Numerical Analysis Project
Manuscript NA-90-06

May 1990

The Nonsymmetric Lanczos Algorithm and Controllability

by

Daniel Boley
Gene Golub

Numerical Analysis Project
Computer Science Department
Stanford University
Stanford, California 94306

The Nonsymmetric Lanczos Algorithm and Controllability

Daniel Boley[1]
Gene Golub[2]

ABSTRACT

We give a brief description of a non-symmetric Lanczos algorithm that does not require strict bi-orthogonality among the generated vectors. We show how the vectors generated are algebraically related to "Controllable Space" and "Observable Space" for a related linear dynamical system. The algorithm described is particularly appropriate for large sparse systems.

1. Introduction

The Lanczos Algorithm was originally proposed by Lanczos [15] 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 [5] [17]. In that case, the convergence of the algorithm, when used to compute eigenvalues, has been extensively analyzed in [14] [16] [20] [21] [22, p270ff]. This algorithm is particularly suited for large sparse matrix problems. A block Lanczos analog has been studied and analyzed by Underwood (cf. Golub and Underwood [10], Cullum and Willoughby [5] and Parlett [17]). However, until recently, the nonsymmetric Lanczos Algorithm has received much less attention. Some recent computational exprience with this algorithm can be found in [4]. 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 [22, p388ff]. More recently, several modifications allowing the Lanczos process

[1]Computer Science Dept., University of Minnesota, Mimeapolis, MN 55455. E-mail: boley@cs.umn.edu. The research reported by this author was supported in part by NSF grant CCR-8813493.

[2]Computer Science Dept., Stanford University, Stanford, CA 94305. E-mail: golub@nanet.stanford.edu. The research reported by this author was supported in part by ARO grant DAAL03-90-G-0105 and in part by NSF grant DCR-8412314."

to continue after such breakdowns have been proposed by Parlett et al [19] and by Gutknecht [11]. The close connection between the modified Non-symmetric Lanczos Algorithm and orthogonal polynomials with respect to indefinite inner products is discussed by Golub and Gutknecht [8] and Boley et. al. [2]. Recently, Parlett [18] noticed the close relation between the Lanczos Algorithm and the controllability- observability structure of dynamical systems. In this paper, we show how the Lanczos Algorithm can be used to construct ***all*** of the subspaces associated with the Controllability/Observability Decomposition of a dynamical system. For example, we show that the coefficients generated in the course of the Lanczos process form the system matrix for the minimal realization of the dynamical system.

The Lanczos Algorithm [15] is an example of a method that generates bases for Krylov subspaces starting with a given vector. In a previous paper [3], we have examined how another closely related method, the Arnoldi Algorithm, may be used to compute the controllable space for a linear time-invariant dynamical system. The Amoldi Algorithm can be thought of as a “one-sided” method, which generates one sequence of vectors that span the controllable space. In this paper, we extend this idea to the use of a two-sided method, the non-symmetric Lanczos Algorithm’ which generates two sequences of vectors spanning the left and right Krylov spaces corresponding to the controllable and the observable spaces. We will demonstrate how the vectors are generated in such a way that we obtain bases not only for the left and right Krylov spaces, but also for the intersections of these spaces and the complementary spaces.

This paper is organized as follows. In Section 2, we give an general algorithmic description of the non-symmetric Lanczos Algorithm, including the modifications we have introduced to continue after a breakdown. Next we describe the connection between the items computed by the Lanczos algorithm and the controllability-observability properties for a linear dynamical system, showing that the minimal realization is computed directly by the Lanczos algorithm. We finish with some computational details, an illustrative numerical example, and some conclusions.

2. Description of the Lanczos Process

We give a brief description of the non-symmetric Lanczos process we have implemented. For clarity, we describe the algorithms at a level of detail appropriate for a MATLAB environment, omitting the specific methods used for the basic linear algebra computations.

We use the following notation, to keep the description concise. Vectors are represented by lower case bold letters ($\mathbf{b}$), matrices by upper case italic (B), and linear spaces by upper face bold ($\mathbf{B}$); all other typefaces are scalars or indices. The notation COLSP$[\mathbf{v}_0,\mathbf{v}_1,\dots]$ denotes the ***column space*** generated by the columns $\mathbf{v}_0,\mathbf{v}_1,\dots$. *If* $\mathbf{v}_i = A\mathbf{v}_{i-1}$ for all i, *so* that $\mathbf{v}_i = A^i\mathbf{v}_0$, *the* sequence of vectors $\mathbf{v}_0, \mathbf{v}_1, \ldots$ is called a ***Krylov sequence,*** and the space $\mathrm{COLSP}[\mathbf{v}_0, \mathbf{v}_1, \ldots]$ is called the right ***Krylov*** space K generated by the vector $\mathbf{v}_0$. We let $\mathbf{K}_i$ denote the truncated space generated by the **first** i + 1 vectors: $K_i \equiv [\mathbf{b}_0, A\mathbf{b}_0, \ldots, A^i\mathbf{b}_0]$. Likewise, we let L denote the ***left Krylov space*** $\mathrm{COLSP}[\mathbf{c}_0, A^{\mathrm{T}}\mathbf{c}_0, \ldots \cdot]$, and $\mathbf{L}_i$ the truncated space generated by $L_i \equiv [\mathbf{c}_0, A^{\mathrm{T}}\mathbf{c}_0, \ldots (A^{\mathrm{T}})^i\mathbf{c}_0]$.

Given an $n \times n$ real matrix A and two real, non-null, n-vectors $\mathbf{b}_0$, $\mathbf{c}_0$, the algorithm **generates two** sequences of vectors $B \equiv [\mathbf{b}_0, \mathbf{b}_1, \cdots] \equiv [B_0, \ldots, B_k]$ and $C \equiv [\mathbf{c}_0, \mathbf{c}_1, \cdots] \equiv [C_0, \ldots, C_k]$ grouped into clusters $B_l \equiv [\mathbf{b}_{j_{l1}+1}, \ldots, \mathbf{b}_{j_l}]$ and $C_l \equiv [\mathbf{c}_{j_{l-1}+1}, \ldots, \mathbf{c}_{j_l}]$, for $l = 1, 2, \cdots, k$. At the start, the clusters are all empty, and as each of the new vectors $\mathbf{b}_i$, $\mathbf{c}_i$ is generated, they are placed-either at the end of the latest cluster or in a new empty cluster, according to the following prescription.

At the i-th stage, the algorithm generates vectors $\mathbf{b}_i$, $\mathbf{c}_i$ as follows. Suppose at this stage we have k-l pairs of complete clusters $B_0, \ldots, B_{k-1}, C_0, \ldots, C_{k-1}$, and a last pair of incomplete clusters B_k, C_k (which may be empty), such that

$$\begin{aligned} &D_l \equiv C_l^{\mathrm{T}} B_l \text{ is nonsingular for } l = 0, \ldots, \text{k-l}, \\ &D_k \equiv C_k^{\mathrm{T}} B_k \text{ is singular or empty}, \\ &C_r^{\mathrm{T}} B_s = 0 \text{ for all } r \neq s. \end{aligned} \tag{1}$$

Then the next vector $\mathbf{b}_i$ is obtained by forming $A\,\mathbf{b}_{i-1}$, and adding multiples of all previous vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$ to enforce the bi-orthogonality condition

$$[C_0, \ldots, C_{k-1}]^{\mathrm{T}}\mathbf{b}_i = [\mathbf{c}_0, \ldots, \mathbf{c}_{j_{k-1}}]^{\mathrm{T}}\mathbf{b}_i = 0. \tag{2}$$

Similarly, $\mathbf{c}_i$ is obtained by forming $A^{\mathrm{T}}\mathbf{c}_{i-1}$ and then adding multiples of all the previous vectors $\mathbf{c}_0, \ldots, \mathbf{c}_{i-1}$ **to** enforce condition

$$[B_0, \ldots, B_{k-1}]^{\mathrm{T}}\mathbf{c}_i = [\mathbf{b}_0, \ldots, \mathbf{b}_{j_{k-1}}]^{\mathrm{T}}\mathbf{c}_i = 0. \tag{3}$$

Then the new vectors $\mathbf{b}_i$, $\mathbf{c}_i$ are appended to the (initially empty) clusters B_k, C_k. The question is whether the cluster pair B_k, C_k is complete in the sense that all subsequent vectors are to be placed in subsequent cluster, or if the cluster pair B_k, C_k is ***incomplete in the*** sense that at least one subsequent pair of vectors will be appended to this cluster pair. If $D_k \equiv C_k^{\mathrm{T}} B_k$ is **nonsingular,** then B_k, C_k ***are*** said to be complete in the above sense, and the vectors $\mathbf{b}_{i+1}$, $\mathbf{c}_{i+1}$ will be placed in the next cluster pair B_{k+1}, C_{k+1}. If D_k is singular, then the cluster pair B_k, C_k is kept "open" to accept at least the next pair of vectors $\mathbf{b}_{i+1}$, $\mathbf{c}_{i+1}$. In the algorithm description (Figure 1), the index k denotes the index of the last cluster pair accepting new vectors; as each cluster is *completed, the* index k is incremented.

For example, when we start' we have all clusters empty, and we fill the initially empty first clusters B_0 and C_0 with $\mathbf{b}_0$, $\mathbf{c}_0$, respectively. If $C_0^T B_0 = \mathbf{c}_0^T \mathbf{b}_0 \neq 0$, then the clusters B_0, C_0 *are* complete, each consisting of a single vector. The next vectors $\mathbf{b}_1$, $\mathbf{c}_1$ will go into the next cluster pair B_1, C_1, respectively. However, if $\mathbf{c}_0^T \mathbf{b}_0 = 0$, we append $\mathbf{b}_1, \mathbf{c}_1$ to clusters B_0, C_0.

The **algorithm** continues to generate vectors until both conditions $\mathbf{K}_i = \mathrm{K}$ and $\mathbf{L}_i = \mathrm{L}$ occur, in contrast to the usual criterion of stopping when at least one of the conditions occur. We summarize the process in Figure 1.

```
Non-Symmetric Lanczos Algorithm.
Input: $n x n$ matrix $A$ and two n-vectors $\mathbf{b}_0, \mathbf{c}_0$.
1. (* initialization *)
        Set all clusters $B_l, C_l$ (for all I) to "empty".
        Set first clusters $B_0 := [\mathbf{b}_0]$, $C_0 := [\mathbf{c}_0]$ (for all $l$) to "empty".
        If $\mathbf{c}_0^T\mathbf{b}_0 = 0$ then set $k := 0$, else set $k := 1$; (* cluster index *)
        Set $i := 0$; (* vector index *)
2. While $\mathbf{b}_i \neq 0$ or $\mathbf{c}_i \neq 0$ do begin
        (* main loop *)
        2.1  (* apply matrix operator to expand Krylov sequence *)
                Set $\hat{\mathbf{z}} := A\mathbf{b}_i$ and $\hat{\mathbf{y}} := A^T\mathbf{c}_i$.
        2.2. (* if incomplete cluster nonempty, orthogonalize within the cluster *)
                If cluster pair $B_k, C_k$ nonempty then
                        fmd coefficient vectors $\mathbf{h}_i, \mathbf{g}_i$ so that
                                $\mathbf{z} := \hat{\mathbf{z}} - B_k\mathbf{h}_i$ is orthogonal to $B_k$ and
                                $\mathbf{y} := \hat{\mathbf{y}} - C_k\mathbf{g}_i$ is orthogonal to $C_k$.
        2.3. (* bi-orthogonalize against previous clusters *)
                Find coefficient vectors $\mathbf{h}'_i, \mathbf{g}'_i$ such that
                        $\mathbf{b}_{i+1} := \mathbf{z} - [B_0, \ldots, B_{k-1}]\mathbf{h}'_i$ is orthogonal to $[C_0, \ldots, C_{k-1}]$ and
                        $\mathbf{c}_{i+1} := \mathbf{y} - [CO, \ldots, C_{k-1}]\mathbf{g}'_i$ is orthogonal to $[B_0, \ldots, B_{k-1}]$.
        2.4. (* append latest vectors to latest cluster. *)
                Set $B_k := [B_k, \mathbf{b}_{i+1}]$
                Set $C_k := [C_k, \mathbf{c}_{i+1}]$
        2.5. (* if latest cluster complete, mark it so *)
                If $C_k^T B_k$ is non-singular, then
                        Set $k := k + 1$.
        2.6. Set $i := i + 1$.
        End While Loop.
```

Figure 1. Modified Non-symmetric Lanczos Algorithm.

The 2 vector sequences that result: $\mathbf{b}_0, \mathbf{b}_1, \ldots$ and $\mathbf{c}_0, \mathbf{c}_1, \ldots$ generate the right and left truncated Krylov spaces, respectively:

$$\mathrm{COLSP}[\mathbf{b}_0, \ldots, \mathbf{b}_i] = \mathbf{K}_i, \text{ for all } i, \tag{4}$$

$$\mathrm{COLSP}[\mathbf{c}_0, \ldots, \mathbf{c}_i] = \mathbf{L}_i \text{ for all } i. \tag{5}$$

The vectors satisfy the bi-orthogonality conditions

$$[\mathbf{c}_0, \ldots, \mathbf{c}_j]^T[\mathbf{b}_{j+1}, \cdots] = 0 \text{ and } [\mathbf{b}_0, \ldots, \mathbf{b}_j]^T[\mathbf{c}_{j+1}, \cdots] = 0 \tag{6}$$

for any j for which it is possible, i.e. for any j for which

$$\text{de\&},\ldots, \mathbf{c}_j]^T[\mathbf{b}_0, \ldots, \mathbf{b}_j] \neq 0 \text{ or equivalently } \det L_j^T K_j \neq 0. \tag{7}$$

Let us denote the indices j for which (7) holds by j_l, $l = 0,1,\ldots$ k-l, and set $j_{-1} \equiv -1$. Then the generated clusters will be delimited by the indices j_l: $B_l \equiv [\mathbf{b}_{j_{l-1}+1}, \ldots, \mathbf{b}_{j_l}]$ and $C_m \equiv [\mathbf{c}_{j_{m-1}+1}, \ldots, \mathbf{c}_{j_m}]$. In terms of these clusters, (6) can be written more simply as a bi-orthogonality condition between clusters:

$$C_m^T B_l = 0 \text{ for all } l,m, \ m \neq l. \tag{8}$$

It also follows from (7) that

$$D_l \equiv C_l^T B_l \text{ is non-singular for } l = 0, \ldots, \text{k-l}. \tag{9}$$

In the classical description of the Algorithm [15] [22, p388ff], (7) usually holds for every j, so that the bi-orthogonality conditions (6) reduce to the simple set of conditions $\mathbf{c}_i^T \mathbf{b}_j = 0$, for all i,j, $i \neq j$. In this case, the clusters are just single vectors, and the algorithm described reduces to the non-symmetric Lanczos algorithm as described in, far example, [22, p388ff].

We note that there are several choices for the stopping condition in step 2. In the past, when this algorithm was used for the eigenproblem, the process was continued until $\mathbf{b}_r = 0$ for some r, or $\mathbf{c}_s = 0$ for some s, which ever occurred first. But in our situation it is useful to continue until both conditions occur, in which case we may have a sequence of zero vectors: $0 = \mathbf{b}_r = \mathbf{b}_{r+1} = \cdots = \mathbf{b}_s$ or $0 = \mathbf{c}_s = \mathbf{c}_{s+1} = \cdots = c,$, *if l >m or km,* respectively. We let $p \equiv \max\{r,s\}$ be the index of the last vector generated.

The resulting vectors generated from this algorithm will satisfy certain important properties that we mention. Let $B = [B_0, \ldots, B_k] = [\mathbf{b}_0, \ldots \mathbf{b}_p]$ and $C = [C_0, \ldots C_k] = [\mathbf{c}_0, \ldots \mathbf{c}_p]$ be the matrices of all the vectors generated, where $p \equiv j_k$ is the index of the last vector generated.

The vector $\mathbf{b}_{i+1}$ in step 2.3 of the algorithm is a linear combination of $A\mathbf{b}_i$ and previous vectors $\mathbf{b}_r$, $r \le i$. Thus the matrix B of generated vectors satisfies

$$AB = BH, \tag{10}$$

where H is a unit upper Hessenberg matrix consisting of all the coefficients $\mathbf{h}_i$, $\mathbf{h}'_i$. Likewise, the matrix C satisfies

$$A^T C = CG, \tag{11}$$

where G is a unit upper Hessenberg matrix, consisting of all the coefficients $\mathbf{g}_i$, $\mathbf{g}'_i$. That is, the *i-th* columns of H and G arc, respectively:

$$H_{\cdot i} = \begin{bmatrix} \mathbf{h}_i' \\ \mathbf{h}_i \\ 1 \\ \mathbf{0} \end{bmatrix} \text{ and } G_{\cdot i} = \begin{bmatrix} \mathbf{g}_i' \\ \mathbf{g}_i \\ 1 \\ 0 \end{bmatrix}$$

where each "1" entry above occupies the i + 1-th position, lying on the sub-diagonal of H and G, respectively, for $i = 0, 1, \ldots, p$. The bi-orthogonal@ conditions (6) (7) become

$$C^T B = D \text{ , a block diagonal matrix with diagonal blocks } D_k \equiv C_k^T B_k. \tag{12}$$

Since $C^T AB = C^T BH = DH$, and $B^T A^T C = B^T CG = D^T G$, we have the relation $G^T D^T = DH$. Since a block diagonal matrix times a upper Hessenberg matrix is block upper Hessenberg, it follows that G and H are block tridiagonal, with the partitioning defined by the cluster dimensions. **This** implies that in computing the coefficients $\mathbf{h}_i$, $\mathbf{g}_i$ or $\mathbf{h}'_i$, $\mathbf{g}'_i$ at each stage, only the last two pairs of clusters B_{k-1}, C_{k-1}, and B_k, C_k, must be used, at least in exact arithmetic. We will discuss below the effect of using approximate floating point arithmetic.

3. Controllability and Observability

We discuss an application of this modified Lanczos process arising from the context of Dynamical Systems Theory. The concepts of controllability and observability are fundamental concepts in Systems Theory, extensively analyzed in standard textbooks (see e.g. [6] [12]). For our purposes, it suffices to state purely algebraic definitions for the relevant spaces: the ***controllable space*** (more correctly ***called the reachable space)*** and ***the unobservable space.***

Consider the SISO (Single Input Single Output) dynamical system

$$\dot{\mathbf{x}} = A\mathbf{x} + \mathbf{b}_0 u \; ; y = \mathbf{c}_0^{\mathrm{T}}\mathbf{x}. \tag{13}$$

Here x is an n-vector function of time t, u, y are scalar functions of t, A is an $n \times n$ constant matrix, and $\mathbf{b}_0$, $\mathbf{c}_0$ **are** constant ***n-vectors.*** From classical systems theory **[6],** it is well known that the ***controllable*** space $\mathbf{S}_c$ can be defined algebraically as the Krylov space K, and the ***unobservable*** space $\mathbf{S}_{\bar{o}}$ is the orthogonal complement to the **Krylov** space L. It is known that, algebraically at least, the observable space $\mathbf{S}_o$ is some space that is complementary to S,, but is not uniquely defined **[6].** In fact, we have the following proposition from **[1], [6]:**

Proposition 1. A complete Controllability-Observability **(Kalman-Gilbert)** decomposition **[13] [7]** is obtained for the system (13) if one has the 4 sets of columns $T_{c\bar{o}}$, T_{co}, $T_{\bar{c}\bar{o}}$, $T_{\bar{c}o}$, such that $[T_{c\bar{o}}, T_{co}]$ forms a basis for the ***controllable space*** $\mathbf{S}_c$, $[T_{c\bar{o}}, T_{\bar{c}\bar{o}}]$ forms a basis for the ***unobservable space*** S,, $T_{c\bar{o}}$ forms a basis for $\mathbf{S}_{c\bar{o}} \equiv \mathbf{S}_c \cap \mathbf{S}_{\bar{o}}$, and $T \equiv [T_{\bar{c}o}, T_{\bar{c}\bar{o}}, T_{co}, T_{c\bar{o}}]$ is a square **non**singular matrix. Applying ***the*** transformation T to ***the*** system (13) yields ***the Controllability-Observability Canonical Form*** **(COCF):**

$$\dot{\tilde{\mathbf{x}}} = \tilde{A}\tilde{\mathbf{x}} + \tilde{\mathbf{b}}_0 u \; ; y = \tilde{\mathbf{c}}_0^{\mathrm{T}}\tilde{\mathbf{x}}, \tag{14}$$

where the new coefficients will have the following special structure:

$$\tilde{A} = T^{-1}AT = \begin{bmatrix} \tilde{A}_{11} & 0 & 0 & 0 \\ \tilde{A}_{21} & \tilde{A}_{22} & 0 & 0 \\ \tilde{A}_{31} & 0 & \tilde{A}_{33} & 0 \\ \tilde{A}_{41} & \tilde{A}_{42} & \tilde{A}_{43} & \tilde{A}_{44} \end{bmatrix}, \tilde{\mathbf{b}}_0 = T^{-1}\mathbf{b}_0 = \begin{bmatrix} 0 \\ 0 \\ \tilde{\mathbf{b}}_{30} \\ \tilde{\mathbf{b}}_{40} \end{bmatrix}, \tilde{\mathbf{c}}_0 = T^{\mathrm{T}}\mathbf{c}_0 = \begin{bmatrix} \tilde{\mathbf{c}}_{10} \\ 0 \\ \tilde{\mathbf{c}}_{30} \\ 0 \end{bmatrix}. \tag{15}$$

$\square$

Note that the eigenvalues of $\tilde{A}$ are the same as A, so that to compute the eigenvalues of $\tilde{A}$, one need only compute the eigenvalues of $\tilde{A}_{ii}$, $i = 1, \ldots, 4$. The eigenvalues of each block $\tilde{A}_{ii}$ may be computed independently of any other block Note also that the ordering of the blocks of T in Proposition 2 is not the standard one, but with this ordering, it will be seen that the three blocks $T_{c\bar{o}}$, T_{co}, $T_{\bar{c}o}$ are obtained directly from the generated Lanczos vectors. The fourth block $T_{\bar{c}\bar{o}}$ ***can be*** obtained by finding a basis for the space of vectors in $\mathbf{S}_{\bar{o}}$ orthogonal to $\mathbf{S}_{c\bar{o}}$, both of which are produced by our implementation of the Lanczos algorithm.

Assume we apply the modified Lanczos algorithm, starting with matrix A and starting vectors $\mathbf{b}_0$, $\mathbf{c}_0$, and generating vectors $\mathbf{b}_0, \ldots, \mathbf{b}_p$, $\mathbf{c}_0, \ldots, \mathbf{c}_p$, grouped into clusters $B_0, \ldots, B_k$, $C_0, \ldots, C_k$, where the last, incomplete, pair of clusters B_k, C_k may or may not be empty. **Then** all the vectors $\mathbf{b}_0, \ldots, \mathbf{b}_p$ will span $\mathbf{S}_c$. We examine how the individual B clusters form bases for relevant parts of S,.

Let B_l he any cluster (possibly incomplete) generated from the Lanczos process, and let $\mathbf{b}_i = \mathbf{b}_{j_{l-1}+1}$ denote the first vector within that cluster. We consider two choices: either $\mathbf{b}_i$ lies within $\mathbf{S}_{c\bar{o}}$ or it does not. This is equivalent to asking whether $\mathbf{b}_i$ is orthogonal to L or not.

Consider the case $\mathbf{b}_i$ lies within $\mathbf{S}_{\bar{o}}$. Since $\mathbf{S}_{\bar{o}}$ is an invariant subspace of the matrix A, *we* have that $A^r\mathbf{b}_i$ will lie within $\mathbf{S}_{\bar{o}}$ for any $r \geq 0$. This means that every subsequent vector $\mathbf{b}_{i+r}$ will he orthogonal to L, ***without any*** need to enforce this condition in step 2.3 (hi-orthogonalization) of the Lanczos Algorithm. Thus the entire cluster B_l started by $\mathbf{b}_i$ will lie in S,, and in particular $C_l^{\mathrm{T}}B_l = 0$. Hence this cluster must he the last cluster B_k and must remain incomplete. That is, $l = k$ and j_{k-1} is the largest value of j for which (7) holds. Thus B_k must lie within $\mathbf{S}_{c\bar{o}}$, and we will show below that no other b vector in any previous cluster can lie within $\mathbf{S}_{c\bar{o}}$ That is, B_k ***will be*** exactly $T_{c\bar{o}}$ of Proposition 2. By analogy, we also have that C_l is not in $\mathbf{S}_{\bar{o}}$ and is orthogonal to K = S,, and so will he exactly $T_{\bar{c}o}$.

Now consider ***the case*** $\mathbf{b}_i$ is not ***in S,,*** i.e. not orthogonal to L = ***span*** $[\mathbf{c}_0, \ldots, \mathbf{c}_p]$. Since $\mathbf{b}_i$ is the first vector in the cluster B_l, *we* have the two relations regarding all the previous vectors

$$[\mathbf{c}_0, \ldots, \mathbf{c}_{i-1}]^{\mathrm{T}}\mathbf{b}_i = [C_0, \ldots, C_{l-1}]^{\mathrm{T}}\mathbf{b}_i = 0, \tag{16}$$

and

$$[\mathbf{c}_0, \ldots, \mathbf{c}_{i-1}]^{\mathrm{T}}[\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}] = [C_0, \ldots, C_{l-1}]^{\mathrm{T}}[B_0, \ldots, B_{l-1}] \text{ is nonsingular.} \tag{17}$$

Since $\mathbf{b}_i$ is not orthogonal to L, we can find the fiit index r such that $((A^{\mathrm{T}})^r\mathbf{c}_0)^{\mathrm{T}}\mathbf{b}_i \neq 0$. By (16), $r \geq i$. This implies that the matrix

$$[(A^{\mathrm{T}})^i\mathbf{c}_0, \ldots, (A^{\mathrm{T}})^r\mathbf{c}_0]^{\mathrm{T}}[\mathbf{b}_i, \ldots, A^{r-i}\mathbf{b}_i] \tag{18}$$

is nonzero on its anti-diagonal and zero above its anti-diagonal, and hence is nonsingular. (We remark that the ***anti-diagonal*** is the diagonal running from the lower left to the upper right of the ***matrix,*** and ***the*** property of being ***all zero above the anti-diagonal*** is called lower ***anti-triangular***.) This in turn implies that the matrix $L_r^{\mathrm{T}}K_r$ is non-singular, which means that the Lanczos process will continue at least through the ***r-th*** vector. The vectors $\mathbf{b}_i, \ldots, \mathbf{b}_r$ and $\mathbf{c}_i, \ldots, \mathbf{c}_r$ will form complete clusters B_l, C_l, respectively, with $D_l = C_l^{\mathrm{T}}B_l$ nonsingular. In fact,

D_l *will* have the same lower anti-triangular structure as (18). Note that no vector in B_l is orthogonal to L; that is, no vector in any complete cluster $B_l, l = 0, \ldots,$ k-l lies in $\mathbf{S}_{c\bar{o}}$.

Thus, if $\mathbf{S}_{c\bar{o}}$ is empty, all the clusters will be complete, but if the space $\mathbf{S}_{c\bar{o}}$ is not empty, there must be a final incomplete cluster B_k whose columns span this space. By carrying out similar reasoning for the C clusters, we **find** that the final incomplete cluster C_k (if any) spans the space of all vectors in L that are orthogonal to K. Thus C_k corresponds to the space S,.

We summarize the results of the above discussion in the following proposition:

Proposition 2. Assume the Non-symmetric **Lanczos** Algorithm is applied to the SISO system **(13),** starting with the system matrix A and vectors $\mathbf{b}_0$, $\mathbf{c}_0$, and the result consists of k-l complete clusters $B_0, \ldots, B_{k-1}, C_0, \ldots, C_{k-1}$, and a **final** (possibly empty) incomplete cluster B_k, and C_k (with any zero vectors deleted), satisfying (1). Then the individual blocks of the matrix T in Proposition 2 yielding the COCF will be defined as follows: $T_{c\bar{o}} = B_k$ will be a basis for the **space** $\mathbf{S}_{c\bar{o}}$, $T_{co} = [B_0, \ldots, B_{k-1}]$, $T_{\bar{c}o} = C_k$, and $T_{\bar{c}\bar{o}}$ **will** *be* a basis for the space orthogonal to $[C_0, \ldots, C_k, B_k]$. []

Hence, defining *T as in* Proposition 2, we can express the various parts of the COCF (15) in terms of the vectors and coefficients generated by the Non-symmetric Lanczos Algorithm (10) (11) (12). From (10) we have $A \cdot [T_{co}, T_{c\bar{o}}] = [T_{co}, T_{c\bar{o}}] \cdot H$, which yields the following structure for *H:*

$$H = \begin{bmatrix} H_{j_{k-1}} & 0 \\ E_{1,j_{k-1}} & H_k \end{bmatrix} = \begin{bmatrix} \tilde{A}_{33} & 0 \\ \tilde{A}_{43} & \tilde{A}_{44} \end{bmatrix}, \text{ where } E_{1,j_{k-1}} = \begin{bmatrix} & 1 \\ 0 & \end{bmatrix}, \tag{19}$$

and $H_{j_{k-1}}$ represents the top left $j_{k-1} \times j_{k-1}$ **part of** H, **corresponding to** $[B_0, \ldots, B_{k-1}]$.

Again referring to **(15),** Proposition 2 yields the following form:

$$\mathbf{b}_{30} = \mathbf{e}_1 = [1,0, \ldots, 0]^{\mathrm{T}}, \text{ and } \tilde{\mathbf{b}}_{40} = \mathbf{0}. \tag{20}$$

We have as **well** that $\tilde{\mathbf{c}}_{10}^{\mathrm{T}} = \mathbf{c}_0^{\mathrm{T}} C_k$ and $\tilde{\mathbf{c}}_{30}^{\mathrm{T}} = \mathbf{c}_0^{\mathrm{T}} [B_0, \ldots, B_{k-1}] = [d_{00}, d_{01}, \ldots, d_{0,j_0}, 0, \ldots, 0]$, **where** $[d_{00}, d_{01}, \ldots, d_{0,j_0}]$ is the first row of the matrix block $D_0 = C_0^{\mathrm{T}} B_0$. Since the blocks D_l **are lower anti-triangular,** $d_{00} = d_{01} = \cdots = d_{0,j_0 - 1} = 0$, so that

$$\tilde{\mathbf{c}}_{30} = d_{0,j_0}\mathbf{e}_{j_0}. \tag{21}$$

The most important part of the dynamical system (13) is the "controllable-observable" part **($\tilde{A}_{33}$),** which yields ***the minimal realization*** **[6]** ***[*****12]:**

$$\dot{\tilde{\mathbf{x}}}_3 = H_{j_{k-1}}\tilde{\mathbf{x}}_3 + \mathbf{e}_1 u \ ; \ y = d_{1,j_1}\mathbf{e}_{j_1}^{\mathrm{T}}\tilde{\mathbf{x}}_3. \tag{22}$$

All the coefficients in the minimal realization (22) can be expressed directly in terms of the coefficients generated by the Lanczos Algorithm. Note that (22) is essentially equivalent to the expression for the minimal realization given by Theorem 7.1 in [**18].** In that Theorem 7.1, the system matrix for the minimal realization is expressed as $H_{j_{k-1}} = \Omega^{-1}\hat{T}$, where $\Omega = [C_0, \ldots, C_{k-1}]^{\mathrm{T}}[B_0, \ldots, B_{k-1}]$ **is the upper left** $j_{k-1}\times j_{k-1}$ part of D, and $\hat{T} = [C_0, \ldots, C_{k-1}]^{\mathrm{T}}A\,[B_0, \ldots, B_{k-1}]$. However, by saving the coefficients ***H, we*** also obtain expressions for the complete COCF, as indicated by Proposition 2 above.

4. Computational Details

In order to obtain a numerically stable algorithm, it is necessary to examine the effect of using finite-precision floating point arithmetic. We outline the principal points in the algorithm where variations should be made to accommodate the use of floating point arithmetic.

In spite of the fact that **bi-orthogonality** conditions (6) (7) hold exactly, computational experience shows that in the approximate floating point arithmetic encountered on digital computers, it is necessary to re-bi-orthogonalize the b, c vectors as they are generated against all previous clusters in order to maintain the bi-orthogonality conditions numerically. Otherwise, numerical cancellation may result in the vectors losing not only orthogonality, but also linear independence.

In step 2.5, we must determine if a cluster is complete by computing the rank of $D_k = C_k^{\mathrm{T}}B_k$. ***The*** **vectors** $\mathbf{b}_{i+1}, \mathbf{c}_{i+1}$ have not been scaled in any way to be close to unit norm in step 2.3, so it is possible that the norms of these vectors will vary widely, thus making the rank determination **difficult.** To substantially reduce this dif'culty, it is useful to **rescale** the vectors as they are formed by slightly modifying step 2.3 as in Figure 2. Then the coefficient matrices ***H,*** G will still be upper Hessenberg and block tridiagonal as before, but the subdiagonal elements will no longer be all 1 s, but rather the **β_i's** and **γ_i's,** respectively. With this modification, the rank determination can be made in a very robust way by using the singular value decomposition **[9,** pp **71, 427ff].**

We remark that if $\mathbf{c}_i^T\mathbf{b}_i \neq 0$ for every i, so that every cluster consists of exactly one vector, then the matrices C^TAB, H, G, will all be tridiagonal. If we rescale the vectors so that $\mathbf{c}_i^T\mathbf{b}_i = 1$, then D will be the Identity matrix, and $C^TAB = H =$ G will be the tridiagonal matrix usually computed by the classical Lanczos algorithm.

In step 2.2, the vector $\mathbf{h}_i$ is obtained by finding the orthogonal projection of $A\mathbf{b}_i$ onto the the span of the vectors currently in B_k, and thus the vector z is the orthogonal projection onto the orthogonal complement of this space. This is a form of a standard linear least squares problem discussed in [9, p217], and for which standard methods exist in LINPACK. Similar comments can be said for the computation of $\mathbf{g}_i$ and $\mathbf{y}$.

In step 2.3, we must compute the oblique projection of the vector z, to satisfy the bi-orthogonality condition (2). That is, the vector $\mathbf{h}'_i$ must satisfy the equation

$$0 = [C_0, \ldots, C_{k-1}]^T \mathbf{b}_{i+1} = [C_0, \ldots, C_{k-1}]^T \left(\mathbf{z} - [B_0, \ldots, B_{k-1}]\mathbf{h}'_i\right).$$

Hence $\mathbf{h}'_i$ is

$$\mathbf{h}'_i = \left([C_0, \ldots, C_{k-1}]^T [B_0, \ldots, B_{k-1}]\right)^{-1} [C_0, \ldots, C_{k-1}]^T \mathbf{z}.$$

Likewise, $\mathbf{g}'_i$ is

$$\mathbf{g}'_i = \left([B_0, \ldots, B_{k-1}]^T [C_0, \ldots, C_{k-1}]\right)^{-1} [B_0, \ldots, B_{k-1}]^T \mathbf{y}.$$

Note that the coefficient matrix $\left([B_0, \ldots, B_{k-1}]^T [C_0, \ldots, C_{k-1}]\right)$ appears in the formulas for both $\mathbf{h}'_i$ and $\mathbf{g}'_i$, and is block diagonal, so that the coefficients for each cluster can be computed independently of one another, in parallel.

We note that the above formulas for the $\mathbf{h}'_i$, $\mathbf{g}'_i$ are as numerically appealing as those for the orthogonal projections in step' 2.2, but -since these are oblique projections, we are not guaranteed that this matrix is well-conditioned.

```
2.3. (* bi-orthogonalize against previous clusters *)
        Find coefficient vectors $\mathbf{h}'_i$, $\mathbf{g}'_i$ and scalars $\beta_i$, $\gamma_i$ such that
            $\beta_i\mathbf{b}_{i+1} := \mathbf{z} - [B_0, \ldots, B_{k-1}]\mathbf{h}'_i$ is orthogonal to $[C_0, \ldots, C_{k-1}]$,
            $\gamma_i\mathbf{c}_{i+1} := \mathbf{y} - [C_0, \ldots, C_{k-1}]\mathbf{g}'_i$ is orthogonal to $[B_0, \ldots, B_{k-1}]$,
            $\|\mathbf{b}_{i+1}\| = \|\mathbf{c}_{i+1}\| = 1$.
```

Figure 2. Modified Step 2.3 with vectors re-scaled.

5. Illustrative Example

We illustrate the algorithm with a small numerical example for which all four blocks $T_{\bar{c}o}, T_{\bar{c}\bar{o}}, T_{co}, T_{c\bar{o}}$ are nonempty. We start with the system (13) where

$$A = \begin{bmatrix} 1.507 & 0.880 & -1.760 & -0.476 & -0.335 \\ 1.324 & 1.435 & 2.321 & -2.483 & -2.352 \\ -1.818 & 0.938 & 0.803 & -0.518 & -0.266 \\ 0.211 & -1.444 & 0.151 & -0.280 & 1.539 \\ -0.130 & 0.947 & 0.710 & -0.462 & -0.465 \end{bmatrix}, \quad \mathbf{b}_0 = \begin{bmatrix} -4.521 \\ 2.294 \\ -0.818 \\ 0.695 \\ 0.380 \end{bmatrix}, \quad \mathbf{c}_0 = \begin{bmatrix} -0.870 \\ 0.037 \\ 4.359 \\ 0.758 \\ -2.582 \end{bmatrix}.$$

Note that $\mathbf{c}_0^T \mathbf{b}_0 = 0$, so that it is immediately seen that the first clusters B_0, C_0 will have at least 2 vectors at the conclusion of the Lanczos process. After the Lanczos algorithm is applied, we obtain the right vectors $\boldsymbol{B}$, coefficients $\boldsymbol{H}$ and left vectors C, with the zero vectors deleted and with the clusters marked:

$$B = \left[\begin{array}{cc|c} -4.521 & -0.411 & -0.195 \\ 2.294 & -0.498 & 0.653 \\ -0.818 & 0.693 & -0.228 \\ 0.695 & -0.291 & 0.688 \\ 0.380 & 0.140 & -0.107 \end{array}\right], H = \left[\begin{array}{cc|c} -0.333 & 0.171 & 0 \\ 12.961 & 2.333 & 0 \\ - & - & \\ \hline 0 & 2.315 & -2.000 \end{array}\right], \text{c} = \left[\begin{array}{cc|c} -0.870 & -0.918 & 0.019 \\ 0.037 & -0.022 & -0.292 \\ 4.359 & 0.056 & -0.230 \\ 0.758 & -0.166 & 0.341 \\ -2.582 & 0.355 & 0.864 \end{array}\right].$$

We have two cluster pairs B_0, B_2 and C_0, C_2, where B_2, C_2 consist of the single last vector of $\boldsymbol{B}$, C, respectively. Note $D_2 = C_2^T B_2 = 0$, ***so*** B_2, C_2 ***are*** incomplete, and $j_{k-1} = 2$. We form the matrix that yields the **Kalman** Canonical Decomposition:

$$T = [T_{\bar{c}o}, T_{\bar{c}\bar{o}}, T_{co}, T_{c\bar{o}}] = \left[\begin{array}{c|c|cc|c} 0.019 & 0.252 & & -0.411 & \\ -0.292 & 0.666 & -4.521\ 2.294 & -0.498 & -0.195\ 0.653 \\ -0.230 & 0.383 & -0.818 & 0.693 & -0.228 \\ 0.341 & -0.362 & 0.695 & -0.291 & 0.688 \\ 0.864 & 0.464 & 0.380 & 0.140 & -0.107 \end{array}\right]$$

When we apply $\boldsymbol{T}$ to (13) we obtain the Canonical Form (15):

$$\tilde{A} = T^{-1}AT = \left[\begin{array}{c|c|cc|c} 1.000 & 0 & 0 & 0 & 0 \\ \hline -4.000 & 2.000 & 0 & 0 & 0 \\ \hline -0.278 & 0 & -0.333 & 0.171 & 0 \\ 1.080 & 0 & 12.961 & 2.333 & 0 \\ \hline 0.500 & 1.000 & 0 & 2.315 & -2.000 \end{array}\right], \quad \tilde{\mathbf{b}}_0 = T^{-1}\mathbf{b}_0 = \left[\begin{array}{c} 0 \\ \hline 0 \\ \hline 1 \\ 0 \\ \hline 0 \end{array}\right]$$

and

$$\tilde{\mathbf{c}}_0^T = \mathbf{c}_0^T T = \left[\begin{array}{c|c|c|c} -3.000 & 0 & 0\,,\,2.777 & 0 \end{array}\right].$$

Note how H appears in the lower right part of $\tilde{A}$.

From the above canonical form, we find that the minimal realization for this system is

$$\dot{\mathbf{x}} = \begin{bmatrix} -0.333 & 0.171 \\ 12.961 & 2.333 \end{bmatrix}\mathbf{x} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}u; \; y = \begin{bmatrix} 0\,,\,2.777 \end{bmatrix}\mathbf{x}.$$

6. Conclusions

We have briefly described a modified two-sided non-symmetric Lanczos process that does not suffer from an incurable breakdown, and showed how the vectors generated from this process yield bases for controllable-observable, controllable-unobservable and uncontrollable-observable spaces for a linear time-invariant dynamical system (13). The coeffkients generated by the Lanczos Algorithm yield directly the minimal realization for such a dynamical system. The efficiency of this method for large scale problems remains to be investigated, but based on the experience with the eigenvalue problem, we believe that these ideas form a basis for further development in this subject.

Acknowledgements

We wish to thank Sylvan Elhay and Martin Gutknecht for many illuminating discussions and insights into the algorithms described in this paper.

REFERENCES

[1] D.L. Boley: "Computing the Kalman Decomposition, *an Optimal* Method"; *IEEE Trans. Auto. Contr* vol. AC-29 no. 1, pp 5 l-53,1984.

[2] D.L. Boley, S. Elhay, G.H. Golub, M. Gutknecht: The Lanczos algorithm and reconstructing indefinite weights; in preparation, 1990.

[3] D.L. Boley, G.H. Golub: The Lanczos Algorithm and Controllability; System ***and Control Letters, vol. 4*** no. 6, pp 317-324, Sept 1984.

[4] J. Cullum, W. **Kerner,** R. Willoughby: A generalized nonsymmetric Lanczos procedure; ***Computer Physics Communications,*** **vol 53,** pp **19-48, 1989.**

[5] J. Cullum, R. Willoughby: ***Lanczos Algorithms for Large Symmetric Eigenvalue Computa-*** tions, ***vol*** I ***Theory,*** Birkhauser Boston, 1985.

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

[7] E.G. Gilbert: Controllability and observability in multivariable control systems; ***SIAM J. Control*** vol 1 pp **128-151, 1963.**

[8] G. Golub, M. Gutknecht: Modified Moments for Indefinite Weight Functions; Stanford Univ. Numerical Analysis Project report NA-89-08, July 1989.

[9] **G.H.Golub,** C. Van Loan: ***Matrix Computations,*** 2nd edition; Johns Hopkins, 1989.

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

[11] M. Gutknecht: A Completed Theory for the Lanczos Algorithm; preprint submitted to ***SIAM J. Matrix Anal.,*** 1989.

[12] T Kailath: ***Linear Systems;*** Prentice Hall, 1980.

[13] R.E. **Kalman:** Mathematical description of linear dynamical systems; ***SIAM J. Control,*** vol 1, pp **152-192, 1963.**

[14] S. Kaniel: "estimates for some computational techniques in linear algebra"; ***Math. Comp. 20,*** pp **369-378, 1966.**

[15] c. Lanczos: "An iteration method for the solution of the eigenvalue problem linear differential and integral operators"; ***J. Res. Natl. Bur. Stand. 45,*** pp **255-282, 1950.**

[16] ***C.C.*** **Paige:** ***The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices;*** Ph.D. Thesis, London Univ., 197 1.

[17] B. Parlett: ***The Symmetric Eigenvalue Problem;*** Prentice Hall, 1980.

[18] B.N. Parlett: Reduction to Tridiagonal Form and Minimal Realizations; preprint submitted to ***SIAM J. Matrix Anal.,*** 1990.

[19] B.N. Parlett, D.R. Taylor, Z-S Liu: A look-ahead Lanczos Algorithm for unsymmetric ***matrices; Math. Comp.*** **vol 44,** pp **105-124, 1985.**

[20] Y. **Saad:** “On the rates of Convergence of the Lanczos and the block Lanczos methods”; ***SIAM J. Num. And. 17,*** pp **687-706, 1980.**

[21] D. Scott: “Analysis of the symmetric Lanczos process”; Univ. of **Calif.,** Berkeley, Electronic Res. Lab. report **UCB/ERL M78/40, 1978.**

[22] J.H. Wilkinson: ***The Algebraic Eigenvalue Problem;*** Clarendon Press, Oxford, 1965.