Numerical Implementation of a Functional Observability Algorithm: A Singular Value Decomposition Approach

Tyrone Fernando
School of Electrical Electronic and Computer Engineering, University of WA, Crawley, WA 6009, Australia.
E-mail: tyrone@ee.uwa.edu.au

Less Jennings
School of Mathematics and Statistics, University of WA, Crawley, WA 6009, Australia.
E-mail: les@maths.uwa.edu.au

Hieu Trinh
School of Engineering, Deakin University, Geelong, VIC 3217, Australia.
E-mail: hmt@deakin.edu.au

***Abstract*—The paper outlines a numerical algorithm to implement the concept of Functional Observability introduced in [6] based on a Singular Value Decomposition approach. The key feature of this algorithm is in outputting a minimum number of additional linear functions of the state vector when the system is Functional Observable, these additional functions are required to design the smallest possible order functional observer as stated in [6].**

I. Introduction

There has been a considerable amount of research carried out on the subject of functional state estimation for well over four decades ever since the concept was introduced in the year 1966 by D.J. Luenberger [1]. Many design algorithms to estimate a desired linear function of the state vector denoted by a matrix L_0 for a dynamical system defined by (A, B, C) have been proposed, see [1]-[5] and also the references therein. In the case of designing a state observer, a designer before proceeding with the design is able to ascertain the existence of an observer by considering the pair (A, C). Similarly by testing Functional Observability of the triple (A, C, L_0), a designer is able to ascertain the existence of a functional observer before proceeding with the design, in a similar way to the state observer design case. The focus of this paper is to present a numerical algorithm for testing the Functional Observability based on a Singular Value Decomposition (SVD) approach. The key feature in this algorithm is in outputting a minimum number of additional functions of the state vector that has to be estimated together with the desired ones which will enable the design of the smallest possible order functional observer as per the problem formulation in [6]. In our implementation of the Functional Observability algorithm, the theory outlined in [6] will be followed and we will detail all the numerical steps involved in computing the expressions used in the theory. This paper is on a direct numerical implementation of the theory detailed in [6] and complements the ideas in it on numerical terms.

II. System Description and Preliminaries

Systems description and notation is identical to that in [6] and it is not repeated here. $A \in \mathbb{R}^{n\times n}$, $C \in \mathbb{R}^{p\times n}$ and $L_0 \in \mathbb{R}^{r\times n}$ are known constant matrices. Functional Observability is defined in [6] as follows:

Definition 1: The triple (A, C, L_0) is *Functional Observable* iff there exists an $L \in \mathbb{R}^{q\times n}$ where $r \leq q \leq n$ such that $\mathcal{R}(L) \supseteq \mathcal{R}(L_0)$ and L satisfies conditions 1 and 2 below:

Condition 1:

$$\operatorname{rank}\begin{bmatrix} LA \\ CA \\ C \\ L \end{bmatrix} = \operatorname{rank}\begin{bmatrix} CA \\ C \\ L \end{bmatrix}, \tag{1a}$$

Condition 2:

$$\operatorname{rank}\begin{bmatrix} sL - LA \\ CA \\ C \end{bmatrix} = \operatorname{rank}\begin{bmatrix} CA \\ C \\ L \end{bmatrix}, s \in \mathbb{C}. \tag{1b}$$

The necessary and sufficient condition for the triple (A, C, L_0) to be Functional Observable is also given in [6]. The algorithm begins by assigning L_0 to L and testing Condition 1 and Condition 2. If both conditions are satisfied then the linear functions defined in L_0 can be estimated and an observer can be designed following the design procedure outlined in [2]. If $L = L_0$ can satisfy Conditions 1 and 2 then the Functional Observability algorithm proposed in this paper would confirm that the triple (A, C, L_0) is Functional Observable and furthermore will output a null vector because no additional functions need to estimated. If $L = L_0$ does not satisfy Conditions 1 and 2, a larger L which includes L_0 may still satisfy the two conditions, hence there is a possibility of estimating the desired functions defined in L_0 by estimating additional linear functions as well as the desired ones. Therefore it is logical for a Functional Observability algorithm to not only ascertain the Functional Observability but also output additional linear functions that need to estimated with the desired ones if (A, C, L_0) is functional observable.

In this paper we will not attempt to justify why the algorithm can confirm the Functional Observability of the triple (A, C, L_0), the justification and the relevant theory is detailed in our submitted paper [6]. Rather in this paper will show how the algorithm can confirm the Functional Observability of

978-1-4244-7456-1/10/$26.00 ©2010 IEEE

the triple (A, C, L_0) numerically, and if so how it calculates the minimum required additional functions that need to be estimated.

III. Computation of Row Space, Row Null Space, Perpendicular Space and the Rank

Computation of row space, row null space and also the rank of a matrix can be performed using the Singular Value Decomposition (SVD) technique. This approach can be easily programmed using standard libraries in numerical software packages. Note that it is the row vectors that is considered as oppose to column vectors because a given linear function is represented by a row vector in our analysis. Applying a SVD for a matrix Ω, the following can be written:

$$\Omega = ESV^T \tag{2}$$

where E and V are unitary matrices, and S is a diagonal matrix consisting of the singular values in descending order along the diagonal. The numerical value of a singular value is taken as zero when they are less than a small tolerance a user may define (default value is chosen as 10^{-8}). After truncating the singular values below a tolerance to zero, equation (2) can be written as

$$\Omega = \left[\begin{array}{c|c} E_1 & E_2 \end{array}\right] \left[\begin{array}{c|c} S_1 & 0 \\ \hline 0 & 0 \end{array}\right] \left[\begin{array}{c} V_1 \\ \hline V_2 \end{array}\right] \tag{3}$$

Now it follows from (3) that $E_2^T\Omega = 0$ and that

$$[[\mathcal{N}(\Omega)]] = E_2^T. \tag{4}$$

Also from (3) we get $\Omega = E_1S_1V_1$ and also since E_1S_1 is nonsingular it follows that

$$[[\mathcal{R}(\Omega)]] = V_1 \tag{5}$$

and also

$$\text{rank}(\Omega) = \text{rows}(V_1). \tag{6}$$

Since $[[\mathcal{R}(\Omega)]] = V_1$ and also V is nonsingular it follows that

$$\left[\left[\mathcal{R}(\Omega)^{\perp}\right]\right] = V_2. \tag{7}$$

Computation of null space, row space and perpendicular space of a matrix will be done according to (4), (5) and (7) respectively. The rank of a matrix will be computed based on (6). The computation of vector spaces and the rank of a matrix will be determined by performing a single SVD on a matrix, so the same error tolerance is used for determining all the three vector spaces and the rank of the matrix.

IV. Algorithmic Details

The Functional Observability algorithm has three parts: in part I of the algorithm a minimum L which includes L_0 that satisfies Condition 1 is found, and in part II of the algorithm a minimum L that also satisfies Condition 2 is found by using the minimum L found in part I, and possibly increasing the size of it. Once both parts of the algorithm have been executed, the resulting L is a minimum L that satisfies both Conditions 1 and 2, and Part III of the algorithm merely outputs the data. The algorithm begins by making the following assignment

$$L = L_0 \tag{8}$$

and then proceed to compute matrices:

$$\tilde{T}_1 = \left[\begin{array}{c} C \\ CA \end{array}\right],\ \tilde{T}_2 = \left[\begin{array}{c} \tilde{T}_1 \\ L \end{array}\right] \text{ and } \tilde{T}_3 = \left[\begin{array}{c} \tilde{T}_2 \\ LA \end{array}\right].$$

$$T_1 = \left[\left[\mathcal{R}(\tilde{T}_1)\right]\right], \tag{9}$$

$$T_2 = \left[\left[\mathcal{R}(\tilde{T}_2)\right]\right], \tag{10}$$

$$T_3 = \left[\begin{array}{c} \tilde{L}A \\ T_2 \end{array}\right], \tag{11}$$

where $\tilde{L}$ is the row of L such that $\tilde{L}A$ does not belong to the row space of T_2. All T_1, T_2 and T_3 are computed after performing SVD on $\tilde{T}_1$ and $\tilde{T}_2$. The number of rows of T_2 and the number of rows of T_3 is then a and b respectively as defined in the following two equations:

$$a = \text{rank}\,(T_2)\,, \tag{12}$$

$$b = \text{rank}\,(T_3)\,. \tag{13}$$

Testing Condition 1 involves just checking if the number of rows of T_2 is the same as the number of rows of T_3, which is equivalent checking if the following condition (i.e. Condition 1) is satisfied

$$\mathcal{R}(T_3) = \mathcal{R}(T_2). \tag{14}$$

A finite iteration is needed to increase the size of L when Condition 1 is not true. In each step of the iterative process the number of rows of L is increased from the previous iteration, and the iteration continues until Condition 1 is satisfied. To find the relevant vectors that can be included to L from the previous iteration, it is necessary to compute the following matrices:

$$\left[\begin{array}{cc} \Phi & \Psi \end{array}\right] = \left[\left[\mathcal{N}\left(\left[\begin{array}{c} T_3A \\ T_3 \end{array}\right]\right)\right]\right], \tag{15}$$

$$\left[\begin{array}{cc} \Theta & \Xi \end{array}\right] = \left[\left[\mathcal{N}\left(\left[\begin{array}{c} \Phi T_3 \\ T_2 \end{array}\right]\right)\right]\right], \tag{16}$$

where Φ and Ψ have b columns each and Θ has number of columns equal to number of rows of Φ. The rows of the following matrix Γ represent the rows that can be added in L without increasing the values of b.

$$\Gamma \;=\; \left[\left[\mathcal{R}(\Phi T_3) \cap \mathcal{R}(\Theta\Phi T_3)^{\perp}\right]\right] = \left[\left[\mathcal{R}(\Theta)^{\perp}\right]\right] \Phi T_3 \tag{17}$$

Further rows that need to be added to L in the same iteration is given by the rows of the following matrix

$$\left[\left[\mathcal{R}(T_3) \cap \mathcal{R}\left(\left[\begin{array}{c} T_2 \\ \Gamma \end{array}\right]\right)^{\perp}\right]\right] \tag{18}$$

Part I of the algorithm computes a minimum dimension matrix L that satisfies Condition 1.

Let L_β be the output of Part I. With the minimum L found in Part I, we now proceed with Part II of the algorithm in

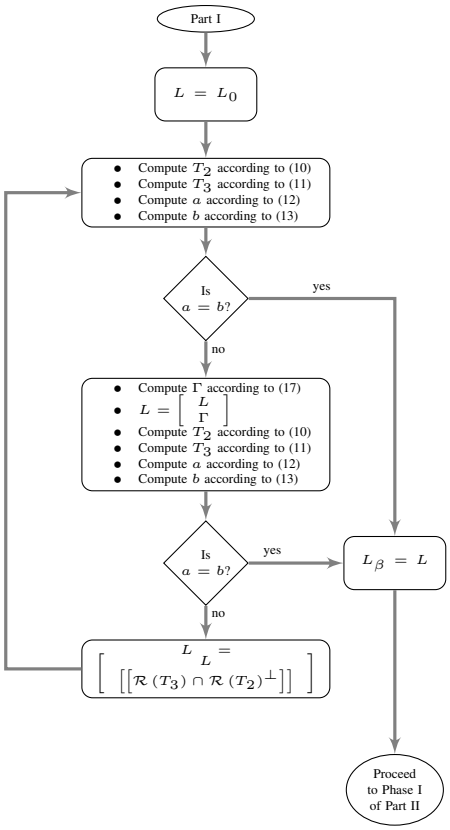

Fig. 1. Part I of Algorithm

order to satisfy Condition 2. Satisfaction of Condition 2 can be tested by checking the maximum difference between the RHS and LHS of Condition 2 denoted by χ

$$\chi = \text{rank}(T_2) - \min_{s \in \text{eig}(D)} \text{rank} \begin{bmatrix} sL - LA \\ T_1 \end{bmatrix} \tag{19}$$

where D is such that

$$LA = \tilde{D}T_1 + DL. \tag{20}$$

Part II of the algorithm has three phases: Phase I, Phase II and Phase III. In Phase I, the algorithm checks if $L = L_\beta$ also satisfies Condition 2. Phase I of Part II is shown below:

If however, $L = L_\beta$ does not satisfy Condition 2 then rows of U shown below are appended to L,

$$U = \Omega_1 CA \tag{21}$$

where Ω_1 such that $\text{cols}(\Omega_1) = \text{rows}(CA^2) = p$ and

$$\begin{bmatrix} \Omega_1 & \Delta_1 \end{bmatrix} = \left[\left[\mathcal{N}\left(\begin{bmatrix} CA^2 \\ T_2 \end{bmatrix}\right)\right]\right]. \tag{22}$$

If $L = \begin{bmatrix} L_\beta \\ U \end{bmatrix}$ satisfies Condition 2 then a search is performed to find the minimum number of rows in U that can be appended to L_β in order to satisfy Condition 2. Suppose U has α rows then it may be necessary to check

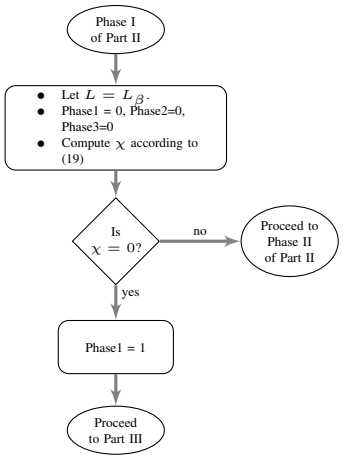

Fig. 2. Phase I of Part II of the Algorithm

upto $\sum_{k=1}^{\alpha} \text{bincoeff}(\alpha, k)$ possibilities of the submatrices of U which include all possible submatrices of U with single rows, all possible submatrices of U with 2 rows, etc upto all possible submatrices of U with α rows (i.e. just U).

If the minimum dimension matrix L is not found in Phase I of Part II or Phase II of Part II then further rows need to be appended to L_β. The rows that should be appended to L_β are rows of the matrix ΥX_2 where

$$\Upsilon = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_{n-a_\beta} \end{bmatrix}, \tag{23}$$

$$X_2 = \left[\left[\mathcal{R}\left(\begin{bmatrix} CA \\ C \\ L_\beta \end{bmatrix}\right)^\perp\right]\right], \tag{24}$$

such that X_2 is orthonomal and $\mu_i, i \in 1, \cdots, n-a_\beta$ represent the left eigenvectors of $X_2 A X_2^T$.

First we try to satisfy Condition 2 by appending to L_β individual rows of Υ post-multiplied by X_2, failing we would test all possible submatrices of two rows of Υ post-multiplied by X_2 and continue until we have considered the whole Υ post-multiplied by X_2 appended to L_β. Here it should be noted that complex eigenvectors have to be taken as a conjugate pair and cannot be considered individually. Conjugate row vector pairs are replaced with only the real part in one of the rows and only the imaginary part in the second row. Note a matrix L that is formed by appending any number of rows of ΥX_2 to L_β does not violate Condition 1. With matrix L formed in this fashion, further rows are appended to L in exactly the same way as in Phase II of Part II of the algorithm. Minimum number of rows of Υ with a minimum number of rows in matrix U which satisfy Condition 2 is a minimum dimension matrix that satisfy both Condition 1 and 2. Phases II and III of Part II is show next:

The final part, Part III of the Algorithm is essentially declaring the output, if any of the variables Phase1 or Phase2

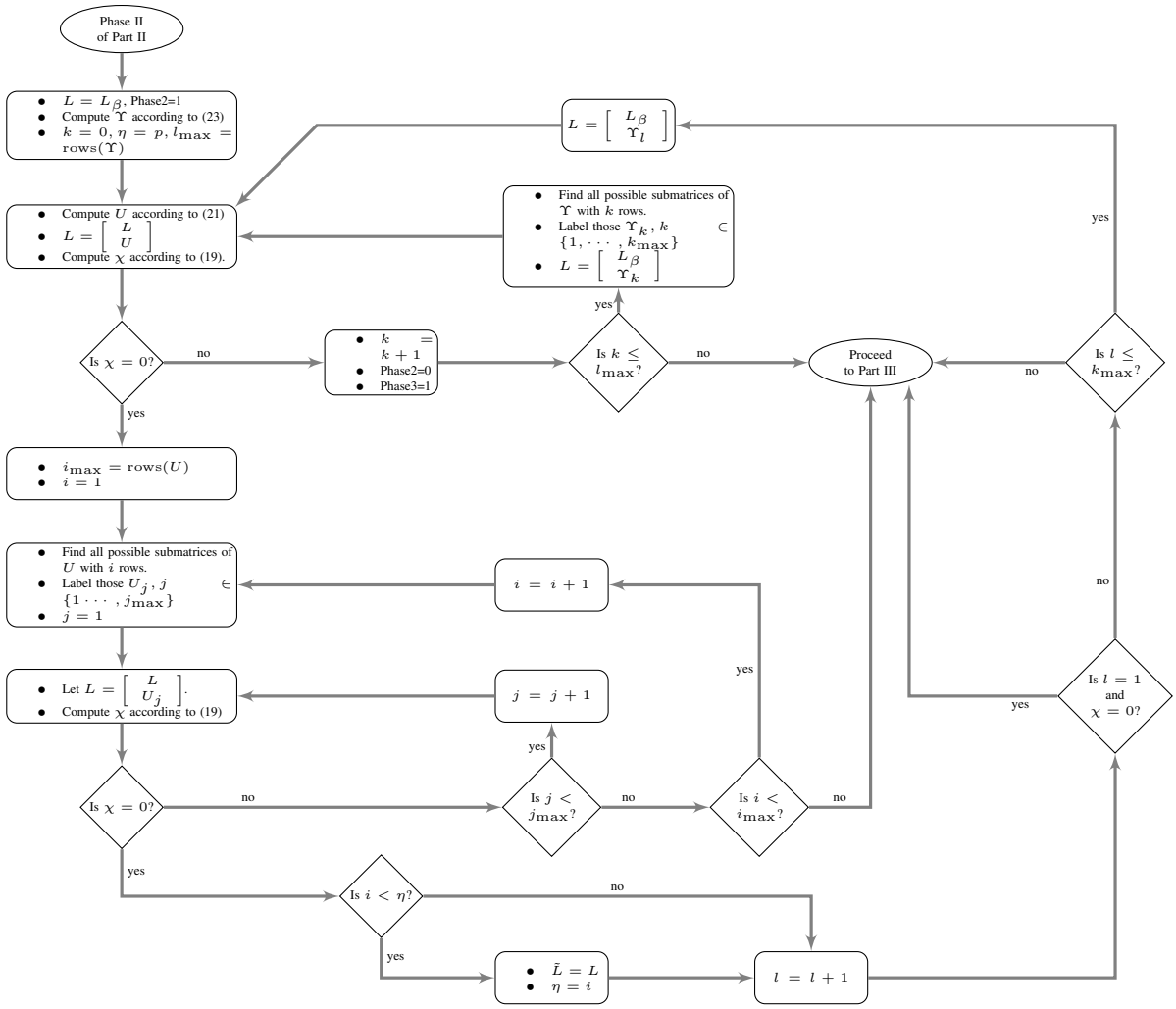

Fig. 3. Phase II of Part II of the Algorithm

or Phase3 has a value of 1 then the triple (A, C, L_0) is functional observable, otherwise it is not. If Phase1 has a value of 1 then $L_{\min} = L_\beta$, if Phase2 has a value of 1 or Phase3 has a value of 1 then $L_{\min} = \tilde{L}$. Part III of the algorithm is shown below:

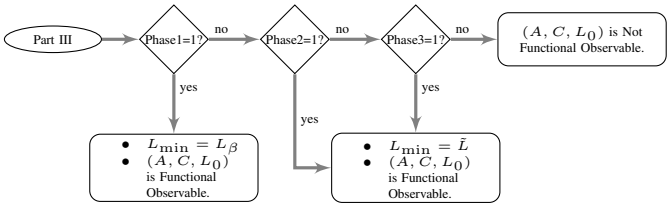

Fig. 4. Part III of the Algorithm

V. Conclusion

The paper has shown the numerical steps involved in the implementation of a Functional Observability algorithm based on a SVD approach. While an algorithm for determining the State Observability can be found in many numerical software packages, an algorithm for determining Functional Observability is not yet available, and inclusion of it in standard numerical libraries can benefit control theorists and practitioners alike.

References

[1] D. G. Luenberger, "Observers for multivariable systems," *IEEE Trans. Automat. Contr.*, vol. 11, pp. 190-197, 1966.

[2] M. Darouach, "Existence and design of functional observers for linear systems," *IEEE Trans. Automat. Contr.*, vol. 45, pp. 940-943, 2000.

[3] J. O'Reilly, *Observers for Linear Systems.* New York, NY: Academic, 1983.

[4] H. Trinh, T. Fernando and S. Nahavandi, "Partial-State Observers for Nonlinear Systems," *IEEE Trans. Automat. Contr.*, vol. 51, pp. 1808 - 1812, 2006.

[5] H. Trinh,Trung Dinh Tran and T.Fernando, "Disturbance Decoupled Observers for Systems With Unknown Inputs," *IEEE Trans. Automat. Contr.*, vol. 53, pp. 2397 - 2402, 2008.

[6] T. Fernando, H. Trinh and L. Jennings, "Functional observability and the design of minimum order functional observers," *IEEE Trans. Automat. Contr.*, vol. 55, pp. 1268 - 1273, 2010.