SIAM J. MATRIX ANAL. APPL.
Vol. 38, No. 2, pp. 401–424
© 2017 Society for Industrial and Applied Mathematics

PRECONDITIONED MULTISHIFT BiCG FOR $\mathcal{H}_2$-OPTIMAL MODEL REDUCTION*

MIAN ILYAS AHMAD†, DANIEL B. SZYLD‡, AND MARTIN B. VAN GIJZEN§

Abstract. We propose the use of a multishift biconjugate gradient method (BiCG) in combination with a suitable chosen polynomial preconditioning, to efficiently solve the two sets of multiple shifted linear systems arising at each iteration of the iterative rational Krylov algorithm (IRKA) [Gugercin, Antoulas, and Beattie, *SIAM J. Matrix Anal. Appl.*, 30 (2008), pp. 609–638] for $\mathcal{H}_2$-optimal model reduction of linear systems. The idea is to construct in advance bases for the two preconditioned Krylov subspaces (one for the matrix and one for its adjoint). By exploiting the shift-invariant property of Krylov subspaces, these bases are then reused inside the model reduction methods for the other shifts. The polynomial preconditioner is chosen to maintain this shift-invariant property. This means that the shifted systems can be solved without additional matrix-vector products. The performance of our proposed implementation is illustrated through numerical experiments.

Key words. model order reduction, IRKA, shifted linear systems, polynomial preconditioning, BiCG

AMS subject classifications. 65F10, 65F50, 41A05

DOI. 10.1137/130914905

1. Introduction.

Numerical simulations are often based on large-scale complex models that are used to measure and control the behavior of some output parameters with respect to a given set of inputs. The goal of model reduction is to approximate this input-output behavior by a much simpler model that can predict the actual behavior. Model reduction is used in many areas, including modeling large-scale dynamical systems [2], and applications such as electronic design, where it is used to predict the behavior of complicated interconnect systems [9].

Consider a single-input single-output (SISO) linear time invariant system,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + bu(t), \\ y(t) &= c^T x(t), \end{aligned} \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$, $b, c \in \mathbb{R}^n$, with n being the order (or dimension) of the system, and $x(t) \in \mathbb{R}^n$, $u(t), y(t) \in \mathbb{R}$ are the state, input, and output, respectively. We assume that the eigenvalues of the matrix A lie in the open left half plane, meaning that the system is stable. In terms of the transfer function representation, (1.1) is given by $G(s) = c^T(sI - A)^{-1}b$.

Model reduction aims at identifying another system, of much smaller dimension $m \ll n$, similar to (1.1), i.e., of the form

*Received by the editors March 29, 2013; accepted for publication (in revised form) by D. P. O'Leary January 4, 2017; published electronically May 25, 2017.
http://www.siam.org/journals/simax/38-2/91490.html
Funding: The work of the second author was supported in part by the U.S. Department of Energy under grants DE-FG02-05ER25672 and DE-SC0016578, and the U.S. National Science Foundation under grants DMS-1115520 and DMS-1418882.
†Max Planck Institute for Dynamics of Complex Technical Systems, 39106 Magdeburg, Germany, and Research Center for Modeling and Simulation, H-12 Campus, Islamabad, Pakistan (imahmad@mpi-magdeburg.mpg.de).
‡Department of Mathematics College of Science and Technology, Temple University, Philadelphia, PA 19122-6094 (szyld@temple.edu).
§Delft Institute of Applied Mathematics, 2628 CD Delft, The Netherlands (M.B.vanGijzen@tudelft.nl).

$$\begin{aligned} \dot{x}_m(t) &= A_m x_m(t) + b_m u(t), \\ y_m(t) &= c_m^T x_m(t), \end{aligned} \tag{1.2}$$

with $A_m \in \mathbb{R}^{m\times m}$, $b_m, c_m, x_m(t) \in \mathbb{R}^m$. The goal is that $y_m(t)$ is close to $y(t)$ in some norm so that the reduced system can be used as a surrogate to the original system. In the frequency domain, this means that the transfer function of the reduced system (1.2) given by $G_m(s) = c_m^T(sI_m - A_m)^{-1}b_m$ approximates $G(s)$ well, in a certain system norm. Here we consider $\mathcal{H}_2$-optimal model reduction where one seeks a stable minimizer to the $\mathcal{H}_2$-norm of $G - G_m$ over all possible choices of stable G_m, and the $\mathcal{H}_2$-norm is defined as

$$\|G\|_{\mathcal{H}_2}^2 := \frac{1}{2\pi}\int_{-\infty}^{\infty} |G(iw)|^2 dw.$$

Projection techniques are often used to compute a reduced system of the form given in (1.2); see, e.g., [5]. In general, projection involves the following steps:

- Compute matrices $V_m \in \mathbb{R}^{n\times m}$ and $W_m \in \mathbb{R}^{n\times m}$, whose columns span the m-dimensional subspaces $\mathcal{V}$ and $\mathcal{W}$, respectively.
- Approximate $x(t)$ by $V_m x_m(t)$.
- Impose the Petrov–Galerkin conditions

$$\begin{aligned} W_m^T(V_m x_m(t) - AV_m x_m(t) - bu(t)) &= 0, \\ y_m(t) &= c^T V_m x_m(t). \end{aligned}$$

The matrices and vectors to be used in the reduced system (1.2) are then given by

$$A_m = (W_m^T V_m)^{-1} W_m^T A V_m, \quad b_m = (W_m^T V_m)^{-1} W_m^T b, \quad c_m = V_m^T c. \tag{1.3}$$

This means that the reduced system depends on the choice of the matrices V_m and W_m, or equivalently on the subspaces $\mathcal{V}$ and $\mathcal{W}$. These subspaces can be constructed in different ways, resulting in a variety of model reduction techniques. For a detailed analysis of these techniques; see, e.g., [2]. In this paper, we consider rational interpolatory projection methods [15, 26, 27] for model reduction, where $\mathcal{V}$ and $\mathcal{W}$ are appropriate (left and right) Krylov subspaces such that the particular choice allows us to enforce certain interpolation conditions and the construction involve matrix-vector products (or solution of shifted systems) of dimension n; see further section 1.1. For another approach, using moment matching and Krylov spaces; see [29].

The concept of projection-based interpolation was introduced in [34] and modified to a computationally efficient setting in [15]. This idea is linked with $\mathcal{H}_2$-optimal approximation of linear systems in [18], where a fixed point iterative framework was used to propose the iterative rational Krylov algorithm (IRKA). Upon convergence, the algorithm produces a reduced system that satisfies interpolation-based necessary conditions for $\mathcal{H}_2$-optimality. IRKA has received considerable attention in the literature and has been extended to more general interpolatory projection methods, including parameterized model order reduction [4], model reduction of linear descriptor systems [19], and interpolatory projection methods for bilinear systems [7, 10].

In this paper, we propose an alternative implementation to methods such as IRKA, by utilizing an appropriately preconditioned biconjugate gradient (BiCG) method [11]. In IRKA and other interpolatory projection methods, at every iteration, two systems,

$$(\sigma I - A)x = b \ \text{ and } \ (\sigma I - A^T)\tilde{x} = c, \tag{1.4}$$

must be solved for every shift frequency $\sigma \in S_m = \{\sigma_1, \ldots, \sigma_m\}$. For each shift, BiCG can solve both of these shifted linear systems simultaneously, and we call this multishift BiCG. While the approximate solutions produced by BiCG (or any other iterative method) are not exact, the results in [6, 36] indicate that the reduced order models obtained with these approximations are optimal for a model which is close to the original problem.

In addition, we take advantage of the shift-invariant property of Krylov subspaces, and thus, use the same space for all shifts simultaneously. In other words, we can build a basis for one of the shifts, called the seed system, and use the same basis for BiCG for the other systems, avoiding any additional matrix-vector products, and thus achieving considerable computational savings [13, 23]. The shift-invariant property can be simply stated as

$$\mathcal{K}_i(A, b) = \mathcal{K}_i((\sigma I - A), b), \tag{1.5}$$

where $\mathcal{K}_i(A, b) := \text{span}\{b, Ab, A^2b, \ldots, A^{i-1}b\}$, i.e., an element of this subspace can be written as $p(A)b$, where $p(z)$ is a polynomial of degree $i - 1$.

A third component of our proposed contribution is the use of specific polynomial preconditioners, which allows us to simultaneously precondition a set of shifted linear systems, without losing the shift-invariance property of the corresponding Krylov subspaces, as considered in [12]. We also discuss a polynomial preconditioner for matrices with a strongly complex spectrum. In this manner, the same preconditioner can be used for all shifts, in contrast to other preconditioners, such as those discussed in [36].

We mention that the use of BiCG for IRKA has already been proposed in [36] and [1]. In the latter reference, a set of dual shifted systems is solved such that the Krylov subspace associated with a pair of shifted systems is updated and reused for the next pair of shifted systems belonging to the same set. Our approach also uses BiCG, as described above, but instead of reusing some vectors, and restarting the computations, we use the same basis as for the seed system, without any additional matrix-vector product, and with an appropriate polynomial preconditioner. We also mention that BiCG was also used for moment matching in [30].

In addition, we note that some of the ideas presented in this paper, such as storing the bases of the seed system, and the use of suitable polynomial preconditioners, could also be applied when using (restarted) GMRES [14] or BiCGStab [13] (or for that matter QMR [12]), but in these cases, of course, one needs to solve the two systems (1.4) (for each shift).

Remark 1.1 (MIMO systems). The shift invariance property given in (1.5) is applicable to systems with fixed b. In case of multi-input multi-output systems, tangential interpolation is often used for model order reduction, in which case there are vectors $b(\sigma)$ that vary with the interpolation points. Therefore, the proposed framework would have to be revisited since it is not directly applicable to the tangential interpolation framework.

Remark 1.2 (Descriptor systems). The general form of the system (1.1) often includes a coefficient matrix $E \in \mathbb{R}^{n\times n}$ in front of the vector $\dot{x}(t)$. The property (1.5) holds if the matrix E is the identity matrix, or if it is nonsingular by considering the Krylov subspace $\mathcal{K}_i(E^{-1}A, E^{-1}b)$, whose basis is built without necessarily explicitly computing E^{-1}. In the case of singular E, one may use the setup given in [19], where the transfer function is first decomposed into strictly proper and polynomial parts and then IRKA-type iterations are used only on the strictly proper part of the transfer function. However, the approach of decomposing a transfer function into strictly

proper and polynomial parts involve the explicit computation of spectral projectors. This decomposition is known to be a difficult task [31]. To avoid this issue, certain important classes of descriptor systems have been considered [19, 20], where the same can be achieved without explicit computation of spectral projectors. We assume, throughout the paper, that the original system can be written in the form of (1.1), and in several places, we comment on the case $E \neq I$.

The remainder of this paper is organized as follows. In section 1.1 we further describe rational Krylov methods linked to $\mathcal{H}_2$-optimal model reduction (in particular IRKA); a shifted variant of BiCG that can simultaneously solve a set of shifted systems is reviewed in section 1.2; in section 2 we explain how this variant of BiCG can be used to give a computationally attractive implementation of IRKA; in section 3 we discuss how to appropriately precondition a set of shifted linear systems; and finally, in section 4 we present numerical experiments that show how efficienct our solution methods for the shifted linear systems in IRKA can be.

1.1. Rational Krylov methods and $\mathcal{H}_2$-optimal model reduction. In this section, we discuss a particular choice of Krylov subspaces for $\mathcal{V}$, $\mathcal{W}$ and its link to interpolation and $\mathcal{H}_2$-optimal approximation. We begin with the following result, where $\mathrm{Ran}(V)$ is the range of the matrix V.

THEOREM 1.1. [19] *Let $S_m = \{\sigma_1, \ldots, \sigma_m\} \subset \mathbb{C}$ be a set of m distinct interpolation points that is closed under conjugation. Suppose that $(\sigma_k I - A)^{-1}$ and $(\sigma_k I_m - A_m)^{-1}$ exist for $k = 1, \ldots, m$, and that*

$$\mathrm{Ran}(V_m) = \mathrm{span}\{(\sigma_1 I - A)^{-1} b, \ldots, (\sigma_m I - A)^{-1} b\},$$
$$\mathrm{Ran}(W_m) = \mathrm{span}\{(\sigma_1 I - A^T)^{-1} c, \ldots, (\sigma_m I - A^T)^{-1} c\}.$$

Then for $G_m(s) = c_m^T (sI_m - A_m)^{-1} b_m$, with $W_m^T V_m = I_m$ and A_m, b_m, c_m as defined in (1.3),

$$G_m(\sigma_k) = G(\sigma_k), \quad G_m'(\sigma_k) = G'(\sigma_k), \quad \text{for } k = 1, \ldots, m, \tag{1.6}$$

where as usual, $G'(\sigma_k)$ stands for the derivative of $G(s)$ with respect to s, evaluated at σ_k.

Note that as long as the inverses $(\sigma_k I - A)^{-1}$ and $(\sigma_k I_m - A_m)^{-1}$, $k = 1, \ldots, m$, exist, the reduced transfer function $G_m(s)$ satisfies (1.6). This also holds for descriptor systems as long as the inverses of $(\sigma_k E - A)^{-1}$ and $(\sigma_k E_m - A_m)^{-1}$, $k = 1, \ldots, m$, exist. An important issue is the selection of interpolation points σ_k. The choice of selecting an optimal set of interpolation points provide a link to the $\mathcal{H}_2$-optimal model reduction problem.

The $\mathcal{H}_2$-optimal model reduction problem, as discussed before, is to identify a stable reduced order system $G_m(s)$, which is the best approximation of $G(s)$ in terms of the $\mathcal{H}_2$-norm, i.e.,

$$G_m(s) = \underset{\substack{\hat{G}_m(s) \text{ is stable} \\ \dim(\hat{G}_m(s)) = m}}{\arg \min} \|G - \hat{G}_m\|_{\mathcal{H}_2}. \tag{1.7}$$

The set of stable mth order dynamical systems is not convex and therefore the problem in (1.7) may have multiple local minimizers. An iterative algorithm may not converge to the global minimizer that can solve (1.7) but may result in a local minimizer. Most approaches for solving (1.7) utilize first-order necessary conditions for local optimality.

Algorithm 1.1. The iterative rational Krylov algorithm, IRKA.

1: **Inputs:** A, b, c, m, and tol
2: **Initialize:** Select $S_m = \{\sigma_1, \ldots, \sigma_m\} \in \mathbb{C}^m$ that are closed under conjugation
3: **while** $error < tol$ **do**
4: Compute,

$$\tilde{v}_1 = (\sigma_1 I - A)^{-1} b, \ldots, \tilde{v}_m = (\sigma_m I - A)^{-1} b \,, \tag{1.10}$$

$$\tilde{w}_1 = (\sigma_1 I - A)^{-T} c, \ldots, \tilde{w}_m = (\sigma_m I - A)^{-T} c \tag{1.11}$$

5: Compute matrices with orthonormal columns V_m and W_m with

$$\mathrm{Ran}(V_m) = \mathrm{span}\{\tilde{v}_1, \ldots, \tilde{v}_m\},$$
$$\mathrm{Ran}(W_m) = \mathrm{span}\{\tilde{w}_1, \ldots, \tilde{w}_m\}$$

6: Compute $A_m = (W_m^T V_m)^{-1} W_m^T A V_m$
7: $error = \|\mathrm{sort}(\lambda(-A_m)) - \mathrm{sort}(S_m)\|$
8: Update the interpolation points, $S_m = \lambda(-A_m)$
9: **end while**
10: **Outputs:** A_m, $b_m = (W_m^T V_m)^{-1} W_m^T b$, $c_m = V_m^T c$.

Gugercin [16] introduced an error expression for the $\mathcal{H}_2$-norm of the error system and observed that the error is small if the reduced order model $G_m(s)$ interpolates $G(s)$ at $-\lambda_i(A)$ and $-\lambda_i(A_m)$, where $\lambda_i(A)$ stands for the ith eigenvalue of A. Since $-\lambda_i(A_m)$ is not known a priori, several approaches were developed to minimize the $\mathcal{H}_2$-error by choosing interpolation points to be $-\lambda_i(A)$ [17, 16]. Gugercin et al. [18] showed that interpolation at $-\lambda_i(A_m)$ is more useful and is a necessary condition for $\mathcal{H}_2$-optimality. The following result gives this condition.

LEMMA 1.2. [18] *Given a stable SISO system $G(s) = c^T(sI - A)^{-1}b$, let $G_m(s) = c_m^T(sI_m - A_m)^{-1}b_m$ be a local minimizer of dimension m for the optimal $\mathcal{H}_2$-model reduction problem* (1.7) *and suppose that $G_m(s)$ has simple poles at $-\sigma_k$, $k=1,\ldots,m$. Then $G_m(s)$ interpolates both $G(s)$ and its first derivative at σ_k, $k=1,\ldots,m$, i.e.,*

$$G_m(\sigma_k) = G(\sigma_k), \quad G_m'(\sigma_k) = G'(\sigma_k), \ \textit{for} \ \ k=1,\ldots,m. \tag{1.8}$$

It was also shown in [18] that the necessary optimality conditions of Hyland–Bernstein [22] and Wilson [35] are equivalent to the conditions in Lemma 1.2 in the case of continuous time SISO systems having simple poles.

The necessary conditions (1.8) are equivalent to the root finding problem of

$$\lambda\left(A_m(S_m)\right) = -S_m, \tag{1.9}$$

where $\lambda(\cdot)$ denotes the eigenvalues, and S_m is the set of required roots, and where we have written A_m as $A_m(S_m)$ to emphasize the dependence of A_m on the interpolation points. An iterative framework is used in [18] to compute the interpolation points (roots) that successively update the roots using $S_m^{i+1} = \lambda(-A_m(S_m^i))$. Using Lemma 1.2, the successive updates $S_m^{i+1} = \lambda(-A_m(S_m^i))$ are possible via the rational Krylov method [26, 27]. This leads to the iterative rational Krylov algorithm (IRKA) [18] which is given schematically in Algorithm 1.1.

We remark that care must be taken so that the sort function in step 7 of Algorithm 1.1 pairs perfectly well the corresponding eigenvalues and their

Algorithm 1.2. The biconjugate gradient (BiCG) algorithm.

Inputs: A, b, c, max_it, and tol
Initialize: $x = z = 0$, $r_x = b$, $r_z = c$, $d_x = d_z = 0$, $\rho_{old} = 1$
$n_b = \|b\|$, $n_c = \|c\|$
for $i = 1$ to max_it **do**
  $\rho = r_z^T r_x$, $\beta = -\rho/\rho_{old}$
  $d_x = r_x - \beta d_x, \qquad d_z = r_z - \bar{\beta} d_z$
  $\alpha = \rho/(d_z^T A d)$
  $x = x + \alpha d_x, \qquad z = z + \bar{\alpha} d_z$
  $r_x = r_x - \alpha A d_x, \qquad r_z = r_z - \bar{\alpha} A^T d_z$
  $error = \max\{\|r_x\|/n_b, \|r_z\|/n_c\}$
  if $error < tol$ **then** break **end**
  $\rho_{old} = \rho$
end for
Outputs: x, z

approximation. We also note that the fact that the shifts are closed under conjugation implies that the matrices V_m and W_m can be kept real by combining each pair of complex conjugate basis vectors to two real basis vectors. These real vectors correspond to the real and imaginary parts of the complex conjugate pair.

1.2. Shifted variants of the BiCG algorithm. In this section, we review a variant of the biconjugate gradient (BiCG) method [11], that utilizes the shift invariant property of Krylov subspaces, (1.5), to solve shifted linear systems. We begin with a brief description of the standard BiCG method.

BiCG is based on the non-symmetric Lanczos algorithm that can simultaneously solve a linear system $Ax = b$ and a transposed system $A^T z = c$. For a given $A \in \mathbb{R}^{n\times n}$ and two vectors b, $c \in \mathbb{R}^n$ with $c^T b \neq 0$, it computes two approximate solutions, x_m and z_m, such that

(i) *A-orthogonal search directions*: Given x_0 and z_0, then for $i = 1, 2, \ldots,$ until convergence,

$$x_i = x_{i-1} + \alpha_i d_{x_i} \quad \text{and} \quad z_i = z_{i-1} + \bar{\alpha}_i d_{z_i},$$

where $\alpha_i \in \mathbb{C}$, $\bar{\alpha}_i$ is its complex conjugate and where $d_{x_i}, d_{z_j} \in \mathbb{C}^n$ are A-orthogonal, that is, $d_{z_j}^T A d_{x_i} = 0$ for $i \neq j$,

(ii) *Orthogonal residuals:* Let $r_{x_i} = b - Ax_i$ and $r_{z_j} = c - A^T z_j$ be the residuals for $i, j = 0, 1, 2, \ldots,$ then $r_{z_j}^T r_{x_i} = 0$ for $i \neq j$.

A version of the BiCG algorithm is given in Algorithm 1.2.

We note that the condition $c^T b \neq 0$ is fundamental. In general, if any iteration index results in $\rho = 0$, Algorithm 1.2 has a breakdown of the BiCG method. We mention, however, that we have not encountered this in our experiments in section 4.

In order to solve a set of shifted linear systems, such as those in (1.10)–(1.11), by using basis vectors associated with A and A^T, we consider a seed system and a shifted linear system, i.e.,

$$Ax = b, \qquad (\sigma I - A)\hat{x} = b, \tag{1.12}$$

where $A \in \mathbb{R}^{n\times n}$, and $x, \hat{x}, b \in \mathbb{R}^n$, while $\sigma \in \mathbb{C}$. Let $x_i \in \mathcal{K}_i(A, b)$ and $\hat{x}_i \in \mathcal{K}_i((\sigma I - A), b)$ be the approximations of the above linear systems obtained after i

iterations of the BiCG algorithm (Algorithm 1.2). We can write them as

$$x_i = p_{i-1}(A)b \quad \text{and} \quad \hat{x}_i = \hat{p}_{i-1}(\sigma I - A)b,$$

where p_{i-1} and $\hat{p}_{i-1}$ are polynomials of degree less than or equal to $i-1$. Then the associated residuals $r_i = b - Ax_i$ and $\hat{r}_i = b - (\sigma I - A)\hat{x}_i$ can be written as

$$r_i = q_i(A)b \quad \text{and} \quad \hat{r}_i = \hat{q}_i(\sigma I - A)b, \tag{1.13}$$

where $q_i(t) = 1 - tp_{i-1}(t)$ and $\hat{q}_i = 1 - t\hat{p}_{i-1}(t)$ are polynomials of degree i such that $q_i(0) = \hat{q}_i(0) = 1$, called the residual polynomials.

It turns out that these residuals are necessarily collinear, as established in [13].

LEMMA 1.3. *Let $r_i = b - Ax_i$ and $\hat{r}_i = b - (\sigma I - A)\hat{x}_i$ be the residuals after the ith iteration of BiCG of the two systems in* (1.12)*. Then, there exists $\hat{\zeta}_i \in \mathbb{C}$, such that $r_i = \hat{\zeta}_i \hat{r}_i$.*

As a consequence, we can write

$$\hat{r}_i = (1/\hat{\zeta}_i) r_i, \tag{1.14}$$

which implies

$$\hat{r}_i = \hat{q}_i(\sigma I - A)b = (1/\hat{\zeta}_i) q_i(A)b.$$

Since $\hat{q}_i(\sigma - t) = (1/\hat{\zeta}_i) q_i(t)$ and $\hat{q}_i(0) = 1$,

$$\hat{\zeta}_i = q_i(\sigma). \tag{1.15}$$

The above expression and (1.14) shows that the residual associated with the shifted linear system in (1.12) can be expressed in terms of the residual associated with the seed linear system $Ax = b$. In the following, we give some expressions of the parameters associated with the shifted BiCG method; these are essentially from [13].

From the BiCG algorithm (Algorithm 1.2 with $d := d_x$, $r := r_x$), we have for $i = 1, 2, \ldots,$

$$d_{i-1} = \frac{1}{\beta_i}(r_i - d_i), \qquad Ad_i = \frac{1}{\alpha_i}(r_i - r_{i+1}),$$

and therefore

$$\begin{aligned} r_i = r_{i-1} - \alpha_{i-1} A d_{i-1} &= r_{i-1} - \alpha_{i-1} A \left(\frac{1}{\beta_i}(r_i - d_i) \right), \\ &= r_{i-1} - \frac{\alpha_{i-1}}{\beta_i} A r_i + \frac{\alpha_{i-1}}{\beta_i \alpha_i}(r_i - r_{i+1}), \end{aligned}$$

which results in the following three-term recurrence expression for the residual,

$$r_{i+1} = -\alpha_i A r_i + \frac{\beta_i \alpha_i}{\alpha_{i-1}} r_{i-1} + \left(1 - \frac{\beta_i \alpha_i}{\alpha_{i-1}} \right) r_i. \tag{1.16}$$

In terms of the polynomial representation used in (1.13), the above expression can be written as

$$q_{i+1}(t) = -\alpha_i t q_i(t) + \frac{\beta_i \alpha_i}{\alpha_{i-1}} q_{i-1}(t) + \left(1 - \frac{\beta_i \alpha_i}{\alpha_{i-1}} \right) q_i(t).$$

Setting $t = \sigma$ and using (1.15), we have

$$\hat{\zeta}_{i+1} = \left(1 - \alpha_i\sigma - \frac{\beta_i\alpha_i}{\alpha_{i-1}}\right)\hat{\zeta}_i + \frac{\beta_i\alpha_i}{\alpha_{i-1}}\hat{\zeta}_{i-1}. \tag{1.17}$$

Since we have collinear residuals (1.14), the residual for the shifted system is

$$\begin{aligned}\hat{r}_{i+1} &= \frac{1}{\hat{\zeta}_{i+1}}\left(-\alpha_i\hat{\zeta}_i A\hat{r}_i + \frac{\beta_i\alpha_i\hat{\zeta}_{i-1}}{\alpha_{i-1}}\hat{r}_{i-1} + (\hat{\zeta}_i - \frac{\hat{\zeta}_i\beta_i\alpha_i}{\alpha_{i-1}})\hat{r}_i\right), \\ &= \frac{\alpha_i\hat{\zeta}_i}{\hat{\zeta}_{i+1}}(\sigma I - A)\hat{r}_i + \frac{\beta_i\alpha_i\hat{\zeta}_{i-1}}{\alpha_{i-1}\hat{\zeta}_{i+1}}\hat{r}_{i-1} + \left(\frac{\hat{\zeta}_i}{\hat{\zeta}_{i+1}} - \frac{\hat{\zeta}_i\beta_i\alpha_i}{\alpha_{i-1}\hat{\zeta}_{i+1}} - \sigma\frac{\alpha_i\hat{\zeta}_i}{\hat{\zeta}_{i+1}}\right)\hat{r}_i.\end{aligned}$$

Now, comparing the above equation with the three-term recurrence (similar to (1.16)) for the shifted system given by

$$\hat{r}_{i+1} = -\hat{\alpha}_i(sI - A)r_i^s + \frac{\hat{\beta}_i\hat{\alpha}_i}{\hat{\alpha}_{i-1}}\hat{r}_{i-1} + \left(1 - \frac{\hat{\beta}_i\hat{\alpha}_i}{\hat{\alpha}_{i-1}}\right)\hat{r}_i,$$

we have

$$\begin{aligned}\hat{\alpha}_i &= -\alpha_i\left(\frac{\hat{\zeta}_i}{\hat{\zeta}_{i+1}}\right), \\ \hat{\beta}_i &= \left(\frac{\alpha_i}{\hat{\alpha}_i}\right)\left(\frac{\hat{\alpha}_{i-1}}{\alpha_{i-1}}\right)\frac{\hat{\zeta}_{i-1}}{\hat{\zeta}_{i+1}}\beta_i = \left(\frac{\hat{\zeta}_{i-1}}{\hat{\zeta}_i}\right)^2\beta_i, \\ \hat{\zeta}_{i+1} &= \left(1 - \alpha_i\sigma - \frac{\beta_i\alpha_i}{\alpha_{i-1}}\right)\hat{\zeta}_i + \frac{\beta_i\alpha_i}{\alpha_{i-1}}\hat{\zeta}_{i-1}.\end{aligned}$$

The above expressions are also true for $m = 0$, if we initialize $\hat{\zeta}_{-1} = 1$. A variant of the BiCG method using a single basis for m shifts is given in Algorithm 1.3, where it is assumed that the seed system is the one which would take the largest number of iterations to converge.

Note that, as observed in [13], when finding an update for the approximate solution of shifted systems, Algorithm 1.3 uses $\hat{x} = \hat{x} + \hat{\alpha}\hat{d}_x$, where $\hat{d}_x = \hat{r}_x - \hat{\beta}\hat{d}_x$. Since $\hat{r}_x$ can be expressed in terms of r_x, no matrix-vector multiplications are required for the shifted systems if the residual r_x is known. In the following we use this important result to propose an iterative algorithm for the $\mathcal{H}_2$-optimal model reduction problem.

2. Iterative shifted BiCG for $\mathcal{H}_2$-optimal model reduction. In this section we develop an implementation of the shifted biconjugate gradient method (MS-BiCG) to compute local minimizers to the $\mathcal{H}_2$-optimal model reduction problem. The proposed approach allows us to compute the complete Krylov subspace associated with all shifts in S_m in advance.

We begin with a simple implementation, Algorithm 2.1. Observe that since this algorithm utilizes a shifted variant of the BiCG method it can deal with a set of shifted systems and transposed shifted systems using only matrix-vector multiplications associated with the seed systems, i.e., systems with no shifts. As already mentioned, this avoids computing matrix-vector products corresponding to each shifted system.

Furthermore, one could change the value of *max_it* from one "inner" iteration to the next, thus obtaining different levels of approximations to the solution of the linear

Algorithm 1.3. Multishift biconjugate gradient (MS-BiCG) algorithm.

Inputs: A, b, c, max_it, tol, and $S_m = \{\sigma_1, \ldots, \sigma_m\} \subset \mathbb{C}^m$
Initialize: $x = z = 0$, $\hat{x} = \hat{z} = O \in \mathbb{R}^{n\times m}$ $r_x = b$, $r_z = c$, $d_x = d_z = 0$, $\hat{d}_x = \hat{d}_z = O \in \mathbb{R}^{n\times m}$, $\alpha_{old} = 1$, $\rho_{old} = 1$, $\hat{\zeta}_{old} = \hat{\zeta} = 1$
$n_b = \|b\|$, $n_c = \|c\|$
for $i = 1$ to max_it **do**
  {Linear system with no shift}
  $\rho = r_z^T r_x$, $\beta = -\rho/\rho_{old}$
  $d_x = r_x - \beta d_x, \qquad d_z = r_z - \bar{\beta} d_z$
  $\alpha = \rho/(d_z^T A d_x)$
  $r_x = r_x - \alpha A d_x, \qquad r_z = r_z - \bar{\alpha} A^T d_z$
  $error = \max\{\|r_x\|/n_b, \|r_z\|/n_c\}$
  if $error < tol$ **then** break **end**
  {Shifted systems}
  for $k = 1, \ldots, m$ **do**
    $\hat{\zeta}_{new}(k) = (1 - \alpha\sigma_k)\,\hat{\zeta}(k) + \left(\frac{\alpha\beta}{\alpha_{old}}\right)\left(\hat{\zeta}_{old}(k) - \hat{\zeta}(k)\right)$
    $\hat{\alpha}(k) = -\alpha\left(\frac{\hat{\zeta}(k)}{\hat{\zeta}_{new}(k)}\right)$
    $\hat{\beta}(k) = \left(\frac{\hat{\zeta}_{old}(k)}{\hat{\zeta}(k)}\right)^2 \beta$
    $\hat{d}_x(:,k) = \left(1/\hat{\zeta}(k)\right) r_x - \hat{\beta}(k)\hat{d}_x(:,k)$
    $\hat{d}_z(:,k) = \left(1/\bar{\hat{\zeta}}(k)\right) r_z - \bar{\hat{\beta}}(k)\hat{d}_z(:,k)$
    $\hat{x}(:,k) = \hat{x}(:,k) + \hat{\alpha}(k)\hat{d}_x(:,k)$,
    $\hat{z}(:,k) = \hat{z}(:,k) + \bar{\hat{\alpha}}(k)\hat{d}_z(:,k)$
  end for
  $\hat{\zeta}_{old} = \hat{\zeta}$, $\hat{\zeta} = \hat{\zeta}_{new}$, $\alpha_{old} = \alpha_i$, $\rho_{old} = \rho$
end for
Outputs: $\hat{x}$, $\hat{z}$

Algorithm 2.1. IRKA with MS-BiCG.

Inputs: A, b, c, m, max_it, and tol
Initialize: $S_m = \{\sigma_1, \ldots, \sigma_m\} \subset \mathbb{C}^m$ that are closed under conjugation
while $error < tol$ **do**
  Call the MS-BiCG algorithm (Algorithm 1.3) to compute $\hat{x}$ and $\hat{z}$ with A, b, c, max_it, tol, and S_m as inputs
  $V_m := [\hat{x}]$, $W_m := [\hat{z}]$
  Compute $A_m = (W_m^T V_m)^{-1} W_m^T A V_m$
  $error = \|\mathrm{sort}(\lambda(-A_m)) - \mathrm{sort}(S_m)\|$
  Update the interpolation points, $S_m = \lambda(-A_m)$
end while
Outputs: A_m, $b_m = (W_m^T V_m)^{-1} W_m^T b$, $c_m = V_m^T c$

systems with the shifted matrices $(\sigma_i I - A)$, $i = 1, \ldots, m$. Again, we refer to [6, 36], which provide a justification of using these approximations.

In our proposed implementation, we divide the tasks of the shifted BiCG algorithm (Algorithm 1.3) into two parts. The first part involves the computation of all the residuals $\mathcal{R}_x = [r_{x_1}, r_{x_2}, \ldots, r_{x_m}]$ and $\mathcal{R}_z = [r_{z_1}, r_{z_2}, \ldots, r_{z_m}]$ of the seed systems

Algorithm 2.2. Part (a): Krylov basis for MS-BiCG.

Inputs: A, b, c, max_it, and tol
Initialize: $r_x = b, r_z = c, d_x = d_z = 0$, and $\rho_{old} = 1$
$n_b = \|b\|$, $n_c = \|c\|$
for $i = 1$ to max_it **do**
  $\rho = r_z^T r_x$, $\beta = -\rho/\rho_{old}$
  $d_x = r_x - \beta d_x, \qquad d_z = r_z - \bar{\beta} d_z$
  $\alpha = \rho/(d_z^T A d_x)$
  $\mathcal{R}_x(:,i) = r_x$, $\mathcal{R}_z(:,i) = r_z$, $\mathcal{A}(i) = \alpha$, $\mathcal{B}(i) = \beta$
  $r_x = r_x - \alpha A d_x, \qquad r_z = r_z - \bar{\alpha} A^T d_z$
  $error = \max\{\|r_x\|/n_b, \|r_z\|/n_c\}$
  if $error < tol$ **then** break **end**
  $\rho_{old} = \rho$
end for
Outputs: $\mathcal{R}_x$, $\mathcal{R}_z$, $\mathcal{A}$, $\mathcal{B}$

Algorithm 2.3. Part (b): MS-BiCG iterations with the stored Krylov basis.

Inputs: $\mathcal{R}_x$, $\mathcal{R}_z$, $\mathcal{A}$, $\mathcal{B}$, $S_m = \{\sigma_1, \ldots, \sigma_m\}$
Initialize: $\hat{\zeta}, \hat{\zeta}_{old} = \text{ones}(m, 1)$
$max\_it = size(\mathcal{R}_x, 2) - 1$, $\alpha_{old} = \mathcal{A}_1$
for $i = 1$ to max_it **do**
  $\alpha = \mathcal{A}(i)$, $\beta = \mathcal{B}(i)$, $r_x = \mathcal{R}_x(:,i)$, $r_z = \mathcal{R}_z(:,i)$
  for $k = 1, \ldots, m$ **do**
    $\hat{\zeta}_{new}(k) = (1 - \alpha\sigma_k)\,\hat{\zeta}(k) + \left(\frac{\alpha\beta}{\alpha_{old}}\right)\left(\hat{\zeta}_{old}(k) - \hat{\zeta}(k)\right)$
    $\hat{\alpha}(k) = -\alpha\left(\frac{\hat{\zeta}(k)}{\hat{\zeta}_{new}(k)}\right)$
    $\hat{\beta}(k) = \left(\frac{\hat{\zeta}_{old}(k)}{\hat{\zeta}(k)}\right)^2 \beta$
    $\hat{d}_x(:,k) = \left(1/\hat{\zeta}(k)\right) r_x - \hat{\beta}(k)\hat{d}_x(:,k)$
    $\hat{d}_z(:,k) = \left(1/\bar{\hat{\zeta}}(k)\right) r_z - \bar{\hat{\beta}}(k)\hat{d}_z(:,k)$
    $\hat{x}(:,k) = \hat{x}(:,k) + \hat{\alpha}(k)\hat{d}_x(:,k)$
    $\hat{z}(:,k) = \hat{z}(:,k) + \bar{\hat{\alpha}}(k)\hat{d}_z(:,k)$
  end for
  $\hat{\zeta}_{old} = \hat{\zeta}$, $\hat{\zeta} = \hat{\zeta}_{new}$, $\alpha_{old} = \alpha_i$
end for
Outputs: $\hat{x}$, $\hat{z}$

and all the scalar parameters $\mathcal{A} = [\alpha_1, \ldots, \alpha_m]$ and $\mathcal{B} = [\beta_i, \ldots, \beta_m]$. The second part uses these residuals to compute the coefficients for the collinearity of the residuals of the shifted systems, and the rest of the IRKA method.

The first part is given by Algorithm 2.2. Note that this algorithm does not compute the approximations x_m and z_m since only the residuals and the constant parameters are required for the solution of shifted linear systems.

The second part of Algorithm 1.3 involves the computation of all search directions $\hat{d}_{x_i}$, and $\hat{d}_{z_i}$ and all the scalar parameters $\hat{\mathcal{E}} = [\hat{\zeta}_1, \ldots, \hat{\zeta}_m]$, $\hat{\mathcal{A}} = [\hat{\alpha}_1, \ldots, \hat{\alpha}_m]$, and $\hat{\mathcal{B}} = [\hat{\beta}_1, \ldots, \hat{\beta}_m]$ to obtain approximations to the shifted and transposed shifted

Algorithm 2.4. IRKA with two-part MS-BiCG.

Inputs: A, b, c, m, max_it, and tol
Initialize: Select $S_m = \{s_1, \ldots, s_m\} \in \mathbb{C}^m$ that are closed under conjugation
Call Algorithm 2.2 with inputs A, b, c, max_it, and tol to obtain $\mathcal{R}$, $\tilde{\mathcal{R}}$, $\mathcal{A}$, $\mathcal{B}$
while $error < tol$ **do**
  Call Algorithm 2.3 with inputs $\mathcal{R}$, $\tilde{\mathcal{R}}$, $\mathcal{A}$, $\mathcal{B}$, and S_m to obtain x_s and $\tilde{x}_s$
  $V_m = x_s$ and $W_m = \tilde{x}_s$
  Compute $A_m = (W_m^T V_m)^{-1} W_m^T A V_m$
  $error = \|\mathrm{sort}(\lambda(-A_m)) - \mathrm{sort}(S_m)\|$
  Update the interpolation points, $S_m = \lambda(-A_m)$
end while
Outputs: A_m, $b_m = (W_m^T V_m)^{-1} W_m^T b$, $c_m = V_m^T c$

systems, and it is given in Algorithm 2.3. This part requires the output of the first part, Algorithm 2.2. The advantage of this division of our work is that the outputs of Algorithm 2.2 can be used by different sets of shifted linear systems without computing them again for each set; see further our numerical experiments in section 4. Of course, we have to keep in mind that by doing so, we are incurring the expense of the additional storage of the computed residuals. The resulting IRKA algorithm with the two-part MS-BICG method is given in Algorithm 2.4

3. Preconditioning of the shifted problem. In practice, one uses a suitably chosen preconditioner to accelerate the convergence of the Krylov subspace methods, such as BiCG. For shifted problems this means that, in the case of right-preconditioning, the iterative method (implicitly or explicitly) solves the system

$$(\sigma I - A)\hat{M}^{-1}\hat{y} = b, \quad \hat{x} = \hat{M}^{-1}\hat{y}\ . \tag{3.1}$$

The multishift Krylov methods like the BiCG-based algorithms described in the previous section, all rely on the shift-invariant property (1.5). *Preconditioned* multi-shift methods should satisfy the same property; that is, all the Krylov subspaces for the preconditioned shifted systems must be the same. Without further assumptions on the preconditioners $\hat{M}$ this is clearly not the case. The question we address in this section is how to construct such preconditioners.

Let us assume that a matrix M independent of σ is given, and that for any shift σ a matrix $\hat{M}$ exists such that

$$\mathcal{K}_i(AM^{-1}, b) = \mathcal{K}_i((\sigma I - A)\hat{M}^{-1}, b). \tag{3.2}$$

Although $\hat{M}$ is not needed to generate a basis for $K_i((\sigma I - A)\hat{M}^{-1}, b)$, it is needed in the computation of the solution $\hat{x}$ from the solution $\hat{y}$ of the right-preconditioned system; see (3.1).

It is easy to see that the condition (3.2) is satisfied if a σ dependent parameter $\hat{\eta}$ and a matrix M independent of σ exist such that

$$(\sigma I - A)\hat{M}^{-1} = \hat{\eta} I - AM^{-1}, \tag{3.3}$$

that is, such that the preconditioned shifted matrix can be written as a shifted preconditioned matrix; see also [12, 23]. Furthermore, we have to choose M such that the matrices $\hat{M}$ are efficient preconditioners for the shifted systems.

We first consider the situation in which we choose M and $\hat{\eta}$ such that the shifted matrix $\sigma I - A$ can be an explicit factor in the right-hand side of (3.3). To illustrate this case, we consider the shift-and-invert (SI) preconditioner $M = \mu I - A$, in which μ is a suitably chosen shift. Substituting in (3.3) gives

$$(\sigma I - A)\hat{M}^{-1} = \hat{\eta} I - A(\mu I - A)^{-1}.$$

After some manipulations, this can be rewritten as

$$(\sigma I - A)\hat{M}^{-1} = (1 + \hat{\eta}) \left(\frac{\mu\hat{\eta}}{1 + \hat{\eta}} I - A \right) (\mu I - A)^{-1}.$$

This further simplifies if we choose $\hat{\eta}$ such that

$$\frac{\mu\hat{\eta}}{1 + \hat{\eta}} = \sigma, \quad \text{i.e., } \hat{\eta} = \frac{\sigma}{\mu - \sigma}.$$

The matrix $\hat{M}$ then becomes

$$\hat{M} = \frac{1}{1 + \hat{\eta}} (\mu I - A).$$

As a result, solving systems with both M and $\hat{M}$ only involves the matrix $\mu I - A$; only one LU decomposition of the matrix $\mu I - A$ has to be computed that can be used for both the preconditioning operations and for computing the solutions of the shifted systems.

We remark that also in the more general case of descriptor systems, the SI preconditioner $M = \mu E - A$ can be used. Applying this preconditioner yields, assuming that $\mu E - A$ is invertible, a set of shifted problems

$$(\hat{\eta} I - A(\mu E - A)^{-1})\hat{y} = b \ , \qquad \hat{x} = \frac{1}{1 + \hat{\eta}} (\mu E - A)^{-1} \hat{y},$$

that can be solved with a shifted Krylov method, such as those discussed in the previous section.

The SI preconditioner has some important drawbacks. Although the preconditioner is very effective for shifted systems with σ close to μ, it is much less effective for shifts further away from μ. Another disadvantage is that computing the LU decomposition of $\mu I - A$ may still be prohibitively expensive for large three-dimensional (3D) problems.

It was observed in [12, 23] that polynomial preconditioners also satisfy (3.3); that is, if M^{-1} is a polynomial in A then for every shift σ, an $\hat{\eta}$ exists such that $\hat{M}^{-1}$ is a polynomial in A. In [23], Jegerlehner proposed a simple linear polynomial preconditioner that satisfies this property, whereas in [12] Freund constructed a polynomial preconditioner for a Helmholtz problem with imaginary shifts using the recurrence for Chebyshev polynomials. Below we will give a general method for computing the polynomial $\hat{M}^{-1} = \hat{p}_N(A)$ and parameter $\hat{\eta}$ for a shift σ, given the polynomial preconditioner $M^{-1} = p_N(A)$.

Let

$$p_N(A) = \sum_{i=0}^{N} \gamma_i A^i \tag{3.4}$$

be the polynomial preconditioner of degree N for A and let $\hat{p}_N(A) = \sum_{i=0}^{N} \hat{\gamma}_i A^i$ be the polynomial preconditioner that as a result is (implicitly) applied to the shifted system

$$(\sigma I - A)\hat{x} = b.$$

In order to determine $\hat{p}_N(A)$ given $p_N(A)$, we write (cf. (3.3))

$$(\sigma I - A)\hat{p}_N(A) = \hat{\eta} I - A p_N(A). \tag{3.5}$$

We need to find the parameters $\hat{\gamma}_i,\ i = 0, \ldots, N$, and $\hat{\eta}$ given σ, as well as $\gamma_i,\ i = 0, \ldots, N$. Substituting the sum (3.4) and $\hat{p}_N(A) = \sum_{i=0}^{N} \hat{\gamma}_i A^i$ into (3.5) gives

$$\sum_{i=0}^{N} \sigma\hat{\gamma}_i A^i - \sum_{i=0}^{N} \hat{\gamma}_i A^{i+1} - \hat{\eta} I + \sum_{i=0}^{N} \gamma_i A^{i+1} = 0.$$

Shifting the second and last sums gives

$$\sum_{i=0}^{N} \sigma\hat{\gamma}_i A^i - \sum_{i=1}^{N+1} \hat{\gamma}_{i-1} A^i - \hat{\eta} I + \sum_{i=1}^{N+1} \gamma_{i-1} A^i = 0.$$

Taking out the terms for $i = 0$ and $i = N + 1$ and combining the other terms yields

$$\sigma\hat{\gamma}_0 I - \hat{\gamma}_N A^{N+1} - \hat{\eta} I + \gamma_N A^{N+1} + \sum_{i=1}^{N} (\sigma\hat{\gamma}_i - \hat{\gamma}_{i-1} + \gamma_{i-1}) A^i = 0.$$

Equating the coefficients for like powers produces

$$\begin{aligned} \hat{\gamma}_N &= \gamma_N, \\ \hat{\gamma}_{i-1} &= \gamma_{i-1} + \sigma\hat{\gamma}_i, \qquad i = N, N-1, \ldots, 1, \\ \hat{\eta} &= \sigma\hat{\gamma}_0. \end{aligned} \tag{3.6}$$

These difference equations can be solved explicitly, obtaining

$$\hat{\gamma}_i = \sum_{j=i}^{N} \gamma_j \sigma^{j-i}, \qquad \hat{\eta} = \sigma \sum_{j=0}^{N} \gamma_j \sigma^j,$$

i.e., we have $\hat{\eta}$ and the coefficients for the polynomial $\hat{p}_N(A)$ in closed form. In practice, we use the recursive formulas (3.6) to compute $\hat{\eta}$ and $\hat{\gamma}_i$, rather than the formulas in closed form. We use the same formulas both for real and imaginary shifts. Note that the recursive formulas correspond to Horner's rule for evaluating polynomials. A good reference that discusses the numerical stability of Horner's rule is [21].

The above procedure explains how to compute the coefficients of $\hat{p}_N(A)$ from a suitably chosen preconditioning polynomial $p_N(A)$ for A. This preconditioning polynomial should be such that

$$A p_N(A) \approx I,$$

or such that the residual polynomial

$$q_{N+1}(A) := I - A p_N(A) \approx O.$$

This is usually accomplished by constructing a polynomial preconditioner such that the residual polynomial $q_{N+1}(t)$ is small in the area in the complex plane that contains the spectrum of the preconditioned matrix. The coefficients of the residual polynomial thus determine the coefficients of the polynomial preconditioner. Let $q_{N+1}(t)$ be given as

$$q_{N+1}(t) = \prod_{i=1}^{N+1} (1 - \omega_i t),$$

in which the parameters ω_i are the reciprocals of the roots of $q_{N+1}(t)$. Furthermore, for residual polynomials the condition $q(0) = 1$ is satisfied. Then, in order to find the coefficients γ_i of the polynomial $p(t) = \sum_{i=0}^{N} \gamma_i t^i$, we first express $q_{N+1}(t)$ as

$$q_{N+1}(t) = 1 - \sum_{i=1}^{N+1} \gamma_{i-1} t^i.$$

The parameters γ_i can then be found from the recursion

$$q_j(t) = q_{j-1}(t) - \omega_j t q_{j-1}(t), \quad j = 1 : N+1, \quad q_0(t) = 1.$$

Many polynomial preconditioners have been proposed in the literature, e.g., [3, 24, 28, 32, 33], and any of these can be chosen as preconditioner for A. A natural choice is to take the ω_j's equal to the reciprocals of the Chebyshev nodes on the interval $[\ell, \upsilon]$, i.e.,

$$\psi_j = (2j-1)/(2(N+1)), \quad \omega_j = \frac{2}{\upsilon + \ell - (\upsilon - \ell)\cos(\pi\psi_j)}, \quad j = 1, \ldots, N+1.$$

If the eigenvalues are real, ℓ and υ are lower and upper bounds, respectively, on the spectrum. If the eigenvalues are complex, ℓ and υ should be chosen equal to the foci of the ellipse that encloses the spectrum.

Unfortunately, many realistic problems with all the eigenvalues in the left half plane, have a strongly complex spectrum for which the resulting ellipse may extend into the right half plane. Similarly, some non-normal matrices have their field of values extending into the right half plane. When ellipses do extend into the right half plane, an SI preconditioner might be needed. Since the residual polynomial has value 1 at the origin, it can not be small inside the ellipse if the ellipse encloses the origin. As a result, the Chebyshev preconditioner will perform poorly. To overcome this problem, we construct for the case where the ellipse encloses the origin a polynomial approximation for the inverse of the shifted matrix $\mu I - A$, hence we compute an approximate SI preconditioner. As a heuristic, we choose the shift equal to half the length of the minor axis, which guarantees that the eigenvalues of $\mu I - A$ are in the right half plane, and that the origin is outside the ellipse.

In the SI preconditioner it is not possible to choose $\mu = 0$ and satisfy the requirement that the left- and right-hand sides of (3.3) share a common factor $I - \sigma A$. Chosing $\mu \neq 0$ yields (apart from a scaling factor) the same preconditioner for all shifts and has as disadvantage that the quality of the preconditioner decreases when μ is chosen further away from σ. The situation is different, however, for a polynomial preconditioner $p_N(A)$. In this case the choice $\mu = 0$ is allowed. The resulting preconditioners $\hat{p}_N(A) = \hat{M}^{-1}$ are different for every shift σ and aim to approximate (apart from a scaling factor) the inverse of $\sigma I - A$. This is possible because for polynomials one can always find a shift $\hat{\eta}$ such that a common factor can be devided out of equation (3.3).

4. Numerical experiments. In this section we present numerical results for four different test problems. The first two test problems, taken from the SLICOT test set [8], are used to demonstrate how our approach can be applied to models with a system matrix with strongly complex eigenvalues. The third and fourth problem are large-scale models for flow in a cylindrical reservoir. These two problems illustrate the performance gain that can be obtained using our approach.

The dimension m of the reduced models is selected to give a good agreement between the frequency response of the full and the reduced model. For all examples we give the magnitude of the frequency responses in the form of Bode plots (norm of the transfer function for different frequency values).

IRKA is always initialized using the harmonic Ritz values after m Arnoldi iterations with a constant starting vector. The IRKA iterations are terminated once the maximum difference in the shifts between two consecutive iterations is smaller than 10^{-5}.

The ellipses around the spectrum are constructed using Khachiyan's method [25] such that they enclose the m harmonic Ritz values that are computed to initialize the IRKA algorithm. We remark that the computational cost of this algorithm is negligible in comparison with the rest of the IRKA computational effort.

To solve the set of linear systems we have used the following techniques:

1. Direct solution (using MATLAB's "\" command): every system is solved individually.
2. BiCG (Algorithm 1.2), without preconditioner, and with polynomial preconditioners of degrees 4, 8, and 16.
3. Shifted BiCG (MS-BiCG, Algorithm 1.3), without preconditioner, and with polynomial preconditioners of degrees 4, 8, and 16.
4. Two-part MS-BiCG (Algorithm 2.4), without preconditioner, and with a polynomial preconditioners of degrees 4, 8, and 16.

For all methods we report the number of IRKA iterations, the total number of matrix-vector multiplications (MATVECs) and the computing time for the complete IRKA calculation. The number of IRKA iterations should be the same for all tabulated methods, if all linear systems are solved to sufficient accuracy; cf. [6]. We also report the dimensions of the Krylov subspaces that need to be stored for two-part MS-BiCG.

The iterative linear equation solvers are terminated for a specific shifted linear system once the corresponding residuals satisfy $\|\hat{r}_{x_i}\| \leq 10^{-8}\|b\|$ and $\|\hat{r}_{z_i}\| \leq 10^{-8}\|c\|$. Note that the norms of the residuals of the shifted systems can be obtained almost for free using (1.14). The maximum number of iterations is set to 10,000.

For every test problem we have checked that the first five decimal digits of the computed shifts are the same for all techniques. This to ensure that possible differences in the response function are caused by the termination criterion of IRKA, not by the numerical differences in the solutions of the linear solvers. We remark that all converged solutions visually yield the same Bode plots.

All computations that are described in this section have been performed using MATLAB 7.13 on a workstation with 32 GB of memory and equipped with an eight-core Xeon processor.

4.1. The ISS problem. The first example we consider is the ISS problem described in [8]. It models the International Space Station. The system matrix A has dimension 270. For the dimension of the reduced model we take $m = 20$.

The Bode plots for the full and for the reduced model are shown in the left panel of Figure 1. The responses of the two models are almost identical. The bounding

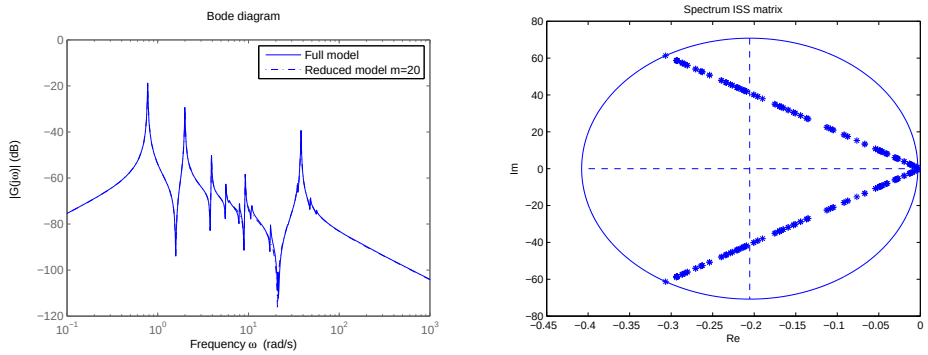

FIG. 1. *ISS problem.*

TABLE 1

Results for the ISS problem. N is the degree of the polynomial preconditioner. Numbers in parentheses are the maximum dimension of the Krylov subspace.

Method, preconditioner	IRKA its.	MATVECs	CPU time [s]
Direct solution	16	-	0.3
BiCG, none	16	83377	12.8
BiCG, $N = 4$	16	314100	22.0
BiCG, $N = 8$	16	480694	27.0
BiCG, $N = 16$	16	821290	41.4
MS-BiCG, none	16	6695	4.4
MS-BiCG, $N = 4$	16	25730	4.8
MS-BiCG, $N = 8$	16	47164	5.2
MS-BiCG, $N = 16$	16	82300	6.3
Two-part MS-BiCG, none	16	270 (270)	2.8
Two-part MS-BiCG, $N = 4$	16	2350 (214)	2.0
Two-part MS-BiCG, $N = 8$	16	4432 (208)	1.9
Two-part MS-BiCG, $N = 16$	16	8384 (192)	1.9

ellipse does not enclose the origin and therefore, it is not necessary to apply a shift to the preconditioner.

In Table 1 we present the results for the different methods. Here, and in the other tables, the numbers in parentheses indicate the dimensions of the Krylov subspaces. Note that the number of MATVECs (matrix-vector multiplications) corresponds to the MATVECS for both the (MS-)BiCG iterations and to the MATVECS that are needed to compute the solutions, i.e., for the multiplications with the polynomials $\hat{p}_N(A)$. These latter computations require a total of $N \times m$ MATVECs per IRKA iteration. This is only a fraction of the total number of MATVECs for BiCG and MS-BiCG. In contrast, almost all the MATVECS for two-part MS-BiCG with polynomial preconditioning are needed for applying the polynomials $\hat{p}_N(A)$.

It can be appreciated that the number of IRKA iterations is the same for all methods. This test problem is too small for an iterative solver to be competitive with the direct solution method. The shifted BiCG algorithm MS-BiCG is about three times faster than standard BiCG. Note that the number of shifts for this problem is 20, which is relatively large. Two-part MS-BiCG gives another improvement of a factor 2. However, two-part MS-BiCG without polynomial preconditioner needs to store 270 basis vectors for the Krylov subspace, which is equal to the problem size. The polynomial preconditioner performs poorly with BiCG and with MS-BiCG. This is indicated by the computing times which go up if the degree of the polynomial

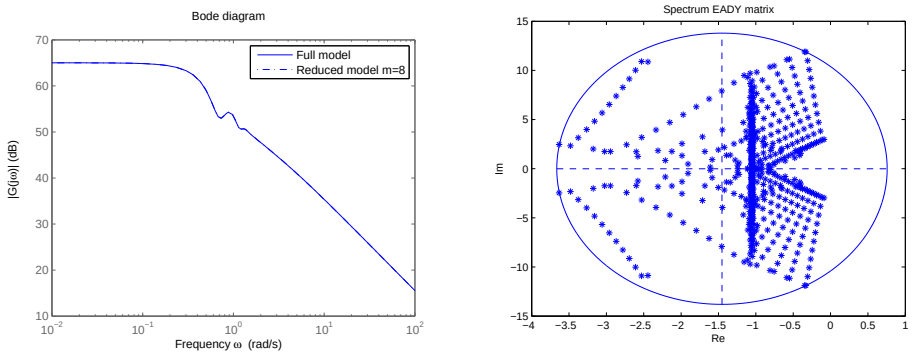

FIG. 2. *Eady problem.*

TABLE 2

Results for the Eady problem. N is the degree of the polynomial preconditioner. Numbers in parentheses are the maximum dimension of the Krylov subspace.

Method, preconditioner	IRKA its.	MATVECs	CPU time [s]
Direct solution	50	-	18.3
BiCG, none	50	141122	64.8
BiCG, $N = 4$	50	235915	65.7
BiCG, $N = 8$	50	373856	63.8
BiCG, $N = 16$	50	621749	147.1
MS-BiCG, none	50	26562	15.6
MS-BiCG, $N = 4$	46	55222	12.2
MS-BiCG, $N = 8$	52	99133	18.7
MS-BiCG, $N = 16$	50	156816	26.9
Two-part MS-BiCG, none	51	578 (578)	7.1
Two-part MS-BiCG, $N = 4$	48	2371 (167)	2.4
Two-part MS-BiCG, $N = 8$	50	4550 (150)	2.7
Two-part MS-BiCG, $N = 16$	50	8678 (134)	3.5

preconditioner is increased. Looking at the numbers of MATVECS, we observe that these almost double if N is doubled, which indicates that the polynomial preconditioner only slightly reduces the number of (MS-)BiCG iterations. This is also apparent from the dimensions of the Krylov subspaces (which equals the number of initial MS-BiCG iterations) whose bases are stored for two-part MS-BiCG. The reason for the poor performance of the polynomial preconditioners is the unfavorable spectrum, which results in an ellipse that is elongated along the imaginary axis and that almost touches the origin. We note, however, that although overall the polynomial preconditioner increases the number of total MATVECS as the degree grows, it does reduce both the storage requirements and the computing time for two-part BiCG by about 30%.

4.2. The EADY problem. The second example we consider is the EADY problem [8], which is a model of the atmospheric storm track. The system matrix A has dimension 598. For the dimension of the reduced model we take $m = 8$.

The left panel of Figure 2 shows the Bode plots for the full and for the reduced model. The right panel shows the spectrum and the bounding ellipse. Although all eigenvalues have a negative real part, the ellipse extends into the right half plane, which makes it necessary to use a shift in the preconditioner.

Table 2 gives the results for the different methods. The number of IRKA iterations, due to small numerical differences in the solutions of the linear systems, varies

slightly for different methods. Since this problem is small, BiCG is not competitive with the direct solver. Simultaneously solving the shifted systems in one IRKA iteration using MS-BiCG gives a method that is slightly faster than the direct method. The use of a low degree polynomial preconditioner gives a small speed-up over plain MS-BiCG. The two-part MS-BiCG method is, for this problem, the fastest: this method in combination with a polynomial preconditioner of degree 4 is almost eight times faster than the direct solver.

For this problem the polynomial preconditioner approximates $(\mu I - A)^{-1}$, with $\mu \neq 0$. This means that the numerical performance (numbers of BiCG iterations) of the polynomial preconditioner cannot be expected to be better than for the exact SI preconditioner. To validate this we have also solved this problem with the SI preconditioner, which results for example in 133 initial MS-BiCG iterations for two-part BiCG, whereas two-part MS-BiCG with a polynomial preconditioner of degree 16 needs 134 initial iterations.

4.3. Convection-diffusion problem. The third test problem models transport in a moving medium. The domain is radial and the medium moves with radial velocity $v = \frac{a}{r}$ towards the well in the center. In this equation r is the distance to the center and a is a constant. We assume that also the diffusion is constant in all directions. A simple model for the concentration C is then given by the following convection-diffusion equation:

(4.1)
$$\frac{\partial}{\partial r}\left(r\frac{\partial C}{\partial r}\right) - a\frac{\partial C}{\partial r} + \frac{1}{r^2}\frac{\partial^2 C}{\partial \theta^2} + \frac{\partial^2 C}{\partial z^2} = \frac{\partial C}{\partial t}, \quad r \in (1, R),\ \theta \in (-\pi, \pi],\ z \in (0, D),\ t > 0.$$

Here, R is the radius of the reservoir, θ is the radial coordinate, z the vertical coordinate, D the depth of the reservoir, and t is the time. The domain has radius $R = 200$ and depth $D = 10$. For the convection parameter we take $a = 4$.

The initial condition is $C(0, r, \theta, z) = 0$. A no-flux condition is imposed at the top and the bottom of the domain:

$$\frac{\partial C(t, r, \theta, 0)}{\partial z} = 0, \quad \frac{\partial C(t, r, \theta, D)}{\partial z} = 0.$$

Furthermore, the periodic boundary condition $C(t, r, 0, z) = C(t, r, 2\pi, z)$ is imposed. We assume that the flux at the well is prescribed and that the concentration at the outer boundary is constant. The corresponding boundary conditions are

$$\frac{\partial C(t, 1, \theta, z)}{\partial r} = -f_{well} \quad \text{and} \quad C(t, R, \theta, z) = 0.$$

The output variable is the concentration in the well C_{well} and the control variable is the flux f_{well}. Discretization with the finite difference method, using an equidistant grid with 500 points in the radial direction, 25 points in the z direction, and 36 points in the angular direction yields a dynamical system of the following form:

$$\dot{C} = AC + bf_{well}, \quad C_{well} = c^T C.$$

The gridsizes are chosen such that the numerical solution is stable, i.e., it does not exhibit spurious oscillations. The number of state variables is 450,000. The matrix A corresponds to a convection-diffusion operator and is therefore nonsymmetric. For the stable grid sizes the eigenvalues of the matrix A are (almost) real.

The dimension of the reduced model m is taken to be 6. Note that due to symmetry the 1D version of the model that takes only the radial direction into account

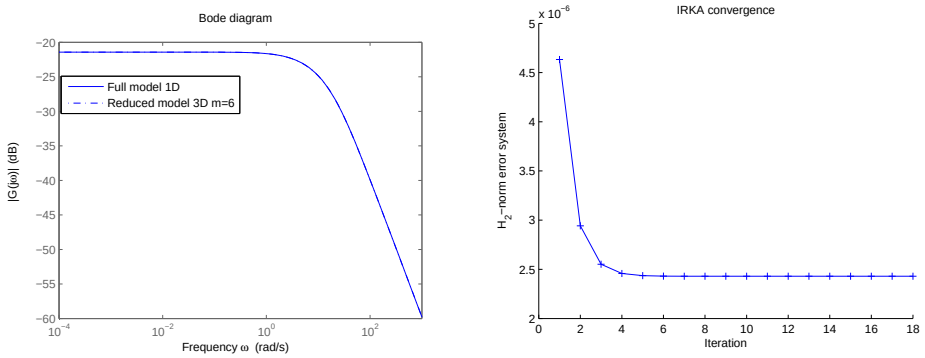

FIG. 3. *Convection-diffusion problem.*

TABLE 3

Results for the convection-diffusion problem. N is the degree of the polynomial preconditioner. Numbers in parentheses are the maximum dimension of the Krylov subspace.

Method	IRKA its.	MATVECs	CPU time [s]
Direct solution	18	-	30645
BiCG, $N = 0$	n.c.	-	-
BiCG, $N = 4$	18	11867	767
BiCG, $N = 8$	18	12798	698
BiCG, $N = 16$	18	14342	596
MS-BiCG, $N = 0$	18	7884	1141
MS-BiCG, $N = 4$	18	5287	398
MS-BiCG, $N = 8$	18	5913	369
MS-BiCG, $N = 16$	18	6896	259
Two-part MS-BiCG, $N = 0$	n.c.	-	-
Two-part MS-BiCG, $N = 4$	18	2797 (473)	229
Two-part MS-BiCG, $N = 8$	18	2484 (180)	139
Two-part MS-BiCG, $N = 16$	18	3445 (101)	129

gives the same response. We exploit this by comparing the Bode plot of the full 1D model with the Bode plot of the reduced 3D model. These are shown in the left panel of Figure 3. The two plots coincide. The right panel shows the $\mathcal{H}_2$-norm of the error system; this is the response of the full 1D model minus the response of the reduced 3D model, as function of the IRKA iteration. The convergence of IRKA in the $\mathcal{H}_2$-norm is monotonic, as mostly observed.

Table 3 gives the results for the different solution techniques. Here and in the sequel, "n.c." indicates no convergence within 10,000 iterations. This occurred for BiCG and two-part MS-BiCG without preconditioning. The number of IRKA iterations is the same for all other tabulated methods.

Iterative solvers are more suited for this large 3D problem than the direct method. The results for BiCG with polynomial preconditioner illustrate this. BiCG with a polynomial preconditioner of degree 16 is about 50 times faster than the direct method. MS-BiCG and two-part MS-BiCG each give a further performance gain of a factor 2. As a result, two-part MS-BiCG with a polynomial preconditioner of degree 16 yields a speed-up of more than a factor of 200 with respect to IRKA with the direct solution method. The polynomial preconditioner is quite effective for this problem. It speeds up MS-BiCG considerably, and it turns BiCG and two-part MS-BiCG from nonconverging into a rapidly converging processes.

The performance gain for the two-part MS-BiCG algorithm comes at the expense of a higher memory consumption than for the other two BiCG variants. Since the bases for both $\mathcal{K}_i(A, b)$ and $\mathcal{K}_i(A^T, c)$ need to be stored, additional memory of $2i$ vectors of the size of the number of state variables is needed. Note that taking the degree of the polynomial preconditioner higher not only reduces the computing time but, more importantly, also reduces the storage requirements. Since the bandwidth of the matrix is 900 (the number of points in angular direction times the number of points in z direction), the storage required for the LU factors of a direct solution method is approximately equivalent to the storage required for 1800 vectors. This is much more than the storage for 202 vectors that is needed to store the basis vectors for the Krylov subspaces for two-part MS-BiCG with $N = 16$.

The example that we have considered in this section could easily be replaced by an equivalent 1D problem. The reason that we have consider the 3D problem, and not simply the equivalent 1D problem, is to illustrate the computational advantage of our approach for large-scale problems. We mention that we have also applied our approach to truly 3D problems, i.e., problems that cannot be replaced by an equivalent 1D problem, with very similar results. However, for these problems we cannot report on the frequency response of the full model, or on the $\mathcal{H}_2$-norm of the error system, since these calculations are too time consuming for large 3D models.

4.4. Potential flow in a cylindrical reservoir. The last example we consider describes flow in a porous medium in a cylindrical reservoir around a well. A mathematical description for this problem is given by the so-called (dimensionless) radial diffusivity equation,

$$\frac{\partial}{\partial r}\left(r\frac{\partial p}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 p}{\partial \theta^2} + \frac{\partial^2 p}{\partial z^2} = \frac{\partial p}{\partial t}, \quad r \in (1, R),\ \theta \in (-\pi, \pi],\ z \in (0, D),\ t > 0. \tag{4.2}$$

In this equation p is the pressure in the reservoir, r is the distance to the center of the well, R is the radius of the reservoir, θ is the radial coordinate, z the vertical coordinate, D the depth of the reservoir, and t is the time. Note that the spatial operator in (4.2) is the Laplace operator in cylindrical coordinates. As in the previous example, we take the radius $R = 200$ and the depth $D = 10$. The initial condition is $p(0, r, \theta, z) = 0$. A no-flow boundary condition is imposed on the top and on the bottom of the reservoir, i.e.,

$$\frac{\partial p(t, r, \theta, 0)}{\partial z} = 0 \ \text{ and } \ \frac{\partial p(t, r, \theta, D)}{\partial z} = 0.$$

Because of the cylindrical domain the periodic boundary condition $p(t, r, 0, z) = p(t, r, 2\pi, z)$ holds. Furthermore, we assume that the pressure in the well is prescribed and that there is no inflow through the outer boundary. In dimensionless variables these boundary conditions read

$$p(t, 1) = p_{well} \ \text{ and } \ \frac{\partial p(t, R, \theta, z)}{\partial r} = 0, \quad t > 0. \tag{4.3}$$

The variable of interest is the outflow from the reservoir into the well (which determines the production rate) given by

$$v_{well} = -\frac{\partial p(t, 1, \theta, z)}{\partial r}.$$

The control variable is p_{well}.

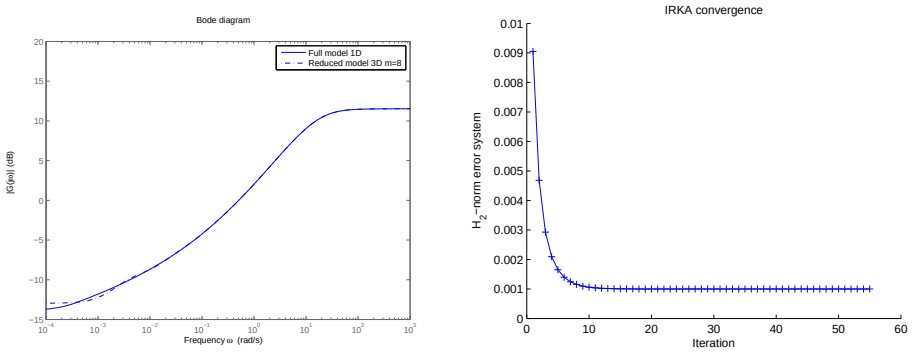

FIG. 4. *Diffusivity equation.*

TABLE 4

Results for the diffusivity equation. N is the degree of the polynomial preconditioner. Numbers in parentheses are the maximum dimension of the Krylov subspace.

Method	IRKA its.	MATVECs	CPU time [s]
Direct solution	55	-	117417
BiCG, $N = 0$	n.c.	-	-
BiCG, $N = 4$	55	122770	8859
BiCG, $N = 8$	55	126433	7221
BiCG, $N = 16$	55	137209	5354
MS-BiCG, $N = 0$	n.c.	-	-
MS-BiCG, $N = 4$	55	63725	4593
MS-BiCG, $N = 8$	55	65179	3720
MS-BiCG, $N = 16$	55	72677	2866
Two-part MS-BiCG, $N = 0$	n.c.	-	-
Two-part MS-BiCG, $N = 4$	55	5720 (792)	751
Two-part MS-BiCG, $N = 8$	55	4843 (147)	403
Two-part MS-BiCG, $N = 16$	55	3586 (83)	359

This problem is discretized with the finite difference method, using the same grid as in the previous example. The number of state variables is also 450,000.

The outflow into the well is calculated using a second-order finite-difference formula. This leads to a dynamical system of the following form:

$$\dot{p} = Ap + bp_{well}, \quad v_{well} = c^T p + dp_{well}.$$

The dimension of the reduced model m is taken to be 8. The matrix A has real eigenvalues, since the diffusivity equation is a Laplace equation and hence A is a discretized Laplacian. However, due to the Neumann boundary conditions and to the cylindrical coordinates, the matrix is nonsymmetric.

Due to symmetry, a 1D model with the same response can be formulated by ignoring the angular and depth dependencies. Figure 4 shows in the left panel the Bode plots of the reduced system and of the 1D model. The right panel shows the $\mathcal{H}_2$-norm of the error system as function of the IRKA iteration number. The decrease of the $\mathcal{H}_2$-norm is monotonic.

Table 4 shows the numerical results. None of the BiCG variants without preconditioning converge. The direct solver is not competitive with any of the preconditioned iterative methods. BiCG with a polynomial preconditioner of degree 16 is more than

20 times faster than the direct method. MS-BiCG gives a further performance gain of a factor 2, and two-part MS-BiCG a further improvement of a factor 8. Two-part MS-BiCG gives a much bigger performance gain for this problem than for the previous problem. This can be explained by the fact that more IRKA iterations are needed for this problem and that m is higher. As a result, the number of shifted systems that need to be solved is about four times higher, and therefore also the gain of reusing the bases for the Krylov subspaces.

The polynomial preconditioner is quite effective for this problem as well. It turns the unpreconditioned BiCG methods into a converging processes. The best performance is achieved for the polynomial preconditioners of the highest degree we considered.

5. Concluding remarks. The computationally most demanding part of IRKA is the solution of sets of shifted systems. In this paper we have proposed variants of IRKA that use preconditioned versions of the shifted BiCG algorithm. In the proposed version, the bases of the two linear systems for a fixed shift are computed, stored, and reused for the systems with the other shifts. We have presented a polynomial preconditioner that can simultaneously be applied to all shifted systems. By means of numerical experiments we have investigated the efficiency of the resulting algorithms.

We have considered two types of test problems. Our first two examples concerned rather small size problems of which the system matrix has strongly complex eigenvalues. For these problems our results indicate that MS-BiCG and two-part MS-BiCG give a significant reduction of computing time relative to BiCG. The performance gain of the polynomial preconditioner turned out to be limited in combination with BiCG and MS-BiCG, but in combination with two-part MS-BiCG the polynomial preconditioner gave a significant performance improvement for both test problems.

The other two examples concerned large-scale problems of which the system matrix has only (almost) real eigenvalues. For such a matrix it is relatively easy to compute a suitable polynomial preconditioner. Our results showed the excellent performance of the preconditioned shifted BiCG methods for these problems.

In general, a higher degree polynomial would be a better preconditioner, but of course at an extra cost. As can be observed in some of our experiments, there is an optimal point after which the additional cost of higher degree offsets the possible gain. In the smaller problems, this occurs at around $N = 8$, and going to $N = 16$ either does not produce any decrease in execution times, or those times increase. In the larger problems, after $N = 8$, there is still a reduction in execution times, but those reductions are not as steep. Therefore, a good heuristic that cover all cases is to choose either $N = 8$ or $N = 16$. The former choice is more conservative, while the latter is a little more aggressive, but in either case, one may very well be near the optimal value.

Acknowledgments. The authors wish to express their appreciation to the anonymous referees for the time and effort they invested in this paper. Their questions and suggestions helped improve it significantly. Thanks go also to Mark Embree, who made valuable comments on an earlier version of this paper. The third author thanks Danny Sorensen for introducing him to the area of model order reduction during a sabbatical stay at the Delft University of Technology. Part of this work was performed while the second author was visiting Delft. The warm hospitality received is greatly appreciated.

REFERENCES

[1] K. Ahuja, E. de Sturler, S. Gugercin, and E. Chang, *Recycling BiCG with an application to model reduction*, SIAM J. Sci. Comput., 34 (2012), pp. A1925–A1949.

[2] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*, SIAM, Philadelphia, 2005.

[3] S. Ashby, T. Manteuffel, and J. Otto, *A comparison of adaptive Chebyshev and least squares polynomial preconditioning for Hermitian positive definite linear systems*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 1–29.

[4] U. Baur, C. A. Beattie, P. Benner, and S. Gugercin, *Interpolatory projection methods for parameterized model reduction*, SIAM J. Sci. Comput., 33 (2011), pp. 2489–2518.

[5] C. A. Beattie and S. Gugercin, *Model reduction by rational interpolation*, in Model Reduction and Algorithms: Theory and Applications, P. Benner, A. Cohen, M. Ohlberger, and K. Willcox, eds., Comput. Sci. Engrg. 15, SIAM, Philadelphia, 2017, pp. 297–334.

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

[7] P. Benner and T. Breiten, *Interpolation-based $\mathcal{H}_2$-model reduction of bilinear control systems*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 859–885.

[8] Y. Chahlaoui and P. V. Dooren, *A collection of benchmark examples for model reduction of linear time invariant dynamical systems*, Technical report, SLICOT Working Note 2002-2, 2002. Available from https://perso.uclouvain.be/paul.vandooren/.

[9] L. Daniel, O. Siong, L. Chay, K. Lee, and J. White, *A multiparameter moment-matching model-reduction approach for generating geometrically parameterized interconnect performance models*, IEEE Trans Comput. Aid. D, 23 (2004), pp. 678–693.

[10] G. Flagg and S. Gugercin, *Multipoint Volterra series interpolation and $\mathcal{H}_2$ optimal model reduction of bilinear systems*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 549–579.

[11] R. Fletcher, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis - Dundee 1975, Lecture Notes in Mathematics 506, G. Watson, ed., Springer, Heidelberg, 1976, pp. 73–89.

[12] R. W. Freund, *On Conjugate Gradient Type Methods and Polynomial Preconditioners for a Class of Complex Non-Hermitian Matrices*, Numer. Math., 57 (1990), pp. 285–312.

[13] A. Frommer, *BiCGStab(ℓ) for families of shifted linear systems*, Computing, 70 (2003), pp. 87–109.

[14] A. Frommer and U. Glassner, *Restarted GMRES for shifted linear systems*, SIAM J. Sci. Comput., 19 (1998), pp. 15–26.

[15] E. Grimme, *Krylov projection methods for model reduction*, Ph.D. thesis, Department of Electrical Engineering, University of Illinois at Urbana-Champain, Urbana, Illinois, 1997.

[16] S. Gugercin, *Projection Methods for Model Reduction of Large-Scale Dynamical Systems*, Ph.D. thesis, Rice University, Houston, Texas, 2002.

[17] S. Gugercin and A. C. Antoulas, *An $\mathcal{H}_2$ error expression for the Lanczos procedure*, in 42nd IEEE Conference on Decision and Control, 2003.

[18] S. Gugercin, A. C. Antoulas, and C. Beattie, *$\mathcal{H}_2$ model reduction for large-scale linear dynamical systems*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 609–638.

[19] S. Gugercin, T. Stykel, and S. Wyatt, *Model reduction of descriptor systems by interpolatory projection methods*, SIAM J. Sci. Comput., 35 (2013), pp. B1010–B1033.

[20] M. Heinkenschloss, D. C. Sorensen, and K. Sun, *Balanced truncation model reduction for a class of descriptor systems with application to the oseen equations*, SIAM J. Sci. Comput., 30 (2008), pp. 1038–1063.

[21] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, Philadelphia, 2002.

[22] D. Hyland and D. Bernstein, *The optimal projection equations for model reduction and the relationships among the methods of Wilson, Skelton, and Moore*, IEEE Trans. Automat. Control, 30 (1985), pp. 1201–1211.

[23] B. Jegerlehner, *Krylov space solvers for shifted linear systems*, Report IUHET-353, Department of Physics, Indiana University, December 1996.

[24] O. Johnson, C. Micchelli, and G. Paul, *Polynomial preconditioning for conjugate gradient calculation*, SIAM J. Numer. Anal., 20 (1983), pp. 362–376.

[25] L. G. Khachiyan, *Rounding of polytopes in the real number model of computation*, Math. Oper. Res., 21 (1996), pp. 307–320.

[26] A. Ruhe, *Rational Krylov algorithms for nonsymmetric eigenvalue problems II. Matrix pairs*, Linear Algebra Appl., 197 (1994), pp. 283–295.

[27] A. Ruhe, *The rational Krylov algorithms for nonsymmetric eigenvalue problems III. Complex shifts for real matrices*, BIT Numer. Math., 34 (1994), pp. 165–176.

[28] Y. Saad, *Practical use of polynomial preconditionings for the conjugate gradient method*, SIAM J. Sci. Stat. Comput., 6 (1985), pp. 865–881.
[29] Z. Strakoš, *Model reduction using the Vorobyev moment problem*, Numer. Algorithms, 51 (2009), pp. 363–379.
[30] Z. Strakoš and P. Tichy, *On efficient numerical approximation of the bilinear form $c^*A^{-1}b$*, SIAM J. Sci. Comput., 33 (2011), pp. 565–587.
[31] T. Stykel, *Low-rank iterative methods for projected generalized Lyapunov equations*, Electron. Trans. Numer. Anal., 30 (2008), pp. 187–202.
[32] H. K. Thornquist, *Fixed-Polynomial Approximate Spectral Transformations for Preconditioning the Eigenvalue Problem*, Ph.D. thesis, Department of Applied Mathematics, Rice University, Houston, Texas, May 2006.
[33] M. B. van Gijzen, *A polynomial preconditioner for the GMRES algorithm*, J. Comput. Appl. Math., 59 (1995), pp. 91–107.
[34] C. D. Villemagne and R. Skelton, *Model reduction using a projection formulation*, Int. J. Control, 40 (1987), pp. 2141–2169.
[35] D. A. Wilson, *Optimum solution of model-reduction problem*, Proc. IEEE, 117 (1970), pp. 1161–1165.
[36] S. Wyatt, *Issues in Interpolatory Model Reduction: Inexact Solves, Second-order Systems and DAEs*, Ph.D. thesis, Department of Mathematics, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, May 2012.