Gerson Beauchamp; Frank L. Lewis

Shuffle algorithm for two-dimensional singular systems with a Fornasini-Marchesini model

Kybernetika, Vol. 27 (1991), No. 3, 243--252

Persistent URL: http://dml.cz/dmlcz/125803

KYBERNETIKA — VOLUME 27 (1991), NUMBER 3

SHUFFLE ALGORITHM FOR TWO-DIMENSIONAL SINGULAR SYSTEMS WITH A FORNASINI-MARCHESINI MODEL

GERSON BEAUCHAMP, FRANK L. LEWIS

The condition for existence of solution for 2-D singular systems is usually stated as the non-singularity of a matrix pencil in two complex variables. The determinant of this matrix pencil equals the characteristic polynomial of the system. Although many authors assume this regularity condition to be satisfied, there has been no apparent effort towards developing computational methods to evaluate such a condition. In this paper we present several, easy to evaluate, sufficient conditions to determine whether the regularity condition is satisfied. The main contribution of the paper is a new *2-D shuffle algorithm* which is a natural extension of that of Luenberger [5]. The algorithm is a useful tool to test for regularity and should also contribute to the study of the geometric structure of 2-D singular systems.

1. INTRODUCTION

By the nature of 2-D systems (e.g. images [7], discrete models for partial differential equations [6], etc.), the independent variables (i, j) may be spatial coordinates which do not need to preserve the notion of time. Due to this fact, 2-D systems may have no natural notion of causality. Singular systems can describe noncausal behavior naturally and are therefore well suited for representing 2-D systems.

This paper extends the shuffle algorithm of Luenberger [5] to the 2-D case. The model under consideration is the singular Fornasini-Marchesini model [3, 4]. In contrast to the algorithm for the 1-D case, the algorithm presented here is quite complex. The level of complexity is due to the fact that the shuffle operations of the algorithm must account for two indices instead of one.

2. PRELIMINARIES

The model under consideration is the 2-D Singular Fornasini-Marchesini Model [3, 4] given by

$$Ex_{i+1,j+1} = Fx_{i+1,j} + Gx_{i,j+1} + Hx_{i,j} + Bu_{i,j} \tag{1a}$$

$$y_{i,j} = Cx_{i,j}\,; \tag{1b}$$

for $0 \leqq i \leqq N_1 - 1$ and $0 \leqq j \leqq N_2 - 1$ where $x \in \mathbb{R}^n$ is the semistate vector, $u \in \mathbb{R}^m$ is the input vector, and $y \in \mathbb{R}^p$ is the output vector. It is assumed that $p \leqq m \leqq n$. In (1) E is a constant square matrix, generally singular, F, G, H, B, and C are constant matrices of appropriate dimensions, and N_1 and N_2 are given integers. The region of interest in the (i, j)-plane is the 2-D region bounded by the rectangle $[0, N_1] \times [0, N_2]$.

The model may also contain inputs like $B_1 u_{i+1,j}$, and $B_2 u_{i,j+1}$ [3]. However, for ease of notation we shall restrict the discussion to model (1) since such input terms are easily incorporated in the algorithm and do not play a crucial role in the development.

Note that if E is nonsingular, then (1) can be solved recursively by iterating forward in both indices i and j given initial conditions $x_{i,0}$; $1 \leqq i \leqq N_1$, and $x_{0,j}$; $0 \leqq j \leqq N_2$ (i.e. the values of $x_{i,j}$ along the (i, j)-axes). Similarly, if F is nonsingular, then the recursion can be performed by iterating forward in i and backward in j given initial conditions x_{i,N_2}; $1 \leqq i \leqq N_1$, and $x_{0,j}$; $0 \leqq j \leqq N_2$. The cases for nonsingular G or H are similar to the above.

In any of these events, (1) may be considered to be a state-space system and solved iteratively provided the appropriate initial conditions are available. The initial conditions for all of these cases can be selected arbitrarily as long as they are appropriate for the given case.

From the above argument it is clear that any of the matrices can play the role of E is the appropriate initial conditions are specified and a suitable input sequence given. This phenomenon is, the basis for the *second shuffle* of the 2-D shuffle algorithm presented in this paper.

If all of the matrices E, F, G, and H are singular the situation is more complex and interesting than those discussed above. In this case the system may still have a solution provided it is regular. Beauchamp [1] has shown that if (1) is regular (i.e. $\det(z_1 z_2 E - z_1 F - z_2 G - H) \not\equiv 0$ for at least one (z_1, z_2)), then a unique solution may be obtained by specifying a suitable set of boundary conditions which involve all of the boundary values. We call *boundary values* those vectors $x_{i,j}$ in a solution sequence to (1) which lie on the perimeter (or boundary) of the region $[0, N_1] \times [0, N_2]$. That is, if a sequence $x_{i,j}$ is a solution to (1), then the vectors $x_{i,0}$, x_{i,N_2}; $1 \leqq i \leqq N_1$, and $x_{0,j}$, $x_{N_1,j}$; $0 \leqq j \leqq N_2$ are the boundary values.

It is also shown in [1] how to compute the solution in a recursive manner. In general, the boundary conditions for this case cannot be specified arbitrarily, but must satisfy certain restrictions in order for them to lead to a unique solution. This matter is discussed thoroughly in [1].

3. EXISTENCE OF SOLUTION

A sequence $x_{i,j}$ is said to be a *solution* to (1) if and only if (1a) holds for a given input sequence $u_{i,j}$.

Theorem 1. There exists a solution to system (1) for all input sequences $u_{i,j}$ if and only if

$$\operatorname{rank}\left[z_1z_2E - z_1F - z_2G - H, B\right] = \operatorname{rank}\left[z_1z_2E - z_1F - z_2G - H\right]. \tag{2}$$

Proof. See [3, 1].

A sufficient condition for existence of solution is given by

Corollary 1. If B is of full row rank (or equivalently invertible), then condition (2) reduces to

$$\det\left(z_1z_2E - z_1F - z_2G - H\right) \not\equiv 0 \tag{3}$$

for at least one (z_1, z_2).

Proof. See [1].

Condition (3) is called the *regularity condition* and it is a sufficient condition for existence of solution for all specified input sequences $\{u_{i,j}\}$ and all input matrices B. A system satisfying (3) is called regular and nonregular otherwise.

The main objective of this paper is to present computational methods to determine when the regularity condition is satisfied. This condition, or similar ones, appear frequently in the literature. However, there seems to be no effort towards actually evaluating it. In what follows we present some necessary and/or sufficient conditions for system (1) to be or not to be regular. These conditions prove to be useful starting points when checking condition (3). The next result provides a necessary condition for system (1) to be nonregular.

Lemma 1. If $\det\left(z_1z_2E - z_1F - z_2G - H\right) \equiv 0$ for all (z_1, z_2), then all of the matrices E, F, G, H are singular.

Proof. Suppose that $\det\left(z_1z_2E - z_1F - z_2G - H\right) \equiv 0$ for all (z_1, z_2) and that either of the matrices E, F, G, H is nonsingular, then the pencil $z_1z_2E - z_1F - z_2G - H$ can be multiplied by the inverse of the nonsingular matrix to obtain a pencil whose determinant cannot be identically equal to zero since the corresponding power of $z_1^j z_2^k$ $(0 \leqq j, k \leqq 1)$ cannot be eliminated identically. This contradicts the assumption. □

Lemma 1 may also be constructed as a sufficient condition for regularity. That is, if for a given system we find that either of the matrices E, F, G, or H is nonsingular, then the system is regular. The next result gives a necessary condition for regularity.

Lemma 2. If $\det\left(z_1z_2E - z_1F - z_2G - H\right) \not\equiv 0$ for at least one (z_1, z_2), then

the array of matrices $[E - F - G - H]$ is of full row rank and the array of matrices $[E^T - F^T - G^T - H^T]^T$ is of full column rank.

Proof. See [1].

Lemma 2 establishes that if either of the two arrays in that statement fails to be of full rank, then the system is nonregular. This is a sufficient condition for nonregularity. Note that the two rank conditions in Lemma 2 are not equivalent. Other sufficient conditions for regularity is given next.

Lemma 3. If at least one of the 1-D pencils $(z_1F + H)$, $(z_2G + H)$, $(z_2E - F)$, $(z_1E - G)$ is regular, then system (1) is regular.

Proof. The pencils $(z_1F + H)$, $(z_2G + H)$ correspond respectively to the special cases $z_2 = 0$ and $z_1 = 0$. Suppose that either of these two pencils is regular, then evaluating $\det(z_1z_2E - z_1F - z_2G - H)$ accordingly at either $z_2 = 0$ or $z_1 = 0$ shows that the determinant cannot equal zero for all (z_1, z_2).

The other two pencils correspond to the limiting cases $z_1 \to \infty$ and $z_2 \to \infty$. Suppose now that either of the two pencils $(z_2E - F)$ or $(z_1E - G)$ is regular, then letting accordingly either $z_1 \to \infty$ or $z_2 \to \infty$ in the original determinant shows that the determinant cannot equal zero for all (z_1, z_2). □

These sufficient conditions are very useful starting points to test for regularity of 2-D pencils since the regularity of one variable pencils (e.g. $(z_1E - G)$) is easily tested using the Luenberger shuffle algorithm [5]. Campbell [2] calls *simply regular* those systems with either $(z_2E - F)$ or $(z_1E - G)$ regular. However, he does not mention the pencils $(z_1F + H)$ and $(z_2G + H)$.

From all these necessary and/or sufficient conditions for regularity it may appear that 2-D systems are generically regular. Nevertheless, nature is not generic; moreover, there are systems whose regularity or nonregularity cannot be determined using the test given above. When all of the above tests fail to provide a conclusive result it may still be desired to determine whether or not the system at hand is regular. In the following section we present a 2-D shuffle algorithm which is a natural extension of that of Luenberger [5] for the 1-D case.

4. THE 2-D SHUFFLE ALGORITHM

The mere evaluation of condition (3) may not, by itself, justify the development of an algorithm that will prove to be far from simple. However, the 2-D shuffle algorithm will prove useful for instance in the computation of subspaces for 2-D systems. The algorithm will also provide a better understanding of the structural properties of 2-D systems and should serve to construct a recursive representation of the system when such a representation exists.

The eventual objective of the algorithm is to obtain an equivalent system with advanced inputs whose E matrix (or the matrix playing the role of E) is nonsingular.

If this is achieved, the algorithm has been successful in determining that the system is indeed regular and a unique solution is guaranteed to exist. Another implication is that the equivalent system thus found may then be solved iteratively provided the appropriate initial conditions are specified and a suitable input sequence given. The action of the algorithm is described next.

Consider system (1) with E singular. Perform a row compression on E to obtain

$$\begin{bmatrix} E_1 \\ 0 \end{bmatrix} x_{i+1,j+1} = \begin{bmatrix} F_1 \\ \bar{F}_1 \end{bmatrix} x_{i+1,j} + \begin{bmatrix} G_1 \\ \bar{G}_1 \end{bmatrix} x_{i,j+1} + \begin{bmatrix} H_1 \\ \bar{H}_1 \end{bmatrix} x_{i,j} + \begin{bmatrix} B_1 \\ \bar{B}_1 \end{bmatrix} u_{i,j} . \tag{4}$$

If at this point either of the matrices $\bar{F}_1$ or $\bar{G}_1$ equals zero, then the appropriate shuffle (i.e. a forward shift of index i or j respectively) of the lower portion of (4) can be performed. The shuffles must be restricted so as to preserve the given shifts of the indices of the vector $x_{i,j}$ in (4).

Consider now the three possible shuffles in (4). If $\bar{F}_1 = 0$, then a forward shift (or advance) on index i would shuffle $-\bar{G}_1$ underneath E_1, $\bar{H}_1$ underneath F_1, and would create a new index-column for $\bar{B}_1$ corresponding to input $u_{i+1,j}$. If $\bar{G}_1 = 0$, then the forward shift is done on index j which would shuffle $-\bar{F}_1$ underneath E_1, $\bar{H}_1$ underneath G_1 and would create a new column for $\bar{B}_1$ corresponding to input $u_{i,j+1}$. In the event that both $\bar{F}_1$ and $\bar{G}_1$ are zero, a forward shift on both indices i and j would shuffle $-\bar{H}_1$ underneath E_1 corresponding to input $u_{i+1,j+1}$. These three constitute the allowed shuffles.

This shuffle operation would then be followed by another row compression on the array containing E_1 and the matrix shuffled under it. In the case that this new "E matrix" is nonsingular the algorithm has determined that the system is regular and the procedure stops.

On the other hand, if all of $\bar{F}_1$, $\bar{G}_1$, and $\bar{H}_1$ equal zero, then the determinant is zero and the algorithm stops, it is concluded that the system is nonregular. It is worthwhile examining one of the allowed shuffles. To wit assume that $\bar{F}_1 = 0$, then the appropriate shuffle takes (4) to

$$\begin{bmatrix} E_1 \\ -\bar{G}_1 \end{bmatrix} x_{i+1,j+1} = \begin{bmatrix} F_1 \\ \bar{H}_1 \end{bmatrix} x_{i+1,j} + \begin{bmatrix} G_1 \\ 0 \end{bmatrix} x_{i,j+1} + \begin{bmatrix} H_1 \\ 0 \end{bmatrix} x_{i,j} + \begin{bmatrix} B_1 \\ 0 \end{bmatrix} u_{i,j} + \begin{bmatrix} 0 \\ \bar{B}_1 \end{bmatrix} u_{i+1,j} . \tag{5}$$

If at this point the new E matrix in (5) (given by $[E_1^{\mathrm{T}} \; -\bar{G}_1^{\mathrm{T}}]^{\mathrm{T}}$) happens to be nonsingular, then (5) becomes a recursive state equation with advanced inputs (e.g. $u_{i+1,j}$) which can be solved by iterating forward in i and forward in j provided the appropriate boundary conditions are given. If $[E_1^{\mathrm{T}} \; -\bar{G}_1^{\mathrm{T}}]^{\mathrm{T}}$ is singular, then a row compression on it would bring (5) to

$$\begin{bmatrix} E_2 \\ 0 \end{bmatrix} x_{i+1,j+1} = \begin{bmatrix} F_2 \\ \bar{F}_2 \end{bmatrix} x_{i+1,j} + \begin{bmatrix} G_2 \\ \bar{G}_2 \end{bmatrix} x_{i,j+1} + \begin{bmatrix} H_2 \\ \bar{H}_2 \end{bmatrix} x_{i,j} + \begin{bmatrix} B_2 \\ \bar{B}_2 \end{bmatrix} u_{i,j} + \begin{bmatrix} B_{1,2} \\ \bar{B}_{1,2} \end{bmatrix} u_{i+1,j} . \tag{6}$$

Note that if neither $\bar{F}_1$ nor $\bar{G}_1$ in (4) is zero, then no shuffle can be performed

while preserving the given indexing. When this is the case *further row compressions* must be performed on either $\bar{F}_1$ or G_1 in order to be able to shuffle. Suppose for instance that $\bar{F}_1$ in (4) has the smallest rank among F_1, G_1, and H_1, and perform a row compression on it to obtain

$$\begin{bmatrix} E_1 \\ 0 \\ 0 \end{bmatrix} x_{i+1,j+1} = \begin{bmatrix} F_1 \\ \bar{F}_1' \\ 0 \end{bmatrix} x_{i+1,j} + \begin{bmatrix} G_1 \\ \bar{G}_1' \\ \bar{G}_1'' \end{bmatrix} x_{i,j+1} + \begin{bmatrix} H_1 \\ \bar{H}_1' \\ \bar{H}_1'' \end{bmatrix} x_{i,j} + \begin{bmatrix} B_1 \\ \bar{B}_1' \\ \bar{B}_1'' \end{bmatrix} u_{i,j} . \tag{7}$$

Since $\bar{F}_1$ was assumed to have smaller rank than $\bar{G}_1$, the matrix $\bar{G}_1''$ will probably have higher rank than its counterpart $\bar{F}_1''$ (obtained if the row compression was performed on $\bar{G}_1$ instead). This is why the row compression is performed on the matrix with smaller rank. Now, the appropriate shuffle yields

$$\begin{bmatrix} E_1 \\ 0 \\ -\bar{G}_1'' \end{bmatrix} x_{i+1,j+1} = \begin{bmatrix} F_1 \\ \bar{F}_1' \\ \bar{H}_1'' \end{bmatrix} x_{i+1,j} + \begin{bmatrix} G_1 \\ \bar{G}_1' \\ 0 \end{bmatrix} x_{i,j+1} +$$

$$+ \begin{bmatrix} H_1 \\ \bar{H}_1' \\ 0 \end{bmatrix} x_{i,j} + \begin{bmatrix} B_1 \\ \bar{B}_1' \\ 0 \end{bmatrix} u_{i,j} + \begin{bmatrix} 0 \\ 0 \\ \bar{B}_1'' \end{bmatrix} u_{i+1,j} . \tag{8}$$

A row compression on $[E_1^{\mathrm{T}} \;\; 0 \;\; (-\bar{G}_1'')^{\mathrm{T}}]^{\mathrm{T}}$ yields

$$\begin{bmatrix} E_2 \\ 0 \end{bmatrix} x_{i+1,j+1} = \begin{bmatrix} F_2 \\ \bar{F}_2 \end{bmatrix} x_{i+1,j} + \begin{bmatrix} G_2 \\ \bar{G}_2 \end{bmatrix} x_{i,j+1} +$$

$$+ \begin{bmatrix} H_2 \\ \bar{H}_2 \end{bmatrix} x_{i,j} + \begin{bmatrix} B_2 \\ \bar{B}_2 \end{bmatrix} u_{i,j} + \begin{bmatrix} B_{1,2} \\ \bar{B}_{1,2} \end{bmatrix} u_{i+1,j} . \tag{9}$$

Notice that this equation has the same notation as (6), but that it was obtained through a different sequence of steps. Thus, the matrices in (9) need not be the same as those in (6).

A comment about this last step is appropriate. Due to the zero in $[E_1^{\mathrm{T}} \;\; 0 \;\; (-\bar{G}_1'')^{\mathrm{T}}]^{\mathrm{T}}$, it is clear that the row compression on (8) cannot yield a nonsingular E_2 in (9). Therefore, the only way in which the algorithm would terminate while obtaining a regular system would be after shuffles of equations like (6) without having to perform the row compression leading to (7).

To perform this sort of shuffles it is required that at least one of the matrices $\bar{F}_2$, $\bar{G}_2$ be zero as noted before. This will happen if and only if $\mathcal{R}^{\perp}([E_1^{\mathrm{T}} \;\; -\bar{G}_1^{\mathrm{T}}]^{\mathrm{T}}) \subset$ $\subset \mathcal{R}^{\perp}([F_1^{\mathrm{T}} \;\; -\bar{H}_1^{\mathrm{T}}]^{\mathrm{T}})$ or $\mathcal{R}^{\perp}([G_1^{\mathrm{T}} \; 0]^{\mathrm{T}})$ in (5). Similar conditions must also hold throughout the algorithm in order for the shuffles to proceed. For instance, for the shuffle leading from (4) to (5) to be possible (i.e. $\bar{F}_1 = 0$) it is necessary and sufficient that $\mathcal{R}^{\perp}(E) \subset \mathcal{R}^{\perp}(F)$.

From this point on, the action of this part of the algorithm follows a similar pattern. The detailed algorithm is given below. The input matrix B is omitted in order to ease on notation.

The 2-D Shuffle Algorithm

Step 0. *Initialize*: Set $k = 0$. Define $E_0 = -E$, $F_0 = F$, $G_0 = G$, $H_0 = H$, $\bar{f}_0 = 0$, $\bar{g}_0 = 0$, $h_0 = 0$, and let all of $\bar{F}'_0$, $\bar{G}'_0$, $\bar{H}'_0$, E^s_0, F^s_0, G^s_0 be null matrices of appropriate dimensions.

Comment 1. The superscript S denotes *shuffled* or *to-be-shuffled* matrices. The matrix E_0 has been set to be "minus" E in order to avoid confusion with sign changes during the algorithm.

Step 1. *Iteration* k: Row compression on $[E_k^{\mathrm{T}}\ 0\ (E_k^s)^{\mathrm{T}}]^{\mathrm{T}}$

$$\begin{bmatrix} E_k & F_k & G_k & H_k \\ 0 & \bar{F}'_k & \bar{G}'_k & \bar{H}'_k \\ E^s_k & F^s_k & G^s_k & 0 \end{bmatrix} \to \begin{bmatrix} E_{k+1} & F_{k+1} & G_{k+1} & H_{k+1} \\ 0 & \bar{F}_{k+1} & \bar{G}_{k+1} & \bar{H}_{k+1} \end{bmatrix}$$

with E_{k+1} of full row rank.

Stop if rank $(E_{k+1}) = n$.
Then, "system is regular", End.
Else, go to Step 2.

Step 2. *Second level row compression*:
Set $\bar{f}_{k+1} = \text{rank}\,(\bar{F}_{k+1})$, $\bar{g}_{k+1} = \text{rank}\,(\bar{G}_{k+1})$.
If $\bar{f}_{k+1} = \bar{g}_{k+1} = 0$, set $\bar{h}_{k+1} = \text{rank}\,(\bar{H}_{k+1})$.
Then, if $\bar{h}_{k+1} > 0$, set $E^s_{k+1} = \bar{H}_{k+1}$, $F^s_k = G^s_k = 0$, let all of $\bar{F}'_{k+1}$, $\bar{G}'_{k+1}$, $\bar{H}'_{k+1}$ be null matrices, set $k = k + 1$, go to Step 1.
Else, Stop, "system is not regular", End.

Else, if $\bar{f}_{k+1} \leqq \bar{g}_{k+1}$ perform a row compression of $\bar{F}_{k+1}$

$$[\bar{F}_{k+1}\ \bar{G}_{k+1}\ \bar{H}_{k+1}] \to \begin{bmatrix} \bar{F}'_{k+1} & \bar{G}'_{k+1} & \bar{H}'_{k+1} \\ 0 & E^s_{k+1} & F^s_{k+1} \end{bmatrix}$$

with $\bar{F}'_{k+1}$ of full row rank. Set $G^s_{k+1} = 0$, $k = k + 1$, go to Step 1.

Comment 2. Note that this includes the case of $\bar{f}_{k+1} = 0$ in which case all $\bar{F}'_{k+1}$, $\bar{G}'_{k+1}$, $\bar{H}'_{k+1}$ must be set to null matrices and proceed accordingly. That is, set $G^s_{k+1} = 0$, $E^s_{k+1} = \bar{G}_{k+1}$, $F^s_{k+1} = \bar{H}_{k+1}$, $k = k + 1$, and go to Step 1.

Else, perform row compression on $\bar{G}_{k+1}$.

$$[\bar{F}_{k+1}\ \bar{G}_{k+1}\ \bar{H}_{k+1}] \to \begin{bmatrix} \bar{F}'_{k+1} & \bar{G}'_{k+1} & \bar{H}'_{k+1} \\ E^s_{k+1} & 0 & G^s_{k+1} \end{bmatrix}$$

with $\bar{G}'_{k+1}$ of full row rank. Set $F^s_{k+1} = 0$, $k = k + 1$, go to Step 1.

Comment 3. Note that this includes the case of $\bar{g}_{k+1} = 0$ for which all $\bar{F}'_{k+1}$, $\bar{G}'_{k+1}$, $\bar{H}'_{k+1}$ are set to null matrices and proceed accordingly. See Comment 2 above.

Note that in order to perform a shuffle during the second step of the algorithm at least one of F^s_k, G^s_k must be zero. That is, we must be able to compress either $\bar{F}_{k+1}$ or $\bar{G}_{k+1}$ or else have either of the equal to zero. If this is not the case, the shuffling

process cannot proceed since there can be no further row compressions. The only possible shuffle would then be by allowing additional columns corresponding to additional indices of the vector $x_{i,j}$, other than the original ones. This is best understood by interpreting the shuffles as advances in either (or both) of the indices i, j as was done at the beginning of this section.

Creating additional columns with indices shifts other than the originals may be undesirable, for then the array of matrices becomes of increasing dimension. An alternative option is to introduce what will be called a *second shuffle* or re-indexing.

This relabeling will correspond to interchanging the roles of the matrices in the algorithm while preserving their role regarding the system equations. The motivation for this interchange is that there is no reason to insist in performing all the shuffles towards E (i.e. the coefficient matrix of $x_{i+1,j+1}$) since the nonsingularity of any of the system matrices (E, F, G, H) will suffice to detect regularity as shown in Lemma 1. The second shuffle is performed as follows.

Step 3. *Second Shuffle*:

If both of $\bar{F}_{k+1}$ and $\bar{G}_{k+1}$ are of full row rank, then set $E^0 = [E_{k+1}^T\ 0]^T$. $F^0 = [F_{k+1}^T\ \bar{F}_{k+1}^T]^T$, $G^0 = [G_{k+1}^T\ \bar{G}_{k+1}^T]^T$, and $H^0 = [H_{k+1}^T\ \bar{H}_{k+1}^T]^T$, perform one of the assignments below, and go to Step 1.

Comment 4. Here the superscript O is read "old" and is used to label the old arrays of matrices.

The three alternative shuffles are

$$\begin{aligned}
&\text{I.}\quad E_k = F^0\,,\quad F_k = E^0\,,\quad G_k = H^0\,,\quad H_k = G^0\,,\\
&\text{II.}\quad E_k = G^0\,,\quad F_k = H^0\,,\quad G_k = E^0\,,\quad H_k = F^0\,,\\
&\text{III.}\quad E_k = H^0\,,\quad F_k = G^0\,,\quad G_k = F^0\,,\quad H_k = E^0\,.
\end{aligned}$$

These are assigned in order, without repeating a previous assignment, and by assigning to E_k the matrix F^0, G^0, or H^0 with the highest rank.

That is, either of the assignments I, II, or III is used if, respectively, F^0, G^0, or H^0 has the highest rank and only if it has not been used previously.

What the second shuffle (i.e. Step 3) does is to interchange the matrices describing the system at the time it is encountered in such a way that the first part of the algorithm (i.e. Steps 1 and 2) remains valid in spite of the new assignment. This is best understood by considering the situation in detail.

Suppose that at some point during the algorithm the system is described by

$$E^0 x_{i+1,j+1} + F^0 x_{i+1,j} + G^0 x_{i,j+1} + H^0 x_{i,j} = 0\,. \tag{10}$$

Recall that the algorithm was initialized with $E_0 = -E$ to avoid confusions with sign changes and that the input has not been considered in the algorithm. Thus, the above is an appropriate system description at any iteration of the algorithm.

Considered now *alternative shuffle* I above. This corresponds to rearranging (10) as

$$F^0 x_{i+1,j} + E^0 x_{i+1,j+1} + H^0 x_{i,j} + G^0 x_{i,j+1} = 0\,. \tag{11}$$

Examining the *allowed shuffles* in (11) it should be clear that they are the same sort of shuffles performed by the first part of the algorithm (i.e. Steps 1 and 2). For instance, to shuffle anything from E^0 to F^0 requires a backward shift on index j. This shift would in turn require the corresponding rows in H^0 to be zero because otherwise these rows would have to go to a non-available column corresponding to $x_{i,j-1}$. Moreover, this shift would shuffle the corresponding rows of G^0 to H^0. These are precisely the operations performed in the first two steps of the algorithm when such a situation arises.

Other shuffle possibilities are satisfied in a similar manner. Therefore, the algorithm would continue to perform normally without a need to define new shuffling rules. The other two assignments in Step 3 can be verified to satisfy the shuffling rules of Steps 1 and 2 as well.

It has been seen that the 2-D shuffle algorithm can stop in several ways. First, if at some iteration it is found that E_k is nonsingular, then the system is regular and the algorithm terminates. Second, if at some iteration it is found that the array $[\bar{F}_{k+1}\ \bar{G}_{k+1}\ \bar{H}_{k+1}] = 0$, then the determinant is zero and the system is nonregular. In these two situations it is said that the algorithm has *terminated successfully* since it has reached a define conclusion regarding regularity. The shuffle algorithm thus provides another test for regularity,

This result is stated as a proposition.

Proposition 7.3.2. If there exists an integer $L \equiv k + 1$ such that E_L is nonsingular then the system is *regular*. Furthermore, if for some k it occurs that

$$[\bar{F}_{k+1}\ \bar{G}_{k+1}\ \bar{H}_{k+1}] = 0\,,$$

then the system is *nonregular*.

Finally, if both $\bar{F}_k$ and $\bar{G}_k$ are of full row rank, the shuffles cannot proceed. In the event that this occurs after going through the three alternative *second shuffles*, the algorithm stops without reaching a definite conclusion about the regularity of the system.

It is seen that the difficulty of the 2-D shuffle algorithm arises from the inability to proceed with the shuffling at certain points during the algorithm. The second shuffle is an attempt to remedy this situation.

5. CONCLUSIONS

Sufficient conditions for regularity of 2-D systems have been presented. A 2-D shuffle algorithm has been developed. This algorithm shall prove useful in the study of the geometric structure of 2-D systems. This may include computation of subspaces and system inversion for 2-D systems. The algorithm may also be used to construct a recursive representation of the system when such representation exists.

(Received November 16, 1990.)

REFERENCES

[1] G. Beauchamp: Algorithms for Singular Systems. Ph. D. Thesis, School of Electrical Engineering, Georgia Institute of Technology, Atlanta, GA 30332, 1990.

[2] S. L. Campbell: Comments on 2-D descriptor systems. Proc. IFAC Workshop on System Structure and Control: State-Space and Polynomial Methods, pp. 249—252, Prague, Sept. 1989.

[3] T. Kaczorek: Existence and uniqueness of solutions and Cayley-Hamilton theorem for general singular model of 2-D systems. Electronics and Electrotechnics, Accademia Polacca della Scienze, Vicolo Doria 2, 00-187 Roma, 1988.

[4] T. Kaczorek: Solvability of the singular Fornasini-Marchesini's model and its reduction to an equivalent 1-D model with variable structure. Electronics and Electrotechnics, Accademia Polacca della Scienze, Vicolo Doria 2, 00-187 Roma, 1988.

[5] D. G. Luenberger: Time-invariant descriptor systems. Automatica *14* (1978), 473—480.

[6] W. Marszalek: Two-dimensional state-space discrete models for hyperbolic partial differential equations. Appl. Math. Modeling *8* (1984), 11—14.

[7] R. P. Roesser: A discrete state-space model for linear image processing. IEEE Trans. Automat. Control *AC-20* (1975), 1—10.

Dr. Gerson Beauchamp, Electrical and Computer Engineering Department, University of Puerto Rico, Mayagüez, Puerto Rico 00709. U.S.A.

Prof. Dr. Frank L. Lewis, Automation and Robotics Research Institute, University of Texas at Arlington, 7300 Jack Newell Blvd. 2, Ft. Worth, Texas 76118. U.S.A.