **A** is nonsingular. In [2], we proved that ML(n)BiCG/ML(n)BiCGStab can solve a singular system almost surely provided that the underlying Krylov subspace contains a solution of (1.1).

Let be given n vectors $\mathbf{q}_1, \ldots, \mathbf{q}_n \in \mathbb{C}^N$, which we call *left starting vectors* or *shadow vectors*. Define

$$\mathbf{p}_k = \left(\mathbf{A}^H\right)^{g_n(k)} \mathbf{q}_{r_n(k)}, \quad k \in \mathbb{N}. \tag{3.1}$$

The following algorithm for the solution of (1.1) is from [1].

Algorithm 3.1. ML(n)BiCG

1. Choose an initial guess $\widehat{\mathbf{x}}_0$ and n vectors $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$.
2. Compute $\widehat{\mathbf{r}}_0 = \mathbf{b} - \mathbf{A}\widehat{\mathbf{x}}_0$ and set $\mathbf{p}_1 = \mathbf{q}_1$, $\widehat{\mathbf{g}}_0 = \widehat{\mathbf{r}}_0$.
3. For $k = 1, 2, \ldots,$ until convergence:
4. $\quad \alpha_k = \mathbf{p}_k^H \widehat{\mathbf{r}}_{k-1} / \mathbf{p}_k^H \mathbf{A}\widehat{\mathbf{g}}_{k-1}$;
5. $\quad \widehat{\mathbf{x}}_k = \widehat{\mathbf{x}}_{k-1} + \alpha_k \widehat{\mathbf{g}}_{k-1}$;
6. $\quad \widehat{\mathbf{r}}_k = \widehat{\mathbf{r}}_{k-1} - \alpha_k \mathbf{A}\widehat{\mathbf{g}}_{k-1}$;
7. $\quad$ For $s = \max(k-n, 0), \ldots, k-1$
8. $\quad\quad \beta_s^{(k)} = -\mathbf{p}_{s+1}^H \mathbf{A} \left(\widehat{\mathbf{r}}_k + \sum_{t=\max(k-n,0)}^{s-1} \beta_t^{(k)} \widehat{\mathbf{g}}_t \right) \Big/ \mathbf{p}_{s+1}^H \mathbf{A}\widehat{\mathbf{g}}_s$;
9. $\quad$ End
10. $\quad \widehat{\mathbf{g}}_k = \widehat{\mathbf{r}}_k + \sum_{s=\max(k-n,0)}^{k-1} \beta_s^{(k)} \widehat{\mathbf{g}}_s$;
11. $\quad$ Compute $\mathbf{p}_{k+1}$ according to (3.1)
12. End