SIAM J. SCI. COMPUT.
Vol. 34, No. 4, pp. A1925–A1949

RECYCLING BICG WITH AN APPLICATION TO MODEL REDUCTION*

KAPIL AHUJA†, ERIC DE STURLER†, SERKAN GUGERCIN†, AND EUN R. CHANG†

Abstract. Science and engineering problems frequently require solving a sequence of dual linear systems. Besides having to store only a few Lanczos vectors, using the biconjugate gradient method (BiCG) to solve dual linear systems has advantages for specific applications. For example, using BiCG to solve the dual linear systems arising in interpolatory model reduction provides a backward error formulation in the model reduction framework. Using BiCG to evaluate bilinear forms—for example, in quantum Monte Carlo (QMC) methods for electronic structure calculations—leads to a quadratic error bound. Since our focus is on sequences of dual linear systems, we introduce *recycling BiCG*, a BiCG method that recycles two Krylov subspaces from one pair of dual linear systems to the next pair. The derivation of recycling BiCG also builds the foundation for developing recycling variants of other bi-Lanczos based methods, such as CGS, BiCGSTAB, QMR, and TFQMR. We develop an augmented bi-Lanczos algorithm and a modified two-term recurrence to include recycling in the iteration. The recycle spaces are approximate left and right invariant subspaces corresponding to the eigenvalues closest to the origin. These recycle spaces are found by solving a small generalized eigenvalue problem alongside the dual linear systems being solved in the sequence. We test our algorithm in two application areas. First, we solve a discretized partial differential equation (PDE) of convection-diffusion type. Such a problem provides well-known test cases that are easy to test and analyze further. Second, we use recycling BiCG in the iterative rational Krylov algorithm (IRKA) for interpolatory model reduction. IRKA requires solving sequences of slowly changing dual linear systems. We analyze the generated recycle spaces and show up to 70% savings in iterations. For our model reduction test problem, we show that solving the problem without recycling leads to (about) a 50% increase in runtime.

Key words. Krylov subspace recycling, deflation, bi-Lanczos method, Petrov–Galerkin formulation, BiCG, model reduction, rational Krylov, $\mathcal{H}_2$ approximation

AMS subject classifications. 65F10, 65N22, 93A15, 93C05

DOI. 10.1137/100801500

1. Introduction.

We focus on solving the sequence of dual linear systems,

$$A^{(j)}x^{(j)} = b^{(j)}, \quad A^{(j)*}\tilde{x}^{(j)} = \tilde{b}^{(j)}, \tag{1.1}$$

where $A^{(j)} \in \mathbb{C}^{n\times n}$ and $b^{(j)}, \tilde{b}^{(j)} \in \mathbb{C}^n$ vary with j, the matrices $A^{(j)}$ are large and sparse, the solution of the dual system is relevant, and the change from a pair of systems to the next is small.

In several application areas, there are important advantages to solving dual linear systems using the BiCG algorithm [23]. BiCG has a short recurrence, so very few Lanczos vectors have to be stored. In addition, using BiCG to solve the dual linear systems arising in interpolatory model reduction provides a backward stable method (with respect to the interpolation conditions) for computing a reduced-order model [12] (see section 5.2). This makes BiCG attractive even for symmetric positive definite (SPD) systems. Furthermore, in several applications, such as QMC algorithms [5], we need to evaluate bilinear forms of the type $u^*A^{-1}w$, where $u, w \in \mathbb{C}^n$

*Submitted to the journal's Methods and Algorithms for Scientific Computing section July 9, 2010; accepted for publication (in revised form) March 6, 2012; published electronically July 12, 2012. This research was supported by the National Science Foundation under grants NSF-EAR 0530643, NSF-DMS 1025327, and NSF-DMS 0645347.
http://www.siam.org/journals/sisc/34-4/80150.html
†Department of Mathematics, Virginia Tech, Blacksburg, VA 24061 (kahuja@math.vt.edu, sturler@vt.edu, gugercin@math.vt.edu, changer@math.vt.edu).

and A is non-Hermitian. Solving dual linear systems for u and w to compute $u^*A^{-1}w$ provides a quadratic error bound [54].

Since BiCG is advantageous for solving dual linear systems and we need to solve a sequence of such systems, we focus on Krylov subspace recycling for BiCG. We refer to our recycling BiCG method as RBiCG. In addition, the BiCG algorithm forms the basis of other popular bi-Lanczos based algorithms like CGS [51], BiCGSTAB [56], QMR [27], and TFQMR [25]. Hence, the derivation of RBiCG is also useful for developing recycling variants of these algorithms [3].

The convergence of Krylov subspace methods for solving a linear system, to a great extent, depends on the spectrum of the matrix, and the deflation of eigenvalues close to the origin usually improves the convergence rate [42, 53]. If the Krylov subspace is augmented with an eigenvector, then the associated eigenvalue is effectively deflated. Likewise, for BiCG, it can be shown that if the dual Krylov subspace, $K^i(A^{(j)*}, \tilde{r}_0)$, is augmented with left eigenvectors, the corresponding right eigenvectors are removed from the primal residual (and vice versa if the primal Krylov subspace is augmented with right eigenvectors) [17]. Therefore, while solving a pair of systems, we select approximate left- and right-invariant subspaces of $A^{(j)}$ (corresponding to small eigenvalues in absolute value), and use these to accelerate the solution of the next pair of systems. This process is called *Krylov subspace recycling*, and leads to faster convergence for the next pair of systems.

For solving a single linear system, "recycling" has been used in the algorithms GCROT [18], based on GCRO [16], and GMRES-DR [42] (an improvement to [41]). GCRO was extended for multiple right-hand sides in [15]. Deflation-based approaches for multiple right-hand sides for a fixed matrix are also proposed in [49, 2, 1]. An extensive analysis of augmentation and deflation approaches for acceleration of restarted methods is provided in [20]. For solving a sequence of linear systems, the idea of recycling a small, judiciously selected subspace was first proposed in [44], where it is applied to the GCROT and the GCRO-DR algorithms. Recycling techniques are adapted to short recurrences in the RMINRES [59] algorithm; see [40] for an improved version. GCROT as in [44], GCRO-DR, and RMINRES all focus on solving a sequence of single systems rather than a sequence of two dual systems, which is the focus here. For a comprehensive discussion of recycling algorithms, see [44]. The recent survey [34] provides a framework for choices in augmentation and deflation approaches and their links and analysis, starting with a recapitulation of the theory from [16] and [59] and extending this to a range of other approaches, such as quasi-minimal residual methods based on QMR [27] and bi-Lanczos-based oblique projection methods. The survey also discusses how the methods presented in [3, 4] and the present paper relate to the given framework.

In addition to testing RBiCG in the iterative rational Krylov algorithm (IRKA) [32] for interpolatory model reduction, we test RBiCG for a model convection-diffusion problem. PDEs of this type are pervasive in science and engineering, they lead to nonsymmetric matrices for which BiCG may be well suited, and they provide well-known test cases that are easy to reproduce and analyze further. Convection-diffusion problems arise, for example, in the Oseen problem (a fixed-point linearization of the Navier–Stokes equations), in chemically reacting flows, heat flow in a medium with transport, and so on. Moreover, any large discretized PDE leads to a potential model reduction problem, for example, for uncertainty quantification, for optimizing an engineering process, or indirectly estimating parameters in the model using measurements. We analyze the generated recycle spaces for both test problems, and we show up to 70% reduction in the iteration count. For our model reduction test problem, using

BiCG instead of RBiCG would take approximately 50% more time to generate the reduced-order model. As recycling is not needed for every pair of linear systems, this means that the improvement in time for those systems where recycling is actually used is substantially larger (see section 6).

To simplify notation, we drop the superscript j in (1.1). At any particular point in the sequence of systems, we refer to $Ax = b$ as the primary system and $A^*\tilde{x} = \tilde{b}$ as the dual system. Throughout the paper, $||\cdot||$ refers to the two-norm, $(\cdot,\cdot)$ refers to the standard inner product, and e_i is the ith canonical basis vector. Unless otherwise stated, we refer to the primary system recycle space and the dual system recycle space collectively as the recycle space.

In the next section, we briefly discuss the BiCG algorithm, and in section 3 we derive the RBiCG algorithm using a previously computed recycle space. How to compute or update such a recycle space efficiently is discussed in section 4. After explaining the basics of interpolatory model reduction, we discuss how RBiCG is applied in IRKA in section 5. We present numerical experiments and results in section 6 and conclusions in section 7.

2. The BiCG algorithm. For the primary system, let x_0 be the initial guess with residual $r_0 = b - Ax_0$. Krylov subspace methods, in general, find approximate solutions by projection onto the Krylov subspace associated with A and r_0 [57]. The ith solution iterate is given by

$$x_i = x_0 + \varrho_i, \tag{2.1}$$

where $\varrho_i \in K^i(A, r_0) \equiv span\{r_0,\ Ar_0,\ A^2r_0,\ \ldots,\ A^{i-1}r_0\}$ is defined by some projection. The BiCG method defines this projection using the Krylov subspace associated with the dual system, leading to two bi-orthogonal bases and a pair of three-term or coupled two-term recurrences. This method is called the bi-Lanczos method [38, 23]. We initialize the Lanczos vectors as follows:

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

Defining $V_i = [v_1\ v_2, \ldots, v_i]$ and $\tilde{V}_i = [\tilde{v}_1\ \tilde{v}_2, \ldots, \tilde{v}_i]$, the $(i+1)$th Lanczos vectors are given by

$$\gamma v_{i+1} = Av_i - V_i\tau \perp \tilde{V}_i, \qquad \tilde{\gamma}\tilde{v}_{i+1} = A^*\tilde{v}_i - \tilde{V}_i\tilde{\tau} \perp V_i,$$

where the scalars γ and $\tilde{\gamma}$ and the vectors τ and $\tilde{\tau}$ are to be determined. This biorthogonality condition leads to a pair of three-term recurrences (see [47]), so that computation of the $(i+1)$th Lanczos vectors requires only the ith and the $(i-1)$th Lanczos vectors. These 3-term recurrences are called the bi-Lanczos relations, and they are defined as follows:

$$AV_i = V_{i+1}\underline{T}_i = V_iT_i + t_{i+1,i}v_{i+1}e_i^T,$$
$$A^*\tilde{V}_i = \tilde{V}_{i+1}\underline{\tilde{T}}_i = \tilde{V}_i\tilde{T}_i + \tilde{t}_{i+1,i}\tilde{v}_{i+1}e_i^T,$$

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 $\underline{\tilde{T}}_i \in \mathbb{C}^{(i+1)\times i}$.

The next step is to find approximate solutions by projection. To exploit the efficiency of short recurrences in the bi-Lanczos algorithm, we use the biorthogonality condition to define the projection. This leads to a Petrov–Galerkin approach. Since

Algorithm 1. *BiCG (adapted from* [57]*).*
1. Choose initial guesses x_0 and $\tilde{x}_0$. Compute $r_0 = b - Ax_0$ and $\tilde{r}_0 = \tilde{b} - A^*\tilde{x}_0$.
2. **if** $(r_0, \tilde{r}_0) = 0$ **then** initialize $\tilde{x}_0$ to a random vector.
3. Set $p_0 = 0$, $\tilde{p}_0 = 0$, and $\beta_0 = 0$. Choose `tol` and `max_itn`.
4. **for** $i = 1 \ldots$ `max_itn` **do**
- $p_i = r_{i-1} + \beta_{i-1} p_{i-1}$.
- $\tilde{p}_i = \tilde{r}_{i-1} + \bar{\beta}_{i-1} \tilde{p}_{i-1}$.
- $q_i = Ap_i$.
- $\tilde{q}_i = A^*\tilde{p}_i$.
- $\alpha_i = (\tilde{r}_{i-1}, r_{i-1})/(\tilde{p}_i, q_i)$.
- $x_i = x_{i-1} + \alpha_i p_i$.
- $\tilde{x}_i = \tilde{x}_{i-1} + \bar{\alpha}_i \tilde{p}_i$.
- $r_i = r_{i-1} - \alpha_i q_i$.
- $\tilde{r}_i = \tilde{r}_{i-1} - \bar{\alpha}_i \tilde{q}_i$.
- **if** $||r_i|| \leq$ `tol` and $||\tilde{r}_i|| \leq$ `tol` **then break**.
- $\beta_i = (\tilde{r}_i, r_i)/(\tilde{r}_{i-1}, r_{i-1})$.

5. **end for**.

the columns of V_i form a basis for $K^i(A, r_0)$, we can define ϱ_i in (2.1) as $\varrho_i = V_i y_i$, and the biorthogonality (or Petrov–Galerkin) condition then implies

$$r_i = b - A(x_0 + \varrho_i) = r_0 - AV_i y_i \perp \tilde{V}_i.$$

The vector y_i is defined by this orthogonality condition. The solution iterate for the dual system, $\tilde{x}_i$, is similarly defined by $\tilde{x}_i = \tilde{x}_0 + \tilde{V}_i \tilde{y}_i$ and $\tilde{r}_i \perp V_i$. Further simplifications lead to the standard BiCG algorithm (Algorithm 1) [23, 57].

Next, we briefly discuss the breakdown conditions in BiCG and their remedies [29, 57]. The first breakdown happens when, at any step i, $\tilde{r}_i^* r_i = 0$. This is a breakdown in the underlying bi-Lanczos algorithm and is referred to as a *serious breakdown*. There exist so-called look-ahead strategies [26, 33] to avoid this breakdown. In addition, the two-term recurrence for the solution update requires a pivotless LDU decomposition of the tridiagonal matrix T_i, which may not always exist. This breakdown is referred to as a breakdown of the *second kind*, and it can be avoided by performing the LDU decomposition with 2×2 block diagonal elements [8]. The breakdown conditions in RBiCG are the same, and similar solutions can be applied. Therefore, and for the sake of brevity, we do not discuss breakdowns for RBiCG separately, and we will assume henceforth in our derivations that breakdowns do not occur. Note that extensive experiments show that BiCG works well, and that breakdowns rarely happen in practice [47, 33].

3. Recycling BiCG: Using a recycle space. In this section, we modify the BiCG algorithm to use a given recycle space. First, we briefly describe the recycling idea used in the GCRO-DR algorithm. After solving the jth primary system in (1.1), GCRO-DR computes the matrices U, $C \in \mathbb{C}^{n \times k}$, such that range$(U)$ is an approximate invariant subspace of $A^{(j)}$, $A^{(j+1)}U = C$, and $C^*C = I$. It then computes an orthogonal basis for the Krylov subspace $K^i\left((I - CC^*)\, A, (I - CC^*)\, r_0\right)$. This produces the Arnoldi relation

$$AV_i = CC^*AV_i + V_{i+1}\underline{H}_i \iff (I - CC^*)AV_i = V_{i+1}\underline{H}_i,$$

where $\underline{H}_i$ is an $(i+1) \times i$ upper Hessenberg matrix. GCRO-DR finds the residual-minimizing solution over the (direct) sum of the recycle space, range(U), and the new search space generated, range(V_i).

In RBiCG, we use the matrix U, derived from an approximate right invariant subspace of $A^{(j)}$, to define the primary system recycle space and compute $C = A^{(j+1)}U$. Similarly, we use the matrix $\tilde{U}$, derived from an approximate left invariant subspace of $A^{(j)}$, to define the dual system recycle space and compute $\tilde{C} = A^{(j+1)*}\tilde{U}$. Instead of C being an orthogonal matrix, U and $\tilde{U}$ are computed such that C and $\tilde{C}$ are biorthogonal; see section 4.3. The number of vectors selected for recycling is denoted by k, and hence, U, $\tilde{U}$, C, and $\tilde{C} \in \mathbb{C}^{n\times k}$. Next, we derive an augmented bi-Lanczos algorithm that computes biorthogonal bases for the primal and dual Krylov subspaces. The two-term recurrence for the solution update in RBiCG is derived in section 3.2.

3.1. The augmented Bi-Lanczos algorithm. The standard 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 $\perp_b$ denotes biorthogonality; this implies that $\tilde{V}_i^* V_i$ is a diagonal matrix. Since we recycle spaces U and $\tilde{U}$, the bi-Lanczos algorithm must be modified to compute the columns of V_i and $\tilde{V}_i$ such that either

$$[U\ V_i] \perp_b \left[\tilde{U}\ \tilde{V}_i\right] \tag{3.1}$$

or

$$[C\ V_i] \perp_b \left[\tilde{C}\ \tilde{V}_i\right]. \tag{3.2}$$

We choose to implement (3.2), because it leads to simpler algebra and hence a more efficient algorithm. It also has the advantage that the RBiCG algorithm has a form similar to the standard BiCG algorithm. Next, we derive the recurrences that implement (3.2), where $C \perp_b \tilde{C}$ has already been satisfied. The latter relation is easy to implement when computing the recycle space. Indeed, we can compute C and $\tilde{C}$ such that $\tilde{C}^*C$ is a real, positive, diagonal matrix; see section 4.3. As in the BiCG algorithm, we assume v_1 and $\tilde{v}_1$ are available from the initial residuals r_0 and $\tilde{r}_0$. We make this statement more precise below. The $(i+1)$th Lanczos vector for the primary system is computed by

$$\gamma v_{i+1} = Av_i - V_i\tau - C\rho \perp \left[\tilde{C}\ \tilde{V}_i\right], \tag{3.3}$$

where γ, τ, and ρ are to be determined. Combining (3.2) and (3.3), we get the following equations:

$$\begin{aligned} \mathcal{D}_c\rho &= \tilde{C}^*Av_i, \\ \mathcal{D}_i\tau &= \tilde{V}_i^*Av_i, \end{aligned} \tag{3.4}$$

where $\mathcal{D}_i = \tilde{V}_i^*V_i$ and $\mathcal{D}_c = \tilde{C}^*C$ are both diagonal matrices and $\mathcal{D}_c$ has real, positive coefficients (see section 4.3). As discussed before, a breakdown in the standard BiCG algorithm because of singular $\mathcal{D}_i$ can be fixed with look-ahead strategies. Assuming breakdowns do not occur, we can solve for τ and ρ in (3.4) and choose a normalization γ; substituting these into (3.3) gives the $(i+1)$th Lanczos vector. Because of the biorthogonality condition (3.2), the full recurrence for v_{i+1} reduces to a $(3+k)$-term

recurrence, where k is the number of columns of C. This implies that the computation of the $(i+1)$th Lanczos vector requires the ith and $(i-1)$th Lanczos vectors and 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; they are given by

$$(I - C\hat{C}^*)AV_i = V_{i+1}\underline{T}_i, \qquad (I - \tilde{C}\check{C}^*)A^*\tilde{V}_i = \tilde{V}_{i+1}\underline{\tilde{T}}_i, \tag{3.5}$$

where

$$\begin{aligned} \hat{C} &= \left[\frac{\tilde{c}_1}{c_1^*\tilde{c}_1} \;\; \frac{\tilde{c}_2}{c_2^*\tilde{c}_2} \;\; \cdots \;\; \frac{\tilde{c}_k}{c_k^*\tilde{c}_k}\right] = \tilde{C}\mathcal{D}_c^{-*} = \tilde{C}\mathcal{D}_c^{-1}, \\ \check{C} &= \left[\frac{c_1}{\tilde{c}_1^*c_1} \;\; \frac{c_2}{\tilde{c}_2^*c_2} \;\; \cdots \;\; \frac{c_k}{\tilde{c}_k^*c_k}\right] = C\mathcal{D}_c^{-1}. \end{aligned} \tag{3.6}$$

Using (3.2), we can rewrite (3.5) as

$$\begin{aligned} A_1V_i &= V_{i+1}\underline{T}_i, \qquad \text{where} \qquad A_1 = (I - C\mathcal{D}_c^{-1}\tilde{C}^*)A(I - C\mathcal{D}_c^{-1}\tilde{C}^*), \\ A_1^*\tilde{V}_i &= \tilde{V}_{i+1}\underline{\tilde{T}}_i, \qquad \text{where} \qquad A_1^* = (I - \tilde{C}\mathcal{D}_c^{-*}C^*)A^*(I - \tilde{C}\mathcal{D}_c^{-*}C^*), \end{aligned} \tag{3.7}$$

since $\tilde{C}^*V_i = 0$ and $C^*\tilde{V}_i = 0$. This new form of the augmented bi-Lanczos relations simplifies the derivation of the recurrence for the RBiCG solution update, because the operators (3.7) are each other's conjugate transpose. Note that the additional orthogonalizations in (3.7) need not be carried out in an actual algorithm (see Algorithm 3.2).

3.2. The solution update for the augmented bi-Lanczos recurrence. The ith solution update in the RBiCG algorithm becomes

$$x_i = x_0 + Uz_i + V_iy_i, \quad \tilde{x}_i = \tilde{x}_0 + \tilde{U}\tilde{z}_i + \tilde{V}_i\tilde{y}_i. \tag{3.8}$$

With recycling, the biorthogonality condition (3.2) defines the Petrov–Galerkin condition,

$$r_i = r_0 - AUz_i - AV_iy_i \perp \left[\tilde{C}\ \tilde{V}_i\right], \quad \tilde{r}_i = \tilde{r}_0 - A^*\tilde{U}\tilde{z}_i - A^*\tilde{V}_i\tilde{y}_i \perp [C\ V_i]\,. \tag{3.9}$$

For the remainder of this section, we focus on the primary system. The derivations for the dual system are analogous. The computation of z_i and y_i can be implemented more efficiently than (3.9) suggests. Defining $\zeta = ||(I - C\hat{C}^*)r_0||$ and $v_1 = \zeta^{-1}(I - C\hat{C}^*)r_0$, we get

$$r_0 = C\hat{C}^*r_0 + \left(I - C\hat{C}^*\right)r_0 = [C\ V_{i+1}]\begin{bmatrix} \hat{C}^*r_0 \\ \zeta e_1 \end{bmatrix}. \tag{3.10}$$

Using the augmented bi-Lanczos relation (3.5) we get

$$A\,[U\ \ V_i]\begin{bmatrix} z_i \\ y_i \end{bmatrix} = [C \quad V_{i+1}]\begin{bmatrix} I & \hat{C}^*AV_i \\ 0 & \underline{T}_i \end{bmatrix}\begin{bmatrix} z_i \\ y_i \end{bmatrix}. \tag{3.11}$$

Substituting (3.10) and (3.11) in (3.9) gives

$$\begin{bmatrix} \tilde{C}^* \\ \tilde{V}_i^* \end{bmatrix}[C\ V_{i+1}]\left(\begin{bmatrix} \hat{C}^*r_0 \\ \zeta e_1 \end{bmatrix} - \begin{bmatrix} I & \hat{C}^*AV_i \\ 0 & \underline{T}_i \end{bmatrix}\begin{bmatrix} z_i \\ y_i \end{bmatrix}\right) = 0. \tag{3.12}$$

Using the biorthogonality condition (3.2) in the above equation we get[1]

$$\begin{bmatrix} \hat{C}^* r_0 \\ \zeta e_1 \end{bmatrix} - \begin{bmatrix} I & \hat{C}^* A V_i \\ 0 & T_i \end{bmatrix} \begin{bmatrix} z_i \\ y_i \end{bmatrix} = 0. \tag{3.13}$$

Therefore, y_i and z_i are given by

$$\begin{aligned} T_i y_i &= \zeta e_1, \\ z_i &= \hat{C}^* r_0 - \hat{C}^* A V_i y_i. \end{aligned} \tag{3.14}$$

Substituting (3.14) in (3.8) leads to the following solution update:

$$x_i = x_0 + U\hat{C}^* r_0 + (I - U\hat{C}^* A) V_i y_i,$$

where y_i is obtained from solving $T_i y_i = \zeta e_1$. All computations here are done with matrix-vector products and $U\hat{C}^* A$ is not computed explicitly.

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

$$\begin{aligned} x_0 &= x_{-1} + U\hat{C}^* r_{-1}, & r_0 &= (I - C\hat{C}^*) r_{-1}, \\ \tilde{x}_0 &= \tilde{x}_{-1} + \tilde{U}\check{C}^* \tilde{r}_{-1}, & \tilde{r}_0 &= (I - \tilde{C}\check{C}^*) \tilde{r}_{-1}, \end{aligned} \tag{3.15}$$

and follow this convention for x_0, $\tilde{x}_0$, r_0, and $\tilde{r}_0$ for the rest of this paper. Let

$$\begin{aligned} T_i &= L_i D_i R_i, \\ G_i &= (I - U\hat{C}^* A) V_i R_i^{-1}, \\ \varphi_i &= \zeta D_i^{-1} L_i^{-1} e_1. \end{aligned}$$

As in the standard BiCG algorithm, an LDU decomposition (without pivoting) of T_i might not always exist. We can avoid this breakdown in the same way as done for BiCG (see section 2). The two-term recurrence for the solution update of the primary system is now given by

$$x_i = x_{i-1} + \varphi_{i,i} G_i e_i \quad \text{for} \quad i \geq 1,$$

where $\varphi_{i,i}$ is the last entry of the vector φ_i, and x_0 is given by (3.15). An analogous update can be derived for the dual system. Note that we never compute any explicit matrix inverse. The matrices under consideration, D_i, L_i, and R_i, are diagonal, lower triangular, and upper triangular, respectively.

This two-term recurrence can be simplified such that T_i is not needed explicitly. To derive further simplifications, we use the operator A_1 (instead of A) and follow steps similar to the ones used in the derivation of BiCG [33]. Algorithm 2 provides an outline of RBiCG. Some algorithmic improvements to make the code faster are not given here; see [3] for further details.

[1] Note that the length of the vector e_1 in (3.13) is one less than that of e_1 in (3.12), although both denote the first canonical basis vector. Also, T_i in (3.13) is $\underline{T}_i$ without the last row, and hence is an $i \times i$ tridiagonal matrix.

Algorithm 2. *RBiCG.*
1. Given U and $\tilde{U}$, compute $\check{C}$ and $\hat{C}$ using (3.6). 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 (3.15).
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**
   - $\diamond$ $p_i = r_{i-1} + \beta_{i-1} p_{i-1}$; $\quad \tilde{p}_i = \tilde{r}_{i-1} + \bar{\beta}_{i-1} \tilde{p}_{i-1}$
   - $\diamond$ $z_i = A p_i$; $\quad \tilde{z}_i = A^* \tilde{p}_i$
   - $\diamond$ $\zeta_i = \hat{C}^* z_i$; $\quad \tilde{\zeta}_i = \check{C}^* \tilde{z}_i$
   - $\diamond$ $q_i = z_i - C\zeta_i$; $\quad \tilde{q}_i = \tilde{z}_i - \tilde{C}\tilde{\zeta}_i$
   - $\diamond$ $\alpha_i = (\tilde{r}_{i-1}, r_{i-1})/(\tilde{p}_i, q_i)$; $\quad \tilde{\alpha}_i = \bar{\alpha}_i$
   - $\diamond$ $\zeta_c = \zeta_c + \alpha_i \zeta_i$; $\quad \tilde{\zeta}_c = \tilde{\zeta}_c + \tilde{\alpha}_i \tilde{\zeta}_i$
   - $\diamond$ $x_i = x_{i-1} + \alpha_i p_i$; $\quad \tilde{x}_i = \tilde{x}_{i-1} + \tilde{\alpha}_i \tilde{p}_i$
   - $\diamond$ $r_i = r_{i-1} - \alpha_i q_i$;; $\quad \tilde{r}_i = \tilde{r}_{i-1} - \tilde{\alpha}_i \tilde{q}_i$
   - $\diamond$ **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. $x_i = x_i - U\zeta_c$; $\quad \tilde{x}_i = \tilde{x}_i - \tilde{U}\tilde{\zeta}_c$.

4. Recycling BiCG: Computing a recycle space. We use the matrices U and $\tilde{U}$ to define the primary and dual system recycle spaces. The recycle space used in solving a linear system is fixed throughout the RBiCG iteration; however, the basis of the recycle space for the next pair of linear systems is updated periodically using the bi-Lanczos vectors. We use harmonic Ritz vectors, with respect to the current Krylov subspace, to approximate left- and right-invariant subspaces cheaply.

We use the following definition [43]. Let S be a subspace of $\mathbb{C}^n$. Then $\lambda \in \mathbb{C}$ is a harmonic Ritz value of A and $0 \neq u \in S$ its corresponding harmonic Ritz vector with respect to the subspace $\mathcal{W} = AS$ if

$$(Au - \lambda u) \perp AS. \tag{4.1}$$

In section 4.1, we derive a small generalized eigenvalue problem whose solution gives the desired approximate invariant subspace. The first pair of systems in our sequence of dual linear systems requires special attention, since there is no recycle space available at the start. We discuss this case in section 4.2. In section 4.3, we describe the construction of the biorthogonal C and $\tilde{C}$ in (3.2) such that $\mathcal{D}_c = \tilde{C}^* C$ has positive real coefficients. Although, the generalized eigenvalue problem derived in section 4.1 is of a small dimension, it would be expensive to set up in a straightforward manner. We show how to set up the problem efficiently using recurrences in section 4.4.

4.1. Computing an approximate invariant subspace. We need a sequence of consecutive Lanczos vectors v_i and $\tilde{v}_i$ and tridiagonal matrices T_i and $\tilde{T}_i$ to build the recycle space. There is a degree of freedom in choosing the scaling of the Lanczos vectors [29, 33, 47]. The following scaling yields $\tilde{T}_i = T_i^*$ (using (3.7) and (4.2)):

$$||v_i|| = 1, \qquad (v_i, \tilde{v}_i) = 1. \tag{4.2}$$

Hence, the Lanczos vectors are computed as follows:

$$v_i = \frac{r_{i-1}}{||r_{i-1}||}, \qquad \tilde{v}_i = \frac{\tilde{r}_{i-1}}{(v_i, \tilde{r}_{i-1})}.$$

T_i can be computed using the residuals and iteration scalars of the RBiCG iteration as follows [2, 3]:

$$T_i = \begin{pmatrix} \frac{1}{\alpha_1} & -\frac{||r_0||}{||r_1||}\cdot\frac{\beta_1}{\alpha_1} & & & \\ -\frac{||r_1||}{||r_0||}\cdot\frac{1}{\alpha_1} & \frac{1}{\alpha_2}+\frac{\beta_1}{\alpha_1} & -\frac{||r_1||}{||r_2||}\cdot\frac{\beta_2}{\alpha_2} & & \\ & & \cdots & & \\ & & \ddots & & -\frac{||r_{i-2}||}{||r_{i-1}||}\cdot\frac{\beta_{i-1}}{\alpha_{i-1}} \\ & & & -\frac{||r_{i-1}||}{||r_{i-2}||}\cdot\frac{1}{\alpha_{i-1}} & \frac{1}{\alpha_i}+\frac{\beta_{i-1}}{\alpha_{i-1}} \end{pmatrix}.$$

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 [59], as it allows us to discard Lanczos vectors periodically. 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. Let V_j and $\tilde{V}_j$ contain the Lanczos vectors generated during the jth cycle,

$$V_j = \left[v_{(j-1)s+1}, \ldots, v_{js}\right], \quad \tilde{V}_j = \left[\tilde{v}_{(j-1)s+1}, \ldots, \tilde{v}_{js}\right].$$

Also, let

$$\Upsilon_j = \begin{bmatrix} v_{(j-1)s} & V_j & v_{js+1} \end{bmatrix}, \quad \tilde{\Upsilon}_j = \begin{bmatrix} \tilde{v}_{(j-1)s} & \tilde{V}_j & \tilde{v}_{js+1} \end{bmatrix},$$

where $v_{(j-1)s}$ and $\tilde{v}_{(j-1)s}$ are the last Lanczos vectors from the previous cycle, and v_{js+1} and $\tilde{v}_{js+1}$ are the first Lanczos vectors from the next cycle. The augmented bi-Lanczos relations for the jth cycle are now given by

$$(I - C\hat{C}^*)AV_j = \Upsilon_j\Gamma_j, \qquad (I - \tilde{C}\check{C}^*)A^*\tilde{V}_j = \tilde{\Upsilon}_j\,\tilde{\Gamma}_j, \tag{4.3}$$

where Γ_j, $\tilde{\Gamma}_j \in \mathbb{C}^{(s+2)\times s}$ are T_j, $\tilde{T}_j$, respectively, with an extra row at the top (corresponding to $v_{(j-1)s}$ and $\tilde{v}_{(j-1)s}$) and at the bottom (corresponding to v_{js+1} and $\tilde{v}_{js+1}$).

The discussion in this paragraph concerns only the primary system. However, an analogous discussion applies to the dual system. Let U define the recycle space available from the previous linear system and let U_{j-1} be the recycle space generated at the end of cycle $(j-1)$ for the current linear system. We want to obtain an improved U_j from V_j, U_{j-1}, and U. It is important to note that U_j is not used for solving the current linear system. At the end of solving the current linear system, the final U_j will be U for the next linear system. There are several choices for selecting U_j [59]. For simplicity, we build U_j from range $([U_{j-1}\ V_j])$.

Based on the choices discussed in the previous two paragraphs, we first define certain matrices, and then we derive the generalized eigenvalue problem whose solution gives the approximate invariant subspace. Let

$$\Phi_j = [U_{j-1}\ V_j], \quad \Psi_j = [C\ C_{j-1}\ \Upsilon_j], \quad H_j = \begin{bmatrix} 0 & B_j \\ I & 0 \\ 0 & \Gamma_j \end{bmatrix},$$

$$\tilde{\Phi}_j = \left[\tilde{U}_{j-1}\ \tilde{V}_j\right], \quad \tilde{\Psi}_j = \left[\tilde{C}\ \tilde{C}_{j-1}\ \tilde{\Upsilon}_j\right], \quad \tilde{H}_j = \begin{bmatrix} 0 & \tilde{B}_j \\ I & 0 \\ 0 & \tilde{\Gamma}_j \end{bmatrix},$$

where $C_{j-1} = AU_{j-1}$, $B_j = \hat{C}^* A V_j$, $\tilde{C}_{j-1} = A^* \tilde{U}_{j-1}$, and $\tilde{B}_j = \check{C}^* A^* \tilde{V}_j$. Then, the augmented bi-Lanczos relations (4.3) lead to

$$A\Phi_j = \Psi_j H_j, \qquad A^* \tilde{\Phi}_j = \tilde{\Psi}_j \tilde{H}_j. \tag{4.4}$$

In RMINRES [59], harmonic Ritz pairs of A with respect to the subspace range$(A\Phi_j)$ have been successfully used to build the recycle space. Since we work in a Petrov–Galerkin framework, it is more intuitive to use harmonic Ritz pairs with respect to the subspace range$(A^*\tilde{\Phi}_j)$, following [9]. This leads to simpler algebra and cheaper computations. Let (λ, u) denote a harmonic Ritz pair of A. Then, we derive λ and $u \in \text{range}(\Phi_j)$ from the condition

$$(Au - \lambda u) \perp \text{range}\left(A^* \tilde{\Phi}_j\right). \tag{4.5}$$

Taking $u = \Phi_j w$ and substituting (4.4) in (4.5) gives

$$\left(A^*\tilde{\Phi}_j\right)^* A\Phi_j w = \lambda \left(A^*\tilde{\Phi}_j\right)^* \Phi_j w \quad \Leftrightarrow \quad \left(\tilde{\Psi}_j \tilde{H}_j\right)^* \Psi_j H_j w = \lambda \left(\tilde{\Psi}_j \tilde{H}_j\right)^* \Phi_j w.$$

Thus, condition (4.5) leads to the generalized eigenvalue problem,

$$\tilde{H}_j^* \tilde{\Psi}_j^* \Psi_j H_j w = \lambda \tilde{H}_j^* \tilde{\Psi}_j^* \Phi_j w. \tag{4.6}$$

Let the columns of W_j be the k right eigenvectors corresponding to the eigenvalues closest to the origin. Then, we take $U_j = \Phi_j W_j$. See [3] for an analogous derivation of the dual system recycle space.

4.2. The first linear system and the first cycle. For the first cycle of the first system, the matrices U, $\tilde{U}$, U_{j-1}, and $\tilde{U}_{j-1}$ are not available. Letting T_1 and $\tilde{T}_1$ denote the tridiagonal matrices for the first cycle, we consider the following eigenvalue problems:

$$T_1 w = \lambda w, \qquad \tilde{T}_1 \tilde{w} = \mu \tilde{w}.$$

Since $\tilde{T}_1 = T_1^*$, we solve for the left and the right eigenvectors of T_1, $\tilde{W}_1$ and W_1, respectively. Hence, we take

$$U_1 = V_1 W_1, \qquad \tilde{U}_1 = \tilde{V}_1 \tilde{W}_1.$$

During the second and subsequent cycles of the first linear system, U_{j-1} and $\tilde{U}_{j-1}$ are available, but C and $\tilde{C}$ are not. Redefining $\Psi_j, \tilde{\Psi}_j, H_j$, and $\tilde{H}_j$, we get the generalized eigenvalue problem (4.6) with

$$\Phi_j = \begin{bmatrix} U_{j-1} & V_j \end{bmatrix}, \quad \Psi_j = \begin{bmatrix} C_{j-1} & \Upsilon_j \end{bmatrix}, \quad H_j = \begin{bmatrix} I & 0 \\ 0 & \Gamma_j \end{bmatrix},$$
$$\tilde{\Phi}_j = \begin{bmatrix} \tilde{U}_{j-1} & \tilde{V}_j \end{bmatrix}, \quad \tilde{\Psi}_j = \begin{bmatrix} \tilde{C}_{j-1} & \tilde{\Upsilon}_j \end{bmatrix}, \quad \tilde{H}_j = \begin{bmatrix} I & 0 \\ 0 & \tilde{\Gamma}_j \end{bmatrix}.$$

For the first cycle of each of the subsequent linear systems (i.e., $j = 1$), C and $\tilde{C}$ are available, while U_{j-1} and $\tilde{U}_{j-1}$ are not. Redefining $\Phi_1, \tilde{\Phi}_1, \Psi_1, \tilde{\Psi}_1, H_1$, and $\tilde{H}_1$, we get the generalized eigenvalue problem (4.6) with

$$\Phi_1 = \begin{bmatrix} U & V_1 \end{bmatrix}, \quad \Psi_1 = \begin{bmatrix} C & \underline{V}_1 \end{bmatrix}, \quad H_1 = \begin{bmatrix} I & B_1 \\ 0 & \underline{T}_1 \end{bmatrix},$$
$$\tilde{\Phi}_1 = \begin{bmatrix} \tilde{U} & \tilde{V}_1 \end{bmatrix}, \quad \tilde{\Psi}_1 = \begin{bmatrix} \tilde{C} & \underline{\tilde{V}}_1 \end{bmatrix}, \quad \tilde{H}_1 = \begin{bmatrix} I & \tilde{B}_1 \\ 0 & \underline{\tilde{T}}_1 \end{bmatrix},$$

where $\underline{V}_1$ and $\underline{\tilde{V}}_1$ denote $[V_1\, v_{s+1}]$ and $[\tilde{V}_1\, \tilde{v}_{s+1}]$, respectively.

4.3. Constructing biorthogonal C_j, $\tilde{C}_j$ and C, $\tilde{C}$. We need to compute the matrices C_j and $\tilde{C}_j$ such that $C_j \perp_b \tilde{C}_j$ at the end of each cycle. After solving the generalized eigenvalue problem (4.6), we set (as initial choice) $U_j = \Phi_j W_j$, $\tilde{U}_j = \tilde{\Phi}_j \tilde{W}_j$, $C_j = AU_j$, and $\tilde{C}_j = A^* \tilde{U}_j$, and we compute the SVD

$$\tilde{C}_j^* C_j = M_j \Sigma_j N_j^*, \tag{4.7}$$

such that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k \geq 0$. Given some tolerance $\texttt{tol} > 0$, we pick p such that $\sigma_p \geq \texttt{tol} > \sigma_{p+1}$ (with both $p = k$ and $p = 0$ possible), and redefine $M_j = [m_1, \ldots, m_p]$ and $N_j = [n_1, \ldots, n_p]$, where m_i and n_i are the left and right singular vectors corresponding to σ_i. Next, we redefine

$$\begin{aligned} U_j &= \Phi_j W_j N_j = [U_{j-1}\ V_j] W_j N_j, \quad \tilde{U}_j = \tilde{\Phi}_j \tilde{W}_j M_j = [\tilde{U}_{j-1}\ \tilde{V}_j] \tilde{W}_j M_j, \\ C_j &= AU_j = A[U_{j-1}\ V_j] W_j N_j, \qquad \tilde{C}_j = A^* \tilde{U}_j = A^* [\tilde{U}_{j-1}\ \tilde{V}_j] \tilde{W}_j M_j. \end{aligned} \tag{4.8}$$

By construction $\tilde{C}_j^* C_j$ is diagonal with real, positive coefficients.[2]

Analogous to the above, at the start of each linear system (after the first), we need to compute C and $\tilde{C}$ such that $C \perp_b \tilde{C}$ (cf. (3.2)), and the diagonal matrix $\mathcal{D}_c = \tilde{C}^* C$ has real, positive coefficients. Taking initially for U and $\tilde{U}$ the final matrices U_j and $\tilde{U}_j$ from the previous pair of linear systems, we compute $C = AU$, $\tilde{C} = A^* \tilde{U}$, and compute the SVD $\tilde{C}^* C = M \Sigma N^*$. After this we proceed as for the computation of C_j and $\tilde{C}_j$.

4.4. Efficiently setting up the generalized eigenvalue problem. The main cost of setting up the generalized eigenvalue problem (4.6) is in computing the matrices

$$\tilde{\Psi}_j^* \Psi_j = \begin{bmatrix} \tilde{C}^* \\ \tilde{C}_{j-1}^* \\ \tilde{\Upsilon}_j^* \end{bmatrix} \begin{bmatrix} C & C_{j-1} & \Upsilon_j \end{bmatrix} = \begin{bmatrix} \mathcal{D}_c & \tilde{C}^* C_{j-1} & 0 \\ \tilde{C}_{j-1}^* C & \Sigma_{j-1} & \tilde{C}_{j-1}^* \Upsilon_j \\ 0 & \tilde{\Upsilon}_j^* C_{j-1} & I \end{bmatrix},$$

$$\tilde{\Psi}_j^* \Phi_j = \begin{bmatrix} \tilde{C}^* \\ \tilde{C}_{j-1}^* \\ \tilde{\Upsilon}_j^* \end{bmatrix} \begin{bmatrix} U_{j-1} & V_j \end{bmatrix} = \begin{bmatrix} \tilde{C}^* U_{j-1} & 0 \\ \tilde{C}_{j-1}^* U_{j-1} & \tilde{C}_{j-1}^* V_j \\ \tilde{\Upsilon}_j^* U_{j-1} & \underline{I} \end{bmatrix},$$

where $\underline{I}$ is the $s \times s$ identity matrix with an extra row of zeros at the top and at the bottom. Most of the blocks in these matrices can be constructed efficiently by exploiting recurrences and various algebraic relations.

The biorthogonality condition (3.2) and the construction of C_j and $\tilde{C}_j$ (4.7)–(4.8) give the following orthogonality relations:

$$\tilde{C}_{j-2} \perp \Upsilon_j, \quad C_{j-2} \perp \tilde{\Upsilon}_j. \tag{4.9}$$

Next, going from top-to-bottom and left-to-right, we analyze each block of $\tilde{\Psi}_j^* \Psi_j$ and $\tilde{\Psi}_j^* \Phi_j$ in terms of its defining recurrences and simplify it using (3.2), (4.3), (4.7), (4.8),

[2] For $p = 0$, no recycle space would be selected. This has never occurred in our experience. Indeed, discarding even one pair of vectors is rare. In our experiments, we use $\texttt{tol} = \texttt{10}^{-6}$; see section 6.2.

and (4.9). Blocks whose efficient computation is obvious or has already been detailed are skipped, and we focus on computations that are at least $O(n)$:

- $$\begin{aligned}\tilde{C}^* C_{j-1} = \tilde{C}^* A U_{j-1} &= \left[\begin{array}{cc} \tilde{C}^* A U_{j-2} & \tilde{C}^* A V_{j-1} \end{array}\right] W_{j-1} N_{j-1} \\ &= \left[\begin{array}{cc} \tilde{C}^* C_{j-2} & \tilde{C}^* \left(C \hat{C}^* A V_{j-1} + \Upsilon_{j-1} \Gamma_{j-1} \right) \end{array}\right] W_{j-1} N_{j-1} \\ &= \left[\begin{array}{cc} \tilde{C}^* C_{j-2} & \mathcal{D}_c B_{j-1} \end{array}\right] W_{j-1} N_{j-1}.\end{aligned}$$

For this first block, we describe its efficient computation in some detail. The cost of computing $\tilde{C}^* C_{j-1}$ by direct multiplication is $O(k^2 n)$; so, it would be expensive. However, the submatrix $\tilde{C}^* C_{j-2}$ is available from the previous cycle, and $\mathcal{D}_c$ is a diagonal matrix independent of the cycle (so both must be computed at most once per linear system). Furthermore, B_{j-1} has been computed during the (augmented) bi-Lanczos iteration. Finally, the matrix-matrix product $[\tilde{C}^* C_{j-2} \;\; \mathcal{D}_c B_{j-1}] W_{j-1} N_{j-1}$ does not involve any $O(n)$ operation. Hence, this block can be computed quite cheaply. We give a brief overview of the cost of the RBiCG algorithm in section 6; for a more detailed derivation, see [4].

- $$\tilde{C}_{j-1}^* C = M_{j-1}^* \tilde{W}_{j-1}^* \left[\begin{array}{c} \tilde{C}_{j-2}^* C \\ \tilde{B}_{j-1}^* \mathcal{D}_c \end{array}\right].$$

The derivation of this block is similar to the previous block. As above, $\tilde{C}_{j-2}^* C$ is available from the previous cycle, and $\tilde{B}_{j-1}$ has been computed during the bi-Lanczos iteration.

- $$\begin{aligned}\tilde{C}_{j-1}^* \Upsilon_j = \tilde{U}_{j-1}^* A \Upsilon_j = M_{j-1}^* \tilde{W}_{j-1}^* \left[\begin{array}{c} \tilde{U}_{j-2}^* \\ \tilde{V}_{j-1}^* \end{array}\right] A \Upsilon_j &= M_{j-1}^* \tilde{W}_{j-1}^* \left[\begin{array}{c} 0 \\ \tilde{V}_{j-1}^* A \Upsilon_j \end{array}\right] \\ &= M_{j-1}^* \tilde{W}_{j-1}^* \left[\begin{array}{c} 0 \\ \left[\tilde{C} \check{C}^* A^* \tilde{V}_{j-1} + \tilde{\Upsilon}_{j-1} \tilde{\Gamma}_{j-1} \right]^* \Upsilon_j \end{array}\right] \\ &= M_{j-1}^* \tilde{W}_{j-1}^* \left[\begin{array}{c} 0 \\ \tilde{\Gamma}_{j-1}^* \tilde{\Upsilon}_{j-1}^* \Upsilon_j \end{array}\right],\end{aligned}$$

where

$$\begin{aligned}\tilde{\Gamma}_{j-1}^* \tilde{\Upsilon}_{j-1}^* \Upsilon_j &= \tilde{\Gamma}_{j-1}^* \left[\begin{array}{c} \tilde{v}_{(j-2)s}^* \\ \tilde{v}_{(j-2)s+1}^* \\ \vdots \\ \tilde{v}_{(j-1)s}^* \\ \tilde{v}_{(j-1)s+1}^* \end{array}\right] \left[\begin{array}{cccccc} v_{(j-1)s} & v_{(j-1)s+1} & \cdots & v_{js} & v_{js+1} \end{array}\right] \\ &= \tilde{\Gamma}_{j-1}^* \left[\begin{array}{ccccc} 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \end{array}\right].\end{aligned}$$

- $\tilde{\Upsilon}_j^* C_{j-1} = \begin{bmatrix} 0 & \tilde{\Upsilon}_j^* \Upsilon_{j-1} \Gamma_{j-1} \end{bmatrix} W_{j-1} N_{j-1}$, where

$$\tilde{\Upsilon}_j^* \Upsilon_{j-1} \Gamma_{j-1} = \begin{bmatrix} 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \Gamma_{j-1}.$$

The derivation of this block is similar to that of the previous block.

- $\tilde{C}^* U_{j-1} = \tilde{C}^* \begin{bmatrix} U_{j-2} & V_{j-1} \end{bmatrix} W_{j-1} N_{j-1} = \begin{bmatrix} \tilde{C}^* U_{j-2} & 0 \end{bmatrix} W_{j-1} N_{j-1}$,

where $\tilde{C}^* U_{j-2}$ is available from the previous cycle (such a block must be computed at most once per linear system).

- $$\begin{aligned} \tilde{C}_{j-1}^* U_{j-1} &= M_{j-1}^* \tilde{W}_{j-1}^* \begin{bmatrix} \tilde{U}_{j-2}^* \\ \tilde{V}_{j-1}^* \end{bmatrix} A \begin{bmatrix} U_{j-2} & V_{j-1} \end{bmatrix} W_{j-1} N_{j-1} \\ &= M_{j-1}^* \tilde{W}_{j-1}^* \begin{bmatrix} \tilde{C}_{j-2}^* U_{j-2} & \tilde{C}_{j-2}^* V_{j-1} \\ \tilde{V}_{j-1}^* C_{j-2} & \tilde{V}_{j-1}^* A V_{j-1} \end{bmatrix} W_{j-1} N_{j-1}, \end{aligned}$$

where $\tilde{C}_{j-2}^* U_{j-2}$ and $\tilde{C}_{j-2}^* V_{j-1}$ are submatrices of $\tilde{\Psi}_{j-1}^* \Phi_{j-1}$, $\tilde{V}_{j-1}^* C_{j-2}$ is a submatrix of $\tilde{\Upsilon}_{j-1}^* C_{j-2}$ and is available from $\tilde{\Psi}_{j-1}^* \Psi_{j-1}$, and $\tilde{V}_{j-1}^* A V_{j-1} = \tilde{T}_{j-1}$.

- $\tilde{C}_{j-1}^* V_j$ is a submatrix of $\tilde{C}_{j-1}^* \Upsilon_j$, and hence, is available from $\tilde{\Psi}_j^* \Psi_j$.

Therefore, only $\tilde{\Upsilon}_j^* U_{j-1}$ needs to be computed explicitly.

5. Model reduction. Consider a single-input/single-output (SISO) linear time-invariant (LTI) system represented as

$$G : \begin{cases} E\,\dot{x}(t) = Ax(t) + bv(t), \\ y(t) = c^* x(t), \end{cases} \qquad \text{or} \qquad G(s) = c^*(sE - A)^{-1} b, \tag{5.1}$$

where $E, A \in \mathbb{R}^{n \times n}$ and $b, c \in \mathbb{R}^n$. The time-dependent functions $v(t)$, $y(t)$: $\mathbb{R} \to \mathbb{R}$ are the input and output of $G(s)$, respectively, and $x(t) : \mathbb{R} \to \mathbb{R}^n$ is the associated state. In (5.1), $G(s)$ is the transfer function of the system: Let $V(s)$ and $Y(s)$ denote the Laplace transforms of $v(t)$ and $y(t)$, respectively. Then, the transfer function $G(s)$ satisfies $Y(s) = G(s)V(s)$. By a common abuse of notation, we denote both the underlying dynamical system and its transfer function with G. The dimension of the underlying state-space, n, is called the dimension of G. Systems of the form (5.1) with extremely large state-space dimension n arise in many applications; see [6] and [37] for a collection of such examples. Simulations in such large scale settings lead 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 which provides a high-fidelity approximation of the input-output behavior of the original model G. Let $r \ll n$ denote the order of the reduced model. The reduced model is represented, similar to (5.1), as

$$G_r(s) : \begin{cases} E_r\,\dot{x}_r(t) = A_r x_r(t) + b_r v(t), \\ y_r(t) = c_r^* x_r(t), \end{cases} \qquad \text{or} \quad G_r(s) = c_r^*(sE_r - A_r)^{-1} b_r, \tag{5.2}$$

where $E_r, A_r \in \mathbb{R}^{r\times r}$ and $b_r, c_r \in \mathbb{R}^r$. In this setting, the common approach is to construct reduced-order models via a Petrov–Galerkin projection. This amounts to choosing two r-dimensional subspaces $\mathcal{V}_r$ and $\mathcal{W}_r$ and matrices $V_r \in \mathbb{R}^{n\times r}$ and $W_r \in \mathbb{R}^{n\times r}$ such that $\mathcal{V}_r = \text{Range}(V_r)$ and $\mathcal{W}_r = \text{Range}(W_r)$. Then, we approximate the full-order state $x(t)$ as $x(t) \approx V_r x_r(t)$ and enforce the Petrov–Galerkin condition,

$$W_r^* \left(EV_r\dot{x}_r(t) - AV_r x_r(t) - b\, v(t)\right) = 0, \quad y_r(t) = c^* V_r x_r(t),$$

leading to a reduced-order model as in (5.2) with

$$E_r = W_r^* E V_r, \quad A_r = W_r^* A V_r, \quad b_r = W_r^* b, \quad \text{and} \quad c_r = V_r^* c. \tag{5.3}$$

As (5.3) illustrates, the quality of the reduced model depends solely on the selection of the two subspaces, $\mathcal{V}_r$ and $\mathcal{W}_r$. In this paper, we will choose $\mathcal{V}_r$ and $\mathcal{W}_r$ to enforce interpolation. For other selections of $\mathcal{V}_r$ and $\mathcal{W}_r$, we refer the reader to [6].

5.1. Interpolatory model reduction. For a given full-order model $G(s)$, the goal of interpolatory model reduction is to construct a reduced-order model $G_r(s)$ via rational interpolation. Here, we focus on Hermite interpolation: given the full-order model (5.1) and a collection of interpolation points (also called shifts) $\sigma_i \in \mathbb{C}$, for $i = 1, \ldots, r$, we must construct a reduced-order system by projection as in (5.3) such that $G_r(s)$ interpolates $G(s)$ and its first derivative at selected interpolation points, i.e.,

$$G(\sigma_i) = G_r(\sigma_i) \qquad \text{and} \qquad G'(\sigma_i) = G_r'(\sigma_i) \quad \text{for} \quad i = 1, \ldots, r.$$

Rational interpolation by projection was first proposed in [19, 61, 62]. How to obtain the required projection was derived in [30] using the rational Krylov method [46]. For the special case of Hermite rational interpolation, the solution of the interpolatory model reduction problem is given in Theorem 5.1. For the more general case, we refer the reader to [30] and the recent survey [7].

THEOREM 5.1. *Given $G(s) = c^*(sE-A)^{-1}b$ and r distinct points $\sigma_1, \ldots, \sigma_r \in \mathbb{C}$, let*

$$V_r = [(\sigma_1 E - A)^{-1}b, \ldots, (\sigma_r E - A)^{-1}b], \qquad W_r^* = \begin{bmatrix} c^*(\sigma_1 E - A)^{-1} \\ \vdots \\ c^*(\sigma_r E - A)^{-1} \end{bmatrix}. \tag{5.4}$$

Using (5.3)*, define the reduced-order model $G_r(s) = c_r^*(sE_r - A_r)^{-1}b_r$. Then $G(\sigma_i) = G_r(\sigma_i)$ and $G'(\sigma_i) = G_r'(\sigma_i)$, for $i = 1, \ldots, r$, provided that $\sigma_i E - A$ and $\sigma_i E_r - A_r$ are invertible for $i = 1, \ldots, r$.*

Theorem 5.1 shows how to solve the interpolatory model reduction problem via projection for given shifts. However, it does not provide a strategy for choosing good/optimal interpolation points. Recently, this issue has been resolved for the special case of optimality in the $\mathcal{H}_2$ norm [32]. The $\mathcal{H}_2$ norm of the dynamical system $G(s)$ is defined as

$$\|G\|_{\mathcal{H}_2} = \left(\frac{1}{2\pi}\int_{-\infty}^{\infty} |\, G(\jmath a)\,|^2\, da\right)^{1/2},$$

where $\jmath = \sqrt{-1}$. The $\mathcal{H}_2$ norm of G is the $2-\infty$ induced norm of the underlying convolution operator from input v to output y, so that for any input $v \in L^2(\mathbb{R}^+)$,

Algorithm 3. *IRKA ([32]).*
1. Make an initial shift selection σ_i for $i = 1, \ldots, r$,
2. $V_r = [(\sigma_1 E - A)^{-1} b, \ldots, (\sigma_r E - A)^{-1} b]$,
3. $W_r = [(\sigma_1 E - A)^{-*} c, \ldots, (\sigma_r E - A)^{-*} c]$,
4. while (not converged)
◇ $A_r = W_r^* A V_r$, $E_r = W_r^* E V_r$,
◇ $\sigma_i \leftarrow -\lambda_i(A_r, E_r)$ for $i = 1, \ldots, r$,
◇ $V_r = [(\sigma_1 E - A)^{-1} b, \ldots, (\sigma_r E - A)^{-1} b]$,
◇ $W_r = [(\sigma_1 E - A)^{-*} c, \ldots, (\sigma_r E - A)^{-*} c]$,
5. $A_r = W_r^* A V_r$, $E_r = W_r^* E V_r$, $b_r = W_r^* b$, $c_r = V_r^* c_r$.

$\|y - y_r\|_{L^\infty} \leq \|G - G_r\|_{\mathcal{H}_2} \|v\|_{L^2}$. To ensure that the output error $y - y_r$ is small in $L^\infty(\mathbb{R}^+)$ uniformly over all inputs v, say, with $\|v\|_{L^2} \leq 1$, we seek a reduced system G_r that makes $\|G - G_r\|_{\mathcal{H}_2}$ small. This leads to the *optimal $\mathcal{H}_2$ model reduction problem*: Given $G(s)$, and a reduced-order $r < n$, find $G_r(s)$ that solves

$$\|G - G_r\|_{\mathcal{H}_2} = \min_{dim(\hat{G}_r)=r} \left\| G - \hat{G}_r \right\|_{\mathcal{H}_2}. \tag{5.5}$$

This problem has been studied extensively [39, 60, 32, 52, 58, 31, 10, 11]. It is a nonconvex optimization problem, which makes finding the global minimum, at best, a hard task. Hence, the common approach is to construct reduced-order models that satisfy, for an interpolatory model reduction framework, the following first-order necessary conditions.

THEOREM 5.2 (see [39, 32]). *Given $G(s)$, let $G_r(s) = c_r^*(sE_r - A_r)^{-1} b_r$ be an $\mathcal{H}_2$-optimal reduced-order model of order r, with distinct poles $\hat{\lambda}_1, \ldots, \hat{\lambda}_r$. Then*

$$G(-\hat{\lambda}_i) = G_r(-\hat{\lambda}_i) \quad \text{and} \quad G'(-\hat{\lambda}_i) = G_r'(-\hat{\lambda}_i) \quad \text{for} \quad i = 1, \ldots, r. \tag{5.6}$$

So, the $\mathcal{H}_2$ optimal approximant $G_r(s)$ is a Hermite interpolant to $G(s)$ at the mirror image of its poles. These poles, the optimal interpolation points, are not known a priori. Hence, IRKA [32], starting from an initial selection of interpolation points, iteratively corrects the interpolation points until (5.6) is satisfied. Algorithm 3 outlines IRKA; for details, see [32].

5.2. Advantages of approximating solutions using a Petrov–Galerkin framework in interpolatory model reduction. The main cost in IRKA is solving multiple linear systems to compute V_r and W_r. If the dimension of the state-space, n, is large, these systems are generally solved only approximately by an iterative solver. In this context, it is important to assess the accuracy of the computed reduced-order model, that is, given the shifts, how accurately the Hermite interpolation problem is solved. This question was studied extensively in [12]. One of the major results, outlined below for our particular case, provides the main motivation for solving the linear systems associated with the corresponding columns of V_r and W_r as pairs of dual linear systems (in the terminology of section 1) using BiCG or RBiCG.

Let $\hat{v}_i$ and $\hat{w}_i$, for $i = 1, \ldots, r$, denote the approximate solutions of $(\sigma_i E - A)v_i = b$ and $(\sigma_i E - A)^* w_i = c$, respectively, with residuals $\eta_i = (\sigma_i E - A)\hat{v}_i - b$ and $\xi_i = (\sigma_i E - A)^* \hat{w}_i - c$. Furthermore, let $\hat{v}_i$, $\hat{w}_i$, η_i, and ξ_i satisfy the Petrov–Galerkin condition that there exist spaces $\mathcal{P}$ and $\mathcal{Q}$ such that $\hat{v}_i \in \mathcal{P}$, $\hat{w}_i \in \mathcal{Q}$, $\eta_i \perp \mathcal{Q}$, and $\xi_i \perp \mathcal{P}$. Define the approximate solution matrices ($\hat{V}_r$ and $\hat{W}_r$), the residual matrices

(R_b and R_c), and the rank-2r matrix (F_{2r}) as follows:

$$\hat{V}_r = [\hat{v}_1\, \hat{v}_2, \ldots, \hat{v}_r], \qquad \hat{W}_r = [\hat{w}_1\, \hat{w}_2, \ldots, \hat{w}_r],$$
$$R_b = [\eta_1\, \eta_2, \ldots, \eta_r], \qquad R_c = [\xi_1\, \xi_2, \ldots, \xi_r],$$
$$F_{2r} = R_b(\hat{W}_r^*\hat{V}_r)^{-1}\hat{W}_r^* + \hat{V}_r^*(\hat{W}_r^*\hat{V}_r)^{-1}R_c.$$

Also, define the inexact reduced-order order quantities

$$\hat{A}_r = \hat{W}_r^* A \hat{V}_r, \quad \hat{E}_r = \hat{W}_r^* E \hat{V}_r, \quad \hat{b}_r = \hat{W}_r^* b, \quad \text{and} \quad \hat{c}_r = \hat{V}_r^* c.$$

Then, the computed reduced-order model $\hat{G}_r(s) = \hat{c}_r^*(s\hat{E}_r - \hat{A}_r)^{-1}\hat{b}_r$ *exactly* interpolates the perturbed full-order model $\hat{G}(s) = c^*(sE - (A + F_{2r}))^{-1}b$, i.e.,

$$\hat{G}(\sigma_i) = \hat{G}_r(\sigma_i) \quad \text{and} \quad \hat{G}'(\sigma_i) = \hat{G}_r'(\sigma_i) \quad \text{for} \quad i = 1, \ldots, r.$$

Hence, iteratively solving the linear systems while satisfying the Petrov–Galerkin condition above yields a backward error for the interpolatory model reduction that is bounded by $\|F_{2r}\|$, which is governed by the norms of the residuals. The latter are easily controlled in the iterative solver. For details, we refer the reader to [12].

The easiest way to satisfy the Petrov–Galerkin condition above is by solving the dual pairs of linear systems using BiCG. Hence, BiCG is particularly suitable for solving the linear systems in IRKA (even for symmetric positive definite matrices). However, as IRKA leads to a sequence of dual linear systems, the RBiCG algorithm can be used to reduce the total run time for solving all linear systems. Moreover, if we solve the dual pairs of linear systems arising in IRKA by RBiCG, the Petrov–Galerkin condition is still satisfied. Hence, the resulting reduced-order model will be an optimal $\mathcal{H}_2$ approximation to a nearby full-order model.

5.3. IRKA using RBiCG. IRKA usually converges fast [32], and after the first few steps of IRKA the updates to the interpolations points are modest. Moreover, for many cases, the (ordered) σ_i also change modestly from one column of V_r (and W_r) to the next. So, we expect IRKA to gain significantly from recycling.

For the special case of $E = I$ in (5.1), alternative solution approaches might be advantageous, as one can solve the linear systems for multiple shifts at once [24, 28, 35, 55]. Effective strategies for Krylov subspace recycling for solving systems of this type for multiple shifts at once (and for multiple right-hand sides) were discussed in [36]. For most model reduction applications, however, $E \neq I$.

We consider three strategies for recycling Krylov subspaces in IRKA. For the first strategy, consider iterations m and $m+1$ in IRKA (the while loop Algorithm 3), with shifts $\sigma_i^{(m)}$ and $\sigma_i^{(m+1)}$, for $i = 1, \ldots, r$ and linear systems,

$$V_r^{(j)} = [(\sigma_1^{(j)}E - A)^{-1}b, \ldots, \ (\sigma_r^{(j)}E - A)^{-1}b],$$
$$W_r^{(j)} = [(\sigma_1^{(j)}E - A)^{-*}c, \ldots, \ (\sigma_r^{(j)}E - A)^{-*}c], \tag{5.7}$$

for $j = m$ or $j = m + 1$. We recycle Krylov subspaces from the ith column of $V_r^{(m)}$ and $W_r^{(m)}$ to the ith column of $V_r^{(m+1)}$ and $W_r^{(m+1)}$. That is, from solving the pair of linear systems $(\sigma_i^{(m)}E - A)v_i^{(m)} = b$ and $(\sigma_i^{(m)}E - A)^* w_i^{(m)} = c$ to solving $(\sigma_i^{(m+1)}E - A)v_i^{(m+1)} = b$ and $(\sigma_i^{(m+1)}E - A)^* w_i^{(m+1)} = c$ for each $i = 1, 2, \ldots, r$. This recycling strategy is effective when the change in a shift from one IRKA step to the next is small. For the second strategy, in a single IRKA step, we recycle

selected Krylov subspaces from solving for one pair of columns of V_r and W_r to the next pair of columns across all the columns of the matrices V_r and W_r. For the third strategy, the first two recycling strategies are combined. Consider solving the system $(\sigma_i^{(m+1)}E - A)v_i^{(m+1)} = b$ and its dual system. From a set of previously generated recycle spaces (distinguished by their shifts), we pick the recycle space from the system with the smallest relative change in σ (and less than a relative tolerance). This ensures that the linear system that generated the recycle space is close to the current one. A natural pool from which to pick the σ defining the recycle space would be $\{\sigma_1^{(m)}, \ldots, \sigma_r^{(m)}, \sigma_1^{(m+1)}, \ldots, \sigma_{i-1}^{(m+1)}\}$. The second and third recycling strategies are effective when the shifts at an IRKA step are clustered.

For the experiments in this paper, r is small, and so the shifts at any particular IRKA step are spread far apart. Hence, we follow the first strategy. That is, for every shift, we recycle Krylov subspaces from one IRKA step to the next. In general, the linear systems corresponding to the relatively large shifts converge fast, and so recycling Krylov subspaces is not useful for them. Therefore, we carry out recycling only for selected, small shifts. We give more details in section 6.2.

5.4. Previous work in recycling for model reduction. Recycling for interpolatory model reduction in the Galerkin setting, i.e., with $W_r = V_r$, has been considered in [14] and [22]. In this setting, there are no dual systems to solve, and therefore approaches based on GCR [21] and GMRES [48] are considered, respectively, for a sequence of (single) linear systems, as opposed to our approach based on BiCG for a sequence of dual linear systems. Also in other respects, the approach for improving the linear solver and the model reduction context is quite different from what we focus on in this paper. In [14], the focus is on efficiently solving linear systems with a fixed coefficient matrix and multiple right-hand sides ($Ax^{(j)} = b^{(j)}$), recycling descent vectors (in GCR). Furthermore, the authors target model reduction with a single interpolation point but interpolating higher derivatives.

6. Results. We first give a brief overview of the overhead in RBiCG. We focus on components with at least $O(n)$ cost, where n is the dimension of the linear system. Furthermore, k is the number of basis vectors in the primal (or dual) recycle space, and s is the number of iterations in a cycle. For every iteration, there is an extra cost of $(8k + 2)n$ flops, mostly from orthogonalizations. At the end of each cycle, there is an extra cost of $(14k^2 + 6ks + 16k + 4)n$ flops, mostly from setting up the generalized eigenvalue problem and computing biorthogonal C_j and $\tilde{C}_j$. Once per linear system, there is an extra cost of $(10k^2 + 28k + 14)n$ flops, mostly from computing biorthogonal C and $\tilde{C}$. A more detailed discussion of the overhead is given in [4]. Note that s and k are much smaller than n. For recycling to be beneficial, the savings in iterations should be sufficient to make up for the overhead. Further in this section, we show that the reduction in the number of iterations for (a pair of) linear systems may be as high as 70%. For our model reduction test problem, we show that computing a reduced model without recycling takes about 50% more time than with recycling.

We test RBiCG on a convection-diffusion problem and on IRKA for interpolatory model reduction. All experiments are done using MATLAB.

6.1. Convection-diffusion. To analyze RBiCG, we use the linear system obtained by finite difference discretization of the partial differential equation

$$-(\mathcal{A}\vartheta_x)_x - (\mathcal{A}\vartheta_y)_y + \mathcal{B}(x, y)\vartheta_x = \mathcal{F},$$

with $\mathcal{A}$ as shown in Figure 6.1 (a), $\mathcal{B}(x,y) = 2e^{2(x^2+y^2)}$, and $\mathcal{F} = 0$ everywhere except in a small square in the center where $\mathcal{F} = 100$ [56]; see Figure 6.1(a). The domain is $(0,1) \times (0,1)$ with Dirichlet boundary conditions

$$\vartheta(0,y) = \vartheta(1,y) = \vartheta(x,0) = 1 \quad \text{and} \quad \vartheta(x,1) = 0.$$

We use the second order central difference scheme with a mesh width of $h = 1/64$, giving a nonsymmetric linear system of 3969 unknowns. The convergence improvement by recycling is similar for a problem that is four times larger. To enable further analysis, we give results for this smaller system size. The primary system right-hand side comes from the PDE. We take the vector of all zeros as the dual system right-hand side. In this case, we are concerned only about the primary system.

To analyze RBiCG, we solve the (same) dual linear systems four times. The recycle space generated during the first run is used for solving the same dual systems a second time, further improving the recycle space, and so on. This is a useful approach for analyzing how well Krylov subspace recycling works, as it excludes the effects of changing matrices and of right-hand sides having different expansions in the eigenvector basis [44]. Hence, it provides an indication for reasonable sequences of systems of how fast the recycle spaces converge and how much recycling approximate invariant subspaces is likely to improve convergence. For this experiment, we take $s = 40$ and $k = 10$. These are chosen based on experience with other recycling algorithms [44, 59]. The relative tolerance for RBiCG is taken as 10^{-8}. The initial guess (for both systems) is a vector of all ones. The linear systems are split-preconditioned by a Crout version of the ILUT preconditioner with a drop tolerance of 0.05 [47]. The generated recycle spaces pertain to the preconditioned linear systems.

Figure 6.1(b) shows the convergence improvement of RBiCG, as it solves the primary system multiple times. For the second run, the reduction in iterations is around 35%. The convergence improves further with each run. Next, we present a brief analysis of the generated recycle spaces. In Table 6.1, we give the cosines of the principal angles between the primary (dual) recycle space and the right (left) invariant subspace associated with the eight eigenvalues of smallest magnitude. As for the recycle spaces, the invariant subspaces are computed for the preconditioned operator. As the recycle space improves, the principal angles tend to zero, and so the cosines tend to one. The table shows that with only a few runs, RBiCG accurately approximates increasingly larger subspaces of the invariant subspace. As a result, we see faster convergence for every new run.

6.2. Model reduction. Our test dynamical system is a semidiscretized heat transfer problem for determining the optimal cooling of steel profiles [45, 13, 50]. We will refer to this model as the *rail model* [45]. The rail model has seven inputs and six outputs. Since we focus on SISO systems in this paper, we choose a SISO subsystem corresponding to the second input and sixth output. The rail model is available with 1357, 5177, 20209, and 79841 unknowns, depending on the mesh size.

As convergence tolerance for IRKA we use a relative change in the shifts of less than 10^{-6}. The matrices A and E of (5.1) are symmetric negative definite and symmetric positive definite (SPD), respectively. Since our shifts are real and positive at every IRKA step, $(\sigma_i^{(m)} E - A)$ is always SPD. Nevertheless, RBiCG is advantageous here because of the backward error formulation discussed in section 5.2. We carry out two sets of experiments that differ in the dimension, r, of the reduced models. We also vary the frequency of computing a recycle space, as a recycle space can be effective for multiple consecutive systems [44, 36] and updating it may be expensive.

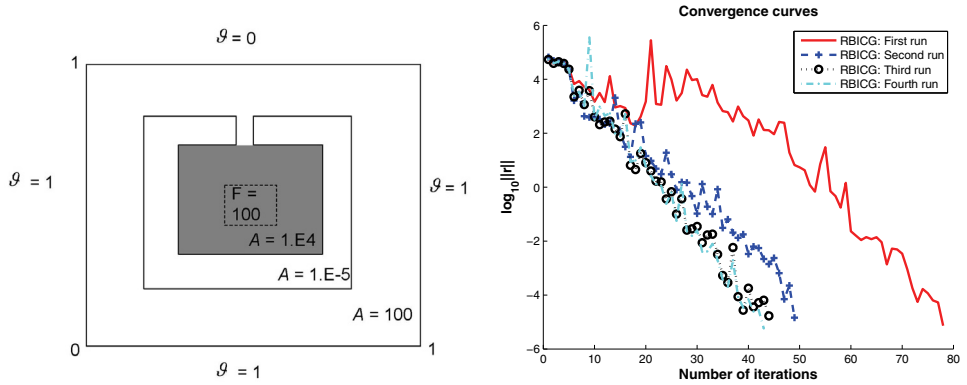

FIG. 6.1. *RBiCG for a convection-diffusion problem.* (a) *The coefficients of the PDE.* (b) *Convergence for preconditioned RBiCG with* $s = 40$ *and* $k = 10$ *for the primary system solved four times to analyze convergence improvement as the recycle space improves.*

TABLE 6.1

Convergence of the recycle space for the convection-diffusion problem as measured by the cosines of the principal angles between the primary (dual) recycle space and the right (left) invariant subspace associated with the eight eigenvalues of smallest magnitude.

Primary system			Dual system		
Start of run 2	Start of run 3	Start of run 4	Start of run 2	Start of run 3	Start of run 4
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
0.9896	1.0000	1.0000	0.9950	1.0000	1.0000
0.3832	1.0000	1.0000	0.9884	1.0000	1.0000
0.1452	0.9983	1.0000	0.7864	0.9844	1.0000
0.0988	0.9437	0.9970	0.6070	0.9206	0.8141
0.0300	0.1869	0.9567	0.4749	0.4118	0.4721

We implement the first recycling strategy from section 5.3 for a few selected shifts. As for the convection-diffusion example, the recycling parameters, s and k, are chosen based on experience with other recycling algorithms [44, 59]. If a pair of linear systems converges in fewer than s iterations, the recycle space is not updated, and we use the previous recycle space for the next pair of systems in the sequence. The relative convergence tolerance for the iterative solves and the tolerance for constructing $\tilde{C}_j$ and C_j in section 4.3 are taken as 10^{-6}. The linear systems are split-preconditioned with an incomplete LU preconditioner with threshold and pivoting (ILUTP) [47]. The drop tolerance varies per problem to avoid ill-conditioning; see Figures 6.2–6.5. The initial guess of the preconditioned system is the solution vector from the previous preconditioned system in the sequence. For the first IRKA step, we take a vector of all zeros as the initial guess. In general, a better initial guess may be based on knowledge of the system and aim to avoid orthogonal initial residuals (Algorithm 1, step 2; Algorithm 2, step 3).

For the first set of experiments, we reduce the models to $r = 6$ degrees of freedom, with 1.00×10^{-5}, 1.38×10^{-4}, 1.91×10^{-3}, 2.63×10^{-2}, 3.63×10^{-1}, and 5.01 as initial shifts.

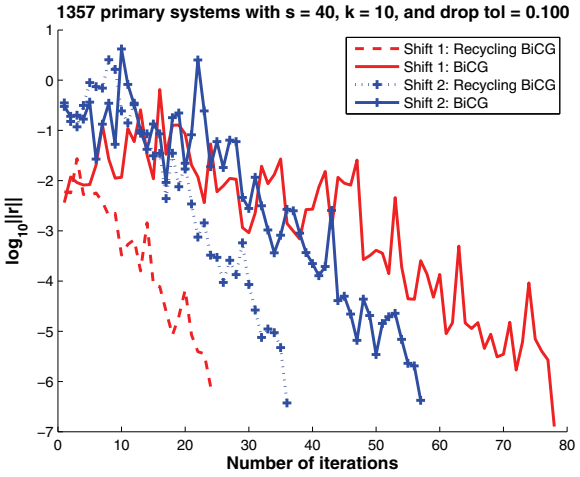

FIG. 6.2. *Convergence of preconditioned RBiCG at the third IRKA step for the* $n = 1357$ *rail model, with* $s = 40$, $k = 10$, *and the preconditioner drop tolerance* 0.1.

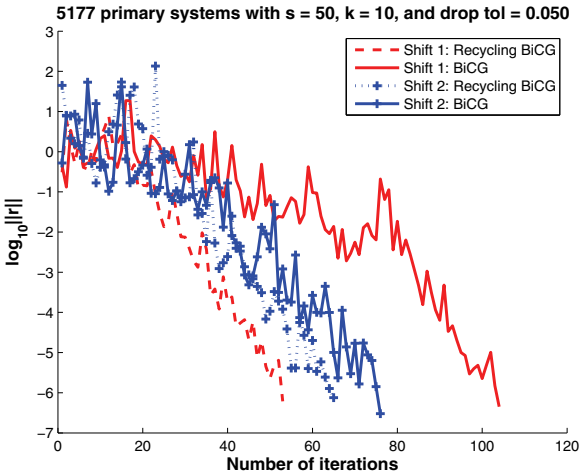

FIG. 6.3. *Convergence of preconditioned RBiCG at the second IRKA step for the* $n = 5177$ *rail model, with* $s = 50$, $k = 10$, *and the preconditioner drop tolerance* 0.05.

We compute a recycle space at every IRKA step. The results for the primary systems at a particular IRKA step (given in the caption) are given in Figures 6.2–6.5. The graphs for the other IRKA steps are similar, as are the graphs for the dual systems. We carry out recycling for the smallest two shifts. Each figure has two solid curves for the linear systems solved without recycling and two dashed-dotted curves for those solved with recycling. It is evident that recycling significantly reduces the number of iterations. The savings in iterations are as high as 70% per system. As discussed in section 5.3, convergence for the remaining four (larger) shifts is rapid, so recycling Krylov subspaces is not useful for these.

Next, we analyze the recycle space generated during the first two IRKA steps for the order 5177 rail model corresponding to the smallest shift. In Table 6.2, we give

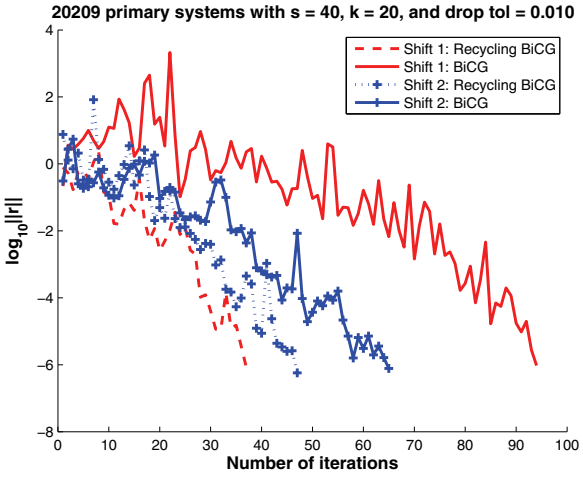

FIG. 6.4. *Convergence of preconditioned RBiCG at the second IRKA step for the* $n = 20209$ *rail model, with* $s = 40$, $k = 20$, *and the preconditioner drop tolerance* 0.01.

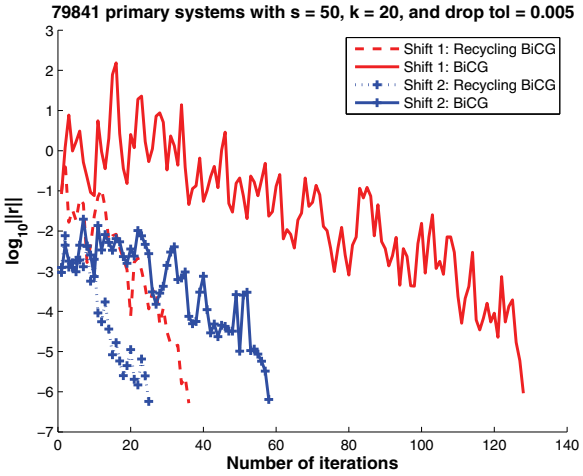

FIG. 6.5. *Convergence of preconditioned RBiCG at the third IRKA step for the* $n = 79841$ *rail model, with* $s = 50$, $k = 20$, *and the preconditioner drop tolerance* 0.005.

the cosines of principal angles between the recycle space and the invariant subspace spanned by eight eigenvectors associated with the eigenvalues of smallest magnitude. As for the recycle space, the invariant subspace is computed for the preconditioned operator. For the primary system, we use the right-invariant subspace. For the dual system, we use the left-invariant subspace. As the recycle space improves, the principal angles tend to zero, and so the cosines tend to one. Consider the results for the primary system. At the first IRKA step and the end of the first cycle, we see that the recycle space captures a subspace of dimension four of the invariant subspace. The recycle space gets more accurate at the end of the second cycle and captures a subspace of dimension seven. For the second IRKA step, we have a new shift,

TABLE 6.2

Convergence of the recycle space for the sequence of linear systems corresponding to the 5177 *rail model and the smallest shift, as measured by the cosines of the principal angles between the primary (dual) recycle space and the right (left) invariant subspace associated with the eight eigenvalues of smallest magnitude. The third column corresponds to the dashed convergence curve in Figure* 6.3.

Primary system				Dual system			
IRKA step 1 $\sigma_1 = 1.000 \times 10^{-5}$		IRKA step 2 $\sigma_1 = 1.834 \times 10^{-5}$		IRKA step 1 $\sigma_1 = 1.000 \times 10^{-5}$		IRKA step 2 $\sigma_1 = 1.834 \times 10^{-5}$	
End of cycle 1	End of cycle 2	Start of cycle 1	End of cycle 1	End of cycle 1	End of cycle 2	Start of cycle 1	End of cycle 1
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000	1.0000	1.0000	1.0000	0.9997	1.0000	1.0000	1.0000
0.9987	1.0000	1.0000	1.0000	0.9765	1.0000	1.0000	1.0000
0.9321	1.0000	1.0000	1.0000	0.4936	1.0000	1.0000	1.0000
0.2257	1.0000	0.9998	0.9999	0.0844	0.9995	0.9997	0.9998
0.0260	0.9997	0.9996	0.9997	0.0231	0.9945	0.9945	0.9989
0.0072	0.7813	0.7799	0.9932	0.0068	0.3439	0.3423	0.9876

TABLE 6.3

The total number of iterations and computation time over all IRKA iterations with BiCG and with RBiCG for the linear systems with the smallest shift. Total time includes the time for all computations to compute the reduced model.

Size	s	k	Drop tol	IRKA steps	Total iteration count			Total time(s)		
					BiCG	RBiCG	Ratio	BiCG	RBiCG	Ratio
20209	40	20	0.01	31	3032	1434	2.11	73.82	54.28	1.36
79841	50	20	0.005	44	6324	2547	2.48	742.83	505.09	1.47

and so the matrix changes. Therefore, at the start of the first cycle, we see a slight deterioration of the recycle space (almost negligible). This recycle space leads to the dashed curve in Figure 6.3. By the end of the first cycle (at the second IRKA step), all eight eigenvectors are captured. The results for the dual system recycle space are similar.

For the second set of experiments, we reduce the models to $r = 3$ degrees of freedom, using as initial shifts $1.00 \times 10^{-5}, 7.08 \times 10^{-3}$, and 5.01. We compute the recycle space at every fifth IRKA step. The results are given in Table 6.3. We implement recycling for the smallest shift only. The linear systems corresponding to the two (larger) shifts converge fast, so recycling Krylov subspaces is not useful for these. *Total iteration count* refers to the sum of iteration counts for solving linear systems over all shifts and all IRKA steps. *Total time* is the time in seconds required by IRKA to converge to the ideal shifts. This includes the time for all IRKA computations as well as all linear solves (BiCG or RBiCG, as the case may be). The table illustrates that computing the reduced model without recycling takes about 50% more time than with recycling. Obviously, the improvement for just the pairs of linear systems where recycling is actually used is substantially larger.

7. Conclusion. We focus on efficiently solving sequences of dual linear systems. For several classes of problems, such as the linear systems arising in interpolatory model reduction, or bilinear forms arising in quantum Monte Carlo methods, the BiCG algorithm has advantages over methods like GMRES that would solve the primary and the dual system separately. For sequences of dual linear systems arising in such problems, it is advantageous to use Krylov subspace recycling for the BiCG algorithm, and for this purpose we propose the RBiCG algorithm. The derivation of RBiCG also provides the foundation for recycling variants of other popular bi-Lanczos based methods, like CGS, BiCGSTAB, QMR, and TFQMR [3].

We have demonstrated the usefulness of RBiCG for interpolatory model reduction using IRKA, an application that may be an important niche for this solver. In addition, we have analyzed and demonstrated the effectiveness of RBiCG for non-symmetric linear systems arising from convection-diffusion problems. This suggests that the RBiCG method may be useful in other areas where solving dual systems in a Petrov–Galerkin sense brings special advantages.

In future work, we plan to extend the use of RBiCG to model reduction for MIMO dynamical systems in a tangential interpolation framework where the right-hand sides are not constant as in the SISO case. In addition, we will investigate the use of RBiCG for evaluating bilinear forms arising in QMC algorithms. Our current results for this look promising.

Acknowledgment. We thank the anonymous reviewers for their careful and helpful suggestions, which greatly helped us improve this paper.

REFERENCES

[1] A. M. Abdel-Rehim, R. B. Morgan, and W. Wilcox, *Deflated BiCGStab for Linear Equations in QCD Problems*, in Proceedings of LAT2007, 2007, pp. 026/1–026/7.

[2] A. M. Abdel-Rehim, A. Stathopoulos, and K. Orginos, *Extending the eigCG Algorithm to Non-symmetric Lanczos for Linear Systems with Multiple Right-hand Sides*, Technical Report WM-CS-2009-06, College of William and Mary, Williamsburg, VA, 2009.

[3] K. Ahuja, *Recycling bi-Lanczos Algorithms: BiCG, CGS, and BiCGSTAB*, Master's thesis, Department of Mathematics, Virginia Tech, Blacksburg, VA, 2009. Available online from http://scholar.lib.vt.edu/theses/available/etd-08252009-161256/.

[4] K. Ahuja, *Recycling Krylov Subspaces and Preconditioners*, Ph.D. thesis, Virginia Tech, Blacksburg, VA, 2011. Available online from http://scholar.lib.vt.edu/theses/available/etd-11112011-010340/.

[5] K. Ahuja, B. K. Clark, E. de Sturler, D. M. Ceperley, and J. Kim, *Improved scaling for quantum Monte Carlo on insulators*, SIAM J. Sci. Comput., 33 (2011), pp. 1837–1859.

[6] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*, Adv. Des. Control 6, SIAM, Philadelphia, 2005.

[7] A. C. Antoulas, C. A. Beattie, and S. Gugercin, *Interpolatory model reduction of large-scale dynamical systems*, in Efficient Modeling and Control of Large-Scale Systems, J. Mohammadpour and K. Grigoriadis, eds., Springer-Verlag, New York, 2010, pp. 3–58.

[8] R. E. Bank and T. F. Chan, *An analysis of the composite step biconjugate gradient method*, Numer. Math., 66 (1993), pp. 295–319.

[9] C. A Beattie, *Harmonic Ritz and Lehmann bounds*, Electron. Trans. Numer. Anal., 7 (1998), pp. 18–39.

[10] C. A. Beattie and S. Gugercin, *Krylov-based minimization for optimal $\mathcal{H}_2$ model reduction*, in Proceedings of the 46th IEEE Conference on Decision and Control, 2007, pp. 4385–4390.

[11] C. A. Beattie and S. Gugercin, *A trust region method for optimal $\mathcal{H}_2$ model reduction*, in Proceedings of the 48th IEEE Conference on Decision and Control and the 28th Chinese Control Conference, 2009, pp. 5370–5375.

[12] C. A Beattie, S. Gugercin, and S. Wyatt, *Inexact solves in interpolatory model reduction*, Linear Algebra Appl., 436 (2012), pp. 2916–2943.

[13] P. Benner, *Solving large-scale control problems*, IEEE Control Systems Magazine, 24 (2004), pp. 44–59.
[14] P. Benner and L. Feng, *Recycling Krylov subspaces for solving linear systems with successively changing right-hand sides arising in model reduction*, in Model Reduction for Circuit Simulation, P. Benner, M. Hinze, and E. Jan W. ter Maten, eds., Lect. Notes Electr. Eng. 74, Springer-Verlag, Dordrecht, 2011, pp. 125–140.
[15] E. de Sturler, *Inner-outer methods with deflation for linear systems with multiple right-hand sides*, in Proceedings of the Householder Symposium on Numerical Algebra, Householder Symposium XIII, Pontresina, Switzerland, 1996, pp. 193–196.
[16] E. de Sturler, *Nested Krylov methods based on GCR*, J. Comput. Appl. Math., 67 (1996), pp. 15–41.
[17] E. de Sturler, *BiCG explained*, in Proceedings of the Householder International Symposium in Numerical Algebra, Householder Symposium XIV, Chateau Whistler, Whistler, BC, Canada, 1999.
[18] E. de Sturler, *Truncation strategies for optimal Krylov subspace methods*, SIAM J. Numer. Anal., 36 (1999), pp. 864–889.
[19] C. de Villemagne and R. E. Skelton, *Model reductions using a projection formulation*, Internat. J. Control, 46 (1987), pp. 2141–2169.
[20] M. Eiermann, O. G. Ernst, and O. Schneider, *Analysis of acceleration strategies for restarted minimal residual methods*, J. Comput. Appl. Math., 123 (2000), pp. 261–292.
[21] S. C. Eisenstat, H. C. Elman, and M. H. Schultz, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983), pp. 345–357.
[22] L. Feng, P. Benner, and J. G. Korvink, *Parametric model order reduction accelerated by subspace recycling*, in Proceedings of the 48th IEEE Conference on Decision and Control and the 28th Chinese Control Conference, 2009, pp. 4328–4333.
[23] R. Fletcher, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis, Lecture Notes in Math. 506, Springer, Berlin, 1976, pp. 73–89.
[24] R. W. Freund, *Solution of shifted linear systems by quasi-minimal residual iterations*, in Numerical Linear Algebra, L. Reichel, A. Ruttan, and R. S. Varga, eds., de Gruyter, Berlin, 1993, pp. 101–121.
[25] R. W. Freund, *A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems*, SIAM J. Sci. Comput., 14 (1993), pp. 470–482.
[26] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal, *An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices*, SIAM J. Sci. Comput., 14 (1993), pp. 137–158.
[27] R. W. Freund and N. M. Nachtigal, *QMR: A quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math., 60 (1991), pp. 315–339.
[28] A. Frommer, *BiCGStab(ℓ) for families of shifted linear systems*, Computing, 70 (2003), pp. 87–109.
[29] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, SIAM, Philadelphia, 1997.
[30] E. J. Grimme, *Krylov Projection Methods for Model Reduction*, Ph.D. thesis, University of Illinois, Champaign, IL, 1997.
[31] S. Gugercin, *An iterative rational Krylov algorithm (IRKA) for optimal $\mathcal{H}_2$ model reduction*, in Proceedings of the Householder Symposium XVI, Seven Springs Mountain Resort, PA, 2005.
[32] S. Gugercin, A. C. Antoulas, and C. A. Beattie, *$\mathcal{H}_2$ model reduction for large-scale linear dynamical systems*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 609–638.
[33] M. H. Gutknecht, *Lanczos-type solvers for nonsymmetric linear systems of equations*, Acta Numer., 6 (1997), pp. 271–397.
[34] M. H. Gutknecht, *Spectral Deflation in Krylov Solvers: A Theory of Coordinate Space Based Methods*, Technical Report 2011-71, Seminar for Applied Mathematics, ETH Zurich, Zurich, Switzerland, 2011.
[35] B. Jegerlehner, *Krylov Space Solvers for Shifted Linear Systems*, Technical report, 1996. Available online at http://arxiv.org/abs/hep-lat/9612014.
[36] M. E. Kilmer and E. de Sturler, *Recycling subspace information for diffuse optical tomography*, SIAM J. Sci. Comput., 27 (2006), pp. 2140–2166.
[37] J. G. Korvink and E. B. Rudnyi, *Oberwolfach benchmark collection*, in Dimension Reduction of Large-Scale Systems, P. Benner, V. Mehrmann, and D. C. Sorensen, eds., Lect. Notes Comput. Sci. Eng. 45, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 311–315.
[38] C. Lanczos, *Solution of systems of linear equations by minimized iterations*, J. Research Nat. Bur. Standards, 49 (1952), pp. 33–53.
[39] L. Meier III and D. Luenberger, *Approximation of linear constant systems*, IEEE Trans.

Automat. Control, 12 (1967), pp. 585–588.
[40] L. A. M. Mello, E. de Sturler, G. H. Paulino, and E. C. N. Silva, *Recycling Krylov subspaces for efficient large-scale electrical impedance tomography*, Comput. Methods Appl. Mech. Engrg., 199 (2010), pp. 3101–3110.
[41] R. B. Morgan, *A restarted GMRES method augmented with eigenvectors*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 1154–1171.
[42] R. B. Morgan, *GMRES with deflated restarting*, SIAM J. Sci. Comput., 24 (2002), pp. 20–37.
[43] C. C. Paige, B. N. Parlett, and H. A. van der Vorst, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Numer. Linear Algebra Appl., 2 (1995), pp. 115–133.
[44] M. L. Parks, E. de Sturler, G. Mackey, D. D. Johnson, and S. Maiti, *Recycling Krylov subspaces for sequences of linear systems*, SIAM J. Sci. Comput., 28 (2006), pp. 1651–1674.
[45] T. Penzl, *Algorithms for model reduction of large dynamical systems*, Linear Algebra Appl., 415 (2006), pp. 322–343.
[46] A. Ruhe, *Rational Krylov algorithms for nonsymmetric eigenvalue problems.* II. *Matrix pairs*, Linear Algebra Appl., 197/198 (1994), pp. 283–295.
[47] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.
[48] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[49] Y. Saad, M. Yeung, J. Erhel, and F. Guyomarc'h, *A deflated version of the conjugate gradient algorithm*, SIAM J. Sci. Comput., 21 (2000), pp. 1909–1926.
[50] J. Saak and P. Benner, *Efficient numerical solution of the LQR-problem for the heat equation*, PAMM Proc. Appl. Math. Mech., 4 (2004), pp. 648–649.
[51] P. Sonneveld, *CGS, a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.
[52] J. T. Spanos, M. H. Milman, and D. L. Mingori, *A new algorithm for L^2 optimal model reduction*, Automatica J. IFAC, 28 (1992), pp. 897–909.
[53] A. Stathopoulos and K. Orginos, *Computing and deflating eigenvalues while solving multiple right-hand side linear systems with an application to quantum chromodynamics*, SIAM J. Sci. Comput., 32 (2010), pp. 439–462.
[54] Z. Strakoš and P. Tichý, *On efficient numerical approximation of the bilinear form $c^*A^{-1}b$*, SIAM J. Sci. Comput., 33 (2011), pp. 565–587.
[55] J. van den Eshof and G. L. G. Sleijpen, *Accurate conjugate gradient methods for families of shifted systems*, Appl. Numer. Math., 49 (2004), pp. 17–37.
[56] H. A. van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.
[57] H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge University Press, Cambridge, UK, 2003.
[58] P. Van Dooren, K. A. Gallivan, and P.-A. Absil, *$\mathcal{H}_2$-optimal model reduction of MIMO systems*, Appl. Math. Lett., 21 (2008), pp. 1267–1273.
[59] S. Wang, E. de Sturler, and G. H. Paulino, *Large-scale topology optimization using preconditioned Krylov subspace methods with recycling*, Internat. J. Numer. Methods Eng., 69 (2006), pp. 2441–2468.
[60] D. A. Wilson, *Optimum solution of model-reduction problem*, Proc. IEE, 117 (1970), pp. 1161–1165.
[61] A. Yousuff and R. E. Skelton, *Covariance equivalent realizations with application to model reduction of large scale systems*, in Control and Dynamic Systems, Vol. 22, C. T. Leondes, ed., Academic Press, Orlando, FL, 1985, pp. 273–348.
[62] A. Yousuff, D. A. Wagie, and R. E. Skelton, *Linear system approximation via covariance equivalent realizations*, J. Math. Anal. Appl., 106 (1985), pp. 91–115.