HAL
open science

On-line structured identification of switching systems with possibly varying orders

Laurent Bako, Guillaume Mercère, Stéphane Lecoeuche

To cite this version:

Laurent Bako, Guillaume Mercère, Stéphane Lecoeuche. On-line structured identification of switching systems with possibly varying orders. European Control Conference 2007, ECC'07, Jul 2007, Kos, Greece. pp.CDROM. hal-00164363

HAL Id: hal-00164363

https://hal.science/hal-00164363v1

Submitted on 20 Jul 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

On-line structured identification of switching systems with possibly varying orders

Laurent Bako*†, Guillaume Mercère‡, Stéphane Lecœuche*†

*Ecole des Mines de Douai - Département Informatique et Automatique, 59508 Douai, France
†Laboratoire d'Automatique, Génie Informatique et Signal - UMR 8146, 59655, France
‡Laboratoire d'Automatique et d'Informatique Industrielle, 86022 Poitiers, France
Email: {bako,lecoeuche}@ensm-douai.fr, guillaume.mercere@univ-poitiers.fr

***Abstract*— This paper is concerned with the identification of piecewise linear MIMO state space systems in a recursive way. The proposed method summons up benefits of recursive parameters estimation, on-line switching times detection and on-line order estimation. A structured identification scheme which applied on-line, allows to track both the extended observability matrix range space and its dimension. This method is used on-line to blindly identify switching systems and to label the different submodels. Since subspace identification methods rely on batch data block matrices, a minimum dwell time in each discrete state is necessary to achieve good performances. Simulation results comfort this point and illustrate the abilities and the benefits of the proposed approach.**

I. Introduction

A great number of complex industrial applications includes more or less hybrid systems. Hybrid systems are commonly thought of as systems which include both discrete-events and continuous phenomena. And yet, the black-box identification of such systems is a still developing research area. The domain is relatively young but since a few years, it has been attracting an increasing research activity. The problem of retrieving both the discrete state and the corresponding dynamical part from only the input-output data measurements is a rather challenging problem.

A large majority of the contributions published in the literature deals with the subclass of Piecewise ARX models. Inspired by pattern recognition techniques, [3] proposes to partition the regressors space into regions on which each linear local model is valid and then, estimate each model by standard LS regression. The solution in addition to requiring a knowledge of the system order is suboptimal since the convergence depends strongly on the initialization step. In [13], an optimal solution to the global problem is presented, converting the identification problem into a linear or quadratic mixed integer programming one as there exist efficient and powerful tools in this area to solve it in an optimal way. Conversely this algorithm suffers from a very high complexity. [17] presents an algebraic approach in which homogenous polynomials are used to realize a segmentation of the regression space into regions which corresponds to the discrete states. This method is ideal only in a noise-free context (see also [5]). It is also possible to cluster in a suboptimal way the generated data by resorting to bayesian inference, the system order and the number of modes being known a priori [6].

Some common characteristics of these methods are that they are mostly iterative, difficult to apply on-line and provide just input-output models with more or less guarantee of optimality. They rely on data partitioning either to compute completely regions of the regression space and then the related parameters [6], [3], [13] or to estimate first the parameters so as to derive the switching times [17].

Input-ouput models certainly describe systems well but there may be inappropriate for many practical analyses of systems especially multivariable systems. Therefore, some authors [12], [2] addressed recently the problem of subspace identification of switching systems in an off-line context. In this case, as the state is generally unkown, there is no available regression vector so that partitioning of the regression space becomes harder. Detection of the switching times seems to be an alternative to deal with switching MIMO systems using subspace methods. Owing to the fact that subspace methods operate on batches of successive measurements, a minimum dwell time assumption in each discrete state is required.

Switching systems may also be regarded as time varying systems. Generally, the data basis used off-line for the identification of such systems may cover only some modes of the systems and not the whole. This situation is probably the weak spot of off-line switching systems identification methods.

In this paper we follow the idea of detection and propose a recursive algorithm which realizes online the multiple tasks of estimation, detection and decision. Firstly, a structured subspace identification scheme is presented which differs from the standard approaches in that it does not require any singular values computation. Furthermore, an extension of that method to both rank and subspace adaptive tracking is straight. The designed method is then applied for online estimation of the submodels of an hybrid MIMO system. For this kind of system, each submodel may be slowly time-varying and from a submodel to another, the order may also change. For this reason, the order is continuously estimated.

The paper is organized as follows. Section II gives the

This research was supported by EC and Regional funded project *Intégration des Modèles multiples pour la Supervision de systèmes Non Stationnaires* (IMS-NS, ARCIR 2004-2007)

problem formulation. The background of the structured subspace identification strategy is developed in section III. The section IV presents the application of the new identification scheme to the on-line switching systems identification and provides simulation results to demonstrate the effectiveness of the scheme.

II. Problem formulation

Consider a system represented by the following discrete state space model

$$\begin{cases} x_{t+1} = \boldsymbol{A}x_t + \boldsymbol{B}u_t \\ \quad y_t = \boldsymbol{C}x_t + \boldsymbol{D}u_t + v_t \end{cases} \tag{1}$$

where $u_t \in \mathbb{R}^{n_u}$, $y_t \in \mathbb{R}^{n_y}$ and $x_t \in \mathbb{R}^{n_x}$ are the input, output and state vectors respectively. $v_t \in \mathbb{R}^{n_y}$ stands for zero-mean white gaussian noise vector. It is also supposed that $E\left[u_i v_j^T\right] = 0 \,\forall\, i,j$ and $E\left[v_i v_j^T\right] = \sigma_v^2 \delta_{ij} \boldsymbol{I}_{n_y}$ where $E\left[\,\right]$ denotes the expected value and δ is the Kronecker delta. Further, the model (1) is assumed to be observable and asymptotically stable. Given a collection of the input-output data of such a system, we are interested in estimating the matrices $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D})$ of the model (1). It is well known that these matrices are not uniquely determined since for any nonsingular matrix $\boldsymbol{T}$, $(\boldsymbol{TAT}^{-1}, \boldsymbol{TB}, \boldsymbol{CT}^{-1}, \boldsymbol{D})$ explains the input-output behaviour of the system in (1) as well.

Subspace based identification methods (SIM) are nowadays considered as good alternatives to the more traditional prediction error or maximum likelihood approaches for directly estimating a state space realization from input-output data. They more precisely rely on the following embedded data equation:

$$\boldsymbol{Y}_{t,f,N} = \boldsymbol{\Gamma}_f \boldsymbol{X}_{t,N} + \boldsymbol{H}_f \boldsymbol{U}_{t,f,N} + \boldsymbol{V}_{t,f,N} \tag{2}$$

where $\boldsymbol{\Gamma}_f$ is the extended observability matrix, $\boldsymbol{H}_f$ the block lower triangular Toeplitz matrix:

$$\boldsymbol{\Gamma}_f = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \\ \vdots \\ \boldsymbol{CA}^{f-1} \end{bmatrix}, \boldsymbol{H}_f = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{CB} & \boldsymbol{D} & \cdots & \boldsymbol{0} \\ \boldsymbol{CAB} & \boldsymbol{CB} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{CA}^{f-2}\boldsymbol{B} & \boldsymbol{CA}^{f-3}\boldsymbol{B} & \cdots & \boldsymbol{D} \end{bmatrix}$$

$X_{t,N} = \begin{bmatrix} x_t & \cdots & x_{t+N-1} \end{bmatrix}$, $f > n_x$ and $\boldsymbol{V}_{t,f,N}$, $\boldsymbol{Y}_{t,f,N}$ and $\boldsymbol{U}_{t,f,N}$ are the block Hankel matrices defined in a similar way as :

$$\boldsymbol{U}_{t,f,N} = \begin{bmatrix} u_t & u_{t+1} & \cdots & u_{t+N-1} \\ u_{t+1} & u_{t+2} & \cdots & u_{t+N} \\ \vdots & \vdots & \ddots & \vdots \\ u_{t+f-1} & u_{t+f} & \cdots & u_{t+f+N-2} \end{bmatrix}$$

There exist many works on subspace identification of systems in the literature [14]. Most of the SIM are implemented using Singular Values Decomposition (SVD) of huge matrices blocks (MOESP, N4SID). The SVD is essentially used to determine the system order and then to compute $\boldsymbol{\Gamma}_f$. The main drawback of such methods is that the SVD is known to be too computationally demanding and difficult to update recursively [8]. Hence, it has been necessary to find SVD alternative algorithms to apply the subspace concept in a recursive framework. Several reliable algorithms have been developed. For example, the IV-PAST method [4], borrowed from [18], have been introduced to track the observability subspace in a coloured noisy framework. More recently, the paper [10] have suggested an identification version of the Propagator Method [11] ordinarily used for subspace tracking in signal array processing. The main advantage of this approach over the previous conception lies in the use of a linear operator and quadratic criteria which lead to recursive least squares implementations for the algorithms. This latter method will be discussed more deeply.

The next section tries to provide a better solution for a more general application of the Propagator Method in MIMO systems identification. Our objective is also to turn it into an efficient tool of deriving the system order and finally a complete structured scheme for identification of switching systems.

III. Adaptation of the Propagator Method to MIMO system identification

A. Propagator Method for system identification

The technique presented hereafter has been firstly introduced in [10] in a recursive identification framework. It consists in the adaptation of the Propagator Method to recursive system identification. For brevity, only its general scheme is presented. Focusing on the data equation (2), the so-called observation matrix $\boldsymbol{\mathcal{Z}}_{t,f,N} = \boldsymbol{Y}_{t,f,N} - \boldsymbol{H}_f \boldsymbol{U}_{t,f,N}$ is first estimated using for example a QR decomposition. Then, a basis of the observability matrix range space is computed from

$$\boldsymbol{\mathcal{Z}}_{t,f,N} = \boldsymbol{\Gamma}_f \boldsymbol{X}_{t,N} + \boldsymbol{V}_{t,f,N} \tag{3}$$

by making a strong use of the observability assumption. It follows indeed from the observability of the system in (1) that $\operatorname{rank}(\boldsymbol{\Gamma}_{n_x}) = \operatorname{rank}(\boldsymbol{\Gamma}_f) = n_x$. Since $\boldsymbol{\Gamma}_f \in \mathbb{R}^{n_y f \times n_x}$ and $n_y f > n_x$, one can find a permutation matrix S which rearranges the rows of $\boldsymbol{\Gamma}_f$ in such a way that n_x linearly independant rows appear in the first n_x positions.

$$\boldsymbol{S}\boldsymbol{\Gamma}_f = \begin{bmatrix} \boldsymbol{\Gamma}_f^{(1)} \\ \boldsymbol{\Gamma}_f^{(2)} \end{bmatrix}$$

with $\boldsymbol{\Gamma}_f^{(1)}$ a square matrix of dimension n_x. As the rows of $\boldsymbol{\Gamma}_f^{(1)}$ span the row space of $\boldsymbol{\Gamma}_f$ there exists a unique matrix $\boldsymbol{P}_f$ [11] such that

$$\boldsymbol{S}\boldsymbol{\Gamma}_f = \begin{bmatrix} \boldsymbol{\Gamma}_f^{(1)} \\ \boldsymbol{P}_f \boldsymbol{\Gamma}_f^{(1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_{n_x} \\ \boldsymbol{P}_f \end{bmatrix} \boldsymbol{\Gamma}_f^{(1)}$$

Since $\boldsymbol{\Gamma}_f^{(1)}$ is invertible, it is possible to obtain an expression of the observability matrix in a particular basis by determining the propagator. If one partitions the whole equation (3) using the same permutation, the following is obtained :

$$\begin{bmatrix} \boldsymbol{\mathcal{Z}}_{t,f,N}^{(1)} \\ \boldsymbol{\mathcal{Z}}_{t,f,N}^{(2)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_{n_x} \\ \boldsymbol{P}_f \end{bmatrix} \boldsymbol{\Gamma}_f^{(1)} \boldsymbol{X}_{t,N} + \begin{bmatrix} \boldsymbol{V}_{t,f,N}^{(1)} \\ \boldsymbol{V}_{t,f,N}^{(2)} \end{bmatrix}$$

Then, an estimate of $\boldsymbol{P}_f$ can be obtained in a noiseless set ($\boldsymbol{V}=0$) by minimizing the following least square criterion

$$\left\| \boldsymbol{\mathcal{Z}}^{(2)}_{t,f,N} - \boldsymbol{P}_f \boldsymbol{\mathcal{Z}}^{(1)}_{t,f,N} \right\|^2_F$$

where $\boldsymbol{\mathcal{Z}}^{(i)}_{t,f,N}$ is a submatrix constructed from $\boldsymbol{\mathcal{Z}}_{t,f,N}$ in the same manner as $\boldsymbol{\Gamma}^{(i)}_f$ is from $\boldsymbol{\Gamma}_f$.
The interested reader is referred to [10] or [9] for more details on that method.

B. Discussion on permutation matrix selection

Selecting the permutation matrix $\boldsymbol{S}$ introduced in the previous subsection is a rather challenging task. Since $\boldsymbol{\Gamma}_f$ is unknown, there is a great problem in recognizing which rows are linearly independent. In [10], the first n_x rows of $\boldsymbol{\Gamma}_f$ have implicitly been assumed to be linearly independent. But, this assumption is accurate only for the MISO class of systems.

To find $\boldsymbol{S}$, it is sufficient to characterize n_x linearly independent rows in $\boldsymbol{\Gamma}_f$. For an observable MISO system, the observability matrix $\boldsymbol{\Gamma}_{n_x}$ is square and invertible so that $\boldsymbol{S}$ is chosen to be the identity matrix. For a general MIMO system, the problem is slightly complicated. Let denote in this case,

$$\gamma_j = \begin{bmatrix} \boldsymbol{C}_j^T \\ \boldsymbol{C}_j^T \boldsymbol{A} \\ \vdots \\ \boldsymbol{C}_j^T \boldsymbol{A}^{f-1} \end{bmatrix} \tag{4}$$

where $\boldsymbol{C}_j^T$ refers to the jth row of $\boldsymbol{C}$. If all the poles are observable from the output y_j, (as it is for MISO systems) then γ_j is necessarily of rank n_x. But that is far from being always the case for general MIMO systems since the other outputs may introduce dynamics in the state vector which would not be visible by y_j. Consider an auxiliary output $\tilde{\mathrm{y}}$ defined as

$$\tilde{\mathrm{y}} = y_1 + \sum_{j=2}^{n_y} \alpha_j \, y_j$$

with α_j real numbers. The objective of this manipulation is to replace one of the outputs (for example y_1) by this auxiliary output from which all the poles of the system may be observable. It is possible to find the numbers α_j such that $\tilde{\boldsymbol{\gamma}} = \gamma_1 + \sum_{j=2}^{n_y} \alpha_j \gamma_j$ is of full rank n_x. An important condition is that $\tilde{\mathrm{y}}$ must be sensitive to all the observable dynamics of the system, whatever output they are acting on. The choice of these coefficients α_j must be so as to guarantee the observability of all the system poles in $\tilde{\mathrm{y}}$. An easy way to choose them consists in generating them randomly as nonzero real numbers. Finally, the data equation (2) can be rewritten as follows

$$\tilde{\boldsymbol{Y}}_{t,f,N} = \tilde{\boldsymbol{\Gamma}}_f \boldsymbol{X}_{t,N} + \tilde{\boldsymbol{H}}_f \boldsymbol{U}_{t,f,N} + \tilde{\boldsymbol{V}}_{t,f,N} \tag{5}$$

with

$$\begin{bmatrix} \tilde{\mathrm{y}} & y_2 & \dots & y_{n_y} \end{bmatrix}^T \leftarrow \begin{bmatrix} y_1 & \dots & y_{n_y} \end{bmatrix}^T \tag{6}$$
$$\tilde{\boldsymbol{C}}_1^T \leftarrow \boldsymbol{C}_1^T + \sum_{j=2}^{n_y} \alpha_j \boldsymbol{C}_j^T, \quad \tilde{\boldsymbol{D}}_1^T \leftarrow \boldsymbol{D}_1^T + \sum_{j=2}^{n_y} \alpha_j \boldsymbol{D}_j^T$$

The matrices $\tilde{\boldsymbol{\Gamma}}_f$ and $\tilde{\boldsymbol{H}}_f$ are defined from $A, B, \tilde{\boldsymbol{C}}, \tilde{\boldsymbol{D}}$ where $\tilde{\boldsymbol{C}}$ and $\tilde{\boldsymbol{D}}$ correspond to the matrices $\boldsymbol{C}$ and $\boldsymbol{D}$ in which the respective first rows have been replaced by $\tilde{\boldsymbol{C}}_1^T$ and $\tilde{\boldsymbol{D}}_1^T$. In the remainder of the paper we will adopt whenever possible, simplified notations as $\boldsymbol{Y} := \tilde{\boldsymbol{Y}}_{t,f,N}$, $\boldsymbol{\Gamma} := \tilde{\boldsymbol{\Gamma}}_f$, $\boldsymbol{\Gamma}^{(1)} := \tilde{\boldsymbol{\Gamma}}^{(1)}_f$. Then, (5) becomes

$$\boldsymbol{Y} = \boldsymbol{\Gamma} \boldsymbol{X} + \boldsymbol{H} \boldsymbol{U} + \boldsymbol{V} \tag{7}$$

The permutation is applied to this equation in the objective to put the sub-matrix $\tilde{\boldsymbol{\gamma}}$ in the first f rows of $\boldsymbol{S}\tilde{\boldsymbol{\Gamma}}_f$. $\boldsymbol{S}$ may be chosen as :

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{\mathcal{I}}(1,:) \\ \boldsymbol{\mathcal{I}}(n_y+1,:) \\ \vdots \\ \boldsymbol{\mathcal{I}}\left((f-1)n_y+1,:\right) \\ \boldsymbol{\mathcal{I}}(2:n_y,:) \\ \boldsymbol{\mathcal{I}}(n_y+2:2n_y,:) \\ \vdots \\ \boldsymbol{\mathcal{I}}\left((f-1)n_y+2:fn_y,:\right) \end{bmatrix}$$

where $\boldsymbol{\mathcal{I}}$ is the identity matrix of order fn_y.

C. Propagator estimation

In this subsection, the problem of the propagator described earlier is considered. By being inspired by the MOESP class of subspace methods, the first step of the propagator estimation consists in eliminating the term $\boldsymbol{HU}$ in the data equation, projecting the whole equation onto the orthogonal complement subspace of the row space of $\boldsymbol{U}$. In this objective, let follow the QR implementation method [16]

$$\begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{Y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{0} \\ \boldsymbol{R}_{21} & \boldsymbol{R}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_1 \\ \boldsymbol{Q}_2 \end{bmatrix} \tag{8}$$

Premultiplying by $\boldsymbol{S}$ and postmultiplying by $\boldsymbol{Q}_2^T$, one gets

$$\boldsymbol{S}\boldsymbol{R}_{22} = \boldsymbol{S}\boldsymbol{Y}\boldsymbol{Q}_2^T = \begin{bmatrix} \boldsymbol{I}_{n_x} \\ \boldsymbol{P} \end{bmatrix} \boldsymbol{\Gamma}^{(1)} \boldsymbol{X} \boldsymbol{Q}_2^T + \boldsymbol{S}\boldsymbol{V}\boldsymbol{Q}_2^T \tag{9}$$

and then

$$\frac{1}{N}\boldsymbol{S}\boldsymbol{R}_{22}\boldsymbol{R}_{22}^T\boldsymbol{S}^T = \begin{bmatrix} \boldsymbol{I}_{n_x} \\ \boldsymbol{P} \end{bmatrix} \underbrace{\frac{1}{N}\left(\boldsymbol{\Gamma}^{(1)}\boldsymbol{X}\right)\boldsymbol{\mathcal{Q}}\left(\boldsymbol{\Gamma}^{(1)}\boldsymbol{X}\right)^T}_{(I)} \begin{bmatrix} \boldsymbol{I}_{n_x} \\ \boldsymbol{P} \end{bmatrix}^T$$
$$+ \underbrace{\frac{1}{N}\boldsymbol{S}\boldsymbol{V}\boldsymbol{\mathcal{Q}}\boldsymbol{V}^T\boldsymbol{S}^T}_{(II)} + \boldsymbol{S}\underbrace{\frac{1}{N}\left(\boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\mathcal{Q}}\boldsymbol{V}^T + \left(\boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\mathcal{Q}}\boldsymbol{V}^T\right)^T\right)}_{(III)}\boldsymbol{S}^T \tag{10}$$

where $\boldsymbol{\mathcal{Q}} = \boldsymbol{Q}_2^T\boldsymbol{Q}_2 = \boldsymbol{I} - \boldsymbol{U}^T\left(\boldsymbol{U}\boldsymbol{U}^T\right)^{-1}\boldsymbol{U}$
The equation (10) can be simplified as follows:

- Using the independence and ergodicity properties of the sequence $\{v_t\}$ and the fact that it is statistically uncorrelated with $\{u_t\}$, one can easily show that $(II) \rightarrow \sigma_v^2 \boldsymbol{I}_{fn_y}$ as $N \rightarrow \infty$.

- On the other hand, by developing (III) using the expression of $\boldsymbol{\mathcal{Q}}$ and writing

$$\frac{1}{N}\boldsymbol{X}\boldsymbol{V}^T = \frac{1}{N}\sum_{t=1}^{N} x_t v_t^T$$
$$= \frac{1}{N}\sum_{t=1}^{N}\left(\boldsymbol{A}^t x_0 \bar{v}_t^T + \sum_{j=1}^{t}\boldsymbol{A}^{j-1}Bu_{t-j}\bar{v}_t^T\right)$$

$\bar{v}_t = \begin{bmatrix} v_t^T & \cdots & v_{t+f-1}^T \end{bmatrix}^T$, and recalling that A is stable, it is straightforward to see that $(III) \to 0$ as $N \to \infty$.
- Similarly, when $N \to \infty$, (I) reduces to

$$(I) = \frac{1}{N}\mathbf{Z}\mathbf{Z}^T \quad \text{with} \quad \mathbf{Z} = \boldsymbol{\Gamma}^{(1)}\boldsymbol{X}$$

Since $\boldsymbol{\Gamma}^{(1)}$ is invertible, one realizes a similar transformation by setting $\mathbf{Z} = \boldsymbol{\Gamma}^{(1)}\boldsymbol{X}$. It is important to notice that assigning a value to $\boldsymbol{S}$ amounts to set a basis of the searched state space representation in (1). Let $\boldsymbol{\Sigma}_{\mathsf{z}} = \frac{1}{N}\mathbf{Z}\mathbf{Z}^T$ be the covariance matrix of z.
Therefore we have :

$$\begin{aligned}\boldsymbol{\Sigma} &:= \frac{1}{N}\boldsymbol{S}\boldsymbol{R}_{22}\boldsymbol{R}_{22}^T\boldsymbol{S}^T \\ &= \begin{bmatrix} \boldsymbol{\Sigma}_{\mathsf{z}} & \boldsymbol{\Sigma}_{\mathsf{z}}\boldsymbol{P}^T \\ \boldsymbol{P}\boldsymbol{\Sigma}_{\mathsf{z}} & \boldsymbol{P}\boldsymbol{\Sigma}_{\mathsf{z}}\boldsymbol{P}^T \end{bmatrix} + \begin{bmatrix} \sigma_v^2\boldsymbol{I} & 0 \\ 0 & \sigma_v^2\boldsymbol{I} \end{bmatrix} \qquad (11) \\ &= \left[\begin{array}{c|c} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \hline \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{array}\right]\end{aligned}$$

where σ_v^2 denotes the variance of $\{v_t\}$. Then the propagator can be estimated by minimising the following cost function

$$\|\boldsymbol{\Sigma}_{21} - \boldsymbol{P}\boldsymbol{\Sigma}_{11}\|_F^2, \qquad (12)$$

since

$$\boldsymbol{P} = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}. \qquad (13)$$

It should be noticed that the estimate obtained here for $\boldsymbol{P}$ may be biased in the presence of noise. An interesting way to suppress this inconvenient may be for example to use the instrumental variable method [8] which provides a theoretical satisfaction. Unfortunately, introducing an instrumental variable would probably not suffice to remove the noise effect and will particularly complicate the choice of the order detecting threshold as we will see in subsection III-D.

Once the propagator is estimated, the extended observability matrix can be obtained as

$$\boldsymbol{\Gamma}_f = \boldsymbol{S}^T\begin{bmatrix} \boldsymbol{I}_{n_x} \\ \boldsymbol{P} \end{bmatrix}. \qquad (14)$$

The matrices $\boldsymbol{A}$ and $\boldsymbol{C}$ of the system (1) are immediately extracted as follows

$$\boldsymbol{A} = \left(\boldsymbol{\Gamma}_f^{\uparrow}\right)^{\dagger}\boldsymbol{\Gamma}_f^{\downarrow}, \quad \boldsymbol{C} = \boldsymbol{\Gamma}_f(1:n_y,:),$$

with $\boldsymbol{\Gamma}_f^{\uparrow} = \boldsymbol{\Gamma}_f(1:(f-1)n_y,:)$, $\boldsymbol{\Gamma}_f^{\downarrow} = \boldsymbol{\Gamma}_f(n_y+1:fn_y,:)$. $\boldsymbol{B}$ and $\boldsymbol{D}$ can then be estimated by a linear regression from $\boldsymbol{C}$ and $\boldsymbol{A}$, assuming the system is asymptotically stable (see [8] for more details).

D. Order estimation

To estimate conveniently the matrix $\boldsymbol{P}$ by (13), an estimate of the system order n_x is essential. Under some mild assumptions, an order estimation procedure is suggested in this subsection. Generally from subspace identification schemes point of view, the order results almost always from an SVD of the matrix $\boldsymbol{R}_{22}$ in (9) for example. To that purpose, [1] defines the information criteria NIC and SVC similarly to the Akaike's criterion and based indeed on singular values. Here, we would like to characterize the order without resort to the very computationally expensive SVD so that an on-line application becomes possible. For that, assume the order of the system (1) has an available upper bound, which is anyway the basic requirement for any subspace identification method (f has to be set greater than n_x). Then, the main idea behind this method relies on exploiting the particularly interesting structure of the matrix $\boldsymbol{\Sigma}$. Firstly, two versions of our method in a deterministic framework are introduced. The stochastic case will be dealt in the last paragraph of this section.

1) Deterministic context: Without noise, $\sigma_v^2\boldsymbol{I}$ from equation (11) does not exist. The underlying idea in the strategy followed is to consider sequentially a submatrix of $\boldsymbol{\Sigma}$ in (11) of the form $\boldsymbol{\Delta}_r = \boldsymbol{\Sigma}(1:r, 1:r)$, r running from r_{min} towards r_{max} with $r_{min} < n_x < r_{max}$ (see fig. 1 for an illustration).

Assumption 1 *The covariance matrix $\boldsymbol{\Sigma}_{\mathsf{z}} = \frac{1}{N}\mathbf{Z}\mathbf{Z}^T$ of the vector $\mathsf{z} = \boldsymbol{\Gamma}^{(1)}x$ is positive definite, which is equivalent to say rank$(\boldsymbol{X}) = n_x$ that is, all the modes are sufficiently excited. Consequently any square submatrix of the form $\boldsymbol{\Delta}_r = \boldsymbol{\Sigma}_z(1:r, 1:r)$ is also positive definite.*

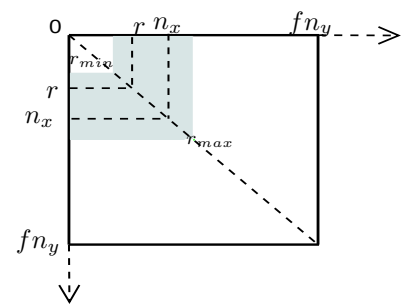


Fig. 1. Sequential slicing of matrix $\boldsymbol{\Sigma}$

In reference to this assumption, $\boldsymbol{\Delta}_r$ is invertible as long as $r \leq n_x$ but becomes non invertible as soon as $r > n_x$. These arguments justify the rank pursuit algorithm described below in two equivalent versions. In each version, a specific decision criterion h_{r+1} associated to every value of r allows to check whether r is the order or not.

Version 1: A first possibility may consist in computing recursively the inverse of $\boldsymbol{\Delta}_r$ extracted from $\boldsymbol{\Sigma}$ going from $r = r_{min}$ towards $r = r_{max}$ until the order is detected.

If $\boldsymbol{\Delta}_r$ is invertible, $\boldsymbol{\Delta}_{r+1}^{-1}$ may be computed using the matrix blocks inversion lemma (see appendix A in [7]) in

the following way :

$$\begin{aligned}\boldsymbol{\Delta}_{r+1}^{-1} &= \begin{bmatrix} \boldsymbol{\Delta}_r & \underline{w}_{r+1} \\ \underline{w}_{r+1}^T & s_{r+1} \end{bmatrix}^{-1} \\ &= h_{r+1}^{-1} \begin{bmatrix} h_{r+1}\boldsymbol{\Delta}_r^{-1} + \underline{\varphi}_{r+1}\underline{\varphi}_{r+1}^T & \underline{\varphi}_{r+1} \\ \underline{\varphi}_{r+1}^T & 1 \end{bmatrix} \end{aligned} \quad (15)$$

with

$$\begin{aligned} \underline{\varphi}_{r+1} &= -\boldsymbol{\Delta}_r^{-1}\underline{w}_{r+1} \in \mathbb{R}^r \\ h_{r+1} &= s_{r+1} + \underline{w}_{r+1}^T\underline{\varphi}_{r+1} \in \mathbb{R} \end{aligned} \quad (16)$$

The partition (15) is possible thanks to the symmetric structure of $\boldsymbol{\Sigma}$. The underlined elements refer to column vectors. Then, it follows from the formula (15) that if $\boldsymbol{\Delta}_r$ is nonsingular, so is $\boldsymbol{\Delta}_{r+1}$ if and only if $h_{r+1} \neq 0$.

Admitting $\boldsymbol{\Delta}_{r_{min}}^{-1}$ is known, an initial value for r is taken as $r = r_{min}$. One proceeds to the computation of h_{r+1}; if $h_{r+1} = 0$, then $\boldsymbol{\Delta}_{r+1}$ is singular and the conclusion $n_x = r$ is drawn; conversely, if $h_{r+1} \neq 0$, the recursion is pursued by computing $\boldsymbol{\Delta}_{r+1}$, and then h_{r+2} and so on. The procedure is stopped when it becomes evident that $\boldsymbol{\Delta}_{r+1}$ is singular. At the end of this loop

$$\boldsymbol{\Sigma}_{11}^{-1} = \boldsymbol{\Sigma}_{\mathsf{z}}^{-1} = \boldsymbol{\Delta}_r^{-1} \quad (17)$$

is known and $\boldsymbol{P}$ is computed as in equation (13).

Version 2: Another recursion option may be led by focusing on the value of $\boldsymbol{P}$ in (13). It presents notably the advantage of providing directly the value of $\boldsymbol{P}$ once the order is detected. Having $\boldsymbol{P}_r$ we would like to estimate $\boldsymbol{P}_{r+1}$. These matrices take the forms

$$\begin{aligned} \boldsymbol{P}_r &= \begin{bmatrix} \underline{w}_{r+1}^T \\ \boldsymbol{W} \end{bmatrix} \boldsymbol{\Delta}_r^{-1} = \begin{bmatrix} \underline{p}_r^T \\ \boldsymbol{\Pi}_r \end{bmatrix} \\ \boldsymbol{P}_{r+1} &= \begin{bmatrix} \boldsymbol{W} & \underline{\kappa} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Delta}_r & \underline{w}_{r+1} \\ \underline{w}_{r+1}^T & s_{r+1} \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \boldsymbol{\Omega}_{r+1} & \underline{\omega}_{r+1} \end{bmatrix} \end{aligned}$$

where $\boldsymbol{W} = \boldsymbol{\Sigma}(r+2 : fn_y, 1 : r)$, $\underline{w}_{r+1}^T = \boldsymbol{\Sigma}(r+1, 1 : r)$, $\underline{\kappa} = \boldsymbol{\Sigma}(r+2 : fn_y, r+1)$, $s_{r+1} = \boldsymbol{\Sigma}(r+1, r+1)$, p_r^T refers to the first row of P_r and Π_r, to the remaining part.

When r is incremented of 1, $\boldsymbol{P}_r$ loses a row and gains a column. Using the matrix identity (15), it is easy to reach the following adaptations laws:

$$\begin{aligned} \underline{\omega}_{r+1} &= \left(\underline{\kappa} - \boldsymbol{\Pi}_r\underline{w}_{r+1}\right)/h_{r+1}, \\ \boldsymbol{\Omega}_{r+1} &= \boldsymbol{\Pi}_r - \underline{\omega}_{r+1}\underline{p}_r^T \end{aligned} \quad (18)$$

The parameter of interest is now $h_{r+1} = s_{r+1} - \underline{p}_r^T\underline{w}_{r+1}$.

It is important to notice that, despite the benefits to compute directly $\boldsymbol{P}$, given the dimensions of the matrices involved, this second version may be sometimes costly in terms of computational time.

2) Stochastic context: In a stochastic context, the term $\sigma_v^2\boldsymbol{I}$ from equation (11) is no longer null. Then, in practice, h_{n_x+1} will probably be greater than zero, but the method could be efficiently performed using a convenient threshold comparison. Naturally, this threshold may depend on the level of the noise acting on the process. It needs also to be related to the system we wish to identify and hence, has to be computed or adapted somehow, particularly in the piecewise systems case (see section IV).

The presence of noise tends to increase all the quantities h_r but a gap is still observable in their values when the iteration process reaches the rank of $\boldsymbol{\Sigma}$ unless the noise is dominant compared to the signal. Note that, owing to the *assumption 1* and the Schur complement theorem (the parameter h_{r+1} is indeed the Schur complement of $\boldsymbol{\Delta}_r$ in $\boldsymbol{\Delta}_{r+1}$), all h_r are positive scalar for $r \leq n_x$. It follows from (11) that

$$\begin{aligned} h_{r+1} = s_{r+1} + \sigma_v^2 \\ - w_{r+1}^T\boldsymbol{\Delta}_r^{-1}\left(I_r + \sigma_v^2\boldsymbol{\Delta}_r^{-1}\right)^{-1} w_{r+1} \end{aligned} \quad (19)$$

If we make the hypothesis that all the eigenvalues of $\boldsymbol{\Delta}_r$ are significantly greater than the noise variance, then the spectral radius of $\sigma_v^2\boldsymbol{\Delta}_r^{-1}$ is in the unit circle. Therefore, expanding the term in brackets at the first order we get the following approximation

$$\begin{aligned} h_{r+1} \approx \left(s_{r+1} - w_{r+1}^T\boldsymbol{\Delta}_r^{-1}w_{r+1}\right) \\ + \left(\sigma_v^2 + \sigma_v^2 w_{r+1}^T\boldsymbol{\Delta}_r^{-2}w_{r+1}\right) \end{aligned} \quad (20)$$

which is composed of the useful signal part and the noise contribution. For $r = n_x$, the first term becomes null so that $h_{n_x+1} \approx \sigma_v^2\left(1 + \varphi_{n_x+1}^T\varphi_{n_x+1}\right)$ where φ_j is defined as in (16). Taking in account these remarks, a threshold can be chosen as

$$\text{Thres}(r) = T_0\left(1 + \varphi_{r+1}^T\varphi_{r+1}\right) \quad (21)$$

where T_0 denotes a constant, supposed to be slightly greater than the noise variance σ_v^2. When the order is detected, we can estimate *a posteriori* the noise variance in the following way since h_{n_x+1} has been computed.

$$\hat{\sigma}_v^2 = \frac{h_{n_x+1}}{1 + \varphi_{n_x+1}^T\varphi_{n_x+1}}$$

E. Recursive update of the matrix $\boldsymbol{\Sigma}$

In the previous parts, a complete off-line identification scheme has been investigated. From now on, we are interested in working out an on-line version of that procedure in order (as we will see in the last section) to apply it to switching systems estimation.

The on-line version of our algorithms relies on the recursive adaptation of the matrix $\boldsymbol{\Sigma}$ (see eq. (11)) through the new captured information. At each time instant, $\boldsymbol{\Sigma}$ is updated first and then, the procedure described above is run.

Assume that a QR factorization as in (8) is known at the instant $\bar{t}$. Then, at $\bar{t}+1$, a new data column is stacked as follows

$$\begin{bmatrix} \sqrt{\lambda}\begin{bmatrix} \boldsymbol{R}_{11}(\bar{t}) & \boldsymbol{0} \\ \boldsymbol{R}_{21}(\bar{t}) & \boldsymbol{R}_{22}(\bar{t}) \end{bmatrix} & \begin{matrix} u_f(\bar{t}+1) \\ y_f(\bar{t}+1) \end{matrix} \end{bmatrix}$$

with $y_f(\bar{t}+1) = \begin{bmatrix} y^T(\bar{t}-f+2) & \dots & y^T(\bar{t}+1) \end{bmatrix}^T$ and $\lambda < 1$ a forgetting factor. It has been shown in [8] that the

last data column can be zeroed out applying a sequence of Givens rotations as

$$\left[\begin{array}{c} \sqrt{\lambda}\left[\begin{array}{cc} \boldsymbol{R}_{11}(\bar{t}) & \mathbf{0} \\ \boldsymbol{R}_{21}(\bar{t}) & \boldsymbol{R}_{22}(\bar{t}) \end{array}\right] \quad \begin{array}{c} u_f(\bar{t}+1) \\ y_f(\bar{t}+1) \end{array} \end{array}\right] \mathbf{G}_1(\bar{t}+1)$$
$$= \left[\begin{array}{ccc} \boldsymbol{R}_{11}(\bar{t}+1) & \mathbf{0} & 0 \\ \boldsymbol{R}_{21}(\bar{t}+1) & \sqrt{\lambda}\boldsymbol{R}_{22}(\bar{t}) & \bar{y}_f(\bar{t}+1) \end{array}\right]$$

Then,

$$\boldsymbol{R}_{22}(\bar{t}+1) = \left[\sqrt{\lambda}\boldsymbol{R}_{22}(\bar{t}) \quad \bar{y}_f(\bar{t}+1)\right]$$

and $\boldsymbol{R}_{22}\boldsymbol{R}_{22}^T$ is updated as

$$\begin{aligned} &\boldsymbol{R}_{22}(\bar{t}+1)\boldsymbol{R}_{22}(\bar{t}+1)^T \\ &= \lambda\boldsymbol{R}_{22}(\bar{t})\boldsymbol{R}_{22}(\bar{t})^T + \underbrace{\bar{y}_f(\bar{t}+1)\bar{y}_f(\bar{t}+1)^T}_{\boldsymbol{\Phi}_R(\bar{t}+1)} \end{aligned}$$

and

$$\boldsymbol{\Sigma}(\bar{t}+1) = \lambda\boldsymbol{\Sigma}(\bar{t}) + \boldsymbol{S}\boldsymbol{\Phi}_R(\bar{t}+1)\boldsymbol{S}^T \tag{22}$$

As already mentioned, it is not of great interest to adapt entirely the matrix $\boldsymbol{\Sigma}$. If the order might vary then, assuming $\boldsymbol{S}$ is suitably set, it is essential to adapt all the first r_{max} columns or rows. After that, the order searching procedure is to be triggered on. Two choices are possible : either it is known that the order will always be greater than a certain number r_{min} and then $\boldsymbol{\Delta}_{rmin}^{-1}$ is recursively adapted (using the matrix inversion lemma) together with Σ, or this information is not available and the procedure is started from $r = 1$.

From $r = r_{min}$, the order searching recursion is then turning on up to the order n_x. The **Algorithm 1** below sums up the whole on-line procedure described previously.

Algorithm 1 On-line subspace tracking algorithm

- `Initialising: set` λ, T_0, f, $\boldsymbol{\Sigma}$, $\boldsymbol{\Delta}_{rmin}^{-1}$
- `FOR` $t = 1, \ldots, \infty$
  1) `Update the QR factorization of the data matrix as in subsection III-E`
  2) `Update` $\boldsymbol{\Sigma}$ `using (22)`
  3) `Update` $\boldsymbol{\Delta}_{rmin}^{-1}$ `using the matrix inversion lemma`
  4) `FOR` $r = r_{min}, \ldots, r_{max}$
     - `Compute` h_{r+1} `and Thres(`r`)`
     - `IF` $h_{r+1} \leq$ `Thres(`r`)` $n_x \leftarrow r$;
     - `Once the order is known, compute the propagator using the formula (13)`
     - `Deduce the system matrices as in subsection III-C` `BREAK;`
     - `ELSE` `Compute` $\boldsymbol{\Delta}_{r+1}^{-1}$ `by the formula (15);` $r \leftarrow r+1$;
     - `ENDIF`
  5) `ENDFOR`
- `ENDFOR`

IV. Application to switching systems estimation

A. Description of the procedure

Consider now a piecewise linear system described by the following state space representation $\mathcal{M}_i$

$$\mathcal{M}_i : \begin{cases} x_{t+1} = \boldsymbol{A}_i x_t + \boldsymbol{B}_i u_t \\ y_t = \boldsymbol{C}_i x_t + \boldsymbol{D}_i u_t + v_t \end{cases} \tag{23}$$

The embedded data equation in (5) still holds as long as all the data involved are generated by the same submodel. But when the system switches between two local models (from a model i to a model j for example), this equation is no longer acceptable in this form as there will be in the matrices $\boldsymbol{U}$ and $\boldsymbol{Y}$ data stemming from two different submodels. The figure below gives an illustration of the changes affecting $\boldsymbol{\Gamma}$ during the transition from i to j.

$$\begin{array}{ccccc} \tau-1 & & \tau & & \tau+1 \\ \left[\begin{array}{c} \boldsymbol{C}_i \\ \boldsymbol{C}_i\boldsymbol{A}_i \\ \vdots \\ \boldsymbol{C}_i\boldsymbol{A}_i^{f-2} \\ \boldsymbol{C}_i\boldsymbol{A}_i^{f-1} \end{array}\right] & \rightarrow & \left[\begin{array}{c} \boldsymbol{C}_i \\ \boldsymbol{C}_i\boldsymbol{A}_i \\ \vdots \\ \boldsymbol{C}_i\boldsymbol{A}_i^{f-2} \\ \mathsf{C}_j\boldsymbol{A}_i^{f-1} \end{array}\right] & \rightarrow & \left[\begin{array}{c} \boldsymbol{C}_i \\ \boldsymbol{C}_i\boldsymbol{A}_i \\ \vdots \\ \mathsf{C}_j\boldsymbol{A}_i^{f-2} \\ \mathsf{C}_j\mathsf{A}_j\boldsymbol{A}_i^{f-2} \end{array}\right] & \rightarrow \ldots \\ \boldsymbol{\Gamma}^i & & \boldsymbol{\Gamma}(\tau) & & \boldsymbol{\Gamma}(\tau+1) \end{array}$$

where matrices in $Roman$ correspond to the model before the switching time τ and matrices in script to the model after the switch.

In the neighborhood $[\tau^-\ ,\ \tau+k]$, $k \leq f$ of a commutation the QR factorization of the data matrix similarly as in (8) is as follows

$$\begin{bmatrix} \boldsymbol{U}^i & \boldsymbol{U}^{mix} \\ \boldsymbol{Y}^i & \boldsymbol{Y}^{mix} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{11} & 0 \\ \boldsymbol{R}_{21} & \boldsymbol{R}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_1^i & \boldsymbol{Q}_1^{mix} \\ \boldsymbol{Q}_2^i & \boldsymbol{Q}_2^{mix} \end{bmatrix}$$

where the superscript *mix* and i refer respectively to mixed data and pure data generated by the i-th submodel. Then, since

$$\left[\begin{array}{cc} \boldsymbol{Y}^i & \boldsymbol{Y}^{mix} \end{array}\right] = \left[\begin{array}{cc} \boldsymbol{\Gamma}^i\boldsymbol{X}^i & \boldsymbol{\Gamma}_x^{mix} \end{array}\right] + \left[\begin{array}{cc} \boldsymbol{H}^i\boldsymbol{U}^i & \boldsymbol{H}_u^{mix} \end{array}\right] + \boldsymbol{V}$$

one gets after some straightforward manipulations

$$\begin{aligned} \boldsymbol{R}_{22} =& \boldsymbol{\Gamma}^i\boldsymbol{X}^i\left(\boldsymbol{Q}_2^i\right)^T + \boldsymbol{V}\boldsymbol{Q}_2^T \\ &+ \underbrace{\left(\boldsymbol{\Gamma}_x^{mix} + \boldsymbol{H}_u^{mix} - \boldsymbol{H}^i\boldsymbol{U}^{mix}\right)\left(\boldsymbol{Q}_2^{mix}\right)^T}_{\text{mixed data}} \end{aligned} \tag{24}$$

where

$$\begin{aligned} \boldsymbol{H}_u^{mix} &= \left[\boldsymbol{H}_1^{mix}u_1^{mix} \quad \cdots \quad \boldsymbol{H}_k^{mix}u_k^{mix}\right] \\ \boldsymbol{\Gamma}_x^{mix} &= \left[\boldsymbol{\Gamma}_1^{mix}x_1^{mix} \quad \cdots \quad \boldsymbol{\Gamma}_k^{mix}x_k^{mix}\right] \end{aligned}$$

As long as only data from one submodel are present in the window, we have

$$\text{rank}\left(\boldsymbol{\Gamma}^i\boldsymbol{X}^i\left(\boldsymbol{Q}_2^i\right)^T\right) = \text{rank}(\boldsymbol{R}_{22}) = n_x^i$$

Hence, the **Algorithm 1** keeps tracking the rank of the current active submodel i until a switch occurs. From equation (24) the presence of mixed data is likely to increase the rank of the matrix $\boldsymbol{R}_{22}$. Hence, any change in the dynamics or in the zeroes of the system will be visible by the order estimation algorithm [1] since it induces a rank increase in the matrix $\boldsymbol{\Gamma}$ (see also [2]).

[1] even if it is not necessarily followed by an order change

Managing the transition period is a rather challenging problem essentially due to the recursive nature of the approach followed. One problem related to this period is for example the risk of a state space basis change underlined in [15]. At each commutation, the state was to be computed in order to bring all submodels in the same basis. Applying the propagator method in the switching systems identification context allows to get rid of this problem since the system matrices obtained remain in the same basis (as long as the permutation does not change as well as the order). We propose to use a procedure in two steps.

- Firstly, the detection of the switch is achieved. The parameters of the previous submodel are memorised and a new model created.
- Secondly, after the estimates have converged, a classification task tests whether the new captured model is already known or not.

The switches detection strategy is based on the assumption that the switching times are separated by a certain minimum delay T_s called the dwell time. This dwell time is supposed to be large enough to include the width f of the data window and to allow the convergence of the estimation procedure. The commutations could be state-driven, time-driven or event-driven provided that the minimum dwell time requirement is satisfied. In practice, when starting the algorithm with random values, a dwell time of about $T_s = (2-\lambda)\, f/\,(1-\lambda)$ samples is generally sufficient for achieving good convergence.

When the system enters the zone of transition, pursuing the update (with mixed data) will corrupt the obtained model for the submodel i. So, once the detection is done, the learning of the submodel i has to be immediately stopped, the final value (just before the switching occurs) of the submodel obtained to be recorded and the **algorithm 1** to be reinitialised with a new model. As, there may exist some delay δ between the actual switching instant τ and the detected one ($\hat{\tau}$), the parameters obtained at $\hat{\tau}-\delta$ are memorised instead.

Instead of creating continuously new models as new dynamics appear, it seems more convenient to classify the different submodels in order to recognise their upcoming occurrences. To that purpose, a classifier based on an appropriate metric (measure of similarity) may be used. Let

$$\underline{\theta}(t) = \text{vec}\left(\left[\begin{array}{c} \boldsymbol{D}(t) \\ \boldsymbol{\Gamma}(t)\boldsymbol{B}(t) \end{array}\right]\right)$$

where vec$(\cdot)$ is the vectorization operator. $\underline{\theta}(t)$ is a vector of dimension $(f+1)n_y n_u$ and characterises at each time the current submodel. The current submodel is compared to the already known submodels which have the same order by analyzing the similarity of $\underline{\theta}(t)$ with known parameters. Let consider for simplicity the euclidean distance as a similarity measure

$$d_i(t)^2 = \underline{e}_i(t)^T \underline{e}_i(t) \tag{25}$$

with $\underline{e}_i(t) = \underline{\theta}(t) - \underline{\theta}_i$ and $\underline{\theta}_i$ is the parameter of some recorded model i. When the new submodel is close enough to some existing submodel, the two submodels are merged. Otherwise, it is saved as being a novel submodel and labelled as different from all the others.

B. Simulation

In order to illustrate the procedure presented above let us consider a numerical simulation. Four linear submodels $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, $\mathcal{M}_4$ respectively of order 2, 2, 4, 3 are switching continuously.

$$\mathcal{M}_1 : \left\{ \begin{array}{ll} \boldsymbol{A}_1 = \left[\begin{array}{cc} 1.23 & -0.49 \\ 1 & 0 \end{array}\right] & \boldsymbol{B}_1 = \left[\begin{array}{c} 2 \\ 0 \end{array}\right] \\ \boldsymbol{C}_1 = \left[\begin{array}{cc} 0.864 & 0.755 \end{array}\right] & \boldsymbol{D}_1 = 1 \end{array} \right.$$

$$\mathcal{M}_2 : \left\{ \begin{array}{ll} \boldsymbol{A}_2 = \left[\begin{array}{cc} 0.36 & -0.5 \\ 0.5 & 0 \end{array}\right] & \boldsymbol{B}_2 = \left[\begin{array}{c} 2 \\ 0 \end{array}\right] \\ \boldsymbol{C}_2 = \left[\begin{array}{cc} 1.62 & 0.1 \end{array}\right] & \boldsymbol{D}_1 = 2 \end{array} \right.$$

$$\mathcal{M}_3 : \left\{ \begin{array}{ll} \boldsymbol{A}_3 = \left[\begin{array}{cccc} 0.563 & -0.31 & 0.46 & -0.314 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] & \boldsymbol{B}_3 = \left[\begin{array}{c} 1 \\ 0 \\ 0 \\ 0 \end{array}\right] \\ \boldsymbol{C}_3 = \left[\begin{array}{cccc} 0 & 0 & 0.1 & 1 \end{array}\right] & \boldsymbol{D}_3 = 0 \end{array} \right.$$

$$\mathcal{M}_4 : \left\{ \begin{array}{ll} \boldsymbol{A}_4 = \left[\begin{array}{ccc} 0.1725 & -0.032 & -0.65 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right] & \boldsymbol{B}_4 = \left[\begin{array}{c} 2 \\ 0 \\ 0 \end{array}\right] \\ \boldsymbol{C}_4 = \left[\begin{array}{ccc} 1.25 & .5 & 0 \end{array}\right] & \boldsymbol{D}_4 = 0 \end{array} \right.$$

The input signal is chosen as a white gaussian zero-mean noise of variance 1.5. The simulation is run with an additive output white noise such that SNR = 35 dB. The switching times are 300, 700, 1200, 1500 etc. We set $f = 7$, $\lambda = 0.9$ and we apply the **algorithm 1**.

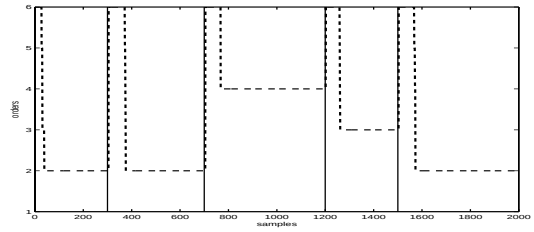
Figure 2 presents the order estimate and the evolution of system poles modulus. At each switching time we can notice that the order estimate increases suddenly up to $f-1$ even if there is no change in the order. This phenomenon is attributable to the presence of mixed data coming from two different submodels. The switching times are estimated with a relatively small delay (about 4 samples). On the other hand, since any transition lasts at least f samples the convergence delay of the order estimation algorithm after a switching occurs is necessarily higher than f (see fig. 2).

Figure 3 compares the two first h_r to the two first singular values of the matrix $\boldsymbol{R}_{22}$ from (8). There is manifestly a great similarity between them since the plots are in phase and seem to differ just by a scaling factor and a bias term.

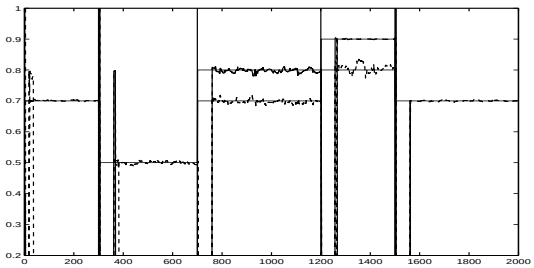
On the figure 4, we represent the evolution of the threshold in (21). That clearly demonstrates the necessity of choosing a model-dependent threshold. To further test the efficiency of the switches detection algorithm, a Monte-Carlo simulation of size 100 has been carried out. Without no additive noise we get 100 % of success in the detection and in the presence of noise (SNR=35dB) the performance changes slightly to be about 98 %.

V. Conclusion

This work demonstrates the possibility to identify on-line MIMO linear piecewise systems in a recursive way using a detection approach. The method proposed may be applied as well to time invariant systems as to slowly time varying systems.

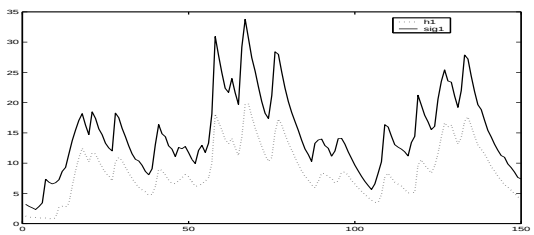


(a)

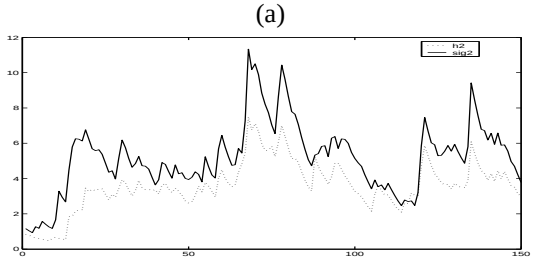


(b)

Fig. 2. (a): on-line stimation of the system order. The vertical lines indicate the switching times - (b): magnitude of local models poles estimated on-line



(a)



(b)

Fig. 3. Comparison of the singular values of matrix R_{22} in (8) and the quantities h_r (a): h_1 and σ_1- (b):h_2 and σ_2

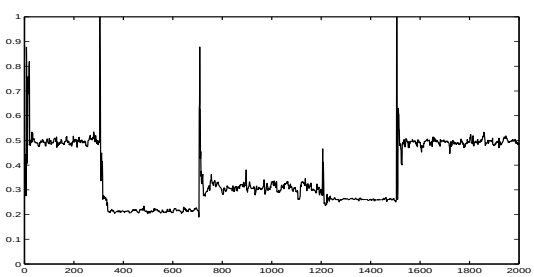


Fig. 4. Evolution of the threshold used for determining the order

A structured identification scheme known as propagator method has been conveniently prepared to be applied for bindly identifying online the submodels orders and parameters. The switching times are easily recognized since they are followed by an increase in the estimated order.

A noteworthy drawback of the off-line methods for piecewise linear systems identification is that the data basis which is used may often be incomplete so that some functioning mode would be ignored. The scheme proposed here allows to discard this problem but requires unfortunately a certain dwell time. Future work will include testing the proposed method on actual application and providing solutions to its underlined weakness.

REFERENCES

[1] Dietmar Bauer. Order estimation for subspace methods. *Automatica*, 37:1561–1573, 2001.

[2] José Borges, Vincent Verdult, Michel Verhaegen, and Miguel Ayala Botto. A switching detection method based on projected subspace classification. In *CDC-ECC05*, 2005.

[3] Giancarlo Ferrari-Trecate, Marco Muselli, Diego Liberati, and Manfred Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39:205–217, 2003.

[4] Tony Gustafsson. Recursive system identification using instrumental variable subspace tracking. In *11th IFAC Symposium on System identification, Fukuoka, Japan*, 1997.

[5] A. Lj. Juloski, W.P.M.H Heemels, G. Ferrari-Trecate, R. Vidal, S. Paoletti, and J.H.G Niessen. Comparison of four procedures for the identification of hybrid systems. In *HSCC*, 2005.

[6] A. Lj. Juloski, S. Weiland, and W.P.M.H. Heemels. A bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, 50, No 10:1520–1533, 2005.

[7] T. Kailath and A. H. Sayed. *Fast Reliable Algorithms for Matrices with Structure*. SIAM, PA, 1999.

[8] Marco Lovera, Tony Gustafsson, and Michel Verhaegen. Recursive subspace identification of linear and non-linear wiener state-space models. *Automatica*, 36:1639–1650, 2000.

[9] Guillaume Mercère. *Contribution à l'identification récursive des systèmes par l'approche des sous espaces*. PhD thesis, Université Scientifique et Technologique de Lille, 2004.

[10] Guillaume Mercère, Stéphane Lecoeuche, and Christian Vasseur. A new recursive method for subspace identification of noisy systems: Eivpm. In *13th IFAC Symposium on System Identification*, 2003.

[11] J. Munier and G. Y. Delisle. Spatial analysis using new properties of the cross spectral matrix. *IEEE Transactions on Signal Processing*, 39:746–749, 1991.

[12] Komi Midzodzi Pekpe, Gilles Mourot, Komi Gasso, and José Ragot. Identification of switching systems using change detection technique in the subspace framework. In *43rd IEEE Conference on Decision and Control, Atlantis, Paradise Island, Bahamas*, 2004.

[13] Jacob Roll, Alberto Bemporad, and Lennart Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40:37–50, 2004.

[14] P. Van Overschee and B. De Moor. *Subspace identification for linear systems. theory, implementation, applications*. Kluwer Academic Publishers, 1996.

[15] Vincent Verdult and Michel Verhaegen. Subspace identification of piecewise linear systems. In *43rd IEEE Conference on Decision and Control December 14-1 7,2004 Atlantis, Paradise Island, Bahamas*, Dec. 2004.

[16] Michel Verhaegen and Patrick Dewilde. Subspace model identification part 1: The output-error state space model identification class of algorithms. *International Journal of Control*, 56, no. 5:1187–1210, 1992.

[17] René Vidal. Identification of pwarx hybrid models with unknown and possibly different orders. In *Proceedings of American Control Conference, 2004.*, 2004.

[18] Bin Yang. Projection approximation subspace tracking. *IEEE Transaction on Signal Processing*, 43, No. 1:95–107, 1995.