A Computational Algebra for Geometric Concepts in Linear System Theory

Bengtsson, Gunnar

1975

Document Version:
Publisher's PDF, also known as Version of record

A COMPUTATIONAL ALGEBRA FOR GEOMETRIC CONCEPTS IN LINEAR SYSTEM THEORY

GUNNAR BENGTSSON

Report 7501 Januari 1975
Lund Institute of Technology
Department of Automatic Control

A COMPUTATIONAL ALGEBRA FOR GEOMETRIC CONCEPTS IN LINEAR SYSTEM THEORY

Gunnar Bengtsson

ABSTRACT

Abstract algebraic concepts appearing in the geometric state space theory are transformed to a computable form. Two basic algorithms consisting of a gaussian procedure and a rearrangement of matrix elements are used. Algebraic expressions and sequences are broken down inte elementary steps in which these two algorithms can be applied. In this way computational algorithms are obtained for (A,B)-invariant subspaces, controllability subspaces, system zeros and minimal system inverses.

This work has been partially supported by the Swedish Board for Technical Development under Contract No. 733553.

TABLE OF CONTENTS

1. INTRODUCTION.

The control theory for linear multivariable systems has during the last few years developed from a mainly matrix-based theory towards a more general algebraic theory [5, 8,12]. Practical use of this theory must be based upon computer-aided design techniques [1,4,9] which permit the designer to utilize the results from theory without knowing the whole algebraic background. Such techniques require fast and accurate algorithms. The algorithmic aspect of the theory is therefore important in practice.

The geometric state space theory, based upon linear maps and vector spaces, was introduced by Wonham and Morse [6,11]. This theory has later been used to develop a rather extensive theory for linear systems [2,3,6,7,10-15] . The results are, however, not always given on a form which directly can be used for computations. A rather typical problem is to evaluate a sequence of the form

$$V_0 = \mathrm{Ker}(C)$$
$$V_{i+1} = V_i \cap A^{-1}(V_i + B) \tag{1.1}$$

where V_i, B denote subspaces and A and C linear maps. The purpose of this report is to transform such sequences into a suitable computable form.

A linear subspace will be represented by a basis on a certain form, denoted unity form. The advantage of this representation is that the previous computations in a sequential procedure like (1.1) can be used to minimize the amount of computation in the next step. Moreover, such problems as orthogonalization, solving linear equations and taking inverses become only a matter of rearrangement of matrix elements.

Two basic algoritms are used in the report. The transformation algoritm (TA) computes a unity basis for a given set of vectors and solves simultaneously a linear equation using a gaussian procedure. The orthogonalization algoritm (OA) computes a unity basis for the orthogonal complement of a given subspace. All the algebraic expressions and sequences are broken down into elementary steps where these two algorithms can be applied. In this way computational algorithms are obtained for invariant subspaces, controllability subspaces, system zeros and minimal system inverses.

The report is organized as follows. In Section 1 some notations are given. Two basic algorithms are presented in Section 2. The concept of (A,B)-invariant subspaces is treated in Section 3. Controllability subspaces are the topic of Section 4. Finally, in Sections 5 and 6, computational algorithms are given for system zeros and minimal inverses

Notations .

The reader is assumed to be familiar with the notations used in [12]. An additional set of notations and symbols are used here to simplify the treatment of matrices and matrix forms.

Let R be a matrix with dimension $n_1 \times m_1$. Then R_{ij}, $R_{\cdot j}$ and $R_{i\cdot}$ are the (i,j):th element, the i:th column and the i:th row respectively in R. Let Q be another matrix with dimension $n_2 \times m_2$. If $n_1 = n_2$, the matrix R * Q is defined as the block matrix [R Q]. Analogously, if $m_1 = m_2$, the matrix $R \wedge Q$ is defined as the block matrix $\begin{bmatrix} R \\ Q \end{bmatrix}$. By $\triangleq$ the left operand is assigned the evaluated value of the right operands, e.g. $r \triangleq r/s$.

A n × m matrix V on the form

$$\left[\begin{array}{ll} 1\ 0\ \ldots\ 0\ 0 & \text{- row } i_1 \\ 0\ 1\ \ldots .0\ 0 & \text{- row } i_2 \\ \ldots\ldots\ldots & \\ 0\ 0\ \ldots\ 0\ 1 & \text{- row } i_m \end{array}\right] \qquad (1.2)$$

where the crosses indicate arbitrary real values, is said to be on unity form. A set of vectors which have the same form as the columns of (1.2) is said to be a unity basis. For a matrix V on unity form, a structure vector $\delta(V)$ is introduced as $\delta(V) = [i_1\ i_2\ ..\ i_m]$ where i_k equals the row number for the position of "1" in column k, cf. (1.2). The columns in V (1.2) may be permuted.
Matrices are used as representatives of subspaces in the sense that the subspace is assumed to be spanned by the column vectors. By a basis matrix we then mean a matrix whose columns are linearly independent. Note that a matrix on unity form is always a basis matrix.

Remark: The following evaluation rules for the symbols * and $\wedge$ follow directly from the definitions

$$A(R * Q) = AR * AQ$$

$$(R \wedge Q)B = RB \wedge QB$$

and in subspace notation

$$Im(R * Q) = Im(R) + Im(Q)$$

$$Ker(R \wedge Q) = Ker(R) \cap Ker(Q)$$

Example 1. The matrix $V = (A(R * Q) \wedge (S * T)) * D$ is the

the block matrix

$$V = \left[\begin{array}{cc:c} AR & AQ & D \\ S & T & \end{array}\right]$$

2. BASIC ALGORITHMS.

Only two basic algorithms are needed to evaluate the algebraic expression considered below. The first algorithm, the Transformation Algoritm (TA), computes a unity basis for a given set of vectors and solves simultaneously a linear equation. The second algorithm, the Orthogonalization Algoritm (OA), computes a unity basis for the orthogonal complement of a given subspace. The latter subspace is assumed to be described by a unity basis.The only operation of this algorithm is rearrangements of matrix elements. Since no other algorithms are used, this means that all the numerical inaccuracies are tied to one single algorithm,namely TA.

The Transformation Algorithm.

Let Q be an n x m matrix of the form

$$Q = N_0 * R \tag{2.1}$$

where the submatrix N_0 is given on unity form with structure $\delta(N_0)$. The purpose is to find a unity basis for Im(Q). The known internal structure of Q should thereby be used. If N denotes this basis, there exists a nonsingular m×m matrix P so that

$$QP = (N_0 * R)P = N * 0 \tag{2.2}$$

where 0 denotes a zero matrix of suitable dimension. Let Y be a given p×m matrix and consider the linear equation

$$XQ = Y \tag{2.3}$$

which shall be solved for X.

Theorem 1. There exists a solution X to the equation (2.3) if and only if

$$(YP)_{\cdot i} = 0 \qquad i \in [s+1,\ldots,m] \tag{2.4}$$

where s is the number of columns in N and P is defined by (2.2). Moreover, if solutions exist, one is given by

$$\begin{aligned} X_{\cdot \delta(N)_j} &= (YP)_{\cdot j} \qquad j \in [1,2,\ldots,s] \\ X_{\cdot k} &= 0 \qquad\qquad \text{elsewhere} \end{aligned} \tag{2.5}$$

Proof. Consider the equation (2.3) and multiply from right by P, i.e.

$$XQP = X(N * 0) = XN * 0 = YP \tag{2.6}$$

It follows dirctly that the last m-s columns of YP must be zero, i.e. (2.4) is a necessary condition for a solution to exist. Sufficiency is shown by construction. Consider X defined by (2.5). It follows that

$$(XN)_{ij} = \sum_{k=1}^{n} X_{ik}N_{kj} = \sum_{r=1}^{s} X_{i\delta(N)_r}N_{\delta(N)_r j}$$

Since N is on unity form it follows from the definition of $\delta(N)$ that

$$N_{\delta(N)_r j} = \begin{cases} 0 & r \neq j \\ 1 & r = j \end{cases}$$

Thus

$$(XN)_{ij} = X_{i\delta(N)_j} \times 1 = (YP)_{ij} \qquad i \in [1,\ldots,p] \quad j \in [1,\ldots,s]$$

Since the last m-s columns in YP are zero, and P is nonsingular, X must be a solution to (2.3). □

Remark: One use of the structure vector $\delta(\cdot)$ is illustrated by this theorem.

Consider now an algorithm which given $Q = N_0 * R$, $\delta(N_0)$ and Y produces N, $\delta(N)$ and $Z = YP$. Such an algorithm computes a unity basis for $\mathrm{Im}(Q)$ and solves the linear equation (2.3) by Theorem 1. Moreover, the computations can be made simultaneously according to the following algorithm. By initialization, let δ denote a generalized structure vector for Q defined by

$$\delta = \delta(N_0) * \delta_0 \qquad \delta_0 = [-1\ -1\ \ldots\ -1]$$

where the negative values indicate the columns that are not on unity form. By use of δ, it is not necessary that the submatrix N_0 appears as in (2.1). The columns of N_0 and R may be permuted.

1^o Set $\delta_j=0$ for all columns $Q_{\cdot j}$ that are zero. Normalize all columns $Q_{\cdot j}$ for which $\delta_j = -1$ so that

$$Q_{\cdot j} \triangleq Q_{\cdot j}/||Q_{\cdot j}|| \qquad Y_{\cdot j} \triangleq Y_{\cdot j}/||Q_{\cdot j}||$$

2^o Zero the rows $\delta(N_0)_i$, i.e. perform for all columns $Q_{\cdot j}$ for which $\delta_j=-1$

$$Q_{\cdot j} \triangleq Q_{\cdot j} - \sum_{\delta_k>0} Q_{\delta_k j} Q_{\cdot k}$$

$$Y_{\cdot j} \triangleq Y_{\cdot j} - \sum_{\delta_k>0} Q_{\delta_k j} Y_{\cdot k}$$

The initialization of the algorithm is thereby concluded.

3^o Find the maximal element (i_s, j_s) in all the columns $Q_{\cdot j}$ for which $\delta_j=-1$. If the maximal element is zero or if δ is nonnegative, then go to 5^o. Otherwize set

$$\delta_{j_s} \triangleq i_s \qquad Q_{\cdot j_s} \triangleq Q_{\cdot j_s}/Q_{i_s j_s} \qquad Y_{\cdot j_s} \triangleq Y_{\cdot j_s}/Q_{i_s j_s}$$

4° Zero the row i_s, i.e. perform for all $j \neq j_s$

$$Q_{\cdot j} \leftarrow Q_{\cdot j} - Q_{i_s j} Q_{\cdot j_s} \qquad Y_{\cdot j} \leftarrow Y_{\cdot j} - Q_{i_s j} Y_{\cdot j}$$

Set $\delta_j = 0$ for all columns that are zero. Go to 3°.

5° Permute the elements of δ so that the s positive numbers are situated in the first s positions in ascending order. Perform the same permutations on the columns of Y and Q. Then

N = s first columns of Q

$Z = Y$

$\delta(N)$ = s first elements of δ

Remark: The following tests of zero have turned out to be favourable. In step 1° first test if R in (2.1) is zero by computing

$$\max_{\delta_j = -1} ||Q_{\cdot j}|| = r < \varepsilon_1$$

If R is zero go directly to step 5°. The column $Q_{\cdot j}$ is considered to be zero if

$$||Q_{\cdot j}||/r < \varepsilon_2$$

In step 2° and 3° use absolute test of zero by

$$||Q_{\cdot j}|| < \varepsilon_2$$

Note that the columns of Q are initially normalize (step 1°).

The known internal structure of Q (2.1) is utilized by the algorithm. The columns of N_0 are taken as the first members of the basis. The remaining vectors are computed from R. In this way $\delta(N_0)$ is used to minimize the amount of computation.

This is of special importance in the sequential procedures described below.

The outcome of the Transformation Algorithm is written formally as

$$(N,Z) = TA(Q,Y) \tag{2.7}$$

where use of internal structure in Q is assumed.

The Orthogonalization Algorithm.

Let $\mathcal{V}$ be an m-dimensional subspace of R^n with basis matrix V. The purpose is to find a suitable basis for $\mathcal{V}^\perp$. If V is given on unity form this is a simple problem. In fact, a unity basis for $\mathcal{V}^\perp$ is obtained by rearranging the elements of V using the structure vector $\delta(V)$.

Assume that V is on unity form. The Orthogonalization Algorithm goes as follows.

1° Set $\delta(V) = [r_1\ r_2\ \ldots\ r_m]$ and let $i_1, i_2, \ldots, i_{n-m}$ be the set of integers $\in [1,2,\ldots,n]$ which are not elements of $\delta(V)$. Arrange these integers so that $i_k < i_{k+1}$.

2° Form a n × (n-m) matrix B and an (n-m)-vector $\delta(B)$ as

$$\delta(B) = [i_1\ i_2\ \ldots\ i_{n-m}]$$

$$B_{i_k j} = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases} \qquad j,k \in [1,2,\ldots,n-m]$$

$$B_{r_k j} = -V_{i_j r_k} \qquad j \in [1,2,\ldots,n-m] \quad k \in [1,2,\ldots,m]$$

5° A basis matrix on unity form for $V^{\perp}$ is given by B with structure $\delta(B)$.

The algorithm is best illustrated by an example.

Example 2. Let V be a basis matrix for V where

$$V = \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ 1 & 0 & 0 \\ \alpha_4 & \alpha_5 & \alpha_6 \\ \alpha_7 & \alpha_8 & \alpha_9 \\ 0 & 1 & 0 \\ \alpha_{10} & \alpha_{11} & \alpha_{12} \\ 0 & 0 & 1 \end{bmatrix}$$

Then V is on unity form with $\delta(V) = [2\ 5\ 7]$. Applying OA to V we get

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -\alpha_1 & -\alpha_4 & -\alpha_7 & -\alpha_{10} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\alpha_2 & -\alpha_5 & -\alpha_8 & -\alpha_{11} \\ 0 & 0 & 0 & 1 \\ -\alpha_3 & -\alpha_6 & -\alpha_9 & -\alpha_{12} \end{bmatrix}$$

$\delta(B) = [1\ \ 3\ \ 4\ \ 6]$

It is immediately verified by taking scalar products that B is a basis matrix for $V^{\perp}$. □

Note the important fact that only rearrangements of matrix elements are performed in OA. No numerical inaccuracies are thus introduced. The outcome of OA is written formally as

$$B = OA(V) \tag{2.8}$$

where V must be on unity form. If $\mathcal{V}$ is spanned by the columns of V which is not on unity form, then a basis for the orthogonal complement is obtained in two steps by

$$(\hat{V},\cdot) = TA(V,\cdot)$$

$$B = OA(\hat{V})$$

where all the numerical inaccuracies are tied to TA.

3. INVARIANT SUBSPACES

Let $A:R^n \to R^n$ and $B:R^m \to R^n$ be two linear maps. A subspace $\mathcal{V} \subset R^n$ is by definition (A,B)-invariant [11] if there exists a linear map $F:R^n \to R^m$ so that

$$(A + BF)\mathcal{V} \subset \mathcal{V} \tag{3.1}$$

A necessary and sufficient condition for such a linear map to exist is [11]

$$A\mathcal{V} \subset \mathcal{V} + \mathcal{B} \tag{3.2}$$

Generally, if $\mathcal{V}$ is (A,B)-invariant there are many linear maps F satisfying (3.1). Denote the class of all such F by $\underline{F}(\mathcal{V})$, i.e.

$$\underline{F}(\mathcal{V}) = \{F \mid (A+BF)\mathcal{V} \subset \mathcal{V}\} \tag{3.3}$$

Let $\mathcal{V} \subset R^n$ be a given subspace spanned by the columns of V. Consider the following algorithm which simultaneously tests (3.2), computes a map F that satisfies (3.1) and computes the restriction $(A+BF)|\mathcal{V}$.

1^o Perform $(\hat{V},\cdot) = TA(V,\cdot)$ and let r and m be the number of

columns in $\hat{V}$ and B respectively.

2° Perform $(S,Y) = TA((\hat{V}*B)^T, (A\hat{V})^T)$ and let s be the number of columns in S.

3° Define a $(r+m)\times r$ matrix X by

$$X_{\delta(S)_j i} = Y_{ij} \qquad i \in [1,\ldots,r] \quad j \in [1,\ldots,s]$$

$$X_{ki} = 0 \qquad \text{elsewhere}$$

and partition the matrix X as $X = X_1 \wedge X_2$ where X_1 is $r\times r$ and X_2 is $m\times r$.

4° We have

(a) Condition (3.2) is satisfied if and only if $Y_{\cdot j} = 0$ for $j \in [s+1,\ldots,r+m]$

(b) A $m\times n$ matrix F in $\underline{F}(V)$ is given by

$$F_{i\delta(\hat{V})_j} = -(X_2)_{ij} \qquad i \in [1,\ldots,m] \; j \in [1,\ldots,r]$$

$$F_{ik} = 0 \qquad \text{elsewhere}$$

(c) The restriction of (A+BF) to V is given by $(A+BF)|V = X_1$

The validity of the algorithm is verified in the following way. First note that $\hat{V}$ produced in step 1° is a basis matrix on unity form for V. In step 2° and 3° the equation

$$A\hat{V} = (\hat{V} * B)X$$

is solved, cf. Theorem 1. A solution to this equation exists if and only if (3.2) is true. Theorem 1 then verifies assertion (a) in step 4°. Moreover, by the partition of X in step 3°

$$A\hat{V} = (\hat{V} * B)(X_1 \wedge X_2) = \hat{V}X_1 + BX_2 \qquad (3.4)$$

Using Theorem 1 once more, it follows that F is a solution to the equation $F\hat{V} = -X_2$ since $\hat{V}$ is on unity form. A substitution into (3.4) yields

$$(A + BF)\hat{V} = \hat{V}X_1$$

which shows that the assertions (b) and (c) hold.

Note that all the computations are made in the first two steps where TA is applied. In the following steps rearrangement of matrix elements are performed by means of the structure vectors $\delta(S)$ and $\delta(\hat{V})$ which are produced by TA.

Maximal Invariant Subspaces.

Let $\mathcal{D}$ be a given subspace of R^n. There is a unique maximal (A,B)-invariant subspace, $\mathcal{V}^M$, contained in $\mathcal{D}$. The subspace $\mathcal{V}^M$ is produced by the following sequence [11]

$$\begin{aligned} \mathcal{V}_0 &= \mathcal{D} \\ \mathcal{V}_i &= \mathcal{V}_{i-1} \cap A^{-1}(\mathcal{V}_{i-1} + \mathcal{B}) \end{aligned} \tag{3.5}$$

If α is the least integer such that $\mathcal{V}_\alpha = \mathcal{V}_{\alpha+1}$, then $\mathcal{V}^M = \mathcal{V}_\alpha$. Moreover, the sequence converges in at most $\dim(\mathcal{D})$ steps. To obtain a form where TA and OA can be applied, take the orthogonal complement of the sequence (3.5):

$$\begin{aligned} \mathcal{V}_0^\perp &= \mathcal{D}^\perp \\ \mathcal{V}_i^\perp &= \mathcal{V}_{i-1}^\perp + A^T(\mathcal{V}_{i-1} + \mathcal{B})^\perp \end{aligned} \tag{3.6}$$

A translation of this sequence to a computational form is made in the following algorithm.

1^o Initialize by setting

$$\begin{aligned} (V_0, \cdot) &= TA(D, \cdot) \\ S_0 &= OA(V_0) \\ (P_0, \cdot) &= TA(V_0 * B, \cdot) \end{aligned}$$

$R_0 = OA(P_0)$

2^o In step i compute

$$(S_i,\cdot) = TA(S_{i-1} * A^T R_{i-1},\cdot)$$
$$V_i = OA(S_i)$$
$$(P_i,\cdot) = TA(V_i * B,\cdot)$$
$$R_i = OA(V_i)$$

If the number of columns in V_i and V_{i-1} are the same set $\alpha = i$, otherwise set $i \leftarrow i+1$ and go to 2^o.

3^o V_α is a basis matrix on unity form for $\mathcal{V}^M$.

To show the equivalence between the algorithm and the sequence, use the facts that V_i, S_i, P_i and R_i are basis matrices for $\mathcal{V}_i$, $\mathcal{V}_i^\perp$, $\mathcal{V}_i + \mathcal{B}$ and $(\mathcal{V}_i + \mathcal{B})^\perp$ respectively.

The advantage of using a procedure like TA in this context is well illustrated by this algorithm. Since $\mathcal{V}_{i-1}^\perp$ is described by a unity basis S_{i-1}, the internal structure $\delta(S_{i-1})$ can be utilized so that only some additional vectors from a basis of $A^T(\mathcal{V}_i + \mathcal{B})^\perp$ are taken in order to form a basis S_i for $\mathcal{V}_i^\perp$, cf. step 2^o. The same argument applies to the computation of $\mathcal{V}_i + \mathcal{B}$. In this way the amount of computation, and thereby also the numerical inaccuracies, are decreased.

4. CONTROLLABILITY SUBSPACES.

Let $A: R^n \to R^n$ and $B: R^m \to R^n$ be a pair of linear maps. A subspace $\mathcal{R}$ of R^n is a controllability subspace [11] if

$$\mathcal{R} = \{A + BF \mid \mathcal{B} \cap \mathcal{R}\} = \mathcal{B}\cap\mathcal{R} + (A+BF)(\mathcal{B}\cap\mathcal{R}) + \ldots + (A+BF)^{n-1}(\mathcal{B}\cap\mathcal{R}) \quad (4.1)$$

for some linear map $F: R^n \to R^m$. It can be shown [12] that $\mathcal{R}$ is a controllability subspace if and only if the following two conditions are satisfied

(i) $\mathcal{R}$ is (A,B)-invariant

(ii) $\mathcal{R} = \{A + BF \mid \mathcal{B} \cap \mathcal{R}\}$ where $F \in \underline{F}(\mathcal{R})$ (4.2)

In order to use (4.2) we must generate the controllable subspace for a given pair (A,B). This is done in a sequential way in the following algorithm.

1° Set $B_0 = B$ and $(S_0,\cdot) = TA(B_0,\cdot)$

2° In step i perform $B_i = AB_{i-1}$ and $(S_i,\cdot) = TA(S_{i-1}*B_i,\cdot)$. If the number of columns in S_{i-1} and S_i are the same set $\alpha = i$ and go to 3°. Otherwize set $i \leftarrow i+1$ and go to 2°.

3° A unity basis for $\{A|\mathcal{B}\}$ is given by S_α.

The reason why TA is not directly applied to the controllability matrix $[B\ AB\ \ldots\ A^{n-1}B]$ is that the latter matrix may require an excessive storage if m and n are large. For instance, if n=20 and m=5, which is rather moderate for a large system, the storage requirement is 4k words for the controllability matrix only. This disadvantage is avoided by the sequential procedure of the algorithm.

The condition (4.2) for controllability subspaces can now be tested. Let $\mathcal{R} = \mathrm{Im}(R)$.

1° Test if $\mathcal{R}$ is (A,B)-invariant by using the first algorithm in section 3. If so, compute an $F \in \underline{F}(\mathcal{R})$ by the same algorithm. If $\mathcal{R}$ is not (A,B)-invariant set $\alpha=-1$ and go to 4°.

2° Compute a unity basis for $\mathcal{B} \cap \mathcal{R}$ by utilizing $\mathcal{B} \cap \mathcal{R} = (\mathcal{B}^{\perp} + \mathcal{R}^{\perp})^{\perp}$:

$(M,\cdot) = TA(B,\cdot)$
$S = OA(M)$
$(N,\cdot) = TA(R,\cdot)$
$V = OA(N)$
$(T,\cdot) = TA(V*S,\cdot)$

$Q = OA(T)$

Then Q is a basis matrix on unity form for $\mathcal{B} \cap \mathcal{R}$.

3° Compute a basis matrix Z for the controllable subspace for the pair (A+BF,Q) by applying the first algorithm in section 4. If the number of columns in Z and N are equal set $\alpha=1$, otherwize $\alpha=-1$. Go to 4°.

4° $\mathcal{R}$ is a controllability subspace if and only if $\alpha=1$.

A maximal controllability subspace $\mathcal{R}^M$ contained in a given subspace $\mathcal{D}$ is given by [11]

$$\mathcal{R}^M = \{ A + BF \mid \mathcal{B} \cap \mathcal{V}^M \} \qquad (4.3)$$

where $\mathcal{V}^M$ is the maximal (A,B)-invariant subspace contained in $\mathcal{D}$ and $F \in \underline{F}(\mathcal{V}^M)$. The computation of $\mathcal{V}^M$ and F is described in section 3. A basis matrix $\bar{B}$ for $\mathcal{B} \cap \mathcal{V}^n$ can be computed as is indicated in step 2° above. The maximal controllability subspace is then obtained as the controllable subspace for the pair $(A+BF,\bar{B})$.

5. SYSTEM ZEROS.

Consider a linear timeinvariant system S(A,B,C), i.e.

$$\dot{x} = Ax + Bu \qquad y = Cx$$

A set of complex numbers, the system zeros, can be defined using the concepts discussed in the preceding sections 2,3 Since the system zeros contain much information about the system and its properties, they are most valuable at a design stage.

Let $\mathcal{V}^M$ and $\mathcal{R}^M$ be the maximal (A,B)-invariant and controllability

respectively contained in Ker(C). Introduce the following polynomials

$$d_v(s) = \text{ch.p. for } (A+BF)|V^M$$
$$d_r(s) = \text{ch.p. for } (A+BF)|R^M \tag{5.1}$$

where $F \in \underline{F}(V^M)$. Since $V^M \supset R^M$ by the maximal property of V^M, it follows that $d_r(s)$ divides $d_v(s)$. Another polynomial $d_z(s)$ is then defined as their qoutient,i.e.

$$d_z(s) = d_v(s)/d_r(s) \tag{5.2}$$

The zeros of the polynomial $d_z(s)$ are defined as the system zeros for the system S(A,B,C) [2,3,7]. The system zeros can be computed straightforwardly by using the algorithms described in the preceding sections. It may, however, be favourable to perform the computations for the maps restricted to V^M since smaller dimension will be involved. In order to explain this more directly, let B^M, V^M and R^M be basis matrices for $B \cap V^M$, V^M and R^M respectively. Then

$$(A + BF)V^M = V^M\bar{A} \tag{5.3}$$

where $\bar{A}$ is a matrix representation of $(A+BF)|V^M$. Moreover, let $\bar{B}$ be a solution of

$$B^M = V^M\bar{B} \tag{5.4}$$

A solution exists since B^M is a basis matrix for $B \cap V^M$. The subspace R^M is equal to the controllable subspace for the pair $(A+BF,B^M)$, cf. (4.3). The controllability matrix Q^M for this pair becomes using (5.3) and (5.4)

$$Q^M = B^M*(A+BF)B^M *\ldots *(A+BF)^{n-1}B^M$$
$$= V^M(\bar{B}* \bar{A}\bar{B}* \ldots..* \bar{A}^{n-1}\bar{B})$$
$$= V^M(\bar{B}* \bar{A}\bar{B}* \ldots. * \bar{A}^{r-1}\bar{B}) .$$

where $r=\dim(V^M)$. Thus

$$R^M = V^M \bar{R}$$

where $\bar{R}$ is a basis matrix for $\bar{R} = \{\bar{A}|\bar{B}\}$. This also means that

$$(A + BF)|R = \bar{A}|\bar{R}$$

The polynomial $d_z(s)$ then equals the quotient of the ch.p. for $\bar{A}$ and $\bar{A}|\bar{R}$, i.e. the system zeros are the uncontrollable modes for the pair $(\bar{A},\bar{B})$.

The system zeros algorithm then goes as follows.

1^o Compute a unity basis D for Ker(C) by

$$(S,.) = TA(C^T,.)$$
$$D = OA(S)$$

2^o Compute the subspace V^M using the second algorithm in section 3.

3^o Compute $\bar{A} = (A+BF)|V^M$ using the first algorithm in section 3.

4^o Compute a basis B^M for $B \cap V^M$ using step 2^o in the second algorithm in section 4. Set the r×d matrix $\bar{B}$, where r= dim(V^M) and d= dim($B \cap V^M$), as

$$\bar{B}_{ij} = (B^M)_{\delta(V^M)_i j} \qquad i \in [1,\ldots,r] \quad j \in [1,\ldots,d]$$

5^o Compute a basis matrix $\bar{R}$ on unity form for the controllable subspace of the pair $(\bar{A},\bar{B})$ using the first algorithm in section 4. Let $\hat{R} = OA(\bar{R})$. Define a $\hat{r}\times\hat{r}$ matrix A_z, where $\hat{r}$ is the number of columns in $\hat{R}$, by

$$(A_z)_{ij} = (\hat{R}^T \bar{A})_{i\delta(\hat{R})_j} \qquad i,j \in [1,\ldots,\hat{r}] \qquad (5.5)$$

6^o The eigenvalues of A_z are then the system zeros for S(A,B,C)

The only step that may need further explanation is step 5°. In order to compute the uncontrollable modes for the pair $(\bar{A},\bar{B})$, a similarity transformation $T^{-1}\bar{A}T$ can be made with $T = \bar{R} * \hat{S}$, where

$$\hat{S}_{\delta(\hat{R})_i i} = 1 \qquad i \in [1,\ldots,\hat{r}]$$

$$\hat{S}_{ki} = 0 \qquad \text{elsewhere}$$

It is easily verfied that $\hat{R}^T\hat{S} = I$. Now, the inverse of T exists and has the form

$$T^{-1} = \hat{X} \wedge \hat{R}^T$$

for some matrix $\hat{X}$. The similarity transformation of $\bar{A}$ then yields

$$T^{-1}\bar{A}T = \begin{bmatrix} \bar{A}_1 & \bar{A}_2 \\ 0 & \bar{A}_z \end{bmatrix}$$

where $\bar{A}_z = \hat{R}^T\bar{A}\hat{S}$. The evaluation of $(\hat{R}^T\bar{A})\hat{S}$ is the same as (5.5) because of the special simple structure of $\hat{S}$.

In the system zero algorithm, the computations have been broken down into a number of elementary steps. In each of these steps maximal use is made of known internal structure. In fact, the algorithm has turned out to be fast and accurate. Note that this algorithm computes the invariant factor of lowest degree for a proper transfer function matrix. In this case the computation is based upon a minimal realization triple (A,B,C). The computation of invariant factors can thus be done without using transformations to Smith- MacMillan or Smith canonical forms.

6. INVERSE SYSTEMS.

System inverse of minimal dynamical order are thouroghly treated in [2,3] . An abbreviated version of the results are given below as a basis for a computational algorithm. Only the construction of minimal left inverses will be considered. A minimal right inverse is obtained using the duality, i.e. by computing a minimal left inverse for the dual system.

Consider the system S(A,B,C), i.e.

$$\dot{x} = Ax + Bu \qquad x(0) = 0$$
$$y = Cx \tag{6.1}$$

where $x \in \mathbb{R}^n$, $u \in R^m$ and $y \in R^p$ and A,B and C are linear maps between the appropriate vector spaces. Let V^M be the maximal (A,B)-invariant subspace contained in Ker(C). By the left invertibility condition, $B \cap V^M = 0$ cf. [12], it follows that R^n can be factorized into independent subspace as

$$R^n = \hat{X} \oplus B \oplus V \tag{6.2}$$

where $\hat{X}$ is any extension space. Transform the system (6.1) by applying a state feedback and a change of basis for the state space as

$$z = T^{-1}x$$

$$u = F x + u_0$$

where $F \in \underline{F}(V^M)$. The matrix T is given by

$$T_* = \hat{X} * B * V^M$$

where $\hat{X}$, B and V^M are basis matrices for $\hat{X}$, B and V^M respectively. The transformed system has the following block structure

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} \bar{A} & 0 \\ \hat{A} & \hat{A} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} \bar{B} \\ 0 \end{bmatrix} u_0$$

$$y = [\bar{C} \quad 0] \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{6.5}$$

$$u = [\bar{F} \quad \hat{F}] \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + u_0$$

There exists a polynomial matrix N(p) [2] such that

$$N(p)y = z_1 \qquad p = d/dt \tag{6.6}$$

i.e. z_1 can be reconstructed from the output without any knowledge of the input u_0. A minimal left inverse is then directly obtained as

$$\begin{aligned} \dot{w} &= \hat{A}w + K(p)y \\ u &= \hat{F}w + M(p)y \end{aligned} \tag{6.7a}$$

where

$$K(p) = \hat{A}N(p)$$

$$M(p) = (\bar{F} + \bar{B}^T(pI - \bar{A}))N(p) \tag{6.7b}$$

The essential computational steps are the construction of V^M and $N(p)$. The subspace V^M is obtained using the second algorithm in section 3. A polynomial matrix N(p) satisfying (6.6) can be computed by a sequence. Let R_i be a set of linear maps satisfying

$$\begin{aligned} R_0 &= I \\ \mathrm{Ker}(R_{i+1}) &= \mathrm{Ker}(R_i) + \bar{C}\bar{A}^i\bar{B} \end{aligned} \tag{6.8}$$

i.e.

$$\mathrm{Ker}(R_i) = \bar{C}\bar{B} + \bar{C}\bar{A}\bar{B} + \ldots + \bar{C}\bar{A}^{i-1}\bar{B}$$

Let α be the first integer such that $\mathrm{Ker}(R_\alpha) = R^p$. Then by successive differentiation of the output in (6.5)

$$R_i y^{(i)} = R_i \bar{C}\bar{A}^i z_1 \qquad i \in [0,1,\ldots,\alpha-1]$$

Hence

$$R(p)y = Qz_1$$

where

$$R(p) = \bigwedge_{i=0}^{\alpha-1} R_i p^i \qquad Q = \bigwedge_{i=0}^{\alpha-1} R_i \bar{C}\bar{A}^i$$

By the special properties of $\bar{A}$, $\bar{B}$ and $\bar{C}$, cf. [2] , it is not difficult to show that Q has a left inverse $\hat{Q}$. Thus

$$N(p) = \hat{Q}R(p)$$

The Minimal Left Inverse Algorithm goes as follows.

1^o Compute a unity basis V^M with structure $\delta(V^M)$ for $\mathcal{V}^M$ using the second algorithm in section 3. Let $r=\dim(\mathcal{V}^*)$.

2^o Perform $(S,Q) = TA(B^*V^M, I_{r+m})$. Let s be the number of columns in S. The system is left invertible if and only if $s = m + r$.

3^o Choose an extension basis $\hat{X}$ using the structure vector $\delta(S)$ as $\hat{X} = [x_1\ x_2\ \ldots\ x_{n-s}]$ where $x_k^T = [0\ldots0\ 1\ 0\ldots0]$ and the position i_k of "1" is such that $i_k \notin \delta(S)$. Note that the number of rows in S which doesn,t contain "1" is exactly n-s, cf. (1.2)

4^o Set $P = \mathrm{block\ diag}(I_{n-s}, Q)$ and perform $(\cdot, \hat{T}) = TA(\hat{X}^*S, P)$. Then $\hat{T} = T^{-1} = (\hat{X} * B * V^M)^{-1}$.

5° Compute

$$\hat{T}\,A\,T = \begin{bmatrix} A_{11} & A_{12} & 0 \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{matrix} n-s \\ m \\ r \end{matrix} \qquad \hat{T}\,B = \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}$$

$$CT = \underset{n-s}{[C_1} \quad \underset{m}{C_2} \quad \underset{r}{0]}$$

Then with the same notations as in (6.7)

$$\bar{A} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad \hat{\bar{A}} = [A_{31} \quad A_{32}] \qquad \hat{A} = A_{33}$$

$$\bar{F} = 0 \qquad \hat{F} = -A_{23} \qquad \bar{B} = \begin{bmatrix} 0 \\ I \end{bmatrix}$$

$$\bar{C} = [C_1 \quad C_2]$$

6° Compute recursively the maps R_i by setting $C_0 = \bar{C}$, $R_0 = I$, $P_0 = 0$ and $H_0 = I$ and $(S_0, Q_0) = TA(C_0^T, H_0)$ and

$$(P_{i+1}, \cdot) = TA(P_i * C_i \bar{B}, \cdot)$$

$$R_{i+1}^T = \emptyset A(P_{i+1})$$

$$H_{i+1} = \text{block diag}(Q_i, I_{r_{i+1}}) \qquad r_{i+1} = \text{no. in } R_{i+1}$$

$$C_{i+1} = C_i \bar{A}$$

$$(S_{i+1}, Q_{i+1}) = TA(S_i * C_{i+1}^T R_{i+1}^T, H_{i+1})$$

Proceed the recursive calculations until in one step α the number of columns in S_α equals $n-r$. Then $\hat{Q} = Q_\alpha^T$ and $R_0, \ldots, R_\alpha$ are computed

7° The minimal left inverse is obtained by substitution into (6.7).

The validity of the algorithm can be shown in a similar way as has been done in the preceding sections, by interpreting the operations of the basic algorithms TA and OA.

7. EXAMPLES

The computer programs that have been developed from the results of this paper will be described in a separate report. To indicate the computing time involved and the accuracy that can be achieved, let us compute the system zeros for a physical system, a drum boiler.

Different types of models for a drum boiler are thoroughly described in [16]. Here we will use a fifth order model from[17]. The linearized equations for a drum boiler around a certain operating point can be written as

$$\dot{x} = Ax + Bu + Gv$$

where the state variables are

x_1 = drum pressure (bar)

x_2 = drum level (m)

x_3 = drum liquid temperature (^{o}C)

x_4 = riser wall temperature (^{o}C)

x_5 = steam quality (%)

The control variables are

u_1 = heat flow to the risers (kJ/s)

u_2 = feedwater flow (kg/s)

and the disturbances are

v = load changes (bar)

Numerical values for A,B,C and G for a power station boiler with a maximum steam flow of about 350t/h are calculated in [17]. The drum pressure is 140 bar and the operating point is 90% full load. From [17]we have

$$A = \begin{bmatrix} -0.129 & 0.000 & 0.396\times10^{-1} & 0.250\times10^{-1} & 0.191\times10^{-1} \\ 0.329\times10^{-2} & 0.000 & -0.779\times10^{-4} & 0.122\times10^{-3} & -0.621 \\ 0.718\times10^{-1} & 0.000 & -0.100 & 0.887\times10^{-3} & -0.385\times10^{1} \\ 0.411\times10^{-3} & 0.000 & 0.000 & -0.822\times10^{-1} & 0.000 \\ 0.361\times10^{-3} & 0.000 & 0.350\times10^{-4} & 0.426\times10^{-4} & -0.743\times10^{-1} \end{bmatrix}$$

$$B = \begin{bmatrix} 0.000 & 0.139\times10^{-2} \\ 0.000 & 0.359\times10^{-4} \\ 0.000 & -0.989\times10^{-2} \\ 0.249\times10^{-4} & 0.000 \\ 0.000 & -0.534\times10^{-5} \end{bmatrix} \qquad G = \begin{bmatrix} 0.995\times10^{-1} \\ -0.318\times10^{-2} \\ -0.232\times10^{-1} \\ 0.000 \\ -0.381\times10^{-3} \end{bmatrix}$$

For different choices of output signals different sets of system zeros are obtained. If $y = [x_1\ x_2]$ are considered as output signals and u as input signal, the system zeros are computed to be

$Z = \{-0.06467, -0.36802\}$

and if $y = [x_1\ x_2\ x_3]$ are considered as output signals, the system zeros become

$Z = \phi$ (empty space)

The computation time in this example is 34 ms on a UNIVAC 1108 computer. The accuracy of the computations can be estimated by comparing with the computations for the corresponding dual system(the dual system has the same system zeros as the original system). The computation in this example turns out to be correct up to 8 digits (single precision is used in the programs).

8. AKNOWLEDGEMENT.

This work has been supported by the Swedish Board for Technical Development under contract no. 733533.

9. REFERENCES

[1] Belletrutti J.J., "Computer aided design and the characteristic locus method" IEE Conf. for Computer Aided Control System Design, Cambrdige (April 1973)

[2] Bengtsson G., " Minimal system inverses for linear multivariable systems", J. Math. An. Appl., 46, 261-274, (May 1974)

[3] Bengtsson G., " A theory for control of linear multivariable systems", Ph.D. Thesis, Lund Inst. of Techn., Div. of Automatic Control (Jan 1974)

[4] Fallside F.- Patel R.V. - Seraji H.," Design of linear multivariable systems using interactive graphics", IEE Conf. on Computer Aided Control System Design, Cambridge (April 1974)

[5] Kalman R.E. - Falb P.L. - Arbib M.A., " Topics in mathematical system theory", Mc Graw Hill, New York (1969)

[6] Morse A.S. - Wonham W.M., "Decoupling and pole assignement by dynamic compensation", SIAM J. Control, 8, 317-337, (Aug 1970)

[7] Morse A.S., "Structural invariants of linear multivariable systems", SIAM J. Control, 11, 446-465 (August 1973)

[8] Rosenbrock H.H., "State space and multivariable control theory", Nelson Ltd, London (1970)

[9] Wieslander J., "Interactive programs for computer aided design", IEE Conf. for Computer Aided Control System Design, Cambridge (April 1974)

[10] Wonham W.M., "Dynamic Observers - geometric theory", IEEE Trans. Automatic Control, AC-15, 258-259, (1970)

[11] Wonham W.M. - Morse A.S.,"Decoupling and pole assignement in linear multivariable systems - geometric theory", SIAM J. Control, 8, 1-18, (Febr 1970)

[12] Wonham W.M. - Morse A.S., "Status of noninteracting control", IEEE Trans. Automatic Control, AC-16, 568-580, (Dec 1971)

[13] Wonham W.M. - Morse A.S., "Feedback invariants of linear multivariable systems", Automatica, 8, 93-100,(1972)

[14] Wonham W.M., " Tracking and regulation in linear multivariable systems", SIAM J. Control, 11,424-437, (1973)

[15] Wonham W.M. - Pearson J.B., "Regulation and internal stabilization in linear multivariable systems", SIAM J. Control, 12, 5-18 (Febr 1974)

[16] Eklund K.: "Linear drum boiler - turbine models", Ph.D. Thesis, Lund Institute of Technology, Div. of Automatic Control (1971)

[17] Eklund K.: "Multivariable control of a boiler: An application of linear quadratic control theory", Report 6901, Lund Institute of Technology, Div. of Automatic Control (1969)