HAL
open science

Modelling of non stationary systems based on a dynamical decision space

Stéphane Lecoeuche, Guillaume Mercère, Habiboulaye Amadou-Boubacar

To cite this version:

Stéphane Lecoeuche, Guillaume Mercère, Habiboulaye Amadou-Boubacar. Modelling of non stationary systems based on a dynamical decision space. 14th IFAC Symposium on System Identification, SYSID'06, Mar 2006, Newcastle, Australia. pp.CDROM. hal-00091332

HAL Id: hal-00091332

https://hal.science/hal-00091332v1

Submitted on 5 Sep 2006

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.

MODELLING OF NON STATIONARY SYSTEMS BASED ON A DYNAMICAL DECISION SPACE

S. Lecœuche * **G. Mercère** ** **H. Amadou-Boubacar** *

* *LAGIS - UMR CNRS 8146*
Département G.I.P., Mines de Douai, 59508 Douai Cedex
Phone: +33(0)327712445, Fax : +33(0)327712980
Email: lecoeuche@ensm-douai.fr
** *Laboratoire d'Automatique et d'Informatique Industrielle*
40 avenue du recteur Pineau, 86022 Poitiers, France
Phone: + 33 (0)549453509, Fax: +33 (0)549454034
Email: guillaume.mercere@univ-poitiers.fr

Abstract: A new approach based on pattern recognition techniques and dedicated to the monitoring of non stationary systems is presented in this paper. More precisely, it consists of a recursive subspace identification algorithm combined with an adaptive classifier set for non stationary environment. The system identification method which provides a recursive estimation of a linear state space model is firstly described. Then, a feature vector representing the system functioning state is extracted from this estimated model. Next, the dynamical clustering algorithm which online learns the functioning modes and continuously determines the current mode of the system is introduced. Its auto adaptive and unsupervised abilities to take into account system modes evolutions are finally emphasized on simulation examples.

Keywords: Monitoring, Pattern Recognition, System identification, Recursive algorithms, Subspace methods, Systems tracking, Non stationary systems, Classification.

1. INTRODUCTION

Supervision systems are more and more used in the framework of time varying processes. The treatment of such processes quite often leads to inopportune alarms and requires a regular maintenance of the references of the monitoring and diagnosis functions. The update of these decision criteria has to be done after establishing the current functioning mode and the definition of its tolerances in relation to the updated reference. Although supervision is an extensive studied research area, there are few works which consider the problem of the reference update. This paper tries to give some solutions to this problem. More precisely, the monitoring of non stationary systems is concerned. The non stationarities (modifying the static and/or dynamic behaviour of the system) considered hereafter are defined as evolutions of the model structure and/or parameters. A convenient way of representing these evolutions consists in considering two points of view:

- functioning modes change,
- functioning modes drift.

The proposed approach is based on the modelling of such systems with a set of several linear models which are representative of all the functioning modes. Thanks to this representation, the mode change can be modeled as a model commutation as illustrated on figure 1 where the functioning modes drift corresponds to a progressive change of the local models parameters. Thus, the global behaviour of the process can be modeled by a succession of modes which offer or not shifts or drifts. Unfortunately, this way of modelling leads to an exhaustive determination of all the models, all the commutation conditions and all their distortions in order to get a reliable tracking.

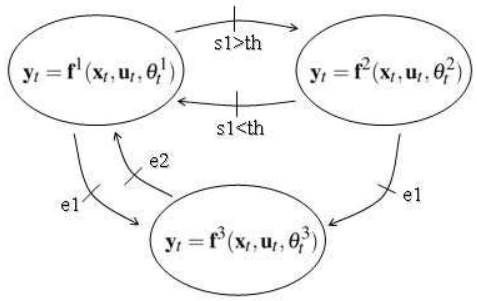

Fig. 1. Modelling based on a set of linear models.

This approach needs furthermore the knowledge of the current system mode any time. In order to not resort to the *a priori* information used by most of techniques based on the residuals or the multimodel identification (Simani *et al.*, 2002), the developed method rests on the pattern recognition where the modes recognition is realised thanks to an online learning.

2. PROBLEM FORMULATION AND APPROACH PRINCIPLE

Based on two consecutive recursive techniques, the considered approach is composed by the following stages :

(1) the extraction of a vector representative of the functioning state from a linear model online identified thanks to the use of a recursive algorithm;
(2) the determination of the functioning mode from a decision subspace continuously adapted by learning, the functioning modes being modelled by an online data clustering technique.

2.1 *Online identification*

Since the study of non stationary systems requires to have access to a reliable representation any time, a recursive identification algorithm is proposed to estimate a linear model of the tracked system at each data acquisition. The considered method has more precisely the goal to online determine a state space model representative of the functioning state of the process under the hypotheses that the order is *a priori* known and that the analysed system is stable and observable. The choice of such a representation can be justified by the fact that the state space realization is particularly adapted to MIMO processes and does not require any canonical parameterization. In this framework, the `EIVPM` algorithm (Mercère *et al.*, 2004) is used for its implementation straightforwardness and its low computational cost (Mercère *et al.*, 2004). From the estimated model, a representative vector of the functioning state is extracted.

2.2 *Dynamical data clustering*

The goal of this stage is to determine the actual functioning modes. The decision is based on a classification space obtained by online learning. Indeed, each mode is characterized by the class model label. The novelty of the proposed approach consists in exploiting a specific classification technique making possible the continuous modelling of the functioning modes. The used dynamical classifier algorithm (Lecœuche and Lurette, 2003) has the capacities to create new classes (unknown modes) and to adapt the known classes models when the data distributions which characterize them evolve or move. The variations resulting from non stationarities thus make the models of the classes, associated with the functioning modes, evolve or move in time. When a new observation is presented at the classification algorithm, the decision space is updated according to the information brought by the observation and the current functioning mode is determined (in terms of closer linear local model).

These two stages will be more precisely introduced in the following two sections.

3. RECURSIVE SUBSPACE IDENTIFICATION

The method considered in this paper lies on the recursive estimation of a linear state space model of fixed structure (the order is *a priori* estimated) by the subspace approach. This approach lets us indeed to online get the state space matrices $[\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}]$ of the linear model in the following noisy framework

$$\mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\tilde{\mathbf{u}}(t) + \mathbf{w}(t) \quad (1)$$
$$\tilde{\mathbf{y}}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\tilde{\mathbf{u}}(t) \quad (2)$$
$$\mathbf{u}(t) = \tilde{\mathbf{u}}(t) + \boldsymbol{\upsilon}(t) \quad (3)$$
$$\mathbf{y}(t) = \tilde{\mathbf{y}}(t) + \mathbf{v}(t) \quad (4)$$

where $\tilde{\mathbf{u}} \in \mathbb{R}^{n_u \times 1}$ and $\tilde{\mathbf{y}} \in \mathbb{R}^{n_y \times 1}$ are respectively the noise free input and output vectors, $\mathbf{x} \in \mathbb{R}^{n_x \times 1}$ the state vector, $\mathbf{w} \in \mathbb{R}^{n_x \times 1}$ the process noise and $\boldsymbol{\upsilon} \in \mathbb{R}^{n_u \times 1}$ and $\mathbf{v} \in \mathbb{R}^{n_y \times 1}$ respectively the input and output measurement noise. The principle of the proposed technique rests on the adaptation of the `MOESP` subspace model identification (SMI) schemes (Verhaegen, 1994) to the recursive estimation problem. This SMI class of algorithms relies on the following two steps:

- an input output (I/O) data matrices compression with the help of a QR factorization (Golub and Van Loan, 1996),
- the use of a singular value decomposition (SVD) (Golub and Van Loan, 1996) of particular submatrices obtained from the previous factorization leading to a consistent estimation of the extended observability matrix.

Since this last stage is not suitable for online implementation due to its computational complexity, several alternative algorithms have been developed in the recursive identification framework (Lovera *et al.*, 2000; Oku and Kimura, 2002; Lovera, 2003). Several studies (Mercère *et al.*, 2004; Mercère *et al.*, 2005) have enhanced some theoretical and practical benefits

of the adaptation of a particular array signal processing technique: the propagator method (Munier and Delisle, 1991). The main advantage of this approach lies in the use of a linear operator and unconstrained and unapproximated quadratic criteria which lead to easy recursive least squares algorithms. Among the different recursive subspace identification methods using the propagator, it has been chosen to apply the `EIVPM` algorithm (Mercère *et al.*, 2004) due to its implementation straightforwardness and its low computational cost. The first phase of this technique consists more precisely in the update of the `Ordinary MOESP` (Verhaegen and Dewilde, 1992) QR factorization at each new data acquisition as follows [1] (Lovera *et al.*, 2000; Mercère *et al.*, 2004):

$$\left[\sqrt{\lambda}\begin{bmatrix}\mathbf{R}_{11}(t) & \mathbf{0}\\ \mathbf{R}_{21}(t) & \mathbf{R}_{22}(t)\end{bmatrix} \begin{matrix}\mathbf{u}_f^+(t+1)\\ \mathbf{y}_f^+(t+1)\end{matrix}\right]\begin{bmatrix}\mathbf{Q}_1(t) & \mathbf{0}\\ \mathbf{Q}_2(t) & \mathbf{0}\\ \mathbf{0} & 1\end{bmatrix} \quad (5)$$

with

$$\mathbf{y}_f^+(t) = \left[\mathbf{y}^T(t) \ \cdots \ \mathbf{y}^T(t+f-1)\right]^T \in \mathbb{R}^{n_y f\times 1}, \quad (6)$$

and $f > n_x$. The update of equation (5) can then be realized by applying Givens rotations (Golub and Van Loan, 1996) in order to bring back the $\mathbf{R}$ factor to a lower triangular form (Lovera *et al.*, 2000). This operation lets us to extract a vector $\mathbf{z}_f^+ \in \mathbb{R}^{n_y f\times 1}$, named the observation vector, obtained by modifying $\mathbf{y}_f^+$ in order to include the information contained in $\mathbf{u}_f^+$ and $\left[\mathbf{R}_{11}^T \ \mathbf{R}_{21}^T\right]^T$. It was more particularly proved in (Mercère *et al.*, 2004) that

$$\mathbf{z}_f^+(t) = \mathbf{\Gamma}_f \mathbf{x}(t) + \mathbf{b}_f^+(t) \quad (7)$$

where $\mathbf{\Gamma}_f$ is the observability matrix:

$$\mathbf{\Gamma}_f = \left[\mathbf{C}^T \ (\mathbf{CA})^T \ \cdots \ \left(\mathbf{CA}^{f-1}\right)^T\right]^T \quad (8)$$

and $\mathbf{b}_f^+$ a vector bringing together all the noise terms. Knowing this observation vector, the `EIVPM` algorithm proposes to recursively estimate a basis of the observability matrix assuming that the studied system is observable. Under this hypothesis, since $\mathbf{\Gamma}_f \in \mathbb{R}^{n_y f\times n_x}$ with $n_y f > n_x$, $\mathbf{\Gamma}_f$ has at least n_x linearly independent rows, which can be gathered in a submatrix $\mathbf{\Gamma}_{f_1}$. The complement $\mathbf{\Gamma}_{f_2}$ of $\mathbf{\Gamma}_{f_1}$ can be expressed as a linear combination of these n_x rows. There is a unique linear operator $\mathbf{P}_f \in \mathbb{R}^{n_x\times(n_y f - n_x)}$, named propagator (Munier and Delisle, 1991), such that

$$\mathbf{\Gamma}_{f_2} = \mathbf{P}_f^T \mathbf{\Gamma}_{f_1}. \quad (9)$$

Furthermore, it is easy to verify that

$$\mathbf{\Gamma}_f = \begin{bmatrix}\mathbf{\Gamma}_{f_1}\\ \mathbf{\Gamma}_{f_2}\end{bmatrix} = \begin{bmatrix}\mathbf{\Gamma}_{f_1}\\ \mathbf{P}_f^T\mathbf{\Gamma}_{f_1}\end{bmatrix} = \begin{bmatrix}\mathbf{I}_{n_x}\\ \mathbf{P}_f^T\end{bmatrix}\mathbf{\Gamma}_{f_1} = \mathbf{E}_o\mathbf{\Gamma}_{f_1}. \quad (10)$$

Thus, since $\operatorname{rank}\left\{\mathbf{\Gamma}_{f_1}\right\} = n_x$,

$$\operatorname{span}_{\text{col}}\left\{\mathbf{\Gamma}_f\right\} = \operatorname{span}_{\text{col}}\left\{\mathbf{E}_o\right\}. \quad (11)$$

Equation (11) implies that it is possible to estimate the observability matrix (in a particular basis) by estimating the propagator. This operator can be determined by applying a data reorganization to (7) so that the first n_x rows of $\mathbf{\Gamma}_f$ are linearly independent

$$\mathbf{z}_f^+(t) = \begin{bmatrix}\mathbf{z}_{f_1}^+(t)\\ \mathbf{z}_{f_2}^+(t)\end{bmatrix} = \begin{bmatrix}\mathbf{I}_{n_x}\\ \mathbf{P}_f^T\end{bmatrix}\mathbf{\Gamma}_{f_1}\mathbf{x}(t) + \begin{bmatrix}\mathbf{b}_{f_1}^+(t)\\ \mathbf{b}_{f_2}^+(t)\end{bmatrix}, \quad (12)$$

then by considering the following quadratic criterion

$$J_{IV}(\mathbf{P}_f) = \left\|\mathbf{R}_{\mathbf{z}_{f_2}^+\boldsymbol{\xi}} - \mathbf{P}_f^T\mathbf{R}_{\mathbf{z}_{f_1}^+\boldsymbol{\xi}}\right\|_F^2 \quad (13)$$

where $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi\times 1}$ ($n_\xi \geq n_x$) is an instrumental variable assumed to be uncorrelated with the noise but sufficiently correlated with $\mathbf{x}$. The minimisation of the cost function (13) can be obtained by adapting the overdetermined instrumental variable technique first proposed in (Friedlander, 1984). The resulting algorithm is named `EIVPM` (Mercère *et al.*, 2004).

Once the observability matrix is estimated, the state space matrices extraction step can be considered. The matrices $\mathbf{A}$ and $\mathbf{C}$ are obtained by using the A-invariance property of $\mathbf{\Gamma}_f$

$$\hat{\mathbf{C}} = \hat{\mathbf{\Gamma}}_f(1:n_y,:) \quad (14)$$

$$\hat{\mathbf{A}} = \hat{\mathbf{\Gamma}}_f(1:n_y(f-1),:)^\dagger\,\hat{\mathbf{\Gamma}}_f(n_y+1:n_y f,:). \quad (15)$$

$\mathbf{B}$ and $\mathbf{D}$ are estimated from a classical linear regression [2] (see, *e.g.*, (Lovera *et al.*, 2000)).

4. ONLINE MODELLING OF MODES

The modes modelling tool treats the data extracted from the identified state space matrices as they arrive. The new information is incorporated continuously in order to redefine the structure of the functioning modes and thus to model continuously the decision space. Taking into account the various situations related to non stationarity environment requires the setting of specific adaptation rules through a continuous learning process. In the area of machine learning, some techniques exist with architectures exploiting incremental learning (Deng and Kasabov, 2003), (Eltoft and deFigueiredo, 1998), (Mouchaweh, 2004). Most of these algorithms present some disadvantages related to a coarse classes modelling and/or limited adaptation capacities in non stationary environment. In order to fill these gaps, two neural algorithms for the dynamic classification of the evolutionary data have been previously developed (Lecœuche and Lurette, 2003; Amadou-Boubacar and Lecœuche, 2005). These algorithms use a multi-prototype approach making possible to accurately model the structure of complex classes. In this paper, the AUDyC network based on a Gaussian modelling is used. Each functioning mode corresponds to a label of a complex class which could be defined by an assembly of

[1] The forgetting factor λ is introduced to weight the past information.

[2] The evolution of the $\mathbf{B}$ and $\mathbf{D}$ matrices is not considered in this paper.

Gaussian prototypes. The activation function of each hidden neuron determines the membership degree μ_j^t of the observation X_t to one prototype Φ_j. In order to obtain a fine classes representation, this one is based on the Mahanalobis distance

$$\mu_j^t = \exp\left(\frac{1}{2}\left(X_t - \overline{X}_j\right)^T * \Sigma_j^{-1} * \left(X_t - \overline{X}_j\right)\right) \quad (16)$$

where $\overline{X}_j$ and Σ_j are respectively the center and the covariance matrix of the prototype. The use of the membership function allows the implementation of the learning rules. With the first acquisition X_1, the network is initialized: creation of the first prototype Φ_1 constituting the first class Ω_1 (first functioning mode). The prototype is parameterized by its center $\overline{X}_1 = X_1$ and an initial covariance matrix Σ_{ini} beforehand selected. Then, according to new acquisitions, various situations can arise by comparison of the membership degree with two fixed thresholds μ_{min} and μ_{max} (resp. limit of prototype and class membership). Each case leads to a specific procedure (see Table 1).

Table 1. Classifier Adaptation rules.

	If	Then
1	$\mu_j^t < \mu_{min} \forall j \in \{1..J\}$	Creation $\Phi_{new} \in \Omega_{new}$
2	$\exists \Phi_j \in !\Omega_i, \mu_{min} < \mu_j^t < \mu_{max}$	Creation $\Phi_{new} \in \Omega_i$
3	$\exists \Phi_j, \mu_{max} < \mu_j^t$	Adaptation $\widehat{\Phi}_j = \Phi_j(X_t)$
4	$\exists \Phi_l \in !\Omega_p \cup$ $\exists \Phi_m \in !\Omega_q, \mu_{min} < \mu_{l,m}^T < \mu_{max}$	Ambiguity $X_t \in \chi_{amb}$

Then, the AUDyC learning process is established in three principal phases:

4.1 *First phase: classification*

The classification stage corresponds to the creation and adaptation of prototypes and classes. In cases 1 and 2 of table 1, the observation X_t is not close to any existing prototype. These cases are similar to a distance rejection which could be used to detect the novelty in the multiclass environment. If the observation is not sufficiently close to any class (case 1), it leads to the creation of a new prototype and a new class corresponding to a new system mode. In the case 2, a new prototype is created and affected to the nearest class in order to contribute to a better definition of the mode model. In situation 3, the observation is rather close to a prototype to take part in its definition. The functioning mode adaptation is then carried out by using the following recursive equations

$$\overline{X}_j^t = \overline{X}_j^{old} + \frac{1}{N_p}(X_t - X_{t-N_p+1}) \quad (17)$$

$$\Sigma_j^t = \Sigma_j^{old} + \Delta X \begin{pmatrix} \frac{1}{N_p} & \frac{1}{N_p(N_p-1)} \\ \frac{1}{N_p(N_p-1)} & \frac{-(N_p+1)}{N_p(N_p-1)} \end{pmatrix} \Delta X^T \quad (18)$$

with $\Delta X = \left[X_t - \overline{X}_j^{old} \quad X_{i-N+1} - \overline{X}_j^{old}\right]$, N_p : prototype size.

4.2 *Second phase: fusion*

The case 4 of table 1 depicts the case of the rejection in ambiguity when an observation is sufficiently close to two or several prototypes (e.g. l, m) to contribute to their structure. The fusion procedure consists in evaluating the similarity of two densities by using an acceptance criterion based on the Kullback-Leibler distance (Zhou and Chellappa, 2004). When this criterion is higher than a threshold, the different classes (e.g. p, q) merge onto a unique new functioning mode.

4.3 *Third phase: evaluation*

The evaluation phase is significant to eliminate the parasite prototypes and classes possibly created by the noise influence. To detect not-representative modes, this phase is based on the cardinality of the models. For example, if the number of allocated data is less than the NP_{min} (NC_{min}) threshold, the prototype (the class) is eliminated .

For more details on the AUDyC network, the reader can consult (Lecœuche and Lurette, 2003). A similar technique based on SVM technique can also be found in (Amadou-Boubacar and Lecœuche, 2005).

5. SIMULATIONS

This last part relates to the application of the suggested method. Only results coming from simulated systems are given. Two cases are more precisely considered in the following subsections.

5.1 *Commutation of functioning modes*

The first simulation shows the abilities of the presented approach to model and to analyze non linear system that could be modelled as a linear model set.

$$\mathbf{x}(t+1) = \mathbf{A}_i\mathbf{x}(t) + \begin{bmatrix} 0 & 0 \\ -0.6 & 0 \\ 0 & 0.5 \end{bmatrix} \tilde{\mathbf{u}}(t) + \mathbf{w}(t) \quad (19)$$

$$\mathbf{y}(t) = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}(t) + \mathbf{v}(t) \quad (20)$$

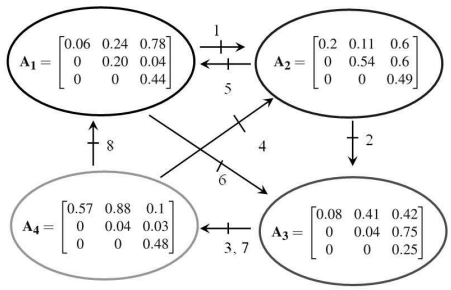

Fig. 2. Evolutions of the **A** matrix.

The system inputs are white Gaussian noises of variance 1. The system is disturbed by an output colored measurement noise (variance 0.03) and by a colored process noise (variance 0.05). During the simulation, the **A** matrix parameters change according to temporal

events. Figure 2 shows the change directions for each numbered event. At the simulation beginning, both algorithms are initialized ($f = 6$, $\lambda = 0.995$) under the assumption[3] that the system order is *a priori* known. Then in a recursive way, the EIVPM technique identifies the process and determines the state matrices corresponding to the linear model best approximating the process. The eigenvalues vp_i^t of the matrix $\hat{\mathbf{A}}$ constitute the X_t observation vector used to determine the functioning mode: $X_t = \left[vp_i^t \; vp_2^t \; vp_3^t\right]$ (see figure 3). In fact, this vector consists in all the monitored parameters. For more complex applications, its dimension could be increased by adding the **B** and **C** coefficients.

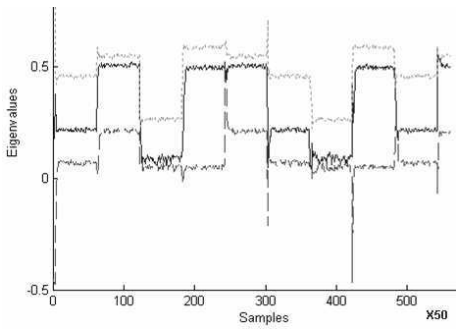

Fig. 3. The eigenvalues of the $\hat{\mathbf{A}}$ matrix online estimated with EIVPM.

The X_t vector observation is then directly sent to the AUDyC network. From this information, the dynamical classifier updates the decision space (functioning modes models) and determines the current functioning mode. For the whole of this study, the parameters are fixed as follows: $\Sigma_{ini} = 0.05, \mu_{min} = 0.015, \mu_{max} = 0.02, NP_{min} = 50, N_P = 500$ and $N_{amb} = 5$. For more information about the choice of the parameters, the reader can refer to (Amadou-Boubacar *et al.*, 2005).

Figure 4.a illustrates the final representation space of the raw data (X_t). In fact, the data are online classified in a sequential way. From the first acquisition, the AUDyC network initializes the first class corresponding to the initial functioning mode of the system. When changes occur, an observation fast drift is detected until stabilization. A new class is created when the number of stable observations exceeds the NP_{min} threshold. This threshold can be interpreted as the minimum duration making it possible the characterization of a stable mode. New classes are created in this way. The figure 4.b gives the final classes locations.

The current functioning mode is determined by using the membership degree of the observation (membership ratio rule). On figure 5, it can be noticed that the observations located between modes are non classified and the class creation (4 first situations) is effective after an extra delay corresponding to the NP_{min} threshold. When the mode is already known, the decision is done quicker.

[3] No mode is *a priori* known.

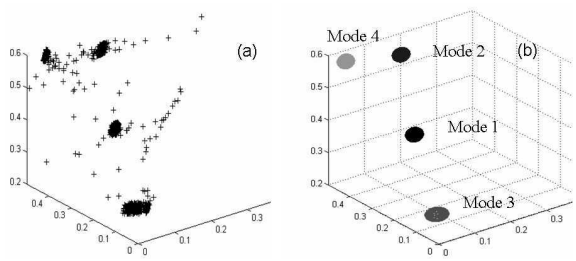

Fig. 4. Modes learning and class jump detection.

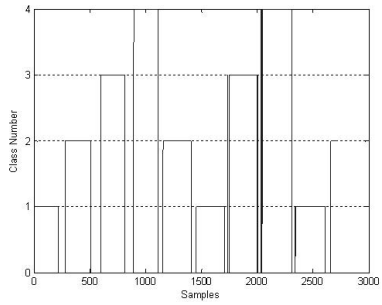

Fig. 5. Classification of the observations.

5.2 *Evolution of a functioning mode*

This second simulation characterizes a progressive mode evolution. The used system has the same representation as the previous (see (19)), only the **A** matrix changes in order to describe two mode jumps followed by a drifting mode

$$\mathbf{A}(t) = \begin{bmatrix} v1 & -0.4 & 0.2 \\ 0 & v2 & -0.5 \\ 0 & 0 & v3 \end{bmatrix} \tag{21}$$

with

$$\left[v1 \; v2 \; v3\right] = \begin{cases} \left[0.8 \; -0.6 \; 0.1\right] & \text{for } 0<t\leq 1000 \\ \left[0.65 \; -0.45 \; 0.1\right] & \text{for } 1000<t\leq 2000 \\ \left[0.65 \; -0.65 \; 0.05\right] & \text{for } 2000<t\leq 2780 \\ \left[0.65 \; -0.65 \; 0.05\right] - \left[0.05 \; 0.05 \; 0.2\right] * \frac{t}{2780} & \text{else.} \end{cases} \tag{22}$$

Figure 6 gives the appearances of the decision space at different instants. The decision space is adapted by the integration of new knowledge.

Concerning the class representing the mode 3, its model is correctly defined at every moment (Figure 6.b to 6.d) by using the prototype adaptation rules. Figure 7 shows that mode commutations have been detected. For the class 3, the current mode is still the same even if its model evolves. Of course, from this point, expert knowledge should be introduced in order to define if this evolution is "normal" (running in periods, chemical transition, ...) or if a progressive failure appears to the process.

6. CONCLUSION

In this paper, a new approach has been presented for the online determination of the functioning mode of

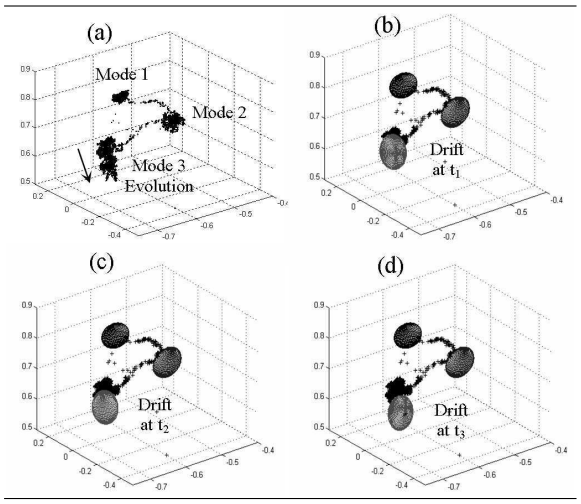

Fig. 6. Class adaptation according to the evolution of a functioning mode.

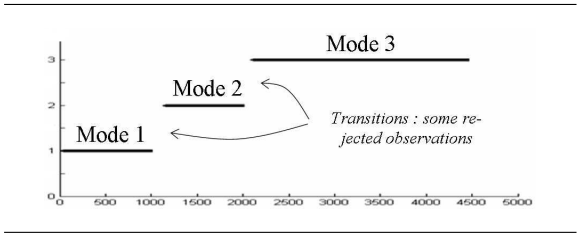

Fig. 7. Observation labels.

a non stationary system. This approach is based on a recursive identification tool coupled with a dynamic classification algorithm. Thanks to its recursive rules, this dynamic classifier online models the functioning modes. The interest of this approach is to take into account commutations and evolutions of system modes without *a priori* knowledge in order to continuously determine the current functioning mode of the system. The simulation results have shown the efficiency of this approach. New tools for jumps and drifts modes detection are being studied in order to propose a complete supervisory module.

7. REFERENCES

Amadou-Boubacar, H. and S. Lecœuche (2005). A new kernel-based algorithm for online clustering. In: *7th International Conference on Artificial Neural Networks*. Warsaw, Poland.

Amadou-Boubacar, H., S. Lecœuche and S. Maouche (2005). AUDyC neural network using a new Gaussian densities merge mechanism. In: *7th International Conference on Adaptive and Natural Computing Algorithms*. Coimbra, Portugal. pp. 155–158.

Deng, D. and N. Kasabov (2003). On-line pattern analysis by evolving self-organizing maps. *Neurocomputing* **51**, 87–103.

Eltoft, T. and R. deFigueiredo (1998). A new neural network for cluster-detection-and-labeling. *IEEE Transactions on Neural Networks* **9**, 1021–1035.

Friedlander, B. (1984). The overdetermined recursive instrumental variable method. *IEEE Transactions on Automatic Control* **4**, 353–356.

Golub, G. H. and C. F. Van Loan (1996). *Matrix Computations*. 3rd ed.. John Hopkins University Press. Baltimore MD.

Lecœuche, S. and C. Lurette (2003). Auto-adaptive and dynamical clustering neural network. In: *13th International Conference on Artificial Neural Networks* (ICANN03, Ed.). Istanbul, Turkey. pp. 350–358.

Lovera, M. (2003). Recursive subspace identification based on projector tracking. In: *13th IFAC Symposium on System Identification*. Rotterdam, The Netherlands.

Lovera, M., T. Gustafsson and M. Verhaegen (2000). Recursive subspace identification of linear and non-linear Wiener state-space models. *Automatica* **36**, 1639–1650.

Mercère, G., S. Lecœuche and C. Vasseur (2005). Sequential correlation based propagator algorithm for recursive subspace identification. In: *16th IFAC World Congress*. Praha, Czech Republic.

Mercère, G., S. Lecœuche and M. Lovera (2004). Recursive subspace identification based on instrumental variable unconstrained quadratic optimization. *International Journal of Adaptive Control and Signal Processing* **18**, 771–797.

Mouchaweh, M. (2004). Diagnostic in real time for evolutionary processes in using pattern recognition and possibility theory. *International Journal of Computational Cognition* **2**(1), 79–112.

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

Oku, H. and H. Kimura (2002). Recursive 4SID algorithms using gradient type subspace tracking. *Automatica* **38**, 1035–1043.

Simani, S., C. Fantuzzi and R.J. Patton (2002). *Model-based fault diagnosis in dynamic systems using identification techniques*. Advances in Industrial Control Series. 1st ed.. London, UK.

Verhaegen, M. (1994). Identification of the deterministic part of MIMO state space models given in innovations form from input output data. *Automatica* **30**, 61–74.

Verhaegen, M. and P. Dewilde (1992). Subspace model identification part 2: analysis of the elementary output-error state space model identification algorithm. *International Journal of Control* **56**, 1211–1241.

Zhou, S. K. and R. Chellappa (2004). Kullback-Leibler distance between two Gaussian densities in reproducing kernel Hilbert space. In: *International Symposium on Information Theory*. Chicago, USA.