CSSP 13:733-758, 1994

Krylov Space Methods on State-Space Control Models

Daniel L. Boley

Department of Computer Science
University of Minnesota
Minneapolis, Minnesota

Abstract

We give an overview of various Lanczos/Krylov space methods and how they are being used for solving certain problems in Control Systems Theory based on state-space models. The matrix methods used are based on Krylov sequences and are closely related to modern iterative methods for standard matrix problems such as sets of linear equations and eigenvalue calculations. We show how these methods can be applied to problems in Control Theory such as controllability, observability and model reduction. All the methods are based on the use of state-space models, which may be very sparse and of high dimensionality. For example, we show how one may compute an approximate solution to a Lyapunov equation arising from discrete-time linear dynamic system with a large sparse system matrix by the use of the Arnoldi Algorithm, and so obtain an approximate Grammian matrix. This has applications in model reduction. The close relation between the matrix Lanczos algorithm and the algebraic structure of linear control systems is also explored.

1 Introduction

Most computational methods presently used for control problems cannot handle very effectively very large problems with hundreds or thousand of states. In many instances, very large state-space models are constructed from physical considerations. In order to manipulate these models effectively, it is usually necessary to either form reduced order state-space models or to compute a frequency domain transfer function description of the system, both of which entail some loss of information. This paper is an attempt to show how techniques designed for classic problems of very large dimensionality in linear algebra may be applicable to certain problems in control. These techniques allow one to operate directly on state-space models without needing to construct frequency domain transfer function descriptions. For example, we may compute a reduced order model for a system in a computationally efficient manner without computing a transfer function description, but operating directly on the original state-space model of possibly very large dimensionality.

Many large control problems involve state space representations of large dimensionality. A major problem in Control is to reduce the dimensionality of the models ("model reduction") or just to analyze

the stability or other properties of such large systems [21]. In the literature of computational methods in linear algebra, much study has been done on methods for solving standard linear algebra problems involving large sparse matrix operators: linear equations using conjugate gradients, least squares also with conjugate gradients, eigenvalues using Lanczos methods, singular values, etc. Many of the techniques are mentioned in [30].

It is only recently that the "technology" developed for large sparse linear algebra problems has been applied to control problems of large dimensionality [20]. Most of the methods are based on the recursive generation of Krylov spaces. These methods are based on the idea of projecting the problem onto ever expanding subspaces generated by the matrices occurring in the problem itself. It has been shown that for many linear algebra problems, the convergence rate is much faster than what would be expected from arbitrary projections. Krylov space methods also allow iterating by expanding the dimension of the target space in a very natural way. The algorithms based on recursive generation of the Krylov spaces can be referred to collectively as Lanczos-type algorithms, and are the basis for this paper.

In this paper, we show some existing examples where Lanczos-type algorithms have been applied to some standard problems in Control, in particular model reduction and controllability/observability. The rest of this paper is organized as follows. In section 2 we describe the basic Lanczos algorithms, the Arnoldi Algorithm and the block variants. In sections 3-5 we show how these algorithms may be used to solve certain model reduction problems in Control. We also show how certain theory in Control actually contributes to our knowledge of the behavior of the nonsymmetric Lanczos algorithm.

2 Krylov Sequence Methods

Most iterative methods for large linear algebra problems are based on the recursive generation of so-called Krylov spaces of ever increasing dimensions, and the projection of the original matrix operator(s) onto these Krylov spaces. The success of these algorithms comes from the fact that as the Krylov space dimension increases, the new projection can be obtained very fast from the previous projection onto the next lower dimensional Krylov space. In the domain of Control, the system matrix for a model can be projected in this way. In fact this is the idea behind most of the methods in this paper.

In this section, we define a Krylov space and describe some of the basic algorithms for computing bases for the Krylov space and the projection of a given matrix operator onto these spaces. In the next section, we apply these algorithms to some standard control problems.

A Krylov Sequence is a sequence of vectors generated by a matrix as follows. Given a $n \times n$ matrix A and a vector $\mathbf{x}$, the k-th Krylov sequence $K(A, \mathbf{x}, k)$ is a sequence of k column vectors:

$$K(A, \mathbf{x}, k) \equiv (\mathbf{x}, A\mathbf{x}, A^2\mathbf{x}, \cdots, A^{k-1}\mathbf{x}) .$$

As k increases, we get a sequence of sequences $K(A, \mathbf{x}, 0) \equiv \mathbf{x}$, $K(A, \mathbf{x}, 1) = (\mathbf{x}, A\mathbf{x})$, $\cdots$. A block Krylov sequence is generated by a matrix A and an $n \times p$ matrix X as follows

$$K(A, X, k) \equiv (X, AX, A^2X, \cdots, A^{k-1}X) ,$$

and the corresponding column space is called the *k-th Krylov space* and is denoted by $\mathcal{K}(A, X, k)$.

The basic theorem for Krylov sequences shows how the rank increases as the vectors are appended.

Theorem 1. Given any $n \times n$ matrix A and $n \times p$ matrix X,

$$\text{RANK}\, K(A, X, k) = \text{RANK}\, K(A, X, k+1)$$

if and only if

$$\text{RANK}\, K(A,X,k) = \max_j \text{RANK}\, K(A,X,j) \equiv \text{RANK}\, K(A,X,n) \equiv \text{RANK}\, K(A,X,\infty).$$

Furthermore, the space $\mathcal{K}(A,X,\infty)$ is invariant under A.

Proof: If $\text{RANK}\, K(A,X,k) = \text{RANK}\, K(A,X,k+1)$, then A^kX is a linear combination of $X,\cdots,A^{k-1}X$. But then $A^{k+j}X$ is the same linear combination of $A^jX,\cdots,A^{k+j-1}X$ for every $j \geq 0$. Hence by induction, $A^{k+j}X$ is a linear combination of $X,\cdots,A^{k-1}X$ for all $j \geq 0$. The maximal rank is achieved in at most n steps. □

As the vectors $X, AX, A^2X, \cdots$ are generated and appended to form ever larger and larger Krylov sequences, the theorem states that the rank increases at every step until it reaches a maximal value.

The Lanczos-type Algorithms are algorithms for recursively generating bases for the Krylov Spaces, together with a matrix of recurrence coefficients. This latter matrix will be in reduced form, such as tridiagonal, and represents the projection of the original matrix operator A onto the Krylov Space, typically of smaller dimension.

The Lanczos Algorithm was originally proposed by Lanczos [47] as a method for the computation of eigenvalues of symmetric and nonsymmetric matrices. The idea was to reduce a general matrix to tridiagonal form, from which the eigenvalues could be easily determined. For symmetric matrices, the Lanczos Algorithm has been studied extensively [17, 53]. In that case, the convergence of the algorithm, when used to compute eigenvalues, has been extensively analyzed in [43, 52, 57, 61], [69, p270ff]. This algorithm is particularly suited for large sparse matrix problems. A block Lanczos analog has been studied and analyzed in [29, 17, 53]. However, until recently, the nonsymmetric Lanczos Algorithm has received much less attention. Some recent computational experience with this algorithm can be found in [16]. Besides some numerical stability problems, the method suffered from the possibility of an incurable breakdown from which the only way to "recover" was to restart the whole process from the beginning with different starting vectors [69, p388ff]. More recently, several modifications allowing the Lanczos process to continue after such breakdowns have been proposed in [35, 34, 55], and a numerical implementation has been developed in [23, 24]. The close connection between the modified Non-symmetric Lanczos Algorithm and orthogonal polynomials with respect to indefinite inner products is discussed in [7, 8, 28]. Recently, in [10, 54] the close relation was observed independently between the Lanczos Algorithm and the controllability- observability structure of dynamical systems.

All the above papers address the Lanczos algorithm with single starting vectors. Recently, a block nonsymmetric Lanczos algorithm was proposed in [44, 45], which is capable of starting with several starting vectors, analogous to the block Arnoldi algorithm, but so far the breakdown situation has not been addressed for the block case.

The Lanczos Algorithm [47] is an example of a method that generates bases for Krylov subspaces starting with a given vector. The Arnoldi Algorithm [4] can be thought of as a "one-sided" method, which generates one sequence of vectors that span the reachable space.

2.1 Arnoldi Algorithm

The first algorithm we will describe is the Arnoldi Algorithm [4, 9, 69], which is a recursive way to generate an orthonormal basis for the Krylov space generated by a given matrix A and vectors X. It will be seen that it is also a way to reduce a given matrix to block upper Hessenberg form $\mathbf{H}$. The algorithm proceeds

by recursively filling in the first few columns in the relation

$$A\mathbf{X} = \mathbf{X}\mathbf{H} \text{ subject to } \mathbf{X}^T\mathbf{X} = I, \tag{1}$$

where $\mathbf{X}$ is an $n \times r_{\max}$ matrix of orthonormal columns spanning the Krylov space of maximal rank $r_{\max}$ and $\mathbf{H}$ is an $r_{\max} \times r_{\max}$ block upper Hessenberg matrix. Unless the starting vector is deficient in certain eigendirections [normally an unusual circumstance], $r_{\max} = n$. The following description is taken from [9], suitably modified to use modified Gram-Schmidt Orthogonalization [30, p218] as suggested in [60].

Block Arnoldi Algorithm [9].

0. Start with $n \times n$ matrix A and $n \times p$ matrix X.
   {*Generate orthonormal vectors* $\mathbf{X}$ *and block upper Hessenberg matrix* $\mathbf{H}$ }
1. Factor $X_0 R =$ QR factorization of X. {*normalization*}
2. For $k = 1, 2, 3, \cdots$, while X_k has nonzero rank,
3. Set $X_k^{(0)} = AX_{k-1}$; {*expand Krylov space*}
4. For $j = 1, \cdots, k$, {*modified Gram-Schmidt orthogonalization*}
5. Set $H_{k-j,k-1} = X_{k-j}^T X_k^{(j-1)}$; {*get coefficients to enforce ortho. cond.*}
6. Set $X_k^{(j)} = X_k^{(j-1)} - X_{k-j} H_{k-j,k-1}$ {*enforce ortho. cond.*};
7. Factor $X_k H_{k,k-1} =$ QR factorization of $X_k^{(k)}$. {*normalize next set of columns*}
8. Collect generated vectors $\mathbf{X} = (\, X_0, X_1, \cdots)$ and coefficients $\mathbf{H} = (\, H_{ij}\,)$.

The QR factorization used in steps 1 and 7 is used to compute an orthonormal basis for the column space of a given matrix. The particular formulation we use is, given an $n \times m$, rank r matrix M with $r \le m \le n$, we compute an $n \times r$ matrix Q of orthonormal columns and an upper triangular $r \times m$ matrix R such that $M = QR$. The method used can be either based on a Gram-Schmidt process or by applying a series of Givens rotations or Householder transformations [30]. With this formulation, the diagonal blocks H_{kk} will be square of progressively smaller (or non-increasing) dimensions as k increases, and the off-diagonal blocks will be rectangular.

It is useful to explain the steps individually, since the same process is used in all the Krylov space algorithms. The purpose of step 3 is to generate the next set of vectors expanding the Krylov space. The purpose of steps 4-6 is to orthogonalize the result against all the previous vectors generated. The process used is a modified Gram-Schmidt orthogonalization, which is mathematically equivalent to the ordinary Gram-Schmidt process, but behaves better numerically [30]. Numerical experience [60, 58] has also shown that it is useful to repeat the steps 4-6 to ensure that the orthogonality condition is satisfied to the precision of the computer. The rank of $X_k^{(k)}$ may be less than that of $X_k^{(0)}$, in which case the matrix block H_{kk} generated by the method will not be square. In fact the method proceeds by reducing the dimension of the blocks progressively to zero. Step 7 is to orthonormalize the vectors once they are made orthogonal to the preceding vectors.

Let us examine the situation during intermediate steps of the algorithm. We denote the items generated after k steps by

$$\mathbf{X}_k \equiv (\, X_0, X_1, \cdots, X_{k-1}\,) \text{ and } \mathbf{H}_k \equiv \begin{pmatrix} H_{00} & H_{01} & \cdots & H_{0,k-1} \\ H_{10} & H_{11} & \ddots & H_{1,k-1} \\ & \ddots & \ddots & \vdots \\ 0 & & H_{k-1,k-2} & H_{k-1,k-1} \end{pmatrix}.$$

Then a simple induction argument demonstrates that at each intermediate stage k, the items generated by that stage satisfy

$$\text{COLSP}\,\mathbf{X}_k = \mathcal{K}(A, X, k), \tag{2}$$

$$\mathbf{X}_k^T A \mathbf{X}_k = \mathbf{H}_k, \tag{3}$$

and

$$A\mathbf{X}_k = \mathbf{X}_k \mathbf{H}_k + X_k H_{k,k-1} \left(0, \cdots, 0, I \right), \tag{4}$$

for every k, where the I in (4) is a square identity matrix of dimension equal to the number of columns in X_k. [Here COLSP M denotes the *column space* of M.] When the maximal rank is reached, the term in (4) involving $X_k H_{k,k-1}$ disappears.

Two particular special cases for this algorithm are worthy of note. The first is the special case when the starting vectors X consists of just a single vector. In this case, the QR factorization (used in steps 1 and 7) reduces to a simple normalization of the vector to unit norm, and the resulting matrix $\mathbf{H}$ will be "scalar" upper Hessenberg. The algorithm is considerably simpler, since all the "H-blocks" are just scalars. The algorithm then proceeds as long as the result from step 6 is nonzero. This algorithm is called simply the *Arnoldi Algorithm*, to distinguish it from the *block* version above.

If the matrix A is symmetric, then so will be the generated $\mathbf{H}$. In other words, $\mathbf{H}$ will be (block) tridiagonal. This implies that in step 5, $H_{k-j,k-1}$ will be zero for $j \geq 3$, so that the loop in steps 4-6 need be carried out only for $j = 1, 2$. However, numerical experience has shown that it is often necessary to reorthogonalize the vectors against all the previous vectors when computing in approximate floating point arithmetic (see the extensive literature reported in [30, Ch. 9]). The resulting algorithm is the *Lanczos Algorithm*, and indeed it is historically the first recursive Krylov sequence method proposed [47].

2.2 Nonsymmetric Lanczos Algorithm

If one starts with the symmetric Lanczos Algorithm and relaxes the condition that the generated vectors be orthonormal, but maintains the condition that the generated matrix of coefficients be tridiagonal, one obtains the *nonsymmetric Lanczos algorithm*, capable of reducing most any matrix to tridiagonal form.

We present three variations of the nonsymmetric Lanczos algorithm. To aid in the exposition, we first present the version assuming no breakdown occurs. Then we present that modifications necessary to handle breakdowns. Finally we present the generalizations to obtain the "block" algorithm. The simple scalar algorithm starts with a general matrix A and two starting vectors $\mathbf{x}_0, \mathbf{y}_0$ such that $\mathbf{y}_0^T \mathbf{x}_0 \neq 0$. The method then proceeds to generate two sequences of vectors

$$\mathbf{X}_k = \left(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_{k-1} \right) \quad \text{and} \quad \mathbf{Y}_k = \left(\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_{k-1} \right) \tag{5}$$

such that for every k we have the following bases for the Krylov spaces

$$\text{COLSP}\,\mathbf{X}_k = \mathcal{K}(A, \mathbf{x}_0, k), \quad \text{COLSP}\,\mathbf{Y}_k = \mathcal{K}(A^T, \mathbf{y}_0, k), \tag{6}$$

and the matrix

$$\mathbf{Y}^T \mathbf{X} = \mathbf{D}_K = \begin{pmatrix} d_0 & & 0 \\ & \ddots & \\ 0 & & d_{k-1} \end{pmatrix} \tag{7}$$

is nonsingular and diagonal. The condition (7) is called the *bi-orthogonality* condition. The method is based on the recursive application of the identities

$$A\mathbf{X}_k = \mathbf{X}_k \mathbf{H}_k + \mathbf{x}_k \left(0, \cdots, 0, 1 \right) \beta_k, \quad A^T \mathbf{Y}_k = \mathbf{Y}_k \mathbf{G}_k + \mathbf{y}_k \left(0, \cdots, 0, 1 \right) \gamma_k,$$

and hence

$$\mathbf{G}_k^T \mathbf{D}_k = \mathbf{Y}_k^T A \mathbf{X}_k = \mathbf{D}_k \mathbf{H}_k, \tag{8}$$

where $\mathbf{H}_k, \mathbf{G}_k$ are upper Hessenberg matrices of coefficients

$$\mathbf{H}_k = \begin{pmatrix} h_{00} & h_{01} & & 0 \\ \beta_1 & h_{11} & \ddots & \\ & \ddots & \ddots & h_{k-2,k-1} \\ 0 & & \beta_{k-1} & h_{k-1,k-1} \end{pmatrix}, \quad \mathbf{G}_k = \begin{pmatrix} g_{00} & g_{01} & & 0 \\ \gamma_1 & g_{11} & \ddots & \\ & \ddots & \ddots & g_{k-2,k-1} \\ 0 & & \gamma_{k-1} & g_{k-1,k-1} \end{pmatrix},$$

generated in a manner analogous to that of the Arnoldi Algorithm. Identity (8) implies that the resulting matrices $\mathbf{H}, \mathbf{G}$ will actually take on the above tridiagonal form. The resulting algorithm is then, as taken from [69, pp388f], but using a modified Gram-Schmidt bi-orthogonalization process analogous the ordinary process used above in the Arnoldi Algorithm.

Simple Nonsymmetric Lanczos Algorithm [69].

0. Start with matrix A and two vectors $\mathbf{x}_0, \mathbf{y}_0$ such that $\mathbf{y}_0^T \mathbf{x}_0 = d_0 \neq 0$
1. For $k = 1, 2, \cdots$, while $\mathbf{x}_k \neq 0$ and/or $\mathbf{y}_k \neq 0$
2. Set $\mathbf{x}_k^{(0)} = A\mathbf{x}_{k-1}$ {*expand Krylov spaces*}
   and $\mathbf{y}_k^{(0)} = A^T \mathbf{y}_{k-1}$.
3. For $j = 1, 2$ {*enforce bi-ortho. cond.*}
4. Set $h_{,k-j,k-1} = d_{k-j}^{-1} \mathbf{y}_{k-j}^T \mathbf{x}_k^{(j-1)}$ and $\mathbf{x}_k^{(j)} = \mathbf{x}_k^{(j-1)} - \mathbf{x}_{k-j} h_{k-j,k-1}$.
   Set $g_{k-j,k-1} = d_{k-j}^{-1} \mathbf{x}_{k-j}^T \mathbf{y}_k^{(j-1)}$ and $\mathbf{y}_k^{(j)} = \mathbf{y}_k^{(j-1)} - \mathbf{y}_{k-j} g_{k-j,k-1}$.
5. Set $\beta_k \mathbf{x}_k = \mathbf{x}_k^{(2)}$ and $\gamma_k \mathbf{y}_k = \mathbf{y}_k^{(2)}$
   where $h_{k,k-1} \equiv \beta_k$, $g_{k,k-1} \equiv \gamma_k$ are scale factors to be chosen.
6. If $(\mathbf{y}_k)^T \mathbf{x}_k = 0$ then we have a *breakdown error.*
7. Set $d_k = \mathbf{y}_k^T \mathbf{x}_k$.

There are two typical choices for the scale factors β_k, γ_k. Choice I is to choose them so that $d_k = 1$, i.e. so that $\mathbf{D}_k = I$. In this case (8) implies that $\mathbf{G}_k^T = \mathbf{H}_k$. Choice II is to set $\beta_k = 1$ and $\gamma_k = 1$. In this case, it is easily verified from (8) that $\mathbf{G}_k = \mathbf{H}_k$. Thus with either choice, the computation of these coefficients in step 4 can be cut in half.

In order to handle the possibility of a breakdown, the method must be modified. Originally a limited recovery method was proposed in [55], but a full recovery method was not proposed until more recently in [8, 54]. Numerical implementations have been described in [23, 24]. The modification is based on the idea of generating the individual vectors to satisfy (6), but grouping the generated vectors into clusters, and enforcing the bi-orthogonality condition only between different clusters. The clusters are denoted $X_0 = (\mathbf{x}_0, \cdots, \mathbf{x}_{k_1-1})$, $X_1 = (\mathbf{x}_{k_1}, \cdots, \mathbf{x}_{k_2-1})$, $\cdots$ and $Y_0 = (\mathbf{y}_0, \cdots, \mathbf{y}_{k_1-1})$, $Y_1 = (\mathbf{y}_{k_1}, \cdots, \mathbf{y}_{k_2-1})$, $\cdots$. The resulting method is the following, taken from [8], but using the modified Gram-Schmidt algorithm as before:

Clustered Nonsymmetric Lanczos Algorithm [8].

0. Start with matrix A, two vectors $\mathbf{x}_0, \mathbf{y}_0$
1. Initialize clusters $X_0 = (\,\mathbf{x}_0\,), Y_0 = (\,\mathbf{y}_0\,)$.
2. Initialize cluster counter $p = 0$ and marker $k_0 = 0$.
3. For $k = 1, 2, \cdots$ while $\mathbf{x}_k \neq 0$ and/or $\mathbf{y}_k \neq 0$

4. If $Y_p^T X_p$ is nonsingular then $\{$*close current cluster and increment counter*$\}$
5. Set $D_p = Y_p^T X_p$ $\{$*next diagonal block*$\}$
6. Increment $p = p + 1$.
7. Set $k_p = k$ $\{$*save the index k corresponding to the beginning of cluster p*$\}$
8. Set $\mathbf{x}_k^{(0)} = A\mathbf{x}_{k-1}$ $\{$*expand Krylov spaces*$\}$
and $\mathbf{y}_k^{(0)} = A^T\mathbf{y}_{k-1}$.
9. For $j = 1, 2, \cdots, p$ $\{$*enforce ortho. cond.*$\}$
10. Set $\mathbf{h}_{p-j,k-1} = D_{p-j}^{-1} Y_{p-j}^T \mathbf{x}_k^{(j-1)}$ and $\mathbf{x}_k^{(j)} = \mathbf{x}_k^{(j-1)} - X_{p-j}\mathbf{h}_{p-j,k-1}$.
Set $\mathbf{g}_{p-j,k-1} = D_{p-j}^{-T} X_{p-j}^T \mathbf{y}_k^{(j-1)}$ and $\mathbf{y}_k^{(j)} = \mathbf{y}_k^{(j-1)} - Y_{p-j}\mathbf{g}_{p-j,k-1}$.
11. Set $\beta_k \mathbf{x}_k = \mathbf{x}_k^{(p)} - \{X_p \mathbf{h}_{p,k-1}\}$,
and $\gamma_k \mathbf{y}_k = \mathbf{y}_k^{(p)} - \{Y_p \mathbf{g}_{p,k-1}\}$,
where β_k, γ_k are scale factors to be chosen, and the braces enclose optional terms (see text).
12. Append to current clusters:
$X_p = (\, X_p, \mathbf{x}_k \,)$,
$Y_p = (\, Y_p, \mathbf{y}_k \,)$.

In step 10, the vectors of coefficients $\mathbf{h}_{p-j,k-1}, \mathbf{g}_{p-j,k-1}$ are chosen so that $\mathbf{x}_k, \mathbf{y}_k$ satisfy the bi-orthogonality conditions

$$\mathbf{x}_k^T (\, Y_0, \cdots, Y_{p-1} \,) = 0, \quad \mathbf{y}_k^T (\, X_0, \cdots, X_{p-1} \,) = 0.$$

The coefficients $\mathbf{h}_{p,k-1}, \mathbf{g}_{p,k-1}$ in step 11 can either be chosen to enforce "internal orthogonality" conditions $X_p^T \mathbf{x}_k = 0, Y_p^T \mathbf{y}_k = 0$ with the result that when the algorithm finishes, the individual clusters will satisfy $X_k^T X_k = I = Y_k^T Y_k$ for each k [10], or else they may be chosen to be zero [54], in which case each diagonal block D_p will be lower *anti*-triangular with Hankel structure. In the latter case, the terms enclosed in braces "{}" are omitted.

At each stage $k = k_p$ we group the vectors generated into matrices, partitioned according to the clusters:

$$\mathbf{X}_k = (\, \mathbf{x}_0, \cdots, \mathbf{x}_{k-1} \,) = (\, X_0, \cdots, X_p \,), \quad \mathbf{Y}_k = (\, \mathbf{y}_0, \cdots, \mathbf{y}_{k-1} \,) = (\, Y_0, \cdots, Y_p \,), \tag{9}$$

where the vectors satisfy the clustered bi-orthogonality conditions $Y_i^T X_j = 0$ for $i \neq j$, i.e.

$$\mathbf{Y}_k^T \mathbf{X}_k = \mathbf{D}_k = \text{DIAG}\ (\, D_0, \cdots, D_p \,), \tag{10}$$

where $D_i = Y_i^T X_i$. Since each cluster i is filled until the corresponding matrix D_i is nonsingular, we see that upon termination of the algorithm, all the diagonal blocks of $\mathbf{D}_k$ will be nonsingular, except the last one, D_p. It will be seen in the next section, from considerations in Control Theory, that in fact D_p will be either entirely zero, or empty. Hence the rank of $\mathbf{D}_k$ will be simply the dimension of DIAG $(\, D_0, \cdots, D_{p-1} \,)$. We also partition the upper Hessenberg matrices of coefficients

$$\mathbf{H}_k = (\, \{H_{ij}\}_{i,j=0}^{p} \,), \text{ with } H_{ij} \text{ satisfying } AX_j = (\, X_0, \cdots, X_{j+1} \,) \begin{pmatrix} H_{0j} \\ \vdots \\ H_{j+1,j} \end{pmatrix}, \tag{11}$$

and

$$\mathbf{G}_k = (\, \{G_{ij}\}_{i,j=0}^{p} \,), \text{ with } G_{ij} \text{ satisfying } A^T Y_j = (\, Y_0, \cdots, Y_{j+1} \,) \begin{pmatrix} G_{0j} \\ \vdots \\ G_{j+1,j} \end{pmatrix}, \tag{12}$$

where the partitioning of $\mathbf{H}_k$, $\mathbf{G}_k$ into $(\{H_{ij}\}_{i,j=0}^{p})$, $(\{G_{ij}\}_{i,j=0}^{p})$ is consistent with the partitioning (9). With this partitioning, we have the formula analogous to (8):

$$\mathbf{G}_k^T \mathbf{D}_k = \mathbf{Y}_k^T A \mathbf{X}_k = \mathbf{D}_k \mathbf{H}_k \tag{13}$$

from which it follows that $\mathbf{H}_k, \mathbf{G}_k$ are block tridiagonal. That is, $H_{ij} = G_{ij} = 0$ for $|i-j| > 1$ and This means that, at least in exact arithmetic, in steps 9-11 j need take on only the values 1,2; the remaining coefficients are all zero.

We remark that in floating point arithmetic, the condition of "nonsingularity" in step 4 is necessarily replaced with "not too ill-conditioned", where the amount of acceptable ill-conditioning depends upon a user-supplied parameter. In this case, the last block D_p will no longer be exactly zero. The block tridiagonal structure of $\mathbf{H}, \mathbf{G}$ will remain, but will have to be especially enforced via complete biorthogonalization. A detailed numerical implementation that addresses most of these issues can be found in [23, 24].

We note that also for this algorithm, the matrices of generated vectors $\mathbf{X}_k, \mathbf{Y}_k$ satisfy (6) for every k. In fact, the Clustered Lanczos Algorithm is based on the idea of generating the vectors one by one so that they satisfy (6) at every stage, enforcing (10) at every stage where it is possible. This is accomplished by filling in the identities (11) and (12).

The block Nonsymmetric Lanczos Algorithm is a generalization of the Simple Nonsymmetric Lanczos Algorithm to take several starting vectors on the left and right. No one has yet proposed a clustering scheme to take care of the breakdowns, so we describe the algorithm assuming breakdowns do not occur. The matrices generated will be just block analogs of those generated by the Simple Nonsymmetric Lanczos Algorithm, with uniform block sizes:

$$\mathbf{X}_k = (X_0, X_1, \cdots, X_{k-1}) \text{ and } \mathbf{Y}_k = (Y_0, Y_1, \cdots, Y_{k-1}) \tag{14}$$

$$\mathbf{D}_k = \begin{pmatrix} D_0 & & 0 \\ & \ddots & \\ 0 & & D_{k-1} \end{pmatrix}.$$

$$\mathbf{H}_k = \begin{pmatrix} H_{00} & H_{01} & & 0 \\ \mathcal{B}_1 & H_{11} & \ddots & \\ & \ddots & \ddots & H_{k-2,k-1} \\ 0 & & \mathcal{B}_{k-1} & H_{k-1,k-1} \end{pmatrix}, \quad \mathbf{G}_k = \begin{pmatrix} G_{00} & G_{01} & & 0 \\ \Gamma_1 & G_{11} & \ddots & \\ & \ddots & \ddots & G_{k-2,k-1} \\ 0 & & \Gamma_{k-1} & G_{k-1,k-1} \end{pmatrix},$$

which satisfy

$$A\mathbf{X}_k = \mathbf{X}_k \mathbf{H}_k + X_k \mathcal{B}_k (0, \cdots, 0, I), \tag{15}$$

$$A^T \mathbf{Y}_k = \mathbf{Y}_k \mathbf{G}_k + Y_k \Gamma_k (0, \cdots, 0, I), \tag{16}$$

$$\mathbf{Y}_k^T \mathbf{X}_k = \mathbf{D}_k.$$

The algorithm is a block analog of the Simple algorithm. The following is the form of the algorithm taken from [44, 45], but using the modified Gram-Schmidt bi-orthogonalization process.

Block Nonsymmetric Lanczos Algorithm [44, 45].

0. Start with matrix A and two sets of vectors X, Y such that $Y^T X$ is nonsingular
1. Compute factorizations $X_0 \mathcal{B}_0 = X$ and $Y_0 \Gamma_0 = Y$.
2. For $k = 1, 2, \cdots$ while $X_k \neq 0$ and/or $Y_k \neq 0$
3. Set $X_k^{(0)} = AX_{k-1}$ {*expand Krylov spaces*}

and $Y_k^{(0)} = A^T Y_{k-1}$.

4. For $j = 1, 2$ $\{$*enforce bi-ortho. cond.*$\}$

5. Set $H_{k-j,k-1} = D_{k-j}^{-1} Y_{k-j}^T X_k^{(j-1)}$ and $X_k^{(j)} = X_k^{(j-1)} - X_{k-j} H_{k-j,k-1}$.
   Set $G_{k-j,k-1} = D_{k-j}^{-1} X_{k-j}^T Y_k^{(j-1)}$ and $Y_k^{(j)} = Y_k^{(j-1)} - Y_{k-j} G_{k-j,k-1}$.

6. Set $X_k \mathcal{B}_k = X_k^{(2)}$ and $Y_k \Gamma_k = Y_k^{(2)}$,
   where $\mathcal{B}_k, \Gamma_k$ are matrices chosen to normalize the vector, as indicated below.

7. If $Y_k^T X_k$ is singular then we have a *breakdown error*, unless $X_k \neq 0$ or $Y_k \neq 0$.

8. Set $D_k = Y_k^T X_k$.

In steps 1 and 6, the normalization matrices $\mathcal{B}_k, \Gamma_k$ are normally chosen to make $D_k = Y_k^T X_k = I$. To derive what they are, we write

$$I = D_k = Y_k^T X_k = \Gamma_k^{-T} (Y_k^{(1)})^T X_k^{(1)} \mathcal{B}_k^{-1}. \tag{17}$$

Among all the possible solutions to this equation, [44] has proposed using the particular choice

$$\Gamma_k^T \mathcal{B}_k = \text{ LU Decomposition of } (Y_k^{(1)})^T X_k^{(1)},$$

and they have further suggested the use of pivoting to enhance numerical stability [45]. However, if $(Y_k^{(1)})^T X_k^{(1)}$ should be exactly singular, then we have a *breakdown* condition, and the only "remedy" that has been proposed for the block algorithm is to start over with different starting vectors.

3 Controllability and Observability

In this section we discuss the concepts of Controllability and Observability and the computation of the minimal realization via the Kalman decomposition. We show how the Lanczos algorithms of the previous section can be used to compute the Kalman decomposition and the minimal realization. We will also see how the Controllability and Observability theory provides understanding about the behavior of the Lanczos Algorithm.

Our discussion is based on time invariant linear systems in continuous time:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \quad \mathbf{y}(t) = C\mathbf{x}(t), \tag{18}$$

and in discrete time:

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k, \quad \mathbf{y}_k = C\mathbf{x}_k, \tag{19}$$

where $\mathbf{x}$, $\mathbf{u}$, $\mathbf{y}$ are, respectively, an n-vector of internal states, an m-vector of inputs, and a p-vector of outputs. The systems (18) and (19) are said to be *stable* if all the eigenvalues of A are, respectively, in the left half plane and within the open unit circle. We define the *controllable subspace* $\mathcal{S}_{\text{c}}$ as the subspace of the $\mathbf{x}$-space from which one can find an input that will drive the state to zero, and the *unobservable* space $\mathcal{S}_{\bar{\text{o}}}$as the subspace of the $\mathbf{x}$-space which one cannot detect using the outputs.

A classical algebraic characterization of these spaces is given by

Theorem 2 [19].

$$\mathcal{S}_{\text{c}} \equiv \mathcal{K}(A, B, \infty), \quad \mathcal{S}_{\bar{\text{o}}} \equiv \text{NULLSPACE} \left\{ \left(K(A^T, C^T, \infty) \right)^T \right\} \equiv \mathcal{K}(A^T, C^T, \infty)^{\perp}, \tag{20}$$

where $\mathcal{S}^\perp$ denotes the *orthogonal complement* of the set $\mathcal{S}$.

□

One may also define the *uncontrollable space* $\mathcal{S}_{\bar{\mathrm{c}}}$ and the *observable space* $\mathcal{S}_{\mathrm{o}}$ as the complements of $\mathcal{S}_{\mathrm{c}}$ and $\mathcal{S}_{\bar{\mathrm{o}}}$, respectively, as well as their mutual intersections:

$$\mathcal{S}_{\mathrm{c}\bar{\mathrm{o}}} = \mathcal{S}_{\mathrm{c}} \bigcap \mathcal{S}_{\bar{\mathrm{o}}}, \quad \mathcal{S}_{\mathrm{co}} = \mathcal{S}_{\mathrm{c}} \bigcap \mathcal{S}_{\mathrm{o}}, \quad \mathcal{S}_{\bar{\mathrm{c}}\bar{\mathrm{o}}} = \mathcal{S}_{\bar{\mathrm{c}}} \bigcap \mathcal{S}_{\bar{\mathrm{o}}}, \quad \mathcal{S}_{\bar{\mathrm{c}}\mathrm{o}} = \mathcal{S}_{\bar{\mathrm{c}}} \bigcap \mathcal{S}_{\mathrm{o}}. \tag{21}$$

The *Kalman Decomposition* [26, 41] of (18), (19) is obtained by applying a similarity transformation $T = (\, T_{\bar{\mathrm{c}}\mathrm{o}}, T_{\bar{\mathrm{c}}\bar{\mathrm{o}}}, T_{\mathrm{co}}, T_{\mathrm{c}\bar{\mathrm{o}}} \,)$ where each T_{ab} is a basis for the corresponding $\mathcal{S}_{ab}$. When T is applied, we obtain the new system (for continuous time)

$$[\dot{\widehat{\mathbf{x}}}](t) = \widehat{A}\widehat{\mathbf{x}}(t) + \widehat{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \widehat{C}\widehat{\mathbf{x}}(t), \tag{22}$$

where

$$\widehat{A} = T^{-1}AT = \begin{pmatrix} \widehat{A}_{\bar{\mathrm{c}}\mathrm{o}} & 0 & 0 & 0 \\ \widehat{A}_{21} & \widehat{A}_{\bar{\mathrm{c}}\bar{\mathrm{o}}} & 0 & 0 \\ \widehat{A}_{31} & 0 & \widehat{A}_{\mathrm{co}} & 0 \\ \widehat{A}_{41} & \widehat{A}_{42} & \widehat{A}_{43} & \widehat{A}_{\mathrm{c}\bar{\mathrm{o}}} \end{pmatrix}, \quad \widehat{B} = T^{-1}B = \begin{pmatrix} 0 \\ 0 \\ \widehat{B}_{\mathrm{co}} \\ \widehat{B}_{\mathrm{c}\bar{\mathrm{o}}} \end{pmatrix}, \quad \widehat{C} = CT = (\, \widehat{C}_{\bar{\mathrm{c}}\mathrm{o}} \quad 0 \quad \widehat{C}_{\mathrm{co}} \quad 0 \,). \tag{23}$$

The minimal realization for (18) is then

$$\dot{\mathbf{z}} = \widehat{A}_{\mathrm{co}}\mathbf{z} + \widehat{B}_{\mathrm{co}}\mathbf{u}, \quad \mathbf{y} = \widehat{C}_{\mathrm{co}}\mathbf{z}, \tag{24}$$

where $\mathbf{x} = T_{co}\mathbf{z}$. Analogous statements apply for discrete time systems.

If we define the spaces $\mathcal{S}_{\bar{\mathrm{c}}}$ and $\mathcal{S}_{\mathrm{o}}$ as the *orthogonal* complements of $\mathcal{S}_{\mathrm{c}}$ and $\mathcal{S}_{\bar{\mathrm{o}}}$, respectively, and we use *orthonormal* bases of resulting intersection spaces in the transformation T, then the resulting T is the best conditioned (in the 2-norm) of all transformations yielding the Kalman Decomposition [6].

As shown in [9] it is evident that the Arnoldi Algorithm starting with A and B computes an orthonormal basis for $\mathcal{S}_{\mathrm{c}}$. If the Arnoldi algorithm is applied to the matrix A^T and starting vectors C^T, we obtain an orthonormal basis for the orthogonal complement of $\mathcal{S}_{\bar{\mathrm{o}}}$. Then the individual spaces (21) may be obtained by numerically computing bases for the intersections and orthogonal complements. One method to compute the intersection and orthogonal complements of two spaces $\mathcal{S}_1$, $\mathcal{S}_2$ given two respective orthonormal bases T_1, T_2, is described in [30, §12.4.4]. Briefly, one computes the *singular value decomposition* (SVD) of $T_1^T T_2$ (see the next section for further discussion on the SVD of a matrix product). If $\mathbf{v}_1, \cdots, \mathbf{v}_r$ are the right singular vectors corresponding to the multiple singular value 1, and $\mathbf{v}_{r+1}, \cdots, \mathbf{v}_s$ are all the remaining right singular vectors, then $\{T_2\mathbf{v}_1, \cdots, T_2\mathbf{v}_r\}$ is an orthonormal basis for the intersection space $\mathcal{S}_1 \bigcap \mathcal{S}_2$, and $\{T_2\mathbf{v}_{r+1}, \cdots, T_2\mathbf{v}_s\}$ is an orthonormal basis for the subspace within $\mathcal{S}_2$ orthogonal to the intersection.

For *single input single output* (SISO) systems, a very special structure arises from the Clustered (or Look-ahead) Nonsymmetric Lanczos Algorithm. The rest of this section is devoted to a discussion of this case. To the author's knowledge, noone has proposed a *Block Clustered (or Look-ahead) Lanczos Algorithm* which would correspond to the *multiple input multiple output* (MIMO) case. For SISO systems, we may apply the Clustered Nonsymmetric Lanczos Algorithm with matrix A and starting vectors $\mathbf{x}_0 = B$ and $\mathbf{y}_0 = C^T$. We will see that not only does this algorithm yield bases for the various spaces, but the theory behind the controllable and observable spaces contributes to our understanding of the behavior of the algorithm. If the algorithm is carried out until both $\mathbf{x}_{k^*}$ and $\mathbf{y}_{k^*}$ are zero, appending zero vectors if necessary, then at the termination of the algorithm we will have the generated vectors grouped into clusters, which are further grouped together as follows

$$\mathbf{X}_{k^*} = (\, X_0 \quad X_1 \quad \cdots X_{p-1} | X_{p^*} \,) = (\, \mathbf{X}_{\widehat{k}} | X_{p^*} \,), \quad \mathbf{Y}_{k^*} = (\, Y_0 \quad Y_1 \quad \cdots Y_{p^*-1} | Y_{p^*} \,) = (\, \mathbf{Y}_{\widehat{k}} | Y_{p^*} \,), \tag{25}$$

where $\widehat{k}$ is the index k corresponding to the last *complete* cluster, i.e. $\widehat{k}$ is the rank of $\mathbf{D}_{k^*}$. It was shown in [10] that indeed at termination of the algorithm the matrix $\mathbf{D}_{k^*}$ (10) will have the form $\mathbf{D}_{k^*} = \text{DIAG}\,(D_0, \cdots, D_{p^*-1}, 0)$. This follows from

Theorem 3 [10, 54]. Let the vectors $\mathbf{X}_{k^*} \equiv (\,\mathbf{x}_0 \quad \cdots \quad \mathbf{x}_{k^*-1}\,)$ and $\mathbf{Y}_{k^*} \equiv (\,\mathbf{y}_0 \quad \cdots \quad \mathbf{y}_{k^*-1}\,)$ be the vectors generated by the Clustered Nonsymmetric Lanczos Algorithm with matrix A and starting vectors $\mathbf{x}_0$, $\mathbf{y}_0$. Then there exists a index r such that the generated vectors can be partitioned in the form

$$\mathbf{X}_{k^*} = (\,\mathbf{X}_r \quad | \quad \Xi_r\,) = (\,\mathbf{x}_0 \quad \cdots \quad \mathbf{x}_{r-1} \quad | \quad \mathbf{x}_r \quad \cdots \quad \mathbf{x}_{k^*-1}\,)$$

$$\mathbf{Y}_{k^*} = (\,\mathbf{Y}_r \quad | \quad \Upsilon_r\,) = (\,\mathbf{y}_0 \quad \cdots \quad \mathbf{y}_{r-1} \quad | \quad \mathbf{y}_r \quad \cdots \quad \mathbf{y}_{k^*-1}\,)$$

such that $\mathbf{D}_r = \mathbf{Y}_r^T \mathbf{X}_r$ is nonsingular and both $\Xi_r^T \mathbf{Y}_{k^*}$ and $\Upsilon_r^T \mathbf{X}_{k^*}$ are identically zero or empty.

Proof: Let $K \equiv K(A, \mathbf{x}_0, k^*)$ and $L = K(A^T, \mathbf{y}_0, k^*)$ be the the right and left Krylov sequences, respectively, corresponding to this application of the Lanczos algorithm. Then (6) can be expressed as

$$\mathbf{X}_{k^*} = KU, \quad \mathbf{Y}_{k^*} = LV, \tag{26}$$

where U, V are some upper triangular nonsingular $k^* \times k^*$ matrices. Then the block diagonal matrix (10) will be

$$\mathbf{D}_{k^*} = \mathbf{Y}_{k^*}^T \mathbf{X}_{k^*} = V^T L^T K U.$$

The matrix $L^T K$ is a leading principal submatrix of the infinite Hankel matrix $[K(A^T, \mathbf{y}_0, \infty)]^T K(A, \mathbf{x}_0, \infty)$ of finite rank r. By the Corollary in [25, p. 206], the leading $r \times r$ submatrix of this infinite Hankel matrix must be nonsingular. Hence $k^* \geq r$, and $\mathbf{D}_r$, the leading $r \times r$ submatrix of $\mathbf{D}_{k^*}$, must also be nonsingular.

The condition in step 4 of the Clustered Nonsymmetric Lanczos Algorithm is equivalent to the condition that $\mathbf{D}_k = \mathbf{Y}_k^T \mathbf{X}_k$ be nonsingular, which is in turn equivalent to the condition that $[K(A^T, \mathbf{y}_0, k)]^T K(A, \mathbf{x}_0, k)$ be nonsingular. Hence for every k for which this condition holds, we have a diagonal partitioning of $\mathbf{D}_{k^*}$ as

$$\mathbf{D}_{k^*} = \begin{pmatrix} \mathbf{D}_k & 0 \\ 0 & \Delta_k \end{pmatrix}$$

with a leading $k \times k$ part $\mathbf{D}_k$ and the remaining part denoted Δ_k. In particular, we have this partitioning for $k = r$.

Since $\text{RANK}\,\mathbf{D}_{k^*} = \text{RANK}\,\mathbf{D}_r$, it follows that $\Delta_r = 0$. □

This theorem, together with Theorem 2, implies that at termination of the Lanczos algorithm, $\mathbf{X}_r = T_{\text{co}}$, $\Xi_r = T_{\text{c}\bar{\text{o}}}$, $\Upsilon_r = T_{\bar{\text{c}}\text{o}}$ will be, respectively, bases for the spaces $\mathcal{S}_{\text{co}}$, $\mathcal{S}_{\text{c}\bar{\text{o}}}$, $\mathcal{S}_{\bar{\text{c}}\text{o}}$. The remaining space, $\mathcal{S}_{\bar{\text{c}}\bar{\text{o}}}$ has a basis $T_{\bar{\text{c}}\bar{\text{o}}}$ which may be computed as a basis for the orthogonal complement of the other three spaces. Just as the transformation T is constructed from the vectors obtained directly from the Lanczos algorithm, the resulting transformed system matrix coefficients are obtained directly from the coefficients generated during the course of the Lanczos Algorithm. As shown in detail in [10], substitute $\mathbf{x} = T_{\text{co}}\mathbf{z} = \mathbf{X}_r\mathbf{z}$ into (18), apply $\mathbf{Y}_{k^*}^T$ to extract $\widehat{A}_{\text{co}}, \widehat{B}_{\text{co}}, \widehat{C}_{\text{co}}$, and use the identities (10), (13) from the Clustered Lanczos Algorithm to obtain the minimal realization (24) of the system (18):

$$\dot{\mathbf{z}} = \mathbf{H}_r \mathbf{z} + \mathbf{e}_1 u, \quad y = (\,\mathbf{e}_1^T D_0 \quad 0 \quad \cdots \quad 0\,)\,\mathbf{z},$$

where $\mathbf{H}_r$ is the leading $r \times r$ upper Hessenberg matrix of coefficients from the Clustered nonsymmetric Lanczos algorithm, D_0 is the leading diagonal block of $\mathbf{D}_r$, and $\mathbf{e}_1 = (\,1 \quad 0 \quad \cdots \quad 0\,)^T$ denotes the initial coordinate unit vector of appropriate dimensions.

4 Model Reduction via Balanced Realization

The *balanced realization* [51] is a method for balancing the gains between inputs and states with those between states and outputs, and isolating the states with small gains. These small gain states can be truncated away without disturbing the system by much [2, 22, 27]. The Lanczos-type algorithms can be used to compute approximate balanced realizations for systems of very large dimensionality, whereas most other methods are restricted to systems of only modest dimensionality.

The balanced realization for stable systems (18) is defined in terms of the *controllability grammian*

$$W_{\rm c} = \int_0^\infty e^{At} B B^T e^{A^T t} dt$$

and the *observability grammian*

$$W_0 = \int_0^\infty e^{A^T t} C^T C e^{At} dt.$$

These grammians are, respectively, the solutions to the Lyapunov equations

$$A W_{\rm c} + W_{\rm c} A^T + B B^T = 0 \tag{27}$$

and

$$A^T W_{\rm o} + W_{\rm o} A + C^T C = 0 \tag{28}$$

If the transformation T is applied to (18) to obtain the model

$$[\dot{\widehat{\mathbf{x}}}](t) = \widehat{A}\widehat{\mathbf{x}}(t) + \widehat{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \widehat{C}\widehat{\mathbf{x}}(t), \tag{29}$$

where

$$\widehat{A} = T^{-1} A T, \quad \widehat{B} = T^{-1} B, \quad \widehat{C} = C T \tag{30}$$

then the grammians will be transformed by contragredient (or congruence) transformations into

$$\widehat{W}_{\rm c} = T^{-1} W_{\rm c} T^{-T}, \quad \widehat{W}_{\rm o} = T^T W_{\rm o} T. \tag{31}$$

The balanced realization may be computed by the following prescription [48, 50]. Define $L_{\rm c} L_{\rm c}^T$, $L_{\rm o} L_{\rm o}^T$ as the Cholesky factorizations of $W_{\rm c}$, $W_{\rm o}$, respectively. Compute the SVD of the product $L_{\rm o}^T L_{\rm c}$ to obtain the factorization $U \Sigma V^T = L_{\rm o}^T L_{\rm c}$, where U, V are orthogonal matrices, and $\Sigma = \textsc{diag}\,(\sigma_1, \cdots, \sigma_n)$ with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0$. Then the particular choice $T = L_{\rm c} V \Sigma^{-1/2}$ will reduce the grammians to the same diagonal form $\widehat{W}_{\rm c} = \widehat{W}_{\rm o} = \Sigma$ [50]. The resulting system (29) with this particular choice for T is the *balanced realization* for the system (18) [51]. Effective methods for solving (27), (28) directly for the Cholesky factors without forming $W_{\rm c}$, $W_{\rm o}$ and for finding the SVD of a matrix product without forming the product were given in [37] and [39, 5], respectively, thus enhancing the numerical accuracy of the results.

Partition the matrices in (30) conformally as

$$\widehat{A} = \begin{pmatrix} \widehat{A}_{11} & \widehat{A}_{12} \\ \widehat{A}_{21} & \widehat{A}_{22} \end{pmatrix}, \quad \widehat{B} = \begin{pmatrix} \widehat{B}_1 \\ \widehat{B}_2 \end{pmatrix}, \quad \widehat{C} = \begin{pmatrix} \widehat{C}_1 & \widehat{C}_2 \end{pmatrix},$$

as well as the grammians

$$\widehat{W}_{\rm c} = \widehat{W}_{\rm o} = \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix},$$

where $\widehat{A}_{11}$, Σ_1 are the leading $r \times r$ part of the respective matrices, for some arbitrary r. Define $F(s) = C(sI - A)^{-1}B = \widehat{C}(sI - \widehat{A})^{-1}\widehat{B}$ as the transfer function for (18) and (29), and $G_1(s) = \widehat{C}_1(sI - \widehat{A}_{11})^{-1}\widehat{B}_1$

as the transfer function of the system obtained when the last $n-r$ states in the balanced realization are truncated away. Then it has been shown (e.g. [22]) that

$$\|G(s) - G_1(s)\|_\infty \leq 2\sum_{r+1}^{n} \sigma_i,$$

where the norm $\|\cdot\|_\infty$ on functions analytic everywhere except at some poles in the left half plane is defined by

$$\|G(s)\|_\infty = \sup_\omega \sigma_{\max} G(j\omega),$$

where $j\omega$ varies over the entire imaginary axis. Thus the truncation of the balanced realization is a reduced order model whose behavior does not differ from that of the original system by very much.

A similar development may be carried out for stable discrete time systems of the form (19). In this case, the grammians are defined as

$$W_{\rm c} = \sum_{i=0}^{\infty} A^i B B^T (A^T)^i, \quad W_{\rm o} = \sum_{i=0}^{\infty} (A^T)^i C^T C A^i,$$

and satisfy the Lyapunov equations

$$A W_{\rm c} A^T + B B^T = W_{\rm c} \tag{32}$$

$$A^T W_{\rm o} A + C^T C = W_{\rm o}. \tag{33}$$

When a transformation T is applied to (19) to obtain the transformed system

$$\widehat{\mathbf{x}}_{k+1} = \widehat{A}\widehat{\mathbf{x}}_k + \widehat{B}\mathbf{u}_k, \quad \mathbf{y}_k = \widehat{C}\mathbf{x}_k, \tag{34}$$

where the matrices are defined by (30), the grammians are again transformed by the same congruence transformations (31). Thus the balanced realization can be computed using the same prescription as for the continuous time case [38]. A similar error bound for truncated balanced realizations was obtained in [2].

It is difficult to solve the Lyapunov equations for very large sparse systems. The Arnoldi Algorithm may be used to compute approximate solutions to such Lyapunov equations (27) (28) (32) (33), which will satisfy certain Galerkin conditions. We will describe these conditions for the discrete time case. The continuous time case is analogous and is treated in [59]. In addition, one can obtain an error bound for the discrete time solution. Similar techniques have been proposed for the more general Sylvester equation $AX - XB = C$ in [56], in which the approximate solutions can be recursively generated as the Arnoldi process advances.

We show how the Arnoldi Algorithm yields an approximate solution to (32), then we show that it satisfies the Galerkin property and an error bound. If the Arnoldi Algorithm is carried out with a matrix A and starting vectors $X = B$, then after k steps, we have the relation (4). We also have that $R = X_0^T B$ is the nonsingular upper triangular matrix constructed in step 1 of the Arnoldi Algorithm. Define

$$G = \sum_{i=0}^{\infty} (\mathbf{H}_k)^i \begin{pmatrix} R \\ 0 \end{pmatrix} \begin{pmatrix} R^T & 0 \end{pmatrix} (\mathbf{H}_k^T)^i \tag{35}$$

so that G satisfies the Lyapunov Equation

$$\mathbf{H}_k G \mathbf{H}_K^T + \begin{pmatrix} R \\ 0 \end{pmatrix} \begin{pmatrix} R^T & 0 \end{pmatrix} = G. \tag{36}$$

Then the approximate solution to (32) is $\widetilde{W} = \mathbf{X}_k G \mathbf{X}_k^T$. One can take advantage of the recursive nature of the Arnoldi Algorithm to use the approximate solution obtained after each step of the Arnoldi Algorithm to generate subsequent approximate solutions rapidly, but for the sake of brevity and clarity we do not describe that process here; this is the basis of the methods in, for example, [56, 60]. This solution satisfies the following theorem, proved in [59] for the continuous time case:

Theorem 4. Define the inner product on $n \times n$ matrices as $\langle X, Y \rangle \equiv tr(X^T Y)$. Define the space $\mathcal{G} = \{\mathbf{X}_k M \mathbf{X}_k^T : \forall M \in \mathcal{R}^{k \times k}\}$. Then the solution $\widetilde{W} = \mathbf{X}_k G \mathbf{X}_k^T$ lies in $\mathcal{G}$ and the residual:

$$\text{RES}\,(\widetilde{W}) \equiv A\widetilde{W}A^T + BB^T - \widetilde{W}$$

is orthogonal to $\mathcal{G}$ with respect to this inner product. Hence $\widetilde{W}$ satisfies a Galerkin condition.

[Here the notation $tr(M)$ stands for the *trace* of M.]

Proof: All we must verify is that $\langle \mathbf{X}_k M \mathbf{X}_k^T, \text{RES}\,(\widetilde{W}) \rangle = 0$ for any M. We make use of the identity $tr(MN) = tr(NM)$ for any matrices M, N and the identity (3). We have

$$\begin{aligned}
\langle \mathbf{X}_k M \mathbf{X}_k^T, \text{RES}\,(\widetilde{W}) \rangle &= tr((\mathbf{X}_k M \mathbf{X}_k^T) A (\mathbf{X}_k G \mathbf{X}_k^T) A^T + (\mathbf{X}_k M \mathbf{X}_k^T) BB^T - (\mathbf{X}_k M \mathbf{X}_k^T)(\mathbf{X}_k G \mathbf{X}_k^T)) \\
&= tr(M(\mathbf{X}_k^T A \mathbf{X}_k) G (\mathbf{X}_k^T A^T \mathbf{X}_k) + M \mathbf{X}_k^T BB^T \mathbf{X}_k - M \mathbf{X}_k^T \mathbf{X}_k G \mathbf{X}_k^T \mathbf{X}_k) \\
&= tr(M \mathbf{H}_k G \mathbf{H}_k^T + M \begin{pmatrix} R \\ 0 \end{pmatrix} (\, R^T \quad 0\,) - MG) \\
&= tr(M(\mathbf{H}_k G \mathbf{H}_k^T + \begin{pmatrix} R \\ 0 \end{pmatrix} (\, R^T \quad 0\,) - G)) \\
&= tr(M \cdot 0) \\
&= 0
\end{aligned}$$

□

In addition to the above Galerkin condition, we can also bound the error $\|W_c - \widetilde{W}\|$. In order to do this, we need the following Theorem from Linear Algebra

Theorem 5. Let $\rho(A)$ denote the spectral radius of a matrix A. If $\rho(A) < \delta < 1$, then there is a constant θ depending only on A and the choice of δ such that for any $k = 1, 2, 3, \cdots$

$$\|A^k\|_2 \leq \theta \delta^k.$$

Furthermore, if A is diagonalizable, then the above also holds for $\delta = \rho(A) < 1$.

Proof: One can find a consistent matrix norm such that $\|A\|_* \leq \delta$ [63]. Since all norms on a finite dimensional space are equivalent, one can find a constant θ such that $\|M\|_2 \leq \theta \|M\|_*$ for all matrices M. The result follows from

$$\|A^k\|_2 \leq \theta \|A^k\|_* \leq \theta \|A\|_*^k \leq \theta \delta^k.$$

If A is diagonalizable, then we have $A^k = PD^kP^{-1}$, for diagonal D, which yields equality above with $\theta = \|P\|_2 \cdot \|P^{-1}\|_2$ and $\delta = \rho$. □

We also need the lemma

Lemma 6. When the Arnoldi Algorithm is run with a matrix A and starting vectors B, then for $0 \leq i < k$

$$A^i B = \mathbf{X}_k \mathbf{H}_k^i \begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

where $R = X_0^T B$ is the upper triangular matrix computed in step 1 of the Arnoldi Algorithm.

Proof: Choose a k and assume $0 \le i < k$. Then we have the identity from (4)

$$A\mathbf{X}_k\mathbf{H}_k^{i-1}\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{X}_k\mathbf{H}_k^i\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix} + X_k H_{k,k-1}(0,\cdots,0,I)\mathbf{H}_k^{i-1}\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{X}_k\mathbf{H}_k^i\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

since

$$(0,\cdots,0,I)\mathbf{H}_k^{i-1}\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix} = 0.$$

The lemma then follows from

$$A^iB = A^i\mathbf{X}_k\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix} = A^{i-1}\mathbf{X}_k\mathbf{H}_k\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \cdots = A^0\mathbf{X}_k\mathbf{H}_k^i\begin{pmatrix} R \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

□

Theorem 7. Suppose the Arnoldi Algorithm is run with matrix A and starting vectors B, where $\rho(A) < \delta < 1$ for some δ. Suppose also that $\rho(\mathbf{H}_k) < \delta < 1$. Then

$$\|W_{\rm c} - \widetilde{W}\|_2 \le \frac{2\|B\|_2^2\theta^2\delta^{2k}}{1-\delta^2},$$

where θ is the constant from Theorem 5.

Proof: By the lemma, for $0 \le i < k$,

$$A^iBB^T(A^T)^i = \mathbf{X}_k\mathbf{H}_k^i\begin{pmatrix} R \\ 0 \end{pmatrix}\begin{pmatrix} R^T & 0 \end{pmatrix}(\mathbf{H}_k^T)^i\mathbf{X}_k^T.$$

Therefore, we have the identity

$$\begin{aligned} W_{\rm c} - \widetilde{W} &= \textstyle\sum_{i=0}^\infty A^iBB^T(A^T)^i - \mathbf{X}_k\sum_{i=0}^\infty\left(\mathbf{H}_k^i\begin{pmatrix} R \\ 0 \end{pmatrix}\begin{pmatrix} R^T & 0 \end{pmatrix}(\mathbf{H}_k^T)^i\right)\mathbf{X}_k^T \\ &= \textstyle\sum_{i=k}^\infty A^iBB^T(A^T)^i - \mathbf{X}_k\sum_{i=k}^\infty\left(\mathbf{H}_k^i\begin{pmatrix} R \\ 0 \end{pmatrix}\begin{pmatrix} R^T & 0 \end{pmatrix}(\mathbf{H}_k^T)^i\right)\mathbf{X}_k^T. \end{aligned}$$

It remains to bound the last sums.

$$\begin{aligned} \left\|\textstyle\sum_{i=k}^\infty A^iBB^T(A^T)^i\right\| &\le \|B\|^2\textstyle\sum_{i=k}^\infty\|A^i\|^2 \\ &\le \|B\|^2\textstyle\sum_{i=k}^\infty\theta^2\delta^{2i} \\ &\le \|B\|^2\theta^2\delta^{2k}\frac{1}{1-\delta^2}. \end{aligned}$$

We have the same bound for the sum in $\mathbf{H}_k^i$, where $\|R\| = \|B\|$. So the final bound on the above sums is as given in the theorem. □

Normally, as k increases, i.e. as we carry out the Arnoldi Algorithm to more steps, the eigenvalues of $\mathbf{H}_k$ should become ever better approximations to those of A, and hence should be bounded by some $\delta < 1$. So as k increases, the bound in Theorem 7 will go to zero, and we obtain ever better approximations to the grammian.

5 Model Reduction via Parameter Matching

An alternative approach to model reduction is to expand the transfer function in a Laurent series, and then to create a lower dimensional model for which the first few terms of its Laurent series matches those from the original model. If

$$F(s) = \sum_{i=0}^{\infty} \frac{F_i}{s^{i+1}} \tag{37}$$

is a transfer function, then the idea is to construct a smaller model which yields a transfer function

$$\widehat{F}(s) = \sum_{i=0}^{\infty} \frac{\widehat{F}_i}{s^{i+1}} \tag{38}$$

for which $\widehat{F}_i = F_i$ for $i = 0, \cdots, k$ for some given k. The parameters F_i are called the *high frequency moments* [68] or *Markov Parameters* [40, 62]. It is a simple consequence of the matrix identity (Neumann Series)

$$(I - M)^{-1} = \sum_{i=0}^{\infty} M^i \text{ whenever } \rho(M) < 1.$$

that the transfer function for (18) and (19) satisfies the identity

$$F(s) = C(sI - A)^{-1}B = \sum_{i=0}^{\infty} \frac{CA^iB}{s^{i+1}} \text{ for } |s| > \rho(A). \tag{39}$$

With respect to (19), if $\mathbf{u}_i = 0$, $i = 1, 2, \cdots$ and $\mathbf{x}_0 = 0$, then the sequence $\{\mathbf{y}_{i+1} = CA^iB\mathbf{u}_0\}$, $i = 0, 1, 2, \cdots$ is the *impulse response* from the impulse $\mathbf{u}_0$. The systems (18) and (19) are *state-space realizations* of the transfer function (39).

In applications, $F(s)$ may be either the usual case of a finite dimensional system of large order or an infinite dimensional system [33]. In either case, the model reduction problem is "solved" by constructing (*realizing*) a lower order model of the form (18) such that $\widehat{C}\widehat{A}^i\widehat{B} = F_i$ for $i = 0, \cdots, k$, for a given k. In the SISO case, such a reduced order realization may be found via computation of a transfer function $\widehat{F}(s)$, expressed as a rational function of polynomials (see e.g. [25, Ch. 15 §10], [18, 32, 42]). Several authors have also looked at the MIMO realization problem (see e.g. [67] and refs. therein, such as [3, 14, 46, 70]. Many of the algorithms involved are recursive algorithms which are intimately related to the Clustered Lanczos Algorithm (see e.g. [11, 32, 54]). We describe two approaches for the MIMO case proposed recently for constructing such a smaller state-space model, both based on the use of a large state-space model.

The first approach was proposed in [33] as a method for finding a finite dimensional approximation to an infinite dimensional system, in the case that the transfer function $F(s)$ is given, but not the expansion (37) nor a state-space realization. In this case, [33] proposed applying the discrete Fourier transform to the sequence $F(1), F(\omega), F(\omega^2), \cdots, F(\omega^{N-1})$ for some $N > k$, where ω is an N-th root of unity to obtain approximations to the first N terms in the series (37). Next they construct a triple of matrices $\widehat{A}, \widehat{B}, \widehat{C}$ such that $\widehat{C}\widehat{A}^i\widehat{B} = F_i$ for $i = 0, \cdots, k$ and $\widehat{C}\widehat{A}^i\widehat{B} = 0$ for $i > k$. They choose the triple $\widehat{A}, \widehat{B}, \widehat{C}$ to be:

$$\widehat{A} = \begin{pmatrix} 0 & & & \\ I & \ddots & & \\ & \ddots & \ddots & \\ & & I & 0 \end{pmatrix}, \quad \widehat{B} = \begin{pmatrix} I \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad \widehat{C} = \begin{pmatrix} F_0 & F_1 & \cdots & F_k \end{pmatrix}$$

We refer the reader to [33] for the detailed derivations, including how they apply further model reduction techniques to the resulting finite dimensional model. In particular, they use the balanced realization as described in the previous section.

The second approach that we describe was proposed by [44, 45], and is based on the theory in [68]. In this case, we start with a state-space model of the form (18) (or (19)) represented by the triple A, B, C of large dimensionality and we try to find a triple $\widehat{A}, \widehat{B}, \widehat{C}$ for which the corresponding series (38), defined by $\widehat{F}_i = \widehat{C}\widehat{A}^i\widehat{B}$, agrees with (37) up to a given number of terms. The reduced order model is constructed by the use of the Block Nonsymmetric Lanczos Algorithm. We assume we have an initial model (18), and we assume for simplicity in presentation that B, C and C^TB all have full rank. The Block Nonsymmetric Lanczos Algorithm is applied with matrix A and starting vectors $X_0\mathcal{B}_0 = B$ and $Y_0\Gamma_0 = C^T$, using the scaling (17), so that $\mathcal{B}_0$ and Γ_0 are also constructed so that $Y_0^TX_0 = I$. After k steps, we have the generated vectors (14) which satisfy

$$\text{COLSP}\,\mathbf{X}_k = \mathcal{K}(A, B, k), \quad \text{COLSP}\,\mathbf{Y}_k = \mathcal{K}(A^T, C^T, k), \text{ and } \mathbf{Y}_k^T\mathbf{X}_k = I.$$

Then the reduced order model is defined by

$$\dot{\mathbf{z}}(t) = \widehat{A}\mathbf{z}(t) + \widehat{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \widehat{C}\mathbf{z}(t), \tag{40}$$

where

$$\widehat{A} = \mathbf{Y}_k^TA\mathbf{X}_k, \quad \widehat{B} = \mathbf{Y}_k^TB, \quad \widehat{C} = C\mathbf{X}_k, \quad \mathbf{z} = \mathbf{Y}_k^T\mathbf{x}. \tag{41}$$

It remains to verify that $\widehat{C}\widehat{A}^i\widehat{B} = F_i$ for sufficiently many values of i.

Theorem 8. Given a system (18), together with the results from k steps of the Block Nonsymmetric Lanczos Algorithm started with A, B, C using scaling (17), and given the reduced order model defined by (40) (41), then

$$\widehat{C}\widehat{A}^i\widehat{B} = CA^iB \text{ for } i = 0, \cdots, 2(k-1)$$

To prove this, we use a lemma:

Lemma 9 [68]. $\widehat{A}^i\widehat{B} = \mathbf{Y}_k^TA^iB$ and $\widehat{(A^T)}^i\widehat{C}^T = \mathbf{X}_k^T(A^T)^iC^T$ for $0 \le i \le k-1$.

Proof of Lemma: The properties (15), (16) are identical to (4), so the proof is analogous to that for Lemma 6.

Proof of Theorem: Choose any i, j such that $i < k$, $j < k$. Then $A^jB \in \mathcal{K}(A, B, k)$, hence we can write $A^jB = \mathbf{X}_kV$ for some matrix V. It follows that

$$\mathbf{X}_k\mathbf{Y}_k^TA^jB = \mathbf{X}_k\mathbf{Y}_k^T\mathbf{X}_kV = \mathbf{X}_kV = A^jB.$$

We then use the lemma to establish

$$\widehat{C}\widehat{A}^i\widehat{A}^j\widehat{B} = CA^i\mathbf{X}_k\mathbf{Y}_k^TA^jB = CA^iA^jB.$$

□

The reduced order model obtained by this method exhibits similar transient response behavior to that for the original model. In order to obtain a model that also exhibits similar steady state behavior, it is necessary to compute a model in which the *low frequency moments* $\widehat{C}\widehat{A}^{-i-1}\widehat{B}$, $i = 0, \cdots, p$, match those for the original system [68]. One such model is obtained by an oblique projection

$$\dot{\mathbf{z}} = L^TAT\mathbf{z} + L^TB\mathbf{u}, \quad \mathbf{y} = CT\mathbf{z}, \tag{42}$$

where the columns of matrices L and T include bases for the spaces $\mathcal{K}(A^{-T}, A^{-T}C^T, p)$ and $\mathcal{K}(A^{-1}, A^{-1}B, p)$, respectively [68], where $M^{-T} \equiv (M^T)^{-1} \equiv (M^{-1})^T$. In order to create models matching both the high frequency moments and the low frequency moments, [64] propose combining the vectors generated from the Block Lanczos method using A, B, C^T with those from the Lanczos method using A^{-1}, $A^{-1}B$, C^T. This is based on the following theorem, whose proof is analogous to that of Theorem 8, which is itself a special case of the following.

Theorem 10 [68]. If L, T are bases for the spaces

$$\text{COLSP}\ (\ (A^T)^{-p}C^T \quad \cdots \quad (A^T)^{+q}C^T\ ),$$

$$\text{COLSP}\ (\ A^{-s}B \quad \cdots \quad A^{+t}B\ ),$$

where $L^TT = I$ and $p + q = s + t$, then the system (42) obtained by the oblique projection satisfies

$$(CT)(L^TAT)^i(L^TB) = CA^iB \text{ for } i = -p - s - 1, \cdots, +q + t + 1.$$

□

This approach has been exploited very successfully in [15, 64, 65] to applications in large flexible space structures. One problem with this method is that the block Nonsymmetric Lanczos Algorithm may break down, if the matrix in step 7 of the algorithm is singular. In practice, this has not been a problem in the applications in [65]. We note that in this use of the algorithm, we do not compute the spaces $\mathcal{K}(A, B, \infty)$ or $\mathcal{K}(A^T, C^T, \infty)$. Rather, we choose a $k_{\max}$ and run the algorithm only that many steps. If $k_{\max}$ is much less that the dimension of the entire spaces $\mathcal{K}(A, B, \infty)$ or $\mathcal{K}(A^T, C^T, \infty)$, it is reasonable not to expect breakdowns in the first $k_{\max}$ steps.

6 Conclusions

In this paper, we have described three variants of the Lanczos Algorithm for generating Krylov sequences and gave three examples of control theory applications where these Lanczos algorithms may be used. These algorithms are particularly suited to solving control problems involving state-space models with sparse matrices of very large order. The development of algorithms for such control problems is still in its infancy, especially since many algorithms for "small" dense state-space models are also of recent origin [49]. We have illustrated the use of the Lanczos algorithms on problems of Controllability, Observability, and Model Reduction. The grammians are useful also for other computations in Optimal Control and $\mathbf{H}_\infty$ Theory (e.g. finding the $\mathbf{H}_2$, $\mathbf{H}_\infty$, and Hankel norms of plants [12, and refs. therein]), hence the methods of this paper can be used to compute approximate solutions to those problems.

We remark that the reduced order systems (42), (40) have transfer functions which are rational functions whose Laurent or MacClaurin series expansions agree with the series expansions of the original transfer functions to a given number of terms. In the space of the transfer functions themselves, the problem of computing low degree rational approximations to given meromorphic functions is a classical problem in Approximation Theory in the complex plane. Many methods operating directly on the coefficients in the expansions of the form (37) (or more general MacClaurin expansions) exist, and many are equivalent to the Lanczos Process [11, 13, 32, 31, 54, and refs. therein], and many use related algorithms on related Hankel matrices, though based on different theory [1, 36, 66]. The methods presented in this paper are distinctive in that they are based on the use of state-space descriptions, which facilitate the generalization of many SISO methods to MIMO systems, and which may often allow one to avoid unnecessary conversions between state-space and frequency domain descriptions and the ensuing computational and numerical difficulties. Space

does not permit us to explore here the relations and equivalences between the state-space approximations and the frequency domain methods based on optimal Hankel, $\mathbf{H}_\infty$, or Padé approximations (mentioned in some of the references), or the close relations with fast Hankel matrix factorization methods and orthogonal polynomials. This will be explored in a future paper.

7 Acknowledgments

The author would like to acknowledge the helpful comments from Youcef Saad, Roland Freund and Franklin Luk, as well as the support from the Minnesota Supercomputer Institute.

References

[1] V. Adamjan, D. Arov, and M. Krein, *Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur-Takagi problem*, Mat. USSR Sbornik, 15 (1971), pp. 31–73.

[2] U. M. Al-Saggaf and G. F. Franklin, *An error bound for a discrete reduced order model of a linear multivariable system*, IEEE Trans. Auto. Contr., AC-32 (1987), pp. 815–819.

[3] A. Antoulas, *New results on the algebraic theory of linear systems: The solution of the cover problems*, Lin. Alg & Appl., 50 (1983), pp. 1–45.

[4] W. E. Arnoldi, *The principle of minimized iterations in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29.

[5] A. W. Bojanczyk, L. W. Ewerbring, F. T. Luk, and P. Van Dooren, *An accurate product SVD algorithm*, Signal Processing, 25 (1991), pp. 189–201.

[6] D. L. Boley, *Computing the Kalman decomposition, an optimal method*, IEEE Trans. Auto. Contr, AC-29 (1984), pp. 51–53. [Correction in AC-36 (1991), p. 1341].

[7] D. L. Boley, R. P. Brent, G. H. Golub, and F. T. Luk, *Error correction via the Lanczos process*, SIAM J. Matrix Anal., (1992), pp. 312–332.

[8] D. L. Boley, S. Elhay, G. H. Golub, and M. H. Gutknecht, *Nonsymmetric Lanczos and finding orthogonal polynomials associated with indefinite weights*, Numerical Algorithms, 1 (1991), pp. 21–44.

[9] D. L. Boley and G. H. Golub, *The Lanczos algorithm and controllability*, Systems & Control Letters, 4 (1984), pp. 317–324.

[10] ——, *The nonsymmetric Lanczos algorithm and controllability*, Systems & Control Letters, 16 (1991), pp. 97–105.

[11] D. L. Boley, T. J. Lee, and F. T. Luk, *The Lanczos algorithm and Hankel matrix factorization*, Lin. Alg. & Appl., 172 (1992), pp. 109–133.

[12] S. P. Boyd and C. H. Barratt, *Linear Controller Design*, Prentice Hall, 1991.

[13] C. Brezinski, *Padé-Type Approximation and General Orthogonal Polynomials*, Birkhäuser Basel, 1980.

[14] A. Bultheel, *Recursive algorithms for the matrix Padé problem*, Math. Comp., 35 (1980), pp. 875–892.

[15] R. R. Craig Jr. and A. L. Hale, *Block-Krylov component synthesis method for structural model reduction*, J. Guid., Contr., & Dyn., 11 (1988), pp. 562–570.

[16] J. Cullum, W. Kerner, and R. Willoughby, *A generalized nonsymmetric Lanczos procedure*, Computer Physics Communications, 53 (1989), pp. 19–48.

[17] J. Cullum and R. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations*, vol. I, Theory, Birkhäuser Boston, 1985.

[18] L. S. de Jong, *Numerical aspects of recursive realization algorithms*, SIAM J. Contr. & Opt., 16 (1978), pp. 646–659.

[19] C. A. Desoer, *A Second Course in Linear Systems*, Van Nostrand Reinhold, 1970.

[20] P. V. Dooren, *Numerical linear algebra techniques for large scale matrix problems in systems and control*, in Proc. 31st Conf. on Dec. & Contr, Tuscon AZ, Dec. 1992. session TM6, to appear.

[21] ——, *Upcoming numerical linear algebra issues in systems and control theory*, TR 1007, Univ. of Minn. IMA, August 1992.

[22] D. F. Enns, *Model reduction with balanced realizations: An error bound and a frequency weighted generalization*, in Proc. 23rd IEEE Conf. Dec. Contr., 1984, pp. 127–132.

[23] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal, *An implementation of the look-ahead Lanczos algorithm for non-hermitian matrices, part I*, math. numerical analysis report 90-10, M.I.T., 1990.

[24] R. W. Freund and N. M. Nachtigal, *An implementation of the look-ahead Lanczos algorithm for non-hermitian matrices, part II*, math. numerical analysis report 90-11, M.I.T., 1990.

[25] F. R. Gantmacher, *Theory of Matrices*, vol. 2, Chelsea, New York, 1959.

[26] E. G. Gilbert, *Controllability and observability in multivariable control systems*, SIAM J. Contr., 1 (1963), pp. 128–151.

[27] K. Glover, *All optimal Hankel norm approximations of linear multivariable systems, and their L_∞ error bounds*, Intl. J. Contr., 39 (1984), pp. 1115–1193.

[28] G. H. Golub and M. H. Gutknecht, *Modified moments for indefinite weight functions*, Numer. Math., 57 (1990), pp. 607–624.

[29] G. H. Golub and R. Underwood, *The block Lanczos method for computing eigenvalues*, in Mathematical Software III, J. Rice, ed., Acad. Press, 1977, pp. 364–377.

[30] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins Univ. Press, 2nd ed., 1989.

[31] W. B. Gragg, *The Padé table and its relation to certain algorithms in numerical analysis*, SIAM Rev., 14 (1972), pp. 1–62.

[32] W. B. Gragg and A. Lindquist, *On the partial realization problem*, Lin. Alg. & Appl., 50 (1985), pp. 277–319.

[33] G. Gu, P. P. Khargonekar, and E. B. Lee, *Approximation of infinite-dimensional systems*, IEEE Trans. Auto. Contr., AC-34 (1989), pp. 610–618.

[34] M. H. Gutknecht, *The unsymmetric Lanczos algorithms and their relations to Padé approximation, continued fractions, the qd algorithm, biconjugate gradient squared algorithms, and fast Hankel solvers.* preprint, 1990.

[35] ——, *A completed theory of the unsymmetric Lanczos process and related algorithms, part I*, SIAM J. Mat. Anal., 13 (1992), pp. 594–639.

[36] M. H. Gutknecht and L. N. Trefethen, *Real polynomial Chebyshev approximation by the Carathéodory-Féjer method*, SIAM J. Numer. Anal., 19 (1982), pp. 358–371.

[37] S. J. Hammarling, *Numerical solution of stable, non-negative definite Lyapunov equations*, IMA J. Numer. Anal., 2 (1982), pp. 303–323.

[38] ——, *Numerical solution of the discrete-time convergent, non-negative definite Lyapunov equation*, Systems & Control Letters, 17 (1991), pp. 137–139.

[39] M. T. Heath, A. J. Laub, C. C. Paige, and R. C. Ward, *Computing the SVD of a product of two matrices*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 1147–1159.

[40] J. Hickin and N. K. Sinha, *Model reduction for linear multivariable systems*, IEEE Trans. Auto. Contr., AC-25 (1980), pp. 1121–1127.

[41] R. E. Kalman, *Mathematical description of linear systems*, SIAM J. Contr., 1 (1963), pp. 152–192.

[42] ——, *On partial realizations, transfer functions and canonical forms*, Acta Polytech. Scand., MA31 (1979), pp. 9–32. Helsinki.

[43] S. Kaniel, *Estimates for some computational techniques in linear algebra*, Math. Comp., 20 (1966), pp. 369–378.

[44] H. M. Kim and R. R. Craig Jr., *Structural dynamics analysis using an unsymmetric block Lanczos algorithm*, International Journal for Numerical Methods in Engineering, 26 (1988), pp. 2305–2318.

[45] ——, *Computational enhancement of an unsymmetric block Lanczos algorithm*, International Journal for Numerical Methods in Engineering, 30 (1990), pp. 1083–1089.

[46] S. Y. Kung, *A new identification and model reduction algorithm via singular value decompositions*, in Proc. 12-th Asilomar Conf. Circ., Syst. Comp., November 1984, pp. 705–714.

[47] C. Lanczos, *An iteration method for the solution of the eigenvalue problem linear differential and integral operators*, J. Res. Natl. Bur. Stand., 45 (1950), pp. 255–282.

[48] A. J. Laub, *On computing "balancing" transformations*, in Proc. 1980 JACC, San Francisco, 1980, pp. FA8–E.

[49] ——, *Numerical linear algebra aspects of control design computations*, IEEE Trans. Auto. Contr., AC-30 (1985), pp. 97–108.

[50] A. J. Laub, M. T. Heath, C. C. Paige, and R. C. Ward, *Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms*, IEEE Trans. Auto. Contr., AC-32 (1987), pp. 115–122.

[51] B. C. Moore, *Principal component analysis in linear systems: Controllability, observability, and model reduction*, IEEE Trans. Auto. Contr., AC-26 (1981), pp. 17–31.

[52] C. C. Paige, *The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices*, PhD thesis, Univ. of London, 1971.

[53] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice Hall, 1980.

[54] ——, *Reduction to tridiagonal form and minimal realizations*, SIAM J. Matr. Anal., 13 (1992), pp. 567–593.

[55] 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.

[56] L. Reichel and D. Y. Hu, *Krylov subspace methods for the Sylvester equation*, Lin. Alg. & Appl., 172 (1992), pp. 283–314.

[57] Y. Saad, *On the rates of convergence of the Lanczos and the block Lanczos methods*, SIAM J. Num. Anal., 17 (1980), pp. 687–706.

[58] ——, *Krylov subspace methods for solving large unsymmetric linear systems*, Math. Comp., 37 (1981), pp. 105–126.

[59] ——, *Numerical solution of large Lyapunov equations*, in Signal Processing, Scattering and Operator Theory, and Numerical Methods. Proc. Intl. Symp. MTNS-89, vol. 3, Birkhäuser, 1990, pp. 503–511.

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

[61] D. Scott, *Analysis of the symmetric Lanczos process*, electronic res. lab. report UCB/ERL M78/40, Univ. of Calif., Berkeley, 1978.

[62] L. S. Shieh and Y. J. Wei, *A mixed method for multivariable system reduction*, IEEE Trans. Auto. Contr., AC-20 (1975), pp. 429–432.

[63] G. W. Stewart and J. G. Sun, *Matrix Perturbation Theory*, Academic Press, 1990.

[64] T. J. Su, *A Decentralized Linear Quadratic Control Design Method for Flexible Structures*, PhD thesis, Univ. of Texas, Austin, 1989.

[65] T. J. Su and R. R. Craig Jr., *Model reduction and control of flexible structures using Krylov vectors*, J. Guid., Contr., & Dyn., 14 (1991), pp. 260–267.

[66] L. N. Trefethen, *Rational Chebyshev approximation on the unit disk*, Numer. Math., 37 (1981), pp. 297–320.

[67] P. Van Dooren, *Numerical aspects of system and control problems*, Journal A, 30 (1987), pp. 25–32. Brussels.

[68] C. D. Villemagne and R. E. Skelton, *Model reduction using a projection formulation*, Intl. J. Contr., 46 (1987), pp. 2141–2169.

[69] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press, 1965.

[70] H. P. Zeiger and A. J. McEwen, *Approximate linear realizations of given dimensions via Ho's algorithm*, IEEE Trans. Auto. Contr., AC-19 (1974), p. 153.