RC21185 05/19/98
Computer Science/Applied Mathematics: Arnoldi and Lanczos Algorithms: Eigenvalue Computations: Linear Solvers: Large Matrices

IBM Research Report

Two-Sided Arnoldi and Nonsymmetric Lanczos Algorithms

Jane Cullum

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, New York
cullumj@watson.ibm.com

Tong Zhang

Computer Science Department
Stanford University
Stanford, CA 94305
tzhang@cs.stanford.edu

IBM Research Division
Almaden · T.J. Watson · Tokyo · Zurich

Two-Sided Arnoldi and Nonsymmetric Lanczos Algorithms

Jane Cullum* Tong Zhang†

Abstract

Many popular iterative algorithms for solving $Ax = b$, for computing eigenvalues $Ax = \lambda x$, or for model reduction of large time-invariant, linear systems are based upon Arnoldi or Lanczos recursions. In this paper we describe a two-sided Arnoldi recursion, define several iterative algorithms which are based upon this recursion, and demonstrate equivalences between these iterative algorithms and corresponding nonsymmetric Lanczos algorithms.

1 Introduction

Many popular iterative algorithms for solving $Ax = b$, for computing eigenvalues $Ax = \lambda x$, or for model reduction of large time-invariant, linear systems are based upon Arnoldi or Lanczos recursions. In this paper we describe a two-sided Arnoldi recursion, define several iterative algorithms which are based upon this recursion, and demonstrate equivalences between these iterative algorithms and corresponding nonsymmetric Lanczos algorithms.

In Section 2 we review Arnoldi and nonsymmetric Lanczos recursions and a result from [6] which demonstrates relationships between popular Arnoldi and nonsymmetric Lanczos methods for solving $Ax = b$. These relationships, however correspond to applications of two methods to different problems and only relate residual norms.

It is natural to ask whether or not one can construct Arnoldi methods with stronger ties to nonsymmetric Lanczos procedures. In this paper we demonstrate that the answer to this question is yes, if we consider two-sided Arnoldi methods. We concentrate on the theoretical properties of the new algorithms and the relationships of these algorithms to corresponding Lanczos methods in exact arithmetic. Computational aspects and implementation issues in finite precision arithmetic will be investigated in another report.

In Section 3, we introduce a two-sided block Arnoldi recursion, Algorithm(3.2). In Section(4), we use Algorithm(3.2) to define a two-sided block Arnoldi eigenvalue-eigenvector algorithm, Algorithm(4.1), which is related to the two-sided Arnoldi eigenvalue algorithm in [23]. We prove that for blocks of size 1, the eigenvalue approximations and corresponding right or left eigenvector approximations generated by Algorithm(4.1) are identical to corresponding approximations generated by a nonsymmetric Lanczos eigenvalue-eigenvector algorithm, Algorithm(4.3).

*IBM T.J. Watson Research Center, Yorktown Heights, NY 10598. e-mail: cullumj@watson.ibm.com
†Computer Science Department, Stanford University, Stanford, CA, e-mail: tzhang@cs.stanford.edu

In Section 5 we use Algorithm(3.2) to define a two-sided block Arnoldi model-reduction procedure, Algorithm(5.1), for Equations(1). We prove that Algorithm(5.1) generates Padé approximations to the transfer function of such a system. Using known results [10], we conclude that for si/so systems that Algorithm(5.1) and a corresponding nonsymmetric Lanczos model reduction procedure, Algorithm(5.2), produce identical approximations to the transfer function. Using the equivalences derived in Section 4, we demonstrate this equivalence without invoking the Padé relationship.

In Section 6, we define a two-sided Arnoldi analog, Algorithm(6.1), of the Full Orthogonal Method (FOM) method for solving $Ax = b$. Assuming that all of the quantities needed in Algorithm(6.1) and in $BiCG$ are well-defined, we prove that these two methods generate identical iterates.

In Section 7, we illustrate the equivalence obtained in Section 5 by applying the model reduction algorithms, Algorithm(5.1) and Algorithm(5.2), to a model used in studying the effects of electro-magnetic fields on the behavior of voltages and currents in electronic circuits. We use the following notation.

1.1 Notation.

$A = (a_{ij})$, $1 \leq i, j \leq n$, $n \times n$ matrix
$A^T = (a_{ji})$, transpose of A, $A^H = (\bar{a}_{ji})$, complex conjugate transpose of A
$D =$ diag $\{d_1, \ldots, d_n\}$, $n \times n$ diagonal matrix
$\lambda_j(A)$, $1 \leq j \leq n$, eigenvalues of A , $w(A) = \{\lambda_j(A), 1 \leq j \leq n\}$
$\mathcal{K}_j(A, b) \equiv span\,\{b, Ab, \ldots, A^{j-1}b\}$, jth Krylov subspace generated by A and b
v_j, jth vector in any sequence of vectors, $V_j = \{v_1, \ldots, v_j\}$
x_j^M = jth iterate generated by method M.
$\|x\| = \sqrt{\sum_{j=1}^n x_j^2}$
e_j,$\hat{e}_j$, jth coordinate vector in m, $m+1$ space, m specified in the context
I_j, $j \times j$ identity matrix
$H(s) = p \times o$ matrix function of complex variable s
s, point in complex plane; integer index in an algorithm
si/so, single input, single output system
mi/mo, multiple input, multiple output system
A, C subscripts denote respectively a matrix A , C
$B, F, Q, G, F2$ superscripts denote respectively, quantities associated with $BiCG$, FOM,QMR, $GMRES$, $FOM2$
$Q(:, m_1 : m_2)$ $(Q(m_1 : m_2, :))$, columns(rows) m_1 through m_2 of Q

2 Arnoldi and Lanczos Recursions

Much of the current research on iterative methods for solving linear equations, $Ax = b$, for computing eigenvalues and eigenvectors, $Ax = \lambda x$, and for obtaining reduced-order models of time-invariant linear systems,

$$C\dot{x} = Gx + Bu, \; y = L^T x, \tag{1}$$

focuses on two sets of Krylov subspace methods and their variants. Each set of methods is based upon recursions which, in exact arithmetic, map a specified matrix A into a family of projection matrices which are then used to construct approximations to a solution of the specified problem.

The first set of methods are based upon Arnoldi recursions. Arnoldi recursions are direct extensions of real symmetric Lanczos recursions to nonsymmetric matrices. The second set of methods are based upon nonsymmetric Lanczos recursions.

Arnoldi and Lanczos recursions generate bases for Krylov subspaces $\mathcal{K}_j(A, r)$, $1 \leq j \leq m$. Nonsymmetric Lanczos recursions also generate bases for Krylov subspaces $\mathcal{K}_j(A^T, l)$. The triplet $\{A, r, l\}$ is problem dependent.

The Arnoldi recursion is a method for generating orthonormal bases. The following implementation uses modified Gram-Schmidt orthogonalization [15]. Other implementations exist [30]. In algorithm definitions we may omit certain subscripts to simplify the expressions or include a redundant variable to make it easier for the reader to compare algorithms. For example, in Algorithm(2.1), $s = j$. We set $Q_{m+1} = Q(:, 1 : m + 1)$, $H_m = H(1 : m, 1 : m)$ and $H_m^{(e)} = H(1 : m + 1, 1 : m)$.

Algorithm 2.1 (*Arnoldi Recursion*)

Inputs: A, r.
Outputs: $[Q_{m+1}, H_m^{(e)}, R_m]$.
$Y = r/\|r\|$.
$s = 1$
Generate Q-vectors:
for $j = 1 : m$.
  $Q(:, s) = Y$.
  $P = AY$.
  Orthogonalize P with respect to all preceding Q:
  Store coefficients in Arnoldi matrix H:
  for $i = 1 : s$
    $H(i, s) = Q(:, i)^T P$.
    $P = P - Q(:, i)H(i, s)$.
  end
  Normalize new Q-vector:
  $H(s + 1, s) = \|P\|$.
  $Y = P/\|P\|$.
  $s = s + 1$.
  if $j = m$; $Q(:, m + 1) = Y, R_m = P$. *end*
end

We can write these recursions in matrix form.

$$AQ_m = Q_{m+1}H_m^{(e)} \text{ where } H_m^{(e)} = \begin{pmatrix} H_m \\ h_{m+1,m}e_m^T \end{pmatrix}. \tag{2}$$

Procedures based upon Arnoldi recursions are admired for their classical properties and theoretical stability, [24, 29, 20]. Orthonormality of the Arnoldi vectors is maintained by explicitly orthogonalizing each new Arnoldi vector with respect to all preceding Arnoldi vectors. Each Arnoldi matrix is a representation of the orthogonal projection of A onto a subspace spanned by Arnoldi vectors. Properties of these projection matrices can be directly related to properties of the original matrix.

The following implementation of a nonsymmetric Lanczos recursion uses modified Gram-Schmidt bi-orthogonalization, In Algorithm (2.2), $V_{m+1} = V(:, 1 : m+1)$, $W_{m+1} = W(:, 1 : m+1)$, and $T^{(e)}_{rm} = T_r(1 : m+1, 1 : m)$. We use a subscript r or l, to indicate, respectively, quantities generated using A or A^T.

Algorithm 2.2 (*Nonsymmetric Lanczos Recursion*)

Inputs: A, r, l
Outputs: $[V_{m+1}, W_{m+1}, T^{(e)}_{rm}, R_{rm}]$.
Initialize:
$v_0 = w_0 = 0$, $\beta_1 = 0$, $\rho_1 = \xi_1 = 1$.
$v_1 = r/\|r\|, w_1 = l/\|l\|$.
Generate Lanczos vectors:
for $j = 1 : m$
  $v_{j+1} = Av_j$.
  $w_{j+1} = A^T w_j$.
  $\alpha_j = w_j^T v_{j+1} / w_j^T v_j$.
  $p_j = v_{j+1} - \alpha_j v_j - \beta_j v_{j-1}$.
  $s_j = w_{j+1} - \alpha_j w_j - (\beta_j \rho_j / \xi_j) w_{j-1}$.
  if $j = m$; $R_{rm} = p_j$ *end*
  Normalize vectors:
  $\rho_{j+1} = \|p_j\|, \quad v_{j+1} = p_j/\rho_{j+1}$
  $\xi_{j+1} = \|s_j\|, \quad w_{j+1} = s_j/\xi_{j+1}$
  Compute other coefficient:
  $\beta_{j+1} = \xi_{j+1} w_{j+1}^T v_{j+1} / w_j^T v_j$.
  Store coefficients in Lanczos matrix T_r:
  $T_r(j, j+1) = \beta_{j+1}$
  $T_r(j, j) = \alpha_j$
  $T_r(j+1, j) = \rho_{j+1}$
end

For any A and starting vectors v_1, w_1, the coefficients in any nonsymmetric Lanczos recursions are chosen so that the two sets of Lanczos vectors $V_m = \{v_1, \ldots, v_m\}$, and $W_m = \{w_1, \ldots, w_m\}$, $m = 1, 2, \ldots$ satisfy the following conditions. For each m, V_m is a basis for the Krylov subspace $\mathcal{K}_m(A, v_1)$, W_m is a basis for the Krylov subspace $\mathcal{K}_m(A^T, w_1)$, and these bases are complex bi-orthogonal, $W_m^T V_m = D_m$ where D_m is a nonsingular diagonal matrix.

The coefficients used in the bi-orthogonalizations define tridiagonal matrices T_{rm} such that for each m, $T_{rm} \equiv W_m^T A V_m$, is a matrix representation of the oblique projection of A onto $\mathcal{K}_m(A, v_1)$ along $\mathcal{K}_m(A^T, w_1)$, [24]. In Algorithm 2.2 we scale the Lanczos vectors to have unit norms [13]. Other variants are discussed in [22, 9, 24]. In Section 4 we use a variant where the vectors are scaled so that each $w_j^T v_j = 1$. We can write the Lanczos recursions in matrix form.

$$\begin{array}{l} AV_m = V_{m+1} {T_{rm}}^{(e)} \\ A^T W_m = W_{m+1} {T_{lm}}^{(e)} \end{array} \quad \text{where } {T_{rm}}^{(e)} \equiv \begin{pmatrix} T_{rm} \\ \rho_{m+1} e_m^T \end{pmatrix}, \tag{3}$$

$T_{lm} = \Phi_m^{-1} T_{rm} \Phi_m$, $\Phi_m = \text{diag}(\phi_1, \ldots, \phi_m)$, $\phi_1 = 1$, and $\phi_m = \phi_{m-1}\rho_m/\xi_m$.

Procedures based upon nonsymmetric Lanczos recursions have been used successfully in a variety of applications. See for example, [10, 8, 18]. However, if for some j, $p_j^T s_j = 0$, then the basic recursions *breakdown* and cannot be continued. This difficulty can be circumvented using *look-ahead* methods [22, 12]. Typically, in practice, complex bi-orthogonality is lost as the recursion accumulates information about the original problem in the Lanczos matrices [2].

Assumptions: In any discussion about any Arnoldi or nonsymmetric Lanczos-based procedure we will assume that all quantities are well-defined; no breakdowns occur; the recursions do not terminate prematurely; and the arithmetic is exact.

2.1 An Arnoldi-Lanczos Equivalence

We are interested in relationships between methods which are based upon Arnoldi recursions and methods which are based upon nonsymmetric Lanczos recursions. In [6] we derived relationships between popular Arnoldi and nonsymmetric Lanczos methods for solving $Ax = b$. Related results for corresponding eigenvalue algorithms were obtained in [7].

The theorem in [6] applies to each of the pairs of methods, *FOM/BiCG* and *GMRES/QMR*. *GMRES* (Generalized Minimal Residuals) and *FOM* (Full Orthogonal Method) are Arnoldi-based methods. *QMR*(Quasi-minimal Residuals) and *BiCG*(Bi-conjugate Gradients) are nonsymmetric Lanczos-based methods. All four methods select iterates from the same Krylov subspaces but subject to different constraints. We use x_0, r_0 to denote the initial x-iterate and residual vector.

In *GMRES* the m^{th}-iterate, $x_m^G = x_0 + \delta_m^G$, is selected so that the norm of the corresponding residual vector, $r_m^G = r_0 - A\delta_m^G$, is minimal over the m^{th} Krylov subspace $\mathcal{K}_m(A, r_0)$. δ_m^G is obtained by solving the following reduced-order least squares problem [26].

$$\delta_m^G \equiv Q_m y_m^G \text{ where } y_m^G \text{ solves } \min_y \| -H_m^{(e)} y + \|r_0\| \hat{e}_1 \|. \tag{4}$$

QMR is a direct analog of *GMRES*. δ_m^Q is obtained by solving the following reduced-order least squares problem [13].

$$\delta_m^Q = V_m y_m^Q \text{ where } y_m^Q \text{ solves } \min_y \| -T_{rm}^{(e)} y + \|r_0\| \hat{e}_1 \|. \tag{5}$$

FOM and *BiCG* are Galerkin methods [24]. In *FOM* each iterate is selected so that the corresponding residual vector, r_m^F, is orthogonal to $\mathcal{K}_m(A, r_0)$. δ_m^F is obtained by solving the following reduced-order linear system [27, 24].

$$\delta_m^F = Q_m y_m^F \text{ where } H_m y_m^F = \|r_0\| e_1. \tag{6}$$

BiCG and *FOM* are similar but not direct analogs. In *BiCG* each iterate is chosen so that the m^{th} residual vector, r_m^B, is orthogonal to the m^{th} Krylov subspace $\mathcal{K}_m(A^T, \tilde{r}_0)$. Usually, $\tilde{r}_0 \equiv r_0$. δ_m^B is obtained by solving the following reduced-order linear system [31, 28, 17].

$$\delta_m^B = V_m y_m^B \text{ where } T_m y_m^B = \|r_0\| e_1. \tag{7}$$

In practice, the convergence of a Krylov iterative method is monitored using estimates of the scaled residual norms $\|r_m\|/\|r_0\|$. From the Arnoldi and nonsymmetric Lanczos matrix recursions, Equations(2) and (3),

$$\|r_m^{Q,B}\| = \|\rho_{m+1} y_m^{Q,B}(m) v_{m+1}^{Q,B}\| \text{ and } \|r_m^{G,F}\| = \|h_{m+1,j} y_m^{G,F}(m) q_{m+1}^{G,F}\|. \tag{8}$$

By construction $\|v_{m+1}^{Q,B}\| = \|q_{m+1}^{G,F}\| = 1$. Therefore, $|y_m^{G,F,Q,B}(m)|$ are estimates of the norms of residual vectors generated by $GMRES$, FOM, QMR, and $BiCG$ and can be used to monitor the convergence.

Typically, residual norm convergence is used to compare the performance of different iterative methods for solving $Ax = b$. Therefore, it is of interest to establish relationships between residual norms generated by different procedures. In [5] we derived such relationships for pairs of methods based upon the same type of recursion. Theorem 2.1 provides relationships between residual norms generated by either of the Arnoldi-Lanczos pairs, $BiCG/FOM$ or $QMR/GMRES$. The $QMR/GMRES$ relationships use the residual vectors of the reduced QMR problems. We define quasi-residual vectors,

$$z_m^{Q,B} = -T_m^{(e)} y^{Q,B} + \|r_0\|\hat{e}_1, \tag{9}$$
$$z_m^{G,F} = -H_m^{(e)} y^{G,F} + \|r_0\|\hat{e}_1. \tag{10}$$

Theorem 2.1 *[6] (Exact Arithmetic). Let A be any $n \times n$ matrix with distinct eigenvalues. Apply $BiCG/QMR$ to $Ax = b$. For $1 \leq j \leq n$,*

$$AV_j^{Q,B} = V_j^{Q,B}(T_j^B)^{(e)} \tag{11}$$

where $V_j^{Q,B}$ and $(T_j^B)^{(e)}$ denote respectively the Lanczos vectors and matrices generated using $v_1^{Q,B} = r_0/\|r_0\| = w_1^{Q,B} = \tilde{r}_0/\|\tilde{r}_0\|$. There exists a $n \times n$ matrix C with the same eigenvalues as A and a vector c such that an application of the $FOM/GMRES$ methods to $Cx = c$ yields

$$\|r_{jC}^F\| = \|z_{jC}^F\| = \|z_{jA}^B\| = \|r_{jA}^B\|, \qquad 1 \leq j \leq n. \tag{12}$$

where z_{jA}^B and z_{jC}^F are quasi-residual vectors generated by $BiCG$ on $Ax = b$ and by FOM on $Cy = c$. Furthermore,

$$\|r_{jC}^G\| = \|z_{jC}^G\| = \|z_{jA}^Q\| \tag{13}$$

where z_{jA}^Q and z_{jC}^G are the quasi-residual vectors generated by QMR on $Ax = b$ and by $GMRES$ on $Cy = c$.

Theorem 2.1 states that any type of residual norm behavior observed using the nonsymmetric Lanczos method $BiCG$ can also be observed using the Arnoldi method FOM on a different problem with the same eigenvalues. The relationships in Theorem 2.1 are interesting but not very satisfying since the norms of the errors generated by the applications of $BiCG$ and FOM are not equal.

3 A Two-Sided Arnoldi recursion

We are interested in relationships between Lanczos and Arnoldi algorithms. In this section we define a block Arnoldi recursion, Algorithm(3.1), and use it to define a two-sided block Arnoldi recursion, Algorithm(3.2). In Sections 4, 5, and 6, we use Algorithm(3.2) to construct two-sided iterative methods and then demonstrate that certain approximations generated by those methods are identical to approximations generated by corresponding nonsymmetric Lanczos methods.

3.1 A Block Arnoldi Recursion

Given a matrix A, a starting block X, and deflation tolerances, ϵ_1 and ϵ_2, we define the following block analog of Algorithm(2.1).

Algorithm 3.1 (*Block Arnoldi Recursion*)

Inputs: A, X, ϵ_1 and ϵ_2.
Outputs: $[Q_m, H_m, R_m, S]$.
Orthonormalize columns in X, deflating if necessary:
$[Y, S, d] = \mathit{deflate}(X, \epsilon_1)$.
Specify size of first Q-block:
$d = \#\mathit{cols}(Y)$.
Initialize counter on Q-vectors :
$s = \#\mathit{cols}(Y)$.
Generate Q-blocks:
for $j = 1 : m$
  Store current block in Q:
  $Q(:, s-d+1 : s) = Y$.
  Generate next block:
  $P = AY$.
  Orthogonalize P with respect to existing Q:
  Store coefficients in Arnoldi matrix H:
  for $i = 1 : s$
    $H(i, s-d+1 : s) = Q(:, i)^T P$.
    $P = P - Q(:, i)H(i, s-d+1 : s)$.
  end
  if $j < m$
    Orthonormalize Y, deflating if necessary:
    $[Y, S_j] = \mathit{deflate}(P, \epsilon_2)$.
    $d_j = \#\mathit{cols}(Y)$.
    Store normalization coefficients in H:
    $H(s+1 : s+d_j, s-d+1 : s) = S_j$.
    Update size of block:
    $d = d_j$.
    Increment counter on number of vectors in Q
    $s = s + d$.
  else

$R_m = P.$
end
end

Algorithm(3.1) incorporates deflation. Given a set of vectors P and a deflation tolerance ϵ, subroutine *deflate* determines an orthonormal, maximally-independent subset Y of P, and a matrix S such that for each column i,

$$\|P(:,i) - YS(:,i)\|_2 \leq \epsilon. \tag{14}$$

If we want to preserve Krylov vector relationships, then it is necessary to eliminate any dependent vectors.

In the discussions we use Q_m to denote the vectors generated in m steps of any Arnoldi recursion. The number of vectors contained in Q_m will depend upon the sizes of the individual q-blocks generated on each step of the recursion. Similarly, H_m is a block Hessenberg matrix with m blocks. R_m is the block residual vector. $E_1(E_m)$ denote the first d_1 (last d_m) columns of I_M where $M \equiv dim(H_m)$ and $d_1(d_m)$ is the size of the first(last) Q-block generated. By construction R_m lies in $span\{E_m\}$. In each statement of any theorem we assume that the size of the resulting Arnoldi matrix is no larger than the size of the original matrix A. We have the following theorem which is a direct result of the constructions in Algorithm(3.1).

Theorem 3.1 *(Exact Arithmetic). Given a matrix A, a starting block X, and exact deflation, $\epsilon_i = 0, i = 1, 2$, m steps of Algorithm(3.1) generate a set of Arnoldi vectors Q_m, a matrix S, a block upper-Hessenberg matrix H_m, and a residual matrix R_m such that*

$$AQ_m = Q_m H_m + R_m E_m^T \tag{15}$$
$$X = Q_m E_1 S. \tag{16}$$

Each set of vectors $Q_j, 1 \leq j \leq m$, forms an orthonormal basis for the subspace, $span\{X, AX, \ldots, A^{j-1}X\}$, and for each $0 \leq k < m-1$, $E_m^T H_m^k E_1 = 0$.

The following theorem demonstrates that the deflation causes only a small perturbation in the recursions.

Theorem 3.2 *(Exact Arithmetic). After m steps of Algorithm(3.1), the outputs Q_m, H_m, R_m, S satisfy the following equalities:*

$$\begin{aligned} &AQ_m = Q_m H_m + R_m E_m^T + \Delta_2, \\ &X = Q_m E_1 S + \Delta_1, \end{aligned} \tag{17}$$

where $\|.\|_2$ of each column of Δ_i is at most ϵ_i $(i = 1, 2)$. For $0 \leq k < m-1$.

$$E_m^T H_m^k E_1 = 0. \tag{18}$$

Proof. The relationship for X is immediate. Equation(17) is valid for $m = k = 0, 1$. Assume it is true for some $1 \leq k-1 < m$,

$$AQ_{k-1} = Q_{k-1}H_{k-1} + R_{k-1}E_{k-1}^T + \Delta_2. \tag{19}$$

Consider Q_k, H_k, and R_k.

$$\begin{aligned} Q_k &= \left[\begin{array}{cc} Q_{k-1} & q_k \end{array} \right] \\ H_k &= \left[\begin{array}{cc} H_{k-1} & h_{1:k-1,k} \\ h_{k,k-1}e_{k-1}^T & h_{k,k} \end{array} \right]. \end{aligned} \tag{20}$$

If the number of columns in any block q_j is d_j. then $h_{j,j-1}$ is a $d_j \times d_{j-1}$ matrix and $h_{j,j}$ is a $d_j \times d_j$ matrix. By definition,

$$p_k \equiv Aq_k - Q_{k-1}h_{1:k-1,k} - q_kh_{k,k} = q_{k+1}h_{k+1,k} + \delta_k \tag{21}$$

where each column of δ_k has norm less than Δ_2. Rewriting Equation(21) in terms of Equations(20), we obtain Equation(17) for $m = k$. Equation(18) is an immediate consequence of the fact that H_m is a block upper-Hessenberg matrix, with block sizes $d_1, \ldots, d_m$. For any $0 \leq j < m-1$, $span\{H_m^j E_1\}$ is contained in $span\{e_1, \ldots e_K\}$ where $K = \sum_{j=1}^{k+1} d_j$. □

Theorem 3.3 *(Exact Arithmetic). Let $\{Q_m, H_m, R_m, S\}$ be generated by applying m steps of Algorithm(3.1) to $\{A, X\}$. For all $0 \leq k < m$,*

$$A^kX = Q_mH_m^kE_1S + (A^k\Delta_1 + \sum_{\ell=0}^{k-1} A^\ell\Delta_2Q_mH_m^{k-1-\ell}E_1S). \tag{22}$$

Proof. The proof is by induction and Theorem 3.2. Clearly, Equation(22) is valid for $k = 0, 1$. Assume that it is valid for some $1 \leq k < m-1$. Using Theorem(3.2), we obtain

$$\begin{aligned} A^{k+1}Q_mE_1 &= A(Q_mH_m^kE_1 + \sum_{\ell=0}^{k-1} A^\ell\Delta_2Q_mH_m^{k-1-\ell}E_1) \\ &= (Q_mH_m + R_mE_m^T + \Delta_2)H_m^kE_1 + A\sum_{\ell=0}^{k-1} A^\ell\Delta_2Q_mH_m^{k-1-\ell}E_1 \\ &= Q_mH_m^{k+1}E_1 + R_mE_m^TH_m^kE_1 + \Delta_2H_m^kE_1 + \sum_{\ell=1}^{k} A^\ell\Delta_2Q_mH_m^{k-\ell}E_1 \\ &= Q_mH_m^{k+1}E_1 + 0 + \sum_{\ell=0}^{k} A^\ell\Delta_2Q_mH_m^{k-\ell}E_1, \end{aligned}$$

Therefore, Equation(22) is valid for $k+1 < m$. □

Corollary 3.1 *Under the assumptions of Theorem 3.3,*

$$A^mX = Q_mH_mE_1S + R_mE_m^TH_m^{m-1}E_1S + (A^m\Delta_1 + \tag{23}$$

$$\sum_{\ell=0}^{m-1} A^\ell\Delta_2Q_mH_m^{m-1-\ell}E_1S). \tag{24}$$

3.2 A Two-Sided Block Arnoldi Recursion

We combine two applications of Algorithm(3.1) with a *merge* computation to obtain a block, two-sided Arnoldi recursion. We use subscripts r, l, to denote, respectively, quantities associated with the application of Algorithm(3.1) to $\{A, X_r\}$ and $\{A^T, X_l\}$.

Algorithm 3.2 (*Two-Sided Block Arnoldi Recursion*)

Inputs: A, X_r, X_l, ϵ_1, ϵ_2.
Outputs: $[Q_r, H_r, R_r, S_r], [Q_l, H_l, R_l, S_l], \bar{H}$.
Apply Algorithm(3.1) to $\{A, X_r\}$ *for* m_r *steps:*
Apply Algorithm(3.1) to $\{A^T, X_l\}$ *for* m_l *steps:*
$\bar{Q} = Q_l^T Q_r$.
if $dim(H_l) \leq dim(H_r)$
  $E_1 \equiv [I, 0, \ldots, 0]^T$, $dim(H_r) \times \#rows(S_r)$ *matrix.*
  $\bar{H} = H_r$.
  Compute Z, $\bar{Q}Z = Q_l^T R_r$.
  Update $\bar{H}$ *by adding* Z *to the last block column of* $\bar{H}$.
else
  $E_1 \equiv [I, 0, \ldots, 0]^T$, $dim(H_l) \times \#rows(S_l)$ *matrix.*
  $\bar{H} = H_l^T$.
  Compute Z, $\bar{Q}^T Z = Q_r^T R_l$.
  Update $\bar{H}$ *by adding* Z^T *to the last block row of* $\bar{H}$.
end

Algorithm(3.2) consists of simultaneously applying Algorithm(3.1) to $\{A, X_r\}$ for m_r iterations to generate $[Q_r, H_r, R_r, S_r]$, and to $\{A^T, X_l\}$ for m_l iterations to generate $[Q_l, H_l, R_l, S_l]$. Depending upon the relative sizes of the resulting right and left Arnoldi matrices, H_r or H_l is modified. This modification or *merge* makes the corresponding modified residual block complex orthogonal to the Arnoldi vectors which correspond to the H-matrix which was not modified. This computation requires linear independence of the rows or columns of $\bar{Q}$ which is used to compute Z. We use $\bar{H}_{rm}$ when $dim(H_{rm}) \geq dim(H_{lm})$, and $\bar{H}_{lm}$ when $dim(H_{lm}) > dim(H_{rm})$. Each one-sided block recursion is well-defined for blocks of any size, and deflation can occur in either or both single-sided recursions at any point in the computations.

If $\bar{Q}$ is invertible, Lemma 3.1 demonstrates that $\bar{H}$ is an oblique projection of A on $\mathcal{K}_m(A, R)$ along $\mathcal{K}_m(A^T, L)$ [25]. The proof is straight-forward and not included.

Lemma 3.1 *For any sets of vectors* $\hat{Q}_l$ *and* $\hat{Q}_r$ *such that* $\bar{Q} \equiv \hat{Q}_l^T \hat{Q}_r$ *is invertible, define* $\hat{Z}_l \equiv \hat{Q}_l \bar{Q}^{-T}$ *and* $\hat{Z}_r \equiv \hat{Q}_r \bar{Q}^{-1}$. *Then* $\hat{Z}_l^T \hat{Q}_r = I$, $\hat{Q}_l^T \hat{Z}_r = I$, *and the matrices,*

$$\begin{aligned} \hat{H}_r &\equiv \bar{Q}^{-1} \hat{Q}_l^T A \hat{Q}_r \\ \hat{H}_l &\equiv \hat{Q}_l^T A \hat{Q}_r \bar{Q}^{-1} \end{aligned} \tag{25}$$

are oblique projections of A *onto* $span\{\hat{Q}_r\}$ *along the* $span\{\hat{Q}_l\}$.

In this paper we focus on relationships between nonsymmetric Lanczos methods and methods based upon the two-sided Arnoldi recursion. If the blocks X_r and X_l are the same size, formally we can define a block version of the nonsymmetric Lanczos recursions, Algorithm(2.2). However, there is nothing to prevent the deflation of a left(right) Lanczos block without any deflation of the corresponding right(left) Lanczos block. If this occurs, then it is not possible to make subsequent Lanczos blocks bi-orthogonal, and the recursions must be terminated. If a *block* nonsymmetric Lanczos procedure is formulated as a band procedure [1], then at each stage it is possible to construct a right and a left Lanczos vector regardless of deflation on either side but the correspondence between a band Lanczos method and a block Arnoldi method is not transparent. Therefore, the Arnoldi-Lanczos equivalences derived in this paper all assume that X_r and X_l are column vectors.

We observe also, that correctly implementing deflation and look-ahead strategies in a band Lanczos algorithm is a nontrivial exercise. The implementation of a corresponding two-sided block Arnoldi method is simpler. Right and left Arnoldi vectors are generated without breakdown so look-ahead steps are not necessary. Deflation in either the right or the left Arnoldi recursion can be handled directly since each new vector is explicitly orthogonalized with respect to all existing vectors generated by that recursion. We cannot, however, eliminate the possibility of a *breakdown*. In any two-sided Arnoldi algorithm near-breakdown would exhibit itself as a poorly-conditioned $\bar{Q} \equiv Q_{lm}^T Q_{rm}$. The primary difficulty with a two-sided Arnoldi approach is that it may require twice as much storage as a one-sided Arnoldi method.

4 Arnoldi and Lanczos Eigenvalue Algorithms

Using Algorithm(3.2) we define the following two-sided Arnoldi eigenvalue-eigenvector algorithm. In Algorithm(4.1) and in the corresponding Lanczos algorithm, Algorithm(4.3), we use $[\mu,\ Q_{rm}u,\ Q_{lm}u]$ to denote all of the eigenvalue, and the right or left eigenvector approximations computed by the procedure. At each iteration i, $m_i = \max(m_{li}, m_{ri})$. Q_{rm} (Q_{lm}) are used in the computation if $\bar{H} = \bar{H}_{rm}(\bar{H}_{lm})$.

Algorithm 4.1 (*Two-Sided Block Arnoldi Eigenvalue Algorithm*)

Inputs: A, R, L.
Outputs: $[\ \mu,\ Q_{rm}u]$ or $[\ \mu,\ Q_{lm}u]$
Initialize:
$m_{r0} = m_{l0} = 0$, $i = 1$. *Specify* $m_{r1}, m_{l1} > 1$.
for $i = 1, 2, \ldots$ *until convergence*
  if $i = 1$
    Use Algorithm (3.2) to generate $\bar{H}_{m_1}$
  else
    Use Algorithm (3.2) to enlarge $\bar{H}_{m_{i-1}}$ to $\bar{H}_{m_i}$.
  end
  if
    $\bar{H}_{m_i} = \bar{H}_{rm_i}$, *compute* $\bar{H}_{m_i} u = \mu u$
  else
    $\bar{H}_{m_i} = \bar{H}_{lm_i}$, *compute* $\bar{H}_{m_i}^T u = \mu u$
  end
  Compute error estimates for relevant μ.

if
  desired μ *are converged, compute* $Q_{rm}u$ or $Q_{lm}u$ *and exit.*
else
  $i = i + 1.$
  Specify $m_{r(i+1)} > m_{ri}$, $m_{l(i+1)} > m_{li}$.
end
end

More sophisticated Arnoldi algorithms incorporate polynomial conditioning to accelerate the convergence, [29, 20, 25]. We could define a corresponding nonsymmetric Lanczos procedure based upon Algorithm(2.2). However, expressions simplify if we use the Lanczos vector scaling which makes $T_r = T_l$, $W_m^T V_m = I_m$, and T_r complex and symmetric.

Algorithm 4.2 (*Nonsymmetric Lanczos Recursion*)

Inputs: A, r, l
Outputs: $[T_m, V_m, W_m]$
Initialize:
$v_0 = w_0 = 0$, $\beta_1 = 0$, $\beta_2 = \sqrt{r^T l}$.
$v_1 = r/\beta_2, w_1 = l/\beta_2$
Generate Lanczos vectors:
for $j = 1 : m$
  $v_{j+1} = Av_j$.
  $w_{j+1} = A^T w_j$.
  $\alpha_j = w_j^T v_{j+1}$.
  $p_j = v_{j+1} - \alpha_j v_j - \beta_j v_{j-1}$.
  $s_j = w_{j+1} - \alpha_j w_j - \beta_j w_{j-1}$.
  if $j = m$; $R_{rm} = p_j$ *end*
  Normalize vectors:
  $\beta_{j+1} = \sqrt{p_j^T s_j}, \quad v_{j+1} = p_j/\beta_{j+1}, \; w_{j+1} = s_j/\beta_{j+1}$
  Store coefficients in Lanczos matrix T:
  $T(j,j) = \alpha_j$, $T(j,j+1) = T(j+1,j) = \beta_{j+1}$
end

The matrix equations(3) become

$$\begin{aligned} AV_m &= V_{m+1} {T_m}^{(e)} \\ A^T W_m &= W_{m+1} {T_m}^{(e)} \end{aligned} \tag{26}$$

Algorithm 4.3 (*Nonsymmetric Lanczos Eigenvalue Algorithm*)

Inputs: A, r, l
Outputs: $[\ \mu,\ V_m u,\ W_m u]$
Initialize:
$m_0 = 0$, $i = 1$. *Specify* $m_1 > 1$.

for $i = 1, 2, \ldots$ *until convergence:*
  if $i = 1$
    Use Algorithm (4.2)*to generate* T_{m_1}
  else
    Use Algorithm(4.2) *to enlarge* $T_{m_{i-1}}$ to T_{m_i}.
  end
  Compute $T_{m_i} u = \mu u$.
  Compute error estimates for relevant μ.
  if
    desired μ *are converged, compute* $[V_m u_\mu,\ W_m u_\mu]$ *and exit.*
  else
    $i = i + 1$,
    Specify $m_{i+1} > m_i$.
  end
end

When L, R are column vectors l, r, and $m_{ri} = m_{li} = m_i$, Algorithm(4.3) and Algorithm(4.1) generate identical eigenvalue approximations. In addition, the right(left) eigenvector approximations are identical if $\bar{H}_m = \bar{H}_{rm}(\bar{H}_{lm})$. We use the notation defined in these algorithms. Since l, r are column vectors we could have used either $\bar{H}_{rm}$ or $\bar{H}_{lm}$ and relationships for both choices are stated in Theorem 4.1.

Theorem 4.1 *(Exact Arithmetic:) Apply Algorithm(4.1) and Algorithm(4.3) to* $\{A, r, l\}$ *with* $m_{li} = m_{ri} = m_i$. *Assume all quantities specified in both algorithms are well-defined. Then for each* $m \equiv m_i$*:*

$$\begin{aligned}
T_m &= W_m^T A V_m = V_m^T A^T W_m \\
\bar{H}_{rm} &= (Q_{lm}^T Q_{rm})^{-1} Q_{lm}^T A Q_{rm} \\
\bar{H}_{lm} &= Q_{lm}^T A Q_{rm} (Q_{lm}^T Q_{rm})^{-1} \\
T_m &= E_m^{-1} \bar{H}_{rm} E_m \ \text{where } V_m = Q_{rm} E_m \\
T_m &= F_m^{-1} \bar{H}_{lm}^T F_m \ \text{where } W_m = Q_{lm} F_m
\end{aligned} \tag{27}$$

If for some m, $\bar{H}_m = \bar{H}_{rm}$, *then for any* $T_m u = \mu u$, $\bar{H}_m y = \mu y$, *corresponding right eigenvector approximations,* $V_m u$ *and* $Q_{rm} y$, *are identical. If* $\bar{H}_m = \bar{H}_{lm}$, *then for any* $T_m u = \mu u$, $\bar{H}_m^T y = \mu y$, *corresponding left eigenvector approximations,* $W_m u$ *and* $Q_{lm} y$, *are identical.*

Proof. We present the proof for $\bar{H}_{rm}$ obtained by modifying H_{rm}. Similar arguments apply to the other case. By construction, the modified residual $\bar{R}_{rm}$ satisfies $Q_{lm}^T \bar{R}_{rm} = 0$. Therefore,

$$Q_{lm}^T A Q_{rm} = Q_{lm}^T (Q_{rm} \bar{H}_{rm} + \bar{R}_{rm} E_{m_r}^T) = Q_{lm}^T Q_{rm} \bar{H}_{rm}. \tag{28}$$

and $\bar{H}_{rm} = (Q_{lm}^T Q_{rm})^{-1} Q_{lm}^T A Q_{rm}$. Similarly, $T_m = (W_m^T V_m)^{-1} W_m^T A V_m = W_m^T A V_m$.
Since V_m and Q_{rm} are bases for the Krylov subspace, $\mathcal{K}_m(A, r)$, there exists a nonsingular matrix E_m such that $V_m = Q_{rm} E_m$. Similarly, there exists a nonsingular matrix F_m such that $W_m = Q_{lm} F_m$. Thus

$$T_m = (W_m^T V_m)^{-1} W_m^T A V_m \tag{29}$$

$$= \left(F_m^T Q_{lm}^T Q_{rm} E_m\right)^{-1} F_m^T Q_{lm}^T A Q_{rm} E_m \tag{30}$$

$$= E_m^{-1} \left(Q_{lm}^T Q_{rm}\right)^{-1} Q_{lm}^T A Q_{rm} E_m \tag{31}$$

$$= E_m^{-1} \bar{H}_{rm} E_m. \tag{32}$$

Let $T_m u = \lambda u$, and define the corresponding right eigenvector approximation, $z_\mu^L \equiv V_m u$, Clearly, $\bar{H}_m E_m u = \lambda E_m u$. Set $t = E_m u$. The corresponding right eigenvector approximation for the two-sided Arnoldi procedure, $z_\mu^A \equiv Q_{rm} t = Q_{rm} E_m u = V_m u = z_\mu^L$. $\Box$

Corollary 4.1 *Given the hypotheses in Theorem 4.1 for some $m = m_i$, compute $\bar{H}_{rm}$ and $\bar{H}_{lm}$. Then*

$$\bar{H}_{lm}(Q_{lm}^T Q_{rm}) = (Q_{lm}^T Q_{rm}) \bar{H}_{rm}. \tag{33}$$

For any $\bar{H}_{rm} t = \mu t$, $\bar{H}_{lm}^T y = \mu y$, $T_m u = \mu u$, the corresponding right and left eigenvector approximations are equal. Specifically,

$$\begin{array}{l} z_{r\mu}^L \equiv V_m u = z_{r\mu}^A \equiv Q_{rm} t. \\ z_{l\mu}^L \equiv W_m u = z_{l\mu}^A \equiv Q_{lm} y. \end{array} \tag{34}$$

Proof. From Theorem 4.1,

$$\begin{array}{l} \bar{H}_{rm} = \left(Q_{lm}^T Q_{rm}\right)^{-1} Q_{lm}^T A Q_{rm}, \\ \bar{H}_{lm} = Q_{lm}^T A Q_{rm} \left(Q_{lm}^T Q_{rm}\right)^{-1} \end{array} \tag{35}$$

Combining Equations(35), we obtain Equation(33). The relationships between the eigenvector approximations follow directly from Theorem 4.1. $\Box$

Ruhe [23] proposes a two-sided Arnoldi method for computing an eigenvalue and corresponding right and left eigenvectors. The focus is on computing a good approximation to a left eigenvector, once a good eigenvalue and right eigenvector approximation have been obtained.

Algorithm 4.4 (*Two-Sided Arnoldi Eigenvalue Algorithm [23]*)

Inputs: A, r.
Outputs: $[\ \mu,\ z_{r\mu}, z_{l\mu}]$
Apply an Arnoldi eigenvalue algorithm to $\{A, r\}$
Select a converged pair $\{\mu, z_{r\mu}\}$
Compute $H_{rm}^H u_{l\mu} = \mu^H u_{l\mu}$. *Set* $y_{l\mu} \equiv Q_{rm} u_{l\mu}$.
Apply an Arnoldi eigenvalue algorithm to $\{A^T, y_{l\mu}\}$
Compute $H_{lm} u_{l\mu} = \mu u_{l\mu}$ $z_{l\mu} = Q_{lm} u_{l\mu}$
Modify one or both right(left) residual vectors so that $Q_{rm}^T \bar{r}_{lm} = 0$, $Q_{lm}^T \bar{r}_{rm} = 0$.
Recompute μ *and eigenvector approximations,* $z_{r\mu}, z_{l\mu}$ using modified H_{rm}, H_{lm}.

Algorithm(4.1) differs from Algorithm(4.4) in several ways. In Algorithm(4.1) the second application of the Arnoldi method is independent of the first application. It does not require a specific input vector which is the output from the first application. Algorithm(4.1) is a block routine designed for computing several eigenvalues and eigenvectors simultaneously. In either method, success depends upon the behavior of the matrix used in the adjustments of the residuals. This requires that rows or columns of $\bar{Q}$, depending upon which Arnoldi recursion is used to compute $\bar{H}$, must be linearly independent.

5 Arnoldi and Lanczos Model Reduction

In this section we use Algorithm(3.2) to define a model reduction algorithm, Algorithm(5.1), for Equations(1). For si/so systems we prove that Algorithm(5.1) and Algorithm(5.2) generate identical reduced-order models.

C and G in Equations(1) are $n \times n$ where n is the order of the system. B is $n \times q$ where q is the number of input variables and L^T is $o \times n$ where o is the number of output variables of the system. We work with the system transfer function $\tilde{H}(s)$ which maps the Laplace transform of the input functions u to the transform of the output functions y, [19].

$$y(s)/u(s) = \tilde{H}(s) \equiv L^T(sC - G)^{-1}B. \tag{36}$$

If $q = o = 1$, then the system is si/so(single input, single output) and $\tilde{H}(s)$ is a rational function of s. In general, a system is mo/mi (multi-input, multi-output) and $\tilde{H}(s)$ is an $o \times q$ matrix. Each entry in $\tilde{H}(s)$ is a si/so transfer function for one of the possible input, output combinations.

Since the iterative methods which we are considering require a fixed matrix F, we work with the following modification of Equations(36). We select an expansion point, s_0, set $s = s_0 + \sigma$, and rewrite Equations(36) as

$$\tilde{H}(s) = \hat{H}(\sigma) \equiv L^T(I + \sigma F)^{-1}R, \tag{37}$$

$$F = (s_0C - G)^{-1}C, \; R = (s_0C - G)^{-1}B. \tag{38}$$

The corresponding system equations are

$$F\dot{x} = -x + Ru, \; y = L^T x, \tag{39}$$

Using Equation(38), formally we can expand $\tilde{H}(s)$ about s_0 as follows. The terms $L^T F^j R$ are called the $moments$ of $\tilde{H}(s)$ expanded about s_0.

$$\tilde{H}(s) = L^T(I + \sigma F)^{-1}R = \sum_{j=0}^{\infty} (-1)^j L^T F^j R \sigma^j. \tag{40}$$

Model reduction typically corresponds to matching moments of a reduced-order system with moments of the original system. See for example, [10, 11]. Given a large system $\{F, L, R\}$, generate a reduced system $\{\bar{F}, \bar{L}, \bar{R}\}$ such that for some $0 \leq k \leq M$,

$$\bar{L}^T \bar{F}^k \bar{R} = L^T F^k R. \tag{41}$$

Model reduction procedures which are based upon nonsymmetric Lanczos recursions produce Padé approximations to the original transfer function [16, 10]. For si/so systems, a p^{th} Padé approximant is a rational function $\tilde{H}_p$ whose numerator and denominator are, respectively, polynomials of degree $p-1$ and p for which $\tilde{H}_p(\sigma) = \tilde{H}(s_0 + \sigma) + O(\sigma^{2p})$. Freund [11] extends the notion of a Padé approximation to mi/mo systems.

Algorithm 5.1 (*Two-Sided Block Arnoldi Model Reduction*)

Inputs: F, R, L, ϵ_1, ϵ_2.
Outputs: $[\bar{H}, \bar{R}, \bar{L}]$.
Apply Algorithm(3.2) to $\{F, R, L\}$ *for* m_r, m_l *steps:*
if $dim(H_{ml}) \leq dim(H_{mr})$
  $E_1 \equiv [I, 0, \ldots, 0]^T$, $dim(\bar{H}) \times \#rows(S_r)$ *matrix.*
  $\bar{R} = E_1 S_r$.
  $\bar{L} = \bar{Q}(1 : \#rows(S_l), :)^T S_l$.
else
  $\bar{R} = \bar{Q}(:, 1 : \#rows(S_r)) S_r$.
  $E_1 \equiv [I, 0, \ldots, 0]^T$, $dim(\bar{H}) \times \#rows(S_l)$ *matrix.*
  $\bar{L} = E_1 S_l$.
end

The following theorem states that if the deflation tolerances used in Algorithm (3.2) are zero, then Algorithm (5.1) generates matrix Padé approximations to the system transfer function.

Theorem 5.1 *(Exact Arithmetic). Assume the deflation in Algorithm(3.2) is exact, Apply the two-sided block Arnoldi model reduction algorithm, Algorithm(5.1), to* $\{F, R, L\}$ *to generate a reduced-order system* $\{\bar{H}, \bar{R}, \bar{L}\}$. *Then the first* $0 \leq k \leq m_l + m_r - 1$ *block moments of the reduced system match the corresponding block moments of the original system:*

$$\bar{L}^T \bar{H}^k \bar{R} = L^T F^k R \text{ for } 0 \leq k \leq m_l + m_r - 1. \tag{42}$$

The proof of Theorem 5.1 uses the following lemmas. We use E_{1_l} and E_{1_r} to denote respectively, the matrix E_1 which appears in the left and the right applications of the Arnoldi recursion in Algorithm(3.2).

Lemma 5.1 *Assume that* $Q_r, \bar{H}_r, \bar{R}_r, Q_l, H_l, R_l$ *satisfy* $AQ_r = Q_r \bar{H}_r + \bar{R}_r E_{m_r}^T$, $A^T Q_l = Q_l H_l + R_l E_{m_l}^T$, $Q_l^T \bar{R}_r = 0$. *For any* Y_l *such that* $E_{m_l}^T Y_l = 0$,

$$Y_l^T H_l^T Q_l^T Q_r = Y_l^T Q_l^T Q_r \bar{H}_r. \tag{43}$$

If instead $A^T Q_l = Q_l \bar{H}_l^T + \bar{R}_l E_{m_l}^T$, $AQ_r = Q_r H_r + R_r E_{m_r}^T$, $Q_r^T \bar{R}_l = 0$, *then for any* Y_r *such that* $E_{m_r}^T Y_r = 0$,

$$Y_r^T H_r^T Q_r^T Q_l = Y_r^T Q_r^T Q_l \bar{H}_l. \tag{44}$$

Proof. We give the proof for Equations(43).

$$\begin{aligned} Y_l^T H_l^T Q_l^T Q_r &= Y_l^T (Q_l H_l)^T Q_r \\ &= Y_l^T (A^T Q_l - R_l E_{m_l}^T)^T Q_r \\ &= Y_l^T (A^T Q_l)^T Q_r \\ &= Y_l^T Q_l^T (A Q_r) \\ &= Y_l^T Q_l^T Q_r H_r. \end{aligned}$$

□

Lemma 5.2 *Let the hypotheses of Lemma 5.1 corresponding to $\bar{H}_r$ be satisfied. Let X_k be arbitrary but consistent matrices and assume that*

$$\begin{aligned} A^k Q_r E_r &= Q_r \bar{H}_r^k E_r + \bar{R}_r X_k \qquad (0 \le k \le m_r) && (45)\\ (A^T)^k Q_l E_l &= Q_l H_l^k E_l \qquad (0 \le k < m_l). && (46) \end{aligned}$$

Furthermore, assume that for $0 \le k < m_l - 1$, $E_{1_l}^T (H_l^T)^k E_{m_l} = 0$. Then for $0 \le k < m_l + m_r$,

$$(Q_l E_{1_l})^T A^k (Q_r E_{1_r}) = E_{1_l}^T Q_l^T Q_r \bar{H}_r^k E_{1_r}. \tag{47}$$

Alternatively, let the hypotheses of Lemma 5.1 corresponding to $\bar{H}_l$ be satisfied. Let X_k be arbitrary but consistent matrices and assume that

$$\begin{aligned} (A^T)^k Q_l E_l &= Q_l (\bar{H}_l^T)^k E_l + \bar{R}_l X_k \qquad (0 \le k \le m_l) && (48)\\ A^k Q_r E_r &= Q_r H_r^k E_r \qquad (0 \le k < m_r). && (49) \end{aligned}$$

Furthermore, assume that for $0 \le k < m_r - 1$, $E_{r_l}^T (H_r^T)^k E_{m_r} = 0$. Then for $0 \le k < m_l + m_r$,

$$(Q_l E_{1_l})^T A^k (Q_r E_{1_r}) = E_{1_l}^T \bar{H}_l^k Q_l^T Q_r E_{1_r}. \tag{50}$$

Proof. We give the proof for Equations(47). By construction for $0 \le k < m_r$,

$$(Q_l E_{1_l})^T A^k (Q_r E_{1_r}) = (Q_l E_{1_l})^T (Q_r \bar{H}_r^k E_{1_r} + \bar{R}_r X_k) = (Q_r^T Q_l E_{1_l})^T \bar{H}_r^k E_{1_r}. \tag{51}$$

For $m_r \le k < m_l + m_r$, let $k = m_r + k_l$. Clearly, $0 \le k_l < m_l$ and

$$\begin{aligned} (Q_l E_{1_l})^T A^k (Q_r E_{1_r}) &= ((A^T)^{k_l} Q_l E_{1_l})^T (A^{m_r} Q_r E_{1_r})\\ &= (Q_l H_l^{k_l} E_{1_l})^T (Q_r \bar{H}_r^{m_r} E_{1_r} + R_r X_{m_r})\\ &= (Q_l H_l^{k_l} E_{1_l})^T (Q_r \bar{H}_r^{m_r} E_{1_r})\\ &= E_{1_l}^T (H_l^T)^{k_l} (Q_l^T Q_r) \bar{H}_r^{m_r} E_{1_r}. \end{aligned}$$

For any $p < k_l$, define $Y_l^p \equiv H_l^p E_{1_l}$. By assumption, $E_{m_l}^T Y_l^p = 0$. If we apply Lemma 5.1 recursively for $p = 1, \ldots, k_l - 1$, we obtain $E_{1_l}^T (H_l^T)^{k_l} (Q_l^T Q_r) = E_{1_l}^T (Q_l^T Q_r) \bar{H}_r^{k_l}$. Therefore,

$$\begin{aligned} (Q_l E_{1_l})^T A^k (Q_r E_{1_r}) &= E_{1_l}^T (H_l^T)^{k_l} (Q_l^T Q_r) \bar{H}_r^{m_r} E_{1_r}\\ &= E_{1_l}^T (Q_l^T Q_r) \bar{H}_r^{k_l} \bar{H}_r^{m_r} E_{1_r}\\ &= (Q_r^T Q_l E_{1_l})^T \bar{H}_r^k E_{1_r}. \end{aligned}$$

□

Proof of Theorem 5.1. We present the proof for $\bar{H}_m = \bar{H}_{rm}$. We set $m = m_r$, where m is the number of blocks generated. By construction, $\bar{R} = R_{rm} - Q_{rm} Z$, $F Q_{rm} = Q_{rm} \bar{H} + \bar{R} E_m^T$, and $Q_{lm}^T \bar{R} = 0$. Therefore, from Theorem 3.3 and Corollary 3.1,

$$\begin{aligned} A^k Q_{rm} E_{1_{rm}} = Q_{rm} \bar{H}^k E_{1_{rm}} + \bar{R} X_k \qquad (0 \le k \le m_r).\\ (A^T)^\ell Q_{lm} E_{1_{lm}} = Q_{lm} H_{lm}^\ell E_{1_{lm}}, \qquad (0 \le \ell < m_l), \end{aligned}$$

From Lemma 5.2, for $0 \leq k \leq m_l + m_r - 1$,

$$(Q_{lm}E_{1_{lm}})^T F^k (Q_{rm}E_{1_{rm}}) = (Q_{rm}^T Q_{lm} E_{1_{lm}})^T \bar{H}^k E_{1_{rm}}. \quad (52)$$

$L = Q_{lm}E_{1_{lm}}S_l$, $R = Q_{rm}E_{1_{rm}}S_r$, $\bar{L} = Q_{rm}^T Q_{lm}E_{1_{lm}}S_l$, $\bar{R} = E_{1_{rm}}S_r$ and $\bar{F} = \bar{H}$. Therefore, for $0 \leq k \leq m_l + m_r - 1$,

$$L^T F^k R = (Q_{lm}E_{1_{lm}}S_l)^T F^k (Q_{mr}E_{1_{rm}}S_r) = (Q_{rm}^T Q_{lm}E_{1_{lm}}S_l)^T \bar{H}^k E_{1_{rm}}S_r = \bar{L}^T \bar{H}^k \bar{R}. \quad (53)$$

□

For si/so systems, we can construct a corresponding nonsymmetric Lanczos model reduction procedure.

Algorithm 5.2 (*Nonsymmetric Lanczos Model Reduction*)

Inputs: F, r, l.
Outputs: $[\bar{T}, \bar{r}, \bar{l}]$.
Apply Algorithm(4.2) to $\{F, r, l\}$ *for* m *steps:*
$\bar{r} = e_1\|r\|$.
$\bar{l} = e_1\|l\|$.
$\bar{T} = T_m$.

Reduced order models generated by Algorithm(5.2) are Padé approximations to the system defined by $\{F, r, l\}$ [10]. Therefore, for si/so systems, Algorithms(5.1) and (5.2) both generate Padé approximations. Since the Padé approximation is unique [3], we can immediately state that for si/so systems, Algorithm(5.1) and Algorithm(5.2) produce identical approximations to the transfer function of the system defined by $\{F, r, l\}$. We must of course assume that all quantities in both algorithms are well-defined. We include a proof of Theorem 5.2 which does not invoke the Padé relationships.

Theorem 5.2 *(Exact Arithmetic). Let* $\{F, r, l\}$ *be a* si/so *system defined by Equations(39). Apply the two-sided Arnoldi model reduction procedure Algorithm(5.1) to* $\{F, r, l\}$ *and let* $\{\bar{H}_A, \bar{r}_A, \bar{l}_A\}$ *denote the reduced system obtained using* $m_l = m_r = m$. *Apply the nonsymmetric Lanczos model reduction procedure, Algorithm(5.2), to* $\{F, r, l\}$ *for* m *steps and let* $\{\bar{T}_L, \bar{r}_L, \bar{l}_L\}$ *denote the reduced system obtained where* $\bar{T}_L = T_m$. *Define the corresponding reduced transfer functions*

$$\tilde{H}_A(\sigma) \equiv \bar{l}_A^T (I + \sigma \bar{H}_A)^{-1} \bar{r}_A \quad (54)$$
$$\tilde{H}_L(\sigma) \equiv \bar{l}_L^T (I + \sigma \bar{H}_L)^{-1} \bar{r}_L. \quad (55)$$

Then for all complex σ,

$$\tilde{H}_A(\sigma) = \tilde{H}_L(\sigma). \quad (56)$$

Proof From Theorem 4.1,

$$\begin{array}{l} \bar{H}_A = (Q_{lm}^T Q_{rm})^{-1} Q_{lm}^T A Q_{rm} \\ V_m = Q_{rm} B_m, \; W_m = Q_{lm} C_m \\ B_m, C_m \text{ nonsingular and lower triangular} \\ T_m = B_m^{-1} \bar{H}_m B_m. \end{array} \tag{57}$$

Therefore,

$$\bar{Q} = Q_{lm}^T Q_{rm} = C_m^{-T} W_m^T V_m B_m^{-1} = C_m^{-T} B_m^{-1}. \tag{58}$$

Since B_m and C_m are triangular, $C_m^{-1} e_1 = \xi_C e_1$ and $B_m^{-1} e_1 = \xi_B e_1$. By construction,

$$\begin{array}{l} s_r = \|r\|, s_l = \|l\|, \\ v_1 = r/\sqrt{l^T r}, \; w_1 = l/\sqrt{l^T r} \\ q_{r1} = r/s_r, q_{l1} = l/s_l. \end{array} \tag{59}$$

Therefore,

$$\xi_C = \sqrt{l^T r}/s_l \text{ and } \xi_B = \sqrt{l^T r}/s_r. \tag{60}$$

Combining Equations(58) and (60) with the constructions in the algorithms,

$$\begin{array}{l} \tilde{H}_A(\sigma) \equiv \bar{l}_A^T (I + \sigma \bar{H}_A)^{-1} \bar{r}_A = \\ s_l E_{1_{lm}}^T \bar{Q} (I + \sigma B_m T_m B_m^{-1})^{-1} E_{1_{rm}} s_r = \\ s_l E_{1_{lm}}^T C_m^{-T} (I + \sigma T_m)^{-1} B_m{}^{-1} E_{1_{rm}} s_r = \\ (l^T r) e_1 (I + \sigma T_m)^{-1} e_1 = \tilde{H}_L(\sigma). \end{array} \tag{61}$$

□

6 Two-Sided Arnoldi and Lanczos for $Ax = b$

We consider a two-sided Arnoldi extension, *FOM2*, of the one-sided Arnoldi *FOM* method for solving $Ax = b$. We prove that *FOM2* is equivalent to the nonsymmetric Lanczos method *BiCG*. We use $x_m^{F2}, r_m^{F2}, e_m^{F2}$ to denote respectively, the m^{th} x-iterate, the m^{th} residual vector, $-Ax_m^{F2} + b$, and the error vector, $\epsilon_m^{F2} = x_m^{F2} - x_{true}$. In Algorithm(6.1), $r_0 = -Ax_0 + b$, x_{true} is the solution of $Ax = b$ and $\tilde{r}_0$ is an arbitrary vector.

Algorithm 6.1 (*FOM2: Two-Sided FOM*)

Inputs: $A, r_0, \tilde{r}_0$
Output: $[x_m, r_m, \epsilon_m]$
Initialize:
$q_1^r = r_0/\|r_0\|$
$q_1^l = \tilde{r}_0/\|\tilde{r}_0\|$
Specify $m_1 > 0$.
for $i = 1, \ldots$ *until convergence*
  if $i = 1$
    Use Algorithm (3.2) with $m_l = m_r = m_1$ *to generate* $\bar{H}_{m_1}$

else

Use Algorithm (3.2) with $m_l = m_r = m_i$ *to enlarge* $\bar{H}_{m_{i-1}}$ *to* $\bar{H}_{m_i}$.

end

Solve $\bar{H}_{m_i} y_{m_i} = \|r_0\| e_1$

Check error estimate, $|y_{m_i}(m)|$ *for convergence*

if

converged, compute $x_m^{F2} = x_0 + Q_{rm} y_{m_i}$, *and exit*

else

$i = i + 1$.

Specify $m_{i+1} > m_i$.

end

end

Theorem 6.1 *(Exact Arithmetic) Apply FOM2 and BiCG to* $Ax = b$, *using the same* $\tilde{r}_0$. *Assuming that all quantities required by both methods are well-defined, for each* $m = m_i$, *the two methods produce the same* x*-iterates, residual vectors and error vectors.*

Proof. We need only prove that for each $m = m_i$,

$$\delta_m^{F2} = Q_{rm} y_m^{F2} = \delta_m^B = V_{rm}^B y_m^B \tag{62}$$

By construction,

$$\bar{H}_m y_m^{F2} = B_m T_m B_m^{-1} y_m^{F2} = \|r_0\| e_1 \tag{63}$$

Therefore,

$$T_m (B_m^{-1} y_m^{F2}) = \|r_0\| B_m^{-1} e_1 \tag{64}$$

But, $\|r_0\| B_m^{-1} e_1 = \sqrt{l^T r} e_1$ so that $y_m^B = B_m^{-1} y_m^{F2}$. Substituting these relationships into Equation(62) we obtain,

$$\delta_m^{F2} = Q_{rm} y_m^{F2} = Q_{rm} B_m y_m^B = V_m y_m^B = \delta_m^B. \tag{65}$$

□

7 A Numerical Example of An Equivalence

The theorems derived in the preceding sections assume that the arithmetic is exact. In Figures 1 and 2 we illustrate Theorem 5.2, the equivalence of the transfer function approximations obtained using the model reduction algorithms, Algorithm(5.1) and Algorithm(5.2). We use a small system $\{F, r, l\}$, $n = 159$, extracted from studies of transfer functions of systems which model interconnects (wires, planes) in VLSI circuits [4].

In each figure, for each method, we plot the errors incurred using each method for $s = j\omega$, $\omega = 2\pi f$ and f the frequency in gigahertz (GHZ). In the VLSI application, the focus is on the errors along the imaginary axis. Specifically, we plot $\log_{10} |real(\tilde{H}(j\omega) - \tilde{H}_M(j\omega))|$ and $\log_{10} |imag(\tilde{H}(j\omega) - \tilde{H}_M(j\omega))|$, where $\tilde{H}_M$ denotes method M with $M = L$, the Lanczos method, and $M = A$, the two-sided Arnoldi method.

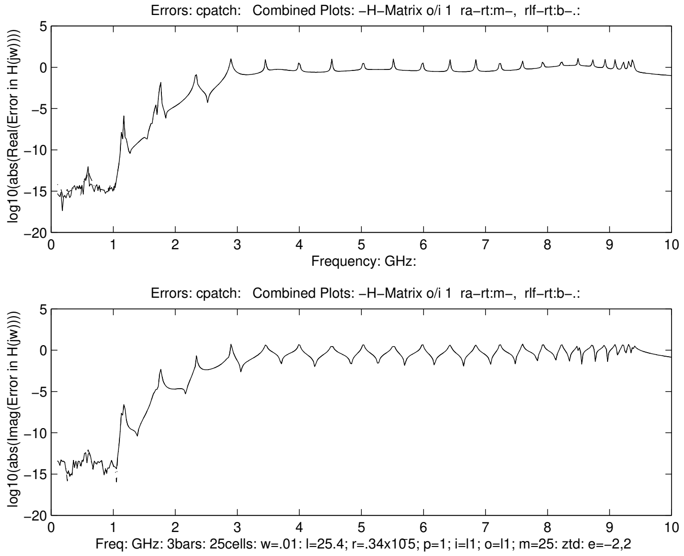

Figure 1: Demonstration of Theorem 5.2: $m = 25$. Errors, $\log_{10}|real(\tilde{H}(j\omega) - \tilde{H}_{L.A}(j\omega))|$ and $\log_{10}|imag(\tilde{H}(j\omega) - \tilde{H}_{L,A}(j\omega))|$ versus Frequency in GHZ. Nonsymmetric Lanczos (-.), Two-Sided Arnoldi (-).

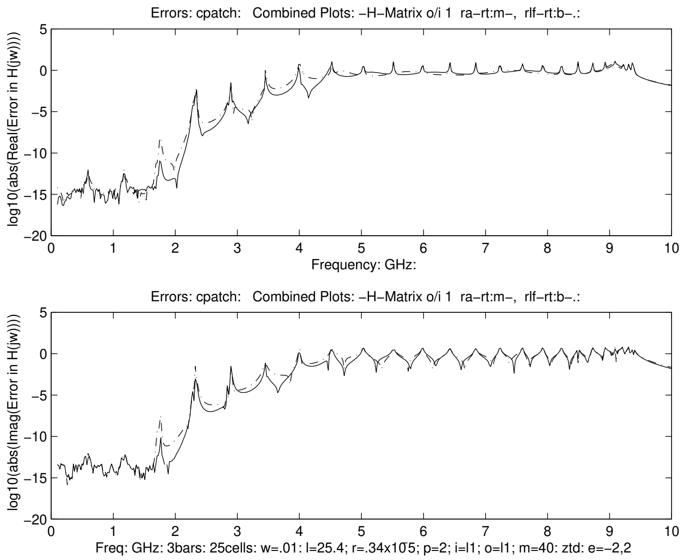

Figure 2: Effects of Finite Precision Arithmetic: $m = 40$. Errors, $\log_{10}|real(\tilde{H}(j\omega) - \tilde{H}_{L.A}(j\omega))|$ and $\log_{10}|imag(\tilde{H}(j\omega) - \tilde{H}_{L,A}(j\omega))|$ versus Frequency in GHZ. Nonsymmetric Lanczos (-.), Two-Sided Arnoldi (-).

The computations were done in finite precision arithmetic using $MATLAB$ [21]. In Figure 1, for $m = 25$, we observe that the errors in the two approximations are equal, providing a demonstration of Theorem 5.2. In Figure 2, however, for $m = 40$, the effects of the finite precision arithmetic are evident. The errors differ because the Lanczos vectors are no longer complex bi-orthogonal. The practicality of two-sided Arnoldi procedures will be explored in another report.

References

[1] J. I. Aliaga, D.L. Boley,R. W. Freund, and V.A. Hernández, *Lanczos-type method for multiple starting vectors* , Numerical Analysis Manuscript No. 96-18, Bell Laboratories, Murray Hill, New Jersey, September 1996.

[2] Z. Bai, *Error analysis of the Lanczos algorithm for nonsymmetric eigenvalue problem*, Math. Comp., 62 (1994), pp. 209–226.

[3] Brezinski, Claude, *Padé-Type Approximation and General Orthogonal Polynomials*, International Series of Numerical Mathematics, vol. 50, Birkhäuser Verlag, Basel, 1980, pp. 35.

[4] Jane Cullum and A.E.Ruehli, *An extension of pseudospectral analysis to nonlinear eigenvalue problems and its use in studying systems with time delays*, IBM Research Report, RC21016, November 1997, IBM Research, Yorktown Heights, NY.

[5] Jane Cullum and Anne Greenbaum, *Relations between Galerkin and norm-minimizing iterative methods for solving linear systems*, SIAM J. Matrix Anal. Appl. 17(2) (1996).

[6] J. Cullum, *Iterative methods for solving Ax = b, GMRES/FOM versus QMR/BiCG* , Advances in Computational Mathematics, 6(1996), pp. 1–24.

[7] J. Cullum, *Arnoldi versus nonsymmetric Lanczos algorithms for solving matrix eigenvalue problems*, BIT 36(1996), pp. 470-493.

[8] J. Cullum, W. Kerner, and R. Willoughby, *A generalized nonsymmetric Lanczos procedure*, Comput. Phys. Comm., 53 (1989), pp. 19–48.

[9] J. Cullum and R. Willoughby, *A practical procedure for computing eigenvalues of large sparse nonsymmetric matrices*, in Large Scale Eigenvalue Problems, J. Cullum and R. Willoughby (eds.), North-Holland, 1986, pp. 193–240.

[10] Peter Feldman and Roland Freund , *Efficient linear circuit analysis by Padé approximation via the Lanczos process.* IEEE Trans. on Computer-Aided Design, 14:639–649, 1995.

[11] R. W. Freund, *Computation of matrix Padé approximations of transfer functions via a Lanczos-type process*, Approximation Theory VIII, Vol.1: Approximation and Interpolation, (C. K. Chui and L. L. Schumaker, eds.), World Scientific Publishing Co., 1995. pp. 215–222.

[12] 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. Stat. Computing, 14,(1993) pp. 137–58.

[13] R. Freund and N. Nachtigal, *QMR: a quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math., 8 (1992), pp. 43–71.

[14] 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.

[15] G. H. Golub and C. F. Van Loan, *Matrix Computations, 2nd ed.* Johns Hopkins University Press, Baltimore, MD, 1989.

[16] M. H. Gutknecht. *The Lanczos process and Padé approximation*, eds. J. D. Brown and M. T. Chu and D. C. Ellison and R. J. Plemmons, *Proceedings of the Cornelius Lanczos International Centenary Conference*, SIAM, Philadelphia, PA, 1994, pp. 61-75.

[17] M. H. Gutknecht. *Variants of BiCGSTAB for matrices with complex spectrum*, SIAM J. Sci. Stat. Comp., 14,(1993) pp. 1020–1033.

[18] W. Kerner, *Large-scale complex eigenvalue problems*, J. Comp. Physics, 85 (1989), pp. 1–85.

[19] Alan J. Laub, Rajini V. Patel, and Paul M. Van Dooren.. *Numerical Linear Algebra Techniques*. IEEE Press, Piscataway, N.J, 1993.

[20] R. B. Lehoucq and D. C. Sorensen, *Deflation techniques for an Implicitly Restarted Arnoldi Iteration*, SIMAX, 17(1996), pp. 789–821.

[21] C. Moler et al., *MATLAB User's Guide*, MathWorks, Inc., 24 Prime Park Way, Natick, MA (1992).

[22] B. N. Parlett, D. R. Taylor, and Z.A. Liu, *A look-ahead Lanczos algorithm for unsymmetric matrices*, Math. Comp., 44 (1985), pp. 105–124.

[23] Axel Ruhe, *The two-sided Arnoldi algorithm for nonsymmetric eigenvalue problems*, *Lecture Notes in Mathematics, vol. 973, Matrix Pencils*, eds., B. Kagstrom and A. Ruhe, pp. 104–120, Springer-Verlag, Heidelberg, 1983.

[24] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Co., Boston, Massachusetts, 1996.

[25] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, Manchester, UK, 1992.

[26] Y. Saad and M. H. Schultz, *GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.

[27] Y. Saad, *Krylov subspace methods for solving unsymmetric linear systems*, Math. Comp., 37 (1981), pp. 105–126.

[28] G.L.G. Sleijpen, H. A. van der Vorst, D.R. Fokkema, *BiCGSTAB(l) and other hybrid Bi-CG methods* , Numerical Algorithms, 7 (1994), pp. 75–109.

[29] D. C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992) pp.357–385.

[30] H. F. Walker, *Implementations of the GMRES method using Householder transformations*, SIAM J. Sci. Stat. Comput. 9(1) (1988), pp. 152–163.

[31] H. A. van der Vorst, *A fast and smoothly converging variant of BiCG for the solutions of nonsymmetric linear systems*, SIAM J. Sci. Stat. Comp., 13 (1992), pp. 631-644.