Max Planck Institute Magdeburg
Preprints

Kapil Ahuja Eric de Sturler Lihong Feng Peter Benner

Recycling BiCGSTAB with an Application to Parametric Model Order Reduction

MPIMD/13-21 **November 16, 2013**

Abstract

Krylov subspace recycling is a process for accelerating the convergence of sequences of linear systems. Based on this technique we have recently developed the recycling BiCG algorithm. We now generalize and extend this recycling theory to BiCGSTAB. Recycling BiCG focuses on efficiently solving sequences of dual linear systems, while the focus here is on efficiently solving sequences of single linear systems (assuming non-symmetric matrices for both recycling BiCG and recycling BiCGSTAB).

As compared to other methods for solving sequences of single linear systems with non-symmetric matrices (e.g., recycling variants of GMRES), BiCG based recycling algorithms, like recycling BiCGSTAB, have the advantage that they involve a short-term recurrence, and hence, do not suffer from storage issues and are also cheaper with respect to the orthogonalizations.

We modify the BiCGSTAB algorithm to use a recycle space, which is built from left and right approximate eigenvectors. Using our algorithm for parametric model order reduction examples gives good results. We show about 40% reduction in iteration count when using recycling BiCGSTAB as compared to the one without recycling.

Impressum:

Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg

Publisher:
Max Planck Institute for
Dynamics of Complex Technical Systems

Address:
Max Planck Institute for
Dynamics of Complex Technical Systems
Sandtorstr. 1
39106 Magdeburg

www.mpi-magdeburg.mpg.de/preprints

1 Introduction

Numerical simulation is an essential tool for solving science and engineering problems. However, simulating large-scale models leads to overwhelming demands on computational resources. This is the main motivation for model reduction. The goal is to produce a surrogate model of much smaller dimension that provides a high-fidelity approximation of the input-output behavior of the original model. Often the models have design parameters associated with them, e.g., boundary conditions, geometry, material properties etc. Changes in these design parameters require generation of new reduced models, which makes the model reduction process very cumbersome. One practical application where such a challenge arises is micro-electro-mechanical systems (MEMS) design [10]. The goal of PMOR [5, 9] is to generate a reduced model such that parametric dependence, as in the original model, is preserved (or retained).

There are various ways of performing PMOR [7, 5, 18]. This includes moment matching, $\mathcal{H}_2$-optimality, and reduced basis approaches. For this work, we focus on moment matching based PMOR because of its flexibility (little limit on the systems properties) and low computational cost in many industrial applications.

Often moment matching based PMOR algorithms require efficient solution of sequences of linear systems of the following type:

$$A^{(\iota)}\ x^{(\iota,\kappa)} = b^{(\iota,\kappa)}, \tag{1}$$

where $A^{(\iota)} \in \mathbb{R}^{n\times n}$ varies with ι; $b^{(\iota,\kappa)} \in \mathbb{R}^n$ varies with both ι and κ; the matrices $A^{(\iota)}$ are large, sparse, and non-symmetric; and the change from one system to the next is small.

Krylov subspace methods are usually used for solving such large and sparse linear systems. For linear systems with non-symmetric matrices, GMRES [20] is one of the first choices, but it is generally not optimal with respect to the runtime. BiCGSTAB [22] is competitive with GMRES, and in many cases performs better than GMRES. Also, it does not suffer from storage issues, which is a problem in GMRES.

Krylov subspaces recycling is a technique for efficient solution of sequences of linear systems. Here, while solving one system in the sequence, approximate invariant subspaces of the matrix are selected and used to accelerate the convergence of the next system in the sequence. Since the matrices in the sequence do not change much, this provides substantial reduction in both iteration and time. See [17] and [4] for more about Krylov subspace recycling.

Here, we have a sequence of linear systems with non-symmetric matrices, and hence, the GCRO-DR [17] algorithm, which is a recycling variant of GMRES, is more suited. However, since there is no optimal method (again, with respect to runtime) for solving linear systems with non-symmetric matrices, there is no optimal method for solving sequences of such linear systems. Like GMRES, GCRO-DR also suffers from storage issues. Hence, we develop a recycling variant of BiCGSTAB. We explore the usage of recycling BiCGSTAB for a PMOR example, and show about 40% reduction in iteration count when using recycling as compared to not using recycling.

To simplify notation, we drop the superscripts ι and κ in (1). Throughout the paper,

$||\cdot||$ refers to the two-norm, $(\cdot,\cdot)$ refers to the inner product, $*$ indicates the conjugate transpose operation, and $\bar{\cdot}$ indicates complex conjugation.

The rest of the paper is divided into six more sections. The bi-Lanczos algorithm [15] and recycling BiCG [4] form the basis of our recycling BiCGSTAB. Hence, we revisit these in Sections 2 and 3, respectively. In Section 3, we also give a new result related to recycling BiCG. Next, we derive recycling BiCGSTAB in Section 4. In Section 5, we analyze the subspaces that can be used in recycling BiCGSTAB. Finally, we perform numerical experiments in Section 6, and give concluding remarks in Section 7.

2 The Bi-Lanczos Algorithm

Consider a primary system $Ax = b$, with x_0 the initial guess and $r_0 = b - Ax_0$ the residual. Also, consider an auxiliary dual system[1] $A^*\tilde{x} = \tilde{b}$, with $\tilde{b}$ a random vector, $\tilde{x}_0$ the initial guess, and $\tilde{r}_0 = \tilde{b} - A^*\tilde{x}_0$ the residual.

Let the columns of $V_i = [v_1\ v_2\ \ldots\ v_i]$ define the basis of the primary system Krylov space $\mathcal{K}^i(A, r_0) \equiv span\{r_0,\ Ar_0,\ A^2r_0,\ \cdots,\ A^{i-1}r_0\}$. Also, let the columns of $\tilde{V}_i = [\tilde{v}_1\ \tilde{v}_2\ \ldots\ \tilde{v}_i]$ define the basis of the dual system Krylov space $\tilde{\mathcal{K}}^i(A^*, \tilde{r}_0) \equiv span\{\tilde{r}_0,\ A^*\tilde{r}_0,\ A^{2*}\tilde{r}_0,\ \cdots,\ A^{(i-1)*}\tilde{r}_0\}$.

The bi-Lanczos algorithm computes columns of V_i and $\tilde{V}_i$ such that, in exact arithmetic, $V_i \perp_b \tilde{V}_i$, where $\perp_b$ is referred to as bi-orthogonality. The columns of V_i and $\tilde{V}_i$ are called Lanczos vectors. There is a degree of freedom in choosing the scaling of the Lanczos vectors [11, 12, 19]. Using the scaling

$$||v_i|| = 1, \qquad (v_i, \tilde{v}_i) = 1, \tag{2}$$

we initialize the Lanczos vectors as follows:

$$v_1 = \tfrac{r_0}{||r_0||}, \quad \tilde{v}_1 = \tfrac{\tilde{r}_0}{(v_1, \tilde{r}_0)}.$$

The $(i+1)$-th Lanczos vectors are given by

$$\begin{array}{rcl} \gamma v_{i+1} &=& Av_i - V_i\tau \perp \tilde{V}_i, \\ \tilde{\gamma}\tilde{v}_{i+1} &=& A^*\tilde{v}_i - \tilde{V}_i\tilde{\tau} \perp V_i, \end{array} \tag{3}$$

where γ and $\tilde{\gamma}$ are unknown scalars, and τ and $\tilde{\tau}$ are unknown vectors. The computation of the $(i+1)$-st Lanczos vectors requires only the i-th and the $(i-1)$-st Lanczos vectors (see [19]). These 3-term recurrences are called the bi-Lanczos relations, and are defined as follows:

$$\begin{array}{rcccl} AV_i &=& V_{i+1}\underline{T}_i &=& V_iT_i + t_{i+1,i}v_{i+1}e_i^T, \\ A^*\tilde{V}_i &=& \tilde{V}_{i+1}\underline{\tilde{T}}_i &=& \tilde{V}_i\tilde{T}_i + \tilde{t}_{i+1,i}\tilde{v}_{i+1}e_i^T, \end{array} \tag{4}$$

where T_i, $\tilde{T}_i$ are $i \times i$ tridiagonal matrices, $t_{i+1,i}$ is the last element of the last row of $\underline{T}_i \in \mathbb{C}^{(i+1)\times i}$, and $\tilde{t}_{i+1,i}$ is the last element of the last row of $\underline{\tilde{T}}_i \in \mathbb{C}^{(i+1)\times i}$.

[1] The system is real, but we are not interested in its solution.

3 Recycling BiCG Revisited

We first introduce a generalization of the bi-Lanczos algorithm [3]. We show that even for a pair of matrices that are not conjugate transpose of each other, one can build bi-orthogonal bases (for the associated two Krylov subspaces) using a short-term recurrence.

Expanding the search space to include a recycle space leads to an augmented bi-orthogonality condition. The augmented bi-Lanczos algorithm, as derived for recycling BiCG [4], computes bi-orthogonal bases for the two Krylov subspaces such that this augmented bi-orthogonality condition is satisfied. Next, we revisit augmented bi-Lanczos [2] and show that it is a special case of generalized bi-Lanczos. Finally, we list the recycling BiCG algorithm from [4].

There are numerous of ways of computing good bases for Krylov subspaces $\mathcal{K}^m(B, v_1)$ and $\mathcal{K}^m(\tilde{B}, \tilde{v}_1)$, where B and $\tilde{B}$ are $n \times n$ general matrices, and v_1 and $\tilde{v}_1$ are any two n dimensional vectors. Let the columns of $V_m = [v_1\ v_2\ \ldots\ v_m]$ and $\tilde{V}_m = [\tilde{v}_1\ \tilde{v}_2\ \ldots\ \tilde{v}_m]$ define one such pair of good bases for $\mathcal{K}^m(B, v_1)$ and $\mathcal{K}^m(\tilde{B}, \tilde{v}_1)$, respectively. We compute these bases using the following, in principle, full recurrences:

$$\beta_{i+1,i} v_{i+1} \quad = \quad B v_i - \beta_{ii} v_i - \beta_{i-1,i} v_{i-1} - \ldots - \beta_{1i} v_1, \tag{5}$$

$$\tilde{\beta}_{i+1,i} \tilde{v}_{i+1} \quad = \quad \tilde{B} \tilde{v}_i - \tilde{\beta}_{ii} \tilde{v}_i - \tilde{\beta}_{i-1,i} \tilde{v}_{i-1} - \ldots - \tilde{\beta}_{1i} \tilde{v}_1, \tag{6}$$

where $i \in \{1, 2, 3, \ldots, m-1\}$ and $\{\beta_{ij}\}$, $\{\tilde{\beta}_{ij}\}$ are scalars to be determined. We assume that for $i < m$, $\mathcal{K}^i(B, v_1)$ is not an invariant subspace of B (similarly, $\mathcal{K}^i(\tilde{B}, \tilde{v}_1)$ is not an invariant subspace of $\tilde{B}$ for $i < m$). We can rewrite (5) as follows:

$$B v_i = \beta_{1i} v_1 + \beta_{2i} v_2 + \ldots + \beta_{i-1,i} v_{i-1} + \beta_{ii} v_i + \beta_{i+1,i} v_{i+1}.$$

Combining these equations, for $i \in \{1, 2, 3, \ldots, m-1\}$, into matrix form we get

$$B[v_1\ v_2 \ldots v_{m-1}] = [v_1\ v_2 \ldots v_{m-2}\ v_{m-1}\ v_m] \begin{bmatrix} \beta_{11} & \beta_{12} & \ldots & \beta_{1,m-1} \\ \beta_{21} & \beta_{22} & \ldots & \beta_{2,m-1} \\ 0 & \beta_{32} & \ldots & \beta_{3,m-1} \\ 0 & 0 & \ldots & \beta_{4,m-1} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & \beta_{m-1,m-1} \\ 0 & 0 & \ldots & \beta_{m,m-1} \end{bmatrix},$$

or

$$B V_{m-1} = V_m \underline{H}_{m-1},$$

where $\underline{H}_{m-1}$ is an $m \times (m-1)$ upper Hessenberg matrix. This result also holds for each $i \in \{1, 2, 3, \ldots, m-1\}$, i.e.,

$$B V_i \quad = \quad V_{i+1} \underline{H}_i. \tag{7}$$

Similarly, using (6) and following the steps above, we get the following relation for the dual system:

$$\tilde{B}\tilde{V}_i \quad = \quad \tilde{V}_{i+1}\underline{\tilde{H}}_i. \tag{8}$$

The scalars $\{\beta_{ij}\}$ and $\{\tilde{\beta}_{ij}\}$ are determined by a choice of constraints. One option is to enforce that the columns of V_i (and $\tilde{V}_i$) are orthonormal vectors (as in the Arnoldi algorithm). Another option, as in the bi-Lanczos algorithm, is to enforce[2]

$$V_i \perp_b \tilde{V}_i, \quad ||v_i|| = 1, \quad \text{and} \quad (v_i, \tilde{v}_i) = 1,$$

or

$$\tilde{V}_i^* V_i = I \quad \text{and} \quad ||v_i|| = 1. \tag{9}$$

If $\tilde{B} = B^*$, then (7), (8), and (9) lead to the bi-Lanczos relations (4), which consist of three-term recurrences. Our goal here is to relax the condition $\tilde{B} = B^*$ and still obtain short-term recurrences.

Theorem 1. *Let $B, \tilde{B} \in \mathbb{C}^{n\times n}$, and let the following conditions hold:*

(a) $B - \tilde{B}^* = \tilde{F}_k\tilde{C}_k^* - C_k F_k^*$, where $C_k, \tilde{C}_k, F_k, \tilde{F}_k \in \mathbb{C}^{n\times k}$,

(b) $\forall x : Bx \perp \tilde{C}_k$, $\forall \tilde{x} : \tilde{B}\tilde{x} \perp C_k$,

(c) $v_1 \perp \tilde{C}_k$, and $\tilde{v}_1 \perp C_k$.

Also, let (9) be used as the set of constraints for (7) and (8). Then, $\beta_{ij} = 0$ and $\tilde{\beta}_{ij} = 0$ for $j > i+1$, which leads to the following three-term recurrences:

$$\begin{aligned} \beta_{i+1,i}v_{i+1} &= Bv_i - \beta_{ii}v_i - \beta_{i-1,i}v_{i-1}, \\ \tilde{\beta}_{i+1,i}\tilde{v}_{i+1} &= \tilde{B}\tilde{v}_i - \tilde{\beta}_{ii}\tilde{v}_i - \tilde{\beta}_{i-1,i}\tilde{v}_{i-1}, \end{aligned}$$

for $i \in \{1, 2, 3, \ldots, m-1\}$.

Proof. Using (b) and (c) we can show that

$$C_k^*\tilde{V}_i = 0 \quad \text{and} \quad \tilde{C}_k^* V_i = 0. \tag{10}$$

We show $C_k^*\tilde{V}_i = 0$ by induction. One can similarly show that $\tilde{C}_k^* V_i = 0$. $C_k^*\tilde{v}_1 = 0$ by (c). Let $C_k^*\tilde{v}_l = 0$ for $l = \{1, 2, \ldots, i\}$, and consider the case $l = i+1$. From (6) we know that

$$\tilde{\beta}_{i+1,i}\tilde{v}_{i+1} = \tilde{B}\tilde{v}_i - \tilde{\beta}_{ii}\tilde{v}_i - \tilde{\beta}_{i-1,i}\tilde{v}_{i-1} - \ldots - \tilde{\beta}_{1i}\tilde{v}_1.$$

Then, $C_k^*\tilde{v}_{i+1} = 0$ since $C_k^*\tilde{B}\tilde{v}_i = 0$ using (b) and $\tilde{\beta}_{li}C_k^*\tilde{v}_l = 0$ for $l \in \{1, 2, \ldots, i\}$ by the induction hypothesis[3]. This proves (10). Multiplying both sides in (7) by $\tilde{V}_i^*$ and using (9) we get

$$\tilde{V}_i^* B V_i = H_i.$$

[2] In this paper, for ease of exposition, we assume breakdowns do not happen. Hence, $(\tilde{v}_i, v_i) \neq 0$.

[3] Note that our earlier assumption, $\mathcal{K}^i(\tilde{B}, \tilde{v}_1)$ is not an invariant subspace of $\tilde{B}$ for $i < m$, shows that $\tilde{\beta}_{i+1,i} \neq 0$.

Substituting (a) in the above equation leads to

$$\begin{array}{rcl} \tilde{V}_i^* \left(\tilde{B}^* + \tilde{F}_k \tilde{C}_k^* - C_k F_k^* \right) V_i & = & H_i \quad \Longleftrightarrow \\ \tilde{V}_i^* \tilde{B}^* V_i + \tilde{V}_i^* \tilde{F}_k \tilde{C}_k^* V_i - \tilde{V}_i^* C_k F_k^* V_i & = & H_i. \end{array}$$

Using (10) we get

$$\begin{array}{rcl} \tilde{V}_i^* \tilde{B}^* V_i & = & H_i \quad \Longleftrightarrow \\ (\tilde{B}\tilde{V}_i)^* V_i & = & H_i. \end{array}$$

Finally, using (8) and (9) in the above equation gives

$$\tilde{H}_i^* = H_i.$$

This implies both H_i and $\tilde{H}_i$ are tridiagonal matrices, and hence $\beta_{ij} = 0$ and $\tilde{\beta}_{ij} = 0$ for $j > i + 1$. □

We now revisit augmented bi-Lanczos [2] and show that it is a special case of generalized bi-Lanczos. The BiCG algorithm is primarily used where the dual system is not auxiliary. That is, one needs to solve both a primary system and a dual system. The RBiCG algorithm was developed to accelerate the convergence of sequences of such systems.

In RBiCG, we use the matrix U to define the primary system recycle space, and compute $C = A^{(\iota+1)}U$, where U is derived from an approximate right invariant subspace of $A^{(\iota)}$ and ι denotes the index of the linear system in the sequence of linear systems; see (1). Similarly, we use the matrix $\tilde{U}$ to define the dual system recycle space, and compute $\tilde{C} = A^{(\iota+1)*}\tilde{U}$, where $\tilde{U}$ is derived from an approximate left invariant subspace of $A^{(\iota)}$. U and $\tilde{U}$ are computed such that C and $\tilde{C}$ are bi-orthogonal (see page 35 of [3]). The number of vectors selected for recycling is denoted by k, and hence, U, $\tilde{U}$, C, and $\tilde{C} \in \mathbb{C}^{n\times k}$.

The bi-Lanczos algorithm was modified to compute the columns of V_i and $\tilde{V}_i$ such that

$$[C \; V_i] \perp_b \left[\tilde{C} \; \tilde{V}_i\right].$$

Using the scaling (2), we initialize the Lanczos vectors as

$$v_1 = \frac{\left(I - C\mathcal{D}_c^{-1}\tilde{C}^*\right) r_0}{\left|\left|\left(I - C\mathcal{D}_c^{-1}\tilde{C}^*\right) r_0\right|\right|}, \quad \tilde{v}_1 = \frac{\left(I - \tilde{C}\mathcal{D}_c^{-1}C^*\right) \tilde{r_0}}{\left(v_1, \left(I - \tilde{C}\mathcal{D}_c^{-1}C^*\right) \tilde{r_0}\right)}.$$

Here $\mathcal{D}_c = \tilde{C}^*C$ is a diagonal matrix (implied by $C \perp_b \tilde{C}$; we also enforce $\mathcal{D}_c$ to have positive, real coefficients). As for the biLanczos algorithm in (3), the $(i+1)$-st Lanczos vectors here are given by

$$\begin{array}{rcl} \gamma v_{i+1} & = & Av_i - V_i\tau - C\rho \perp \left[\tilde{C} \; \tilde{V}_i\right], \\ \tilde{\gamma}\tilde{v}_{i+1} & = & A^*\tilde{v}_i - \tilde{V}_i\tilde{\tau} - \tilde{C}\tilde{\rho} \perp [C \; V_i], \end{array}$$

where γ, $\tilde{\gamma}$, τ, $\tilde{\tau}$, ρ, and $\tilde{\rho}$ are to be determined. The computation of the $(i+1)$-st Lanczos vector for the primary system now requires the i-th and $(i-1)$-st Lanczos vectors and C (see [2]). This gives a $(3+k)$-term recurrence, where k is the number of columns of C. Similarly, we get a $(3+k)$-term recurrence for computing the Lanczos vectors for the dual system. We refer to this pair of $(3+k)$-term recurrences as the augmented bi-Lanczos relations, and they are given by

$$\begin{aligned}(I - C\hat{C}^*)AV_i &= V_{i+1}\underline{T}_i,\\ (I - \tilde{C}\check{C}^*)A^*\tilde{V}_i &= \tilde{V}_{i+1}\underline{\tilde{T}}_i,\end{aligned}$$

where

$$\begin{aligned}\hat{C} = \left[\frac{1}{c_1^*\tilde{c}_1}\tilde{c}_1 \;\; \frac{1}{c_2^*\tilde{c}_2}\tilde{c}_2 \;\; \cdots \;\; \frac{1}{c_k^*\tilde{c}_k}\tilde{c}_k\right] &= \tilde{C}\mathcal{D}_c^{-1},\\ \check{C} = \left[\frac{1}{\tilde{c}_1^* c_1}c_1 \;\; \frac{1}{\tilde{c}_2^* c_2}c_2 \;\; \cdots \;\; \frac{1}{\tilde{c}_k^* c_k}c_k\right] &= C\mathcal{D}_c^{-1}.\end{aligned} \tag{11}$$

Theorem 2. *Let $v_1 = \eta(I - C\mathcal{D}_c^{-1}\tilde{C}^*)r_0$, $\tilde{v}_1 = \tilde{\eta}(I - \tilde{C}\mathcal{D}_c^{-1}C^*)\tilde{r}_0$, $B = (I - C\mathcal{D}_c^{-1}\tilde{C}^*)A$, and $\tilde{B} = (I - \tilde{C}\mathcal{D}_c^{-1}C^*)A^*$, where η, $\tilde{\eta}$ are scalars and C, $\tilde{C} \in \mathbb{C}^{n\times k}$ s.t. $\mathcal{D}_c = \tilde{C}^*C$ is a diagonal matrix with positive, real coefficients. Also, let (9) be used as the set of constraints for (7) and (8). Then, $\beta_{ij} = 0$ and $\tilde{\beta}_{ij} = 0$ for $j > i+1$, which leads to the following short-term recurrences:*

$$\begin{aligned}\beta_{i+1,i}v_{i+1} &= Bv_i - \beta_{ii}v_i - \beta_{i-1,i}v_{i-1},\\ \tilde{\beta}_{i+1,i}\tilde{v}_{i+1} &= \tilde{B}\tilde{v}_i - \tilde{\beta}_{ii}\tilde{v}_i - \tilde{\beta}_{i-1,i}\tilde{v}_{i-1},\end{aligned}$$

for $i \in \{1, 2, 3, \ldots, m-1\}$.

Proof. We show that conditions (a) – (c) of Theorem 1 are satisfied. This demonstrates that augmented bi-Lanczos is a special case of generalized bi-Lanczos. We have $B, \tilde{B} \in \mathbb{C}^{n\times n}$ such that

$$\begin{aligned}B - \tilde{B}^* &= A - C\mathcal{D}_c^{-1}\tilde{C}^*A - A + AC\mathcal{D}_c^{-1}\tilde{C}^*\\ &= \left(AC\mathcal{D}_c^{-1}\right)\tilde{C}^* - C\left(A^*\tilde{C}\mathcal{D}_c^{-1}\right)^*.\end{aligned}$$

Defining $F = A^*\tilde{C}\mathcal{D}_c^{-1}$ and $\tilde{F} = AC\mathcal{D}_c^{-1}$ we get

$$B - \tilde{B}^* = \tilde{F}\tilde{C}^* - CF^* \quad \text{where} \quad C, \tilde{C}, F, \tilde{F} \in \mathbb{C}^{n\times k}.$$

Hence (a) is satisfied. For any $\tilde{x}$ consider the following:

$$\begin{aligned}C^*\tilde{B}\tilde{x} &= C^*(I - \tilde{C}\mathcal{D}_c^{-1}C^*)A^*\tilde{x}\\ &= (C^* - \mathcal{D}_c\mathcal{D}_c^{-1}C^*)A^*\tilde{x} = 0.\end{aligned}$$

Similarly, for any x consider the following:

$$\begin{aligned}\tilde{C}^*Bx &= \tilde{C}^*(I - C\mathcal{D}_c^{-1}\tilde{C}^*)Ax\\ &= (\tilde{C}^* - \mathcal{D}_c\mathcal{D}_c^{-1}\tilde{C}^*)Ax = 0.\end{aligned}$$

Hence (b) is satisfied. Similarly, for v_1 and $\tilde{v}_1$ chosen in the theorem, $\tilde{C}^*v_1 = 0$ and $C^*\tilde{v}_1 = 0$. Hence, (c) is satisfied. ∎

Algorithm 1. *RBiCG [4]*
1. Given U (also $C = AU$) and $\tilde{U}$ (also $\tilde{C} = A^*\tilde{U}$) s.t. $C \perp_b \tilde{C}$, compute $\check{C}$ and $\hat{C}$ using (11). If U and $\tilde{U}$ are not available, then initialize U, $\tilde{U}$, $\check{C}$, and $\hat{C}$ to empty matrices.
2. Choose x_{-1}, $\tilde{x}_{-1}$ and compute x_0, $\tilde{x}_0$, r_0, and $\tilde{r}_0$ using (12).
3. **if** $(r_0, \tilde{r}_0) = 0$ **then** initialize $\tilde{x}_{-1}$ to a random vector.
4. Set $p_0 = 0$, $\tilde{p}_0 = 0$, and $\beta_0 = 0$. Choose `tol` and `max_itn`.
5. **for** $i = 1 \ldots$ `max_itn` **do**
   - $\diamond$ $p_i = r_{i-1} + \beta_{i-1} p_{i-1}$; $\quad \tilde{p}_i = \tilde{r}_{i-1} + \bar{\beta}_{i-1}\tilde{p}_{i-1}$
   - $\diamond$ $z_i = Ap_i$; $\quad \tilde{z}_i = A^*\tilde{p}_i$
   - $\diamond$ $\zeta_i = \hat{C}^* z_i$; $\quad \tilde{\zeta}_i = \check{C}^*\tilde{z}_i$;
   - $\diamond$ $q_i = z_i - C\zeta_i$; $\quad \tilde{q}_i = \tilde{z}_i - \tilde{C}\tilde{\zeta}_i$
   - $\diamond$ $\alpha_i = (\tilde{r}_{i-1}, r_{i-1})/(\tilde{p}_i, q_i)$; $\quad \tilde{\alpha}_i = \bar{\alpha}_i$
   - $\diamond$ $\zeta_c = \zeta_c + \alpha_i\zeta_i$; $\quad \tilde{\zeta}_c = \tilde{\zeta}_c + \tilde{\alpha}_i\tilde{\zeta}_i$
   - $\diamond$ $x_i = x_{i-1} + \alpha_i p_i$ $\quad \tilde{x}_i = \tilde{x}_{i-1} + \tilde{\alpha}_i\tilde{p}_i$
   - $\diamond$ $r_i = r_{i-1} - \alpha_i q_i$ $\quad \tilde{r}_i = \tilde{r}_{i-1} - \tilde{\alpha}_i\tilde{q}_i$
   - $\diamond$ **if** $||r_i|| \leq$ `tol` and $||\tilde{r}_i|| \leq$ `tol` **then break**
   - $\diamond$ $\beta_i = (\tilde{r}_i, r_i)/(\tilde{r}_{i-1}, r_{i-1})$
6. **end for**
7. $x_i = x_i - U\zeta_c$; $\quad \tilde{x}_i = \tilde{x}_i - \tilde{U}\tilde{\zeta}_c$

For ease of future derivations, we introduce a slight change of notation. Let x_{-1} and $\tilde{x}_{-1}$ be the initial guesses and $r_{-1} = b - Ax_{-1}$ and $\tilde{r}_{-1} = \tilde{b} - A^*\tilde{x}_{-1}$ the corresponding initial residuals. We define

$$\begin{aligned} x_0 &= x_{-1} + U\hat{C}^* r_{-1}, \qquad r_0 = (I - C\hat{C}^*)r_{-1}, \\ \tilde{x}_0 &= \tilde{x}_{-1} + \tilde{U}\check{C}^*\tilde{r}_{-1}, \qquad \tilde{r}_0 = (I - \tilde{C}\check{C}^*)\tilde{r}_{-1}, \end{aligned} \tag{12}$$

and follow this convention for x_0, $\tilde{x}_0$, r_0, and $\tilde{r}_0$ for the rest of the paper. Algorithm 1 gives recycling BiCG (also termed as RBiCG) algorithm from [4].

4 Recycling BiCGSTAB

In RBiCG [3, 4], the iteration vectors p, $\tilde{p}$, r, and $\tilde{r}$ are updated using the following recurrences:

$$\begin{aligned} p_i &= r_{i-1} + \beta_{i-1} p_{i-1}, \quad \tilde{p}_i &= \tilde{r}_{i-1} + \tilde{\beta}_{i-1}\tilde{p}_{i-1}, \\ r_i &= r_{i-1} - \alpha_i B p_i, \qquad \tilde{r}_i &= \tilde{r}_{i-1} - \tilde{\alpha}_i \tilde{B}\tilde{p}_i, \end{aligned}$$

where $B = (I - C\hat{C}^*)A$ and $\tilde{B} = (I - \tilde{C}\check{C}^*)A^*$. We first give the polynomial representations of these iteration vectors.

Theorem 3. *For the primary system*

$$r_i = \Theta_i(B)r_0, \quad p_i = \Pi_{i-1}(B)r_0,$$

where $\Theta_i(K)$ and $\Pi_{i-1}(K)$ are i-th and $(i-1)$-st degree polynomials in K with the following polynomial recurrences:

$$\begin{aligned}\Theta_i(K) &= \Theta_{i-1}(K) - \alpha_i K \Pi_{i-1}(K),\\ \Pi_{i-1}(K) &= \Theta_{i-1}(K) + \beta_{i-1}\Pi_{i-2}(K).\end{aligned}$$

Similarly, for the dual system

$$\tilde{r}_i = \bar{\Theta}_i(\tilde{B})\tilde{r}_0, \quad \tilde{p}_i = \bar{\Pi}_{i-1}(\tilde{B})\tilde{r}_0,$$

where $\bar{\Theta}_i(K)$ and $\bar{\Pi}_i(K)$ satisfy the following polynomial recurrences:

$$\begin{aligned}\bar{\Theta}_i(K) &= \bar{\Theta}_{i-1}(K) - \bar{\alpha}_i K \bar{\Pi}_{i-1}(K),\\ \bar{\Pi}_{i-1}(K) &= \bar{\Theta}_{i-1}(K) + \bar{\beta}_{i-1}\bar{\Pi}_{i-2}(K).\end{aligned}$$

Proof. This can be proved by induction. Follow the steps as done in [21] (Section 2; pages 37–40), but use B instead of A and $\tilde{B}$ instead of A^*. □

From RBiCG we know $r_i \perp \tilde{r}_j$ for $j < i$. Using Theorem 3 we get that

$$(\bar{\Theta}_j(\tilde{B})\tilde{r}_0, \Theta_i(B)r_0) = 0 \text{ for } j < i.$$

This implies $\Theta_i(B)r_0 \perp \mathcal{K}^i(\tilde{B}, \tilde{r}_0)$, where $\tilde{r}_0$, $\tilde{B}\tilde{r}_0$, $\cdots$, $\tilde{B}^{i-1}\tilde{r}_0$ span the subspace $\mathcal{K}^i(\tilde{B}, \tilde{r}_0)$. As observed in [22], the above orthogonality conditions must be satisfied by other bases of $\mathcal{K}^i(\tilde{B}, \tilde{r}_0)$, too. So, other polynomials can be used as well [23]. That is,

$$(\bar{\Omega}_j(\tilde{B})\tilde{r}_0, \Theta_i(B)r_0) = 0 \text{ for } j < i. \tag{13}$$

Similar to what is done in [22], we define

$$\bar{\Omega}_i(\tilde{B}) \quad = \quad (I - \bar{\omega}_1\tilde{B})(I - \bar{\omega}_2\tilde{B})\ldots(I - \bar{\omega}_i\tilde{B}),$$

where ω_i is selected to minimize the residual r_i w.r.t. ω_i. Then, as first proposed in [21], instead of (13) we use the following form of inner product:

$$(\tilde{r}_0, \Omega_j(B)\Theta_i(B)r_0) = 0 \text{ for } j < i,$$

with

$$\Omega_i(B) \quad = \quad (I - \omega_1 B)(I - \omega_2 B)\ldots(I - \omega_i B).$$

This inner product does not require the transpose of B, and hence, is appropriate when there is no dual system to solve. Computing the inner product in this fashion, we obtain the recycling BiCGSTAB algorithm (similar to the way BiCGSTAB is obtained from BiCG in [22]). We term our recycling BiCGSTAB as RBiCGSTAB. The algorithm is given in Algorithm 2.

Algorithm 2. *RBiCGSTAB*
1. Given U (also $C = AU$) and $\tilde{U}$ (also $\tilde{C} = A^*\tilde{U}$) s.t. $C \perp_b \tilde{C}$, compute $D_c = C\tilde{C}^*$, $Z = (I - U\mathcal{D}_c^{-1}\tilde{C}^*A)$, and $B = (I - C\mathcal{D}_c^{-1}\tilde{C}^*)A$.
2. Choose $\tilde{b}$, x_{-1}, and $\tilde{x}_{-1}$, and compute x_0, $\tilde{x}_0$, r_0, and $\tilde{r}_0$ using (12).
3. **if** $(r_0, \tilde{r}_0) = 0$ **then** initialize either $\tilde{b}$ or x_{-1} or $\tilde{x}_{-1}$ to a random vector.
4. Set $p_0 = 0$ and $\beta_0 = 0$. Choose `tol` and `max_itn`.
5. **for** $i = 1 \ldots$ `max_itn` **do**
   - $\diamond$ $p_i = r_{i-1} + \beta_{i-1}p_{i-1} - \beta_{i-1}\omega_{i-1}Bp_{i-1}$
   - $\diamond$ $\alpha_i = \frac{(\tilde{r}_0,\ r_{i-1})}{(\tilde{r}_0,\ Bp_i)}$
   - $\diamond$ $s_i = r_{i-1} - \alpha_i Bp_i$
   - $\diamond$ $t_i = Bs_i$
   - $\diamond$ $\omega_i = \frac{(s_i,t_i)}{(t_i,t_i)}$
   - $\diamond$ $x_i = x_{i-1} + \alpha_i Zp_i + \omega_i Zs_i$
   - $\diamond$ $r_i = r_{i-1} - \alpha_i Bp_i - \omega_i Bs_i$
   - $\diamond$ **if** $||r_i|| \leq$ `tol` **then break**
   - $\diamond$ $\beta_i = \frac{(\tilde{r}_0,\ r_i)}{(\tilde{r}_0,\ r_{i-1})} \cdot \frac{\alpha_i}{\omega_i}$
6. **end for**

5 Analysis

For BiCG, it has been shown that including a left eigenvector into the search space leads to the removal of the corresponding right eigenvector from the right residual (and vice versa) [8]. In our experiments we demonstrate that recycling left eigenvectors may improve the convergence rate in the RBiCGSTAB algorithm. We consider two examples. The first example is a 1600×1600 linear system that we obtain from vertex centered finite volume discretization of the PDE

$$-(\mathsf{p}\mathsf{u}_\mathsf{x})_\mathsf{x} - (\mathsf{q}\mathsf{u}_\mathsf{y})_\mathsf{y} + \mathsf{r}\mathsf{u}_\mathsf{x} + \mathsf{s}\mathsf{u}_\mathsf{y} + \mathsf{t}\mathsf{u} = \mathsf{f},$$

on the unit square with $\mathsf{p} = 1, \mathsf{q} = 1, \mathsf{r} = 10, \mathsf{s} = -10, \mathsf{t} = 0$, and $\mathsf{f} = 0$. We use the following boundary conditions: $\mathsf{u}_{\text{south}} = 1, \mathsf{u}_{\text{west}} = 1, \mathsf{u}_{\text{north}} = 0, \mathsf{u}_{\text{east}} = 0$. We do not use a preconditioner in this example, the initial guess is a vector of all ones, and the relative convergence tolerance is 10^{-10}.

For the second example, the linear system is obtained by finite difference discretization of the partial differential equation [22]

$$-(\mathsf{A}\mathsf{v}_\mathsf{x})_\mathsf{x} - (\mathsf{A}\mathsf{v}_\mathsf{y})_\mathsf{y} + \mathsf{B}(\mathsf{x},\mathsf{y})\mathsf{v}_\mathsf{x} = \mathsf{F},$$

with A as shown in Figure 1, $\mathsf{B}(\mathsf{x},\mathsf{y}) = 2\mathsf{e}^{2(\mathsf{x}^2+\mathsf{y}^2)}$, and $\mathsf{F} = 0$ everywhere except in a small square in the center where $\mathsf{F} = 100$ (see Figure 1). The domain is $(0,1) \times (0,1)$ with Dirichlet boundary conditions

$$\mathsf{v}(0,\mathsf{y}) = \mathsf{v}(1,\mathsf{y}) = \mathsf{v}(\mathsf{x},0) = 1,$$
$$\mathsf{v}(\mathsf{x},1) = 0.$$

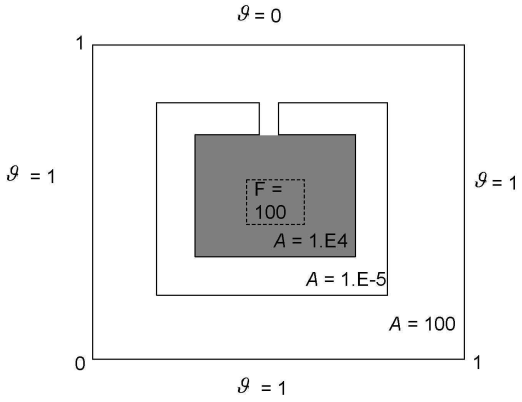


Figure 1: Coefficients for the PDE.

The discretization leads to a linear system with 16129 unknowns. We use an ILUTP [19] preconditioner with a drop tolerance of 0.2 (split-preconditioned). The initial guess is 0.5 times a vector of all ones, and the relative convergence tolerance is 10^{-8}.

For each example we do three experiments. First, we solve the system without recycling. Second, we use the right eigenvectors (corresponding to the largest magnitude eigenvalues) to span the recycle space. This is implemented by setting $\tilde{U} = U$. Finally, we use both left and right eigenvectors (again, corresponding to the largest magnitude eigenvalues) to span the recycle space.

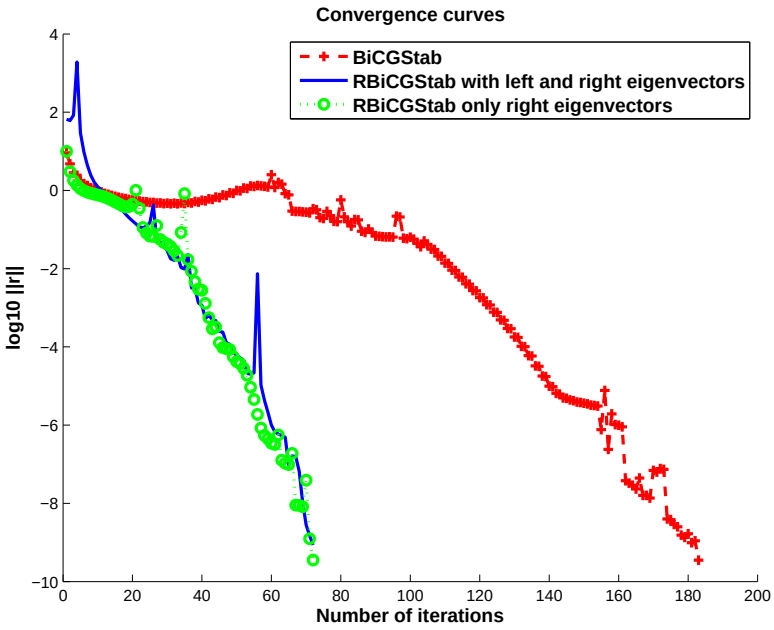
For the first example's second set of experiments, we use five exact right eigenvectors computed using the MATLAB function `eigs`. For the first example's third set of experiments, we use five exact left eigenvectors and five exact right eigenvectors (for a total of ten), again computed using the MATLAB function `eigs`.

For the second example's second set of experiments, we use twenty approximate right eigenvectors obtained by solving the problem once with RBiCG. For the second example's third set of experiments, we use twenty approximate left eigenvectors and twenty approximate right eigenvectors (for a total of forty), again obtained by solving the problem once with RBiCG.

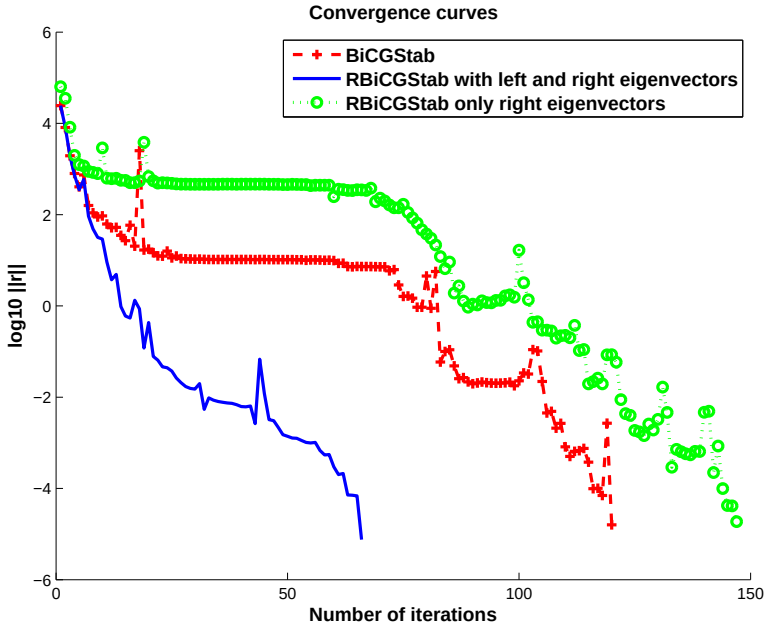
The results are shown in Figures 2 (a) and (b). For the first example, using right eigenvectors or using both left and right eigenvectors works equally well. However, for the second example, we see that using only right eigenvectors leads to convergence that is worse than BiCGSTAB without recycling, and much worse than RBiCGSTAB using both left and right eigenvectors.

6 Application to PMOR

Our test dynamical system comes from a silicone nitride membrane model [6]. Such a membrane can be part of many devices, for e.g., a gas sensor chip, a microthruster, an optimical filter etc. We use the moment matching based PMOR algorithm, described



(a) Example 1: Left eigenvectors not needed, but recycling effective.



(b) Example 2: Left eigenvectors needed for recycling to be effective.

Figure 2: Convergence curves for two examples using RBiCGSTAB. The 2nd example demonstrates that recycling left eigenvectors may improve the convergence rate in the RBiCGSTAB algorithm.

in [10], to compute a reduced model. This leads to a sequence of linear systems of the form (1) and size 60020. Whenever the matrix changes in the sequence, we call RBiCG to perform the linear solve. This helps to approximate both the left and right eigenvectors, which are not easily available from the RBiCGSTAB iterations[4]. The primary system right-hand side comes from the PDE. We take a vector of all ones as the dual system right-hand side. We call RBiCGSTAB for all remaining systems with the same matrix. This corresponds to linear systems where only the right-hand sides change. This is an effective strategy because it has been shown that the recycle space can be useful for multiple consecutive systems [17, 14, 16]. For this experiment, we take s, which is the parameter used in RBiCG[5], as 25 and $k = 20$. These are chosen based on experience with other recycling algorithms [17].

The number of iterations required to solve systems 1 through 63 are given in Figure 3. The peaks in the recycling BiCGSTAB plot correspond to when the matrix changes and RBiCG is called (three times). For all other steps, when only the right-hand side changes, RBiCGSTAB is called. We compare our results with BiCGSTAB. The RBiCG and RBiCGSTAB combination requires about 40% fewer total iterations. This demonstrates the effectiveness of recycling Krylov subspaces for PMOR.

7 Conclusion

For solving linear systems with non-symmetric matrices, BiCGSTAB is one of the best available algorithms. As compared with GMRES, which is the most commonly used algorithm for such linear systems, it has the advantage of having to work with a short term recurrence, and hence, does not suffer from storage issues.

For solving sequences of linear systems with non-symmetric matrices, it is advantageous to use Krylov subspace recycling for the BiCGSTAB algorithm, and hence we propose the RBiCGSTAB algorithm. We have demonstrated the usefulness of RBiCGSTAB for a parametric model order reduction example.

In the future, we plan to do timing experiments for the current example. We also plan to test RBiCGSTAB for other application areas (e.g., acoustics problems).

References

[1] A. M. Abdel-Rehim, W. Wilcox, and R. B. Morgan. Deflated BiCGStab for linear equations in QCD problems. In *Proceedings of Science, LAT2007*, pages 026/1–026/7, 2007.

[2] K. Ahuja. Recycling bi-Lanczos algorithms: BiCG, CGS, and BiCGSTAB. Master's thesis, Department of Mathematics, Virginia Tech, August 2009. Advised by

[4] Sometimes left eigenvectors are available from right eigenvectors [1, 16].

[5] Instead of using all the Lanczos vectors to update the recycle space, we update the recycle space periodically. This strategy keeps the memory requirements modest. The iteration process between two updates of the recycle space is referred to as a cycle. The length of the cycle, s, refers to the number of iterations between updates.

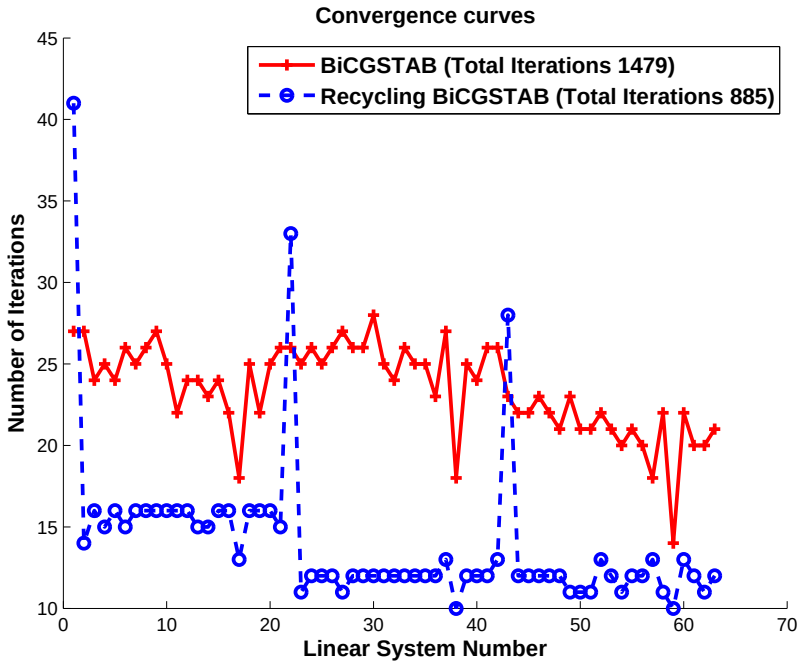


Figure 3: Comparison of iteration count when using RBiCGSTAB and BiCGSTAB as the linear solvers for PMOR.

E. de Sturler. Available from `http://scholar.lib.vt.edu/theses/available/etd-08252009-161256/`.

[3] K. Ahuja. *Recycling Krylov Subspaces and Preconditioners*. PhD thesis, Department of Mathematics, Virginia Tech, October 2011. Advised by E. de Sturler. Available from `http://scholar.lib.vt.edu/theses/available/etd-11112011-010340/`.

[4] K. Ahuja, E. de Sturler, S. Gugercin, and E. Chang. Recycling BiCG with an application to model reduction. *SIAM Journal on Scientific Computing*, 34(4):A1925–A1949, 2012.

[5] U. Baur, C. Beattie, P. Benner, and S. Gugercin. Interpolatory projection methods for parameterized model reduction. *SIAM Journal on Scientific Computing*, 33(5):2489–2518, 2011.

[6] T. Bechtold, D. Hohlfeld, E.B. Rudnyi, and M. Günther. Efficient extraction of thin-film thermal parameters from numerical models via parametric model order reduction. *Journal of Micromechanics and Microengineering*, 20(4):045030 (13pp), 2010.

[7] P. Benner, S. Gugercin, and K. Willcox. A Survey of Model Reduction Methods for Parametric Systems. Technical Report MPIMD/13-14, Max Planck Institute Magdeburg, 2013.

[8] E. de Sturler. BiCG explained. In *Householder Symposium XIV, Proceedings of the Householder International Symposium in Numerical Algebra, Chateau Whistler, Whistler, BC, Canada*, June 13–19, 1999.

[9] L. Feng, P. Benner, and J. Korvink. Parametric model order reduction accelerated by subspace recycling. In *Proceedings of 48th IEEE Conference on Decision & Control and 28th Chinese Control Conference*, pages 4328–4333, 2009.

[10] L. Feng, P. Benner, and J. G. Korvink. Subspace recycling accelerates the parametric macro-modeling of MEMS. *International Journal for Numerical Methods in Engineering*, 94(1):84–110, 2013.

[11] A. Greenbaum. *Iterative Methods for Solving Linear Systems*. SIAM, 1997.

[12] M. H. Gutknecht. Lanczos-type solvers for nonsymmetric linear systems of equations. *Acta Numerica*, 6:271–397, 1997.

[13] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[14] M. E. Kilmer and E. de Sturler. Recycling subspace information for diffuse optical tomography. *SIAM Journal on Scientific Computing*, 27(6):2140–2166, 2006.

[15] C. Lanczos. Solution of systems of linear equations by minimized iterations. *Journal of Research of the National Bureau of Standards*, 49:33–53, 1952.

[16] R. B. Morgan and D. A. Nicely. Restarting the nonsymmetric Lanczos algorithm. Unpublished manuscript, 2010.

[17] M. L. Parks, E. de Sturler, G. Mackey, D. D. Johnson, and S. Maiti. Recycling Krylov subspaces for sequences of linear systems. *SIAM Journal on Scientific Computing*, 28(5):1651–1674, 2006.

[18] A. T. Patera and G. Rozz. Reduced basis approximation and a posteriori error estimation for parameterized partial differential equations. *Version 1.0, Copyright MIT 2006, to appear in (tentative rubric) MIT Pappalardo Graduate Monographs in Mechanical Engineering. Available at http://augustine.mit.edu*.

[19] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 3600 Market Street, Philadelphia, PA 19104-2688, USA, 2nd edition, 2003.

[20] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[21] P. Sonneveld. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 10(1):36–52, 1989.

[22] H. A. van der Vorst. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992.

[23] S.-L. Zhang. GPBi-CG: Generalized product-type methods based on Bi-CG for solving nonsymmetric linear systems. *SIAM Journal on Scientific Computing*, 18(2):537–551, 1997.