De l'identification des systèmes (hybrides et à sortie binaire) à l'extraction de motifs

Abdelhak Goudjil

Normandie Université

THESE

Pour obtenir le diplôme de doctorat

Spécialité : Automatique, Signal, Productique, Robotique

Préparée au sein de l'Université de Caen Normandie

De l'identification des systèmes (hybrides et à sortie binaire) à l'extraction de motifs

Présentée et soutenue par
Abdelhak GOUDJIL

Thèse soutenue publiquement le 07 décembre 2017 devant le jury composé de		
Monsieur Mohammed M'SAAD	Professeur des universités, ENSICAEN.	Directeur de thèse
Monsieur Rachid MALTI	Professeur des universités, Université de Bordeaux.	Rapporteur
Monsieur Tarek RAISSI	Professeur des universités, Conservatoire National des Arts et Métiers.	Rapporteur
Madame Houria SIGUERDIDJANE	Professeur des universités, Centrale Supélec.	Examinateur
Monsieur Mathieu POULIQUEN	Maître de conférences, Université de Caen Normandie.	Examinateur
Monsieur Eric PIGEON	Maître de conférences, Université de Caen Normandie.	Examinateur

Thèse dirigée par Pr. Mohammed M'SAAD, laboratoire d'Automatique de Caen

UNIVERSITÉ CAEN NORMANDIE

MIIS

LAC

Remerciements

Ce manuscrit est le résultat d'un travail de recherche de près de trois ans. En préambule, je veux adresser tous mes remerciements aux personnes avec lesquelles j'ai pu échanger et qui m'ont aidé pour réaliser ce travail.

En commençant par remercier tout d'abord Monsieur **Mohammed M' SAAD**, Professeur à l'ENSICAEN, d'avoir accepté de diriger ce travail, pour son soutien et pour ses encouragements. Je tiens également à remercier et exprimer mes profonds respects aux membres du jury d'avoir acceptés de juger et d'examiner ce travail.

Je tiens à exprimer toute ma reconnaissance à Monsieur **Mathieu POULIQUEN**, à Monsieur **Eric PIGEON** et à Monsieur **Olivier GÉHAN**, Maîtres de Conférences à l'université de Caen et à l'ENSICAEN, d'avoir accepté d'encadrer mes travaux de thèse avec professionnalisme, rigueur, patience, disponibilité et générosité. Ils m'ont guidé tout au long de ces trois années et m'ont fait gravir avec succès toutes les étapes qui jalonnent les trois années de thèse. Encore merci pour votre confiance et votre sympathie.

Je tiens à présent à remercier tous les membres de l'équipe Automatique qui m'ont permis de réaliser cette thèse dans les meilleures conditions, pour leurs encouragements et les services qu'ils m'ont rendus avec gentillesse tout au long de ces trois années.

Enfin, j'adresse mes plus sincères remerciements à ma famille, mes parents, mes frères, mes sœurs et tous mes proches et amis, qui m'ont accompagné, aidé, soutenu et encouragé tout au long de la réalisation de ce travail.

Table des matières

Introduction générale

Contexte et motivations

Les travaux de thèse présentés dans ce manuscrit s'inscrivent dans le cadre de deux axes de recherche actifs comme l'illustre la figure 1. Le premier axe de recherche concerne l'étude de certains problèmes d'identification des systèmes non linéaires. L'identification des systèmes est une opération qui permet à partir de données entrée/sortie, d'extraire un modèle dynamique du système. Ce modèle permet une meilleure compréhension du système, dans l'optique de faire du contrôle ou du diagnostic. L'identification des systèmes n'est pas un axe de recherche nouveau, il s'est développé et est utilisé tant dans la recherche que dans des applications industrielles depuis des décennies. Les méthodes d'identification les plus courantes sont dédiées à l'identification des systèmes linéaires. Aujourd'hui l'identification des systèmes linéaires est arrivée à maturité et elle fait l'objet de nombreuses applications dans l'industrie. En comparaison, l'identification des systèmes non linéaires est encore en phase de développement et reste à ce jour un axe de recherche loin d'être épuisé. Ceci est dû à la nature complexe des systèmes non linéaires et à l'existence d'une grande variété de non-linéarités. L'intérêt de l'identification des systèmes non linéaires réside dans le besoin de piloter des systèmes avec plus de précision et d'efficacité.

Nous nous intéressons particulièrement aux problèmes d'identification de deux variétés de systèmes non linéaires. La première variété est celle des systèmes dynamiques hybrides. Un système dynamique hybride est un système décrit par une interaction entre des phénomènes à évolution continue dans le temps et des phénomènes à évolution discrète dans le temps. Il s'agit d'une famille de sous-systèmes linéaires gouvernés par un état discret qui indique le sous-système actif à chaque instant. Les systèmes hybrides sont utilisés dans de nombreuses applications (biologie, robotique, réseaux, $\cdots$) pour approximer des dynamiques non linéaires via la commutation d'une manière adéquate entre un ensemble de sous-systèmes linéaires évoluant chacun dans une zone de fonctionnement spécifique.

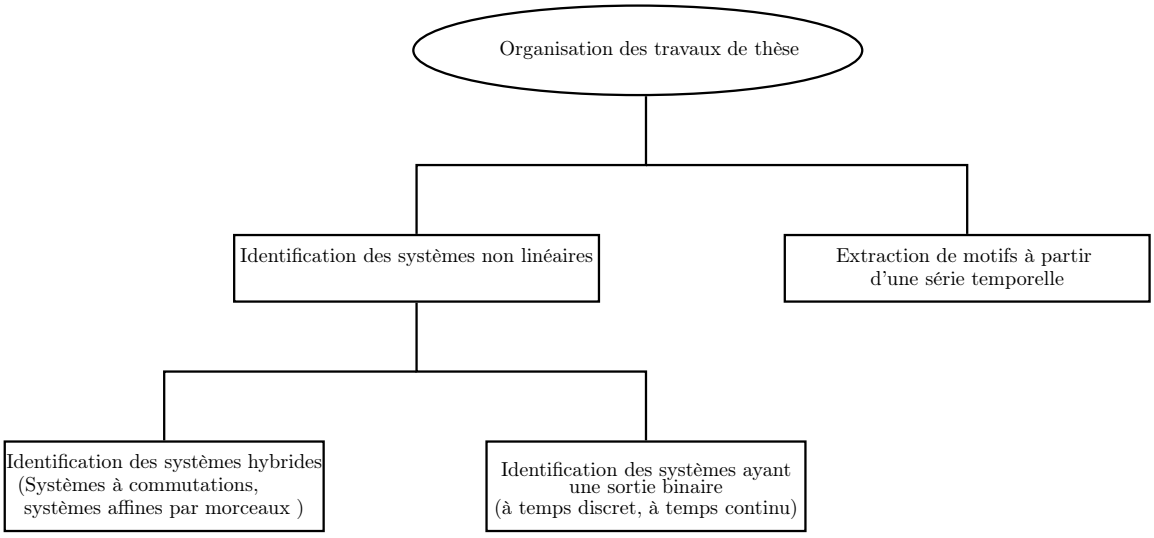

FIGURE 1 – Structuration des travaux de thèse.

A titre d'exemple, dans un système de contrôle d'énergie dans une maison intelligent, l'évolution dynamique de la température est un phénomène continu mais, cette évolution continue est en interaction avec des phénomènes à évolution discrète dans le temps comme l'ouverture/fermeture des fenêtres, des portes, l'allumage etc $\cdots$ ([109]).

La deuxième variété de systèmes non linéaires est celle des systèmes dynamiques linéaires ayant une sortie binaire. La sortie binaire résulte du fait de la présence d'une non-linéarité produite par un capteur binaire. L'identification d'un tel système est un problème délicat du fait de la binarisation du signal de sortie du système. L'étude du problème d'identification des systèmes linéaires ayant une sortie binaire est motivée par certaines applications dans plusieurs domaines telles que : l'utilisation des capteurs intelligents dans les systèmes de surveillance, l'utilisation des capteurs binaires dans les processus chimiques afin d'indiquer le niveau de la pression ou le niveau du liquide et l'utilisation des capteurs binaires dans les systèmes de commutation dans le domaine de réseaux et télécommunications.

Le deuxième axe de recherche développé dans ce manuscrit concerne l'extraction de motifs à partir de données. Les travaux consacrés à cet axe sont développés dans le cadre du projet CASSIE (Customized Artificial Skills for Semantic driven Interoperable objects Experiences), dont les objectifs principaux consistent à développer et disséminer une plateforme d'usages interopérables d'objets connectés, pilotée par l'utilisateur en langage

naturel. Le problème de l'extraction de motifs à partir des données a suscité l'intérêt de nombreux chercheurs dès son apparition au début des années 1990. L'extraction de motifs s'est développée dans le cadre de la fouille des données et c'est devenue un problème classique. Son développement est motivé par l'accroissement des capacités de stockage et de gestion de grandes masses de données, ainsi que la nécessité imposée par le besoin de valoriser les données accumulées dans des bases de données. L'extraction de motifs a pour but de présenter à l'utilisateur final des connaissances et des informations utiles à partir de grandes masses de données. Dans certains cas, l'extraction de motifs peut aussi constituer une alternative de stockage de données en gardant uniquement les informations utiles afin de consommer nettement moins d'espace de stockage.

Organisation et contribution

Ce manuscrit est organisé en trois parties de la façon suivante :

- **Partie I : Identification des systèmes hybrides.**

Un système hybride est un système dynamique commutant entre différents modes de fonctionnement, le plus souvent supposés linéaires. En identification des systèmes hybrides, nous distinguons notamment les deux classes très populaires suivantes : l'identification des systèmes linéaires à commutations et l'identification des systèmes affines par morceaux. Un système linéaire à commutations est défini comme un ensemble de sous-systèmes linéaires dynamiques indexés par un état discret qui indique le sous-système actif à chaque instant. La sortie du système à chaque instant est la sortie de l'un de ces sous-systèmes. Le mode de commutation entre les sous-systèmes peut-être complètement arbitraire. Un système affine par morceaux est un système linéaire à commutations particulier où le mode de commutation est déterminé par une partition polyédrique de l'espace de régression.

En terme de contenu, dans cette première partie, nous présentons dans un premier temps un état de l'art sur les méthodes d'identification des systèmes à commutations et les méthodes d'identification des systèmes affines par morceaux (cf. chapitre 1). La majorité des méthodes existantes décrivent généralement les systèmes linéaires à commutations par un modèle de régression de type ARX à commutations et les systèmes affines par morceaux par un modèle de régression de type ARX affine par morceaux. Peu de méthodes décrivent les systèmes linéaires à commutations (respectivement les systèmes affines par morceaux) par un modèle du type erreur de sortie à commutations (respectivement un

modèle du type erreur de sortie affine par morceaux). De plus, la majorité des méthodes solutionne le problème d'identification sous l'hypothèse usuelle de bruit stochastique. La prise en compte de cette hypothèse nécessite une connaissance à priori de la caractéristique statistique du bruit affectant la sortie du système. Dans certains cas, le bruit ne peut être modélisé comme un processus stochastique (erreur de modélisation, perturbation déterministe basse fréquence, etc $\cdots$). Une solution alternative plus appropriée suivant le contexte est de considérer une hypothèse sur l'existence uniquement d'une borne supérieure sur l'amplitude du bruit. Nos travaux dans cette partie s'inscrivent dans le cadre de cette solution alternative. Afin d'aborder la résolution de ce problème d'identification sous l'hypothèse du bruit borné, les méthodes d'identification ensemblistes "Set-Membership" sont considérées. Nous nous intéressons plus particulièrement aux algorithmes ellipsoïdaux qui présentent des avantages certains en terme d'une faible complexité de calcul et d'une simplicité de mise en œuvre en temps réel.

Nous proposons dans cette partie un algorithme d'identification en temps réel des systèmes à commutations soumis à un bruit borné (cf. chapitre 2). L'algorithme consiste en deux étapes, à chaque instant, la première étape permet de classifier un point de données en l'associant au sous-système actif à l'instant considéré et la deuxième étape permet de mettre à jour le vecteur de paramètres associé au sous-système actif. À la différence de la plupart des méthodes disponibles dans la littérature, nous effectuons une analyse de convergence et de comportement de l'algorithme. Nous proposons aussi plusieurs extensions de l'algorithme soit pour l'identification des systèmes affines par morceaux (cf. chapitre 3), l'identification des systèmes à commutations décrits par un modèle du type erreur de sortie à commutations (cf. chapitre 4) et l'identification des systèmes à commutations multi-entrées/multi-sorties (cf. chapitre 2). Les travaux présentés dans cette partie font l'objet des publications suivantes : [73], [74], [75], [77] et [76].

- **Partie II : Identification des systèmes ayant une sortie binaire.**

Dans cette deuxième partie, nous intéressons à l'identification des systèmes ayant une sortie binaire en exploitant des techniques de la classification supervisée. Ce problème d'identification des systèmes ayant une sortie binaire est délicat à résoudre à cause de la nature binaire du signal de sortie. Afin d'aborder ce problème d'identification, nous introduisons un point de vue original pour l'identification des systèmes ayant une sortie binaire. Nous formulons le problème d'identification des systèmes ayant une sortie binaire comme un problème de classification binaire : le vecteur d'observation est interprété comme un

point dans l'espace et la sortie binaire du système est interprétée comme l'étiquette d'une classe. Cette formulation est basée sur un résultat d'identifiabilité et permet d'envisager l'utilisation d'une méthode de classification supervisée. Plusieurs méthodes de la classification supervisée sont disponibles dans la littérature, nous nous intéressons plus particulièrement aux SVMs car ils présentent des propriétés et des performances intéressantes. Le succès des SVMs est justifié par les fondements théoriques solides, la capacité de traiter un grand volume de données avec une charge de calcul raisonnable et la robustesse des SVMs vis-à-vis du niveau de bruit.

En terme de contenu, dans cette partie, nous présentons dans un premier temps les fondements mathématiques des SVMs en détaillant les équations aboutissant à l'écriture du problème d'optimisation primal et son écriture duale (cf. chapitre 5). Dans un second temps, nous proposons deux algorithmes d'identification basés sur l'utilisation des SVMs. Le premier algorithme est dédié à l'identification des systèmes à temps discret ayant une sortie binaire (cf. chapitre 6). L'algorithme permet de surmonter certaines difficultés des méthodes disponibles dans la littérature. Le deuxième algorithme est dédié à l'identification des systèmes à temps continu ayant une sortie binaire (cf. chapitre 7). À noter qu'à notre connaissance, aucun algorithme d'identification n'a été proposé dans la littérature pour l'identification des systèmes à temps continu ayant une sortie binaire. Les travaux présentés dans cette partie font l'objet des publications suivantes : [78] et [156].

- **Partie III : Extraction de motifs à partir des données.**

Dans cette partie, nous détaillons nos travaux au sein du projet CASSIE sur l'extraction de motifs à partir des données. Nos travaux se concentrent principalement dans le Work Package 1 "IA ambiante et Objets connectés". Conformément aux objectifs initiaux énoncés par les différents partenaires dans le projet (Digital Airways, L&J, Popsicube, Bodycap, Laboratoire d'Informatique de Grenoble, GREYC), notre objectif est la mise en œuvre de stratégies de traitement automatique, non supervisé, de grandes masses de données issues de capteurs de différentes natures. Il s'agit de réaliser l'agrégation et la modélisation de ces données via l'extraction de motifs récurrents. La connaissance de ces motifs (leurs occurrences, leurs profils, leurs fréquences d'apparition, etc $\cdots$) doit notamment permettre d'identifier des activités, des comportements habituels ou la détection d'événements anormaux afin de les signaler à l'utilisateur. Le développement de ce moteur de traitement de données est réalisé notamment en collaboration avec Digital Airways afin que les résultats de ce moteur soient exploitables dans un contexte industriel par Digital Airways.

En terme de contenu, dans cette partie, nous présentons dans un premier temps un bref aperçu sur les algorithmes d'extraction de motifs disponibles dans la littérature (cf. chapitre 8) et un état de l'art sur les techniques de la classification non supervisée (cf. chapitre 9). Dans un second temps, nous proposons un algorithme d'extraction de motifs à partir d'une série de données temporelles (cf. chapitre 10). L'algorithme d'extraction de motifs permet de rechercher et de localiser des motifs et les retourner sous forme d'une bibliothèque de motifs. Ces motifs représentent potentiellement des comportements récurrents utiles et interprétables. L'algorithme d'extraction de motifs consiste en briques logicielles écrites en C/C++ afin de pouvoir assurer un maximum de portabilités. Plusieurs versions de ces briques logiciels ont été livrées à Digitil Airways dans le cadre du projet CASSIE, avec entre 2 versions successives, une réduction de la charge de calcul, une simplification de la mise en œuvre et une amélioration des performances en terme d'extraction de motifs et de capacité de traitement de données.

Première partie

De la classification à l'identification des systèmes hybrides

Chapitre 1

État de l'art sur l'identification des systèmes hybrides

Un système hybride est un système décrit par une interaction entre une partie à évolution continue dans le temps et une partie à évolution discrète dans le temps. Un tel système est régi par l'agrégation d'un nombre fini de sous-systèmes de nature différente. Une variété de classes des systèmes hybrides sont disponibles dans la littérature. Dans ce chapitre, nous présentons un état de l'art sur l'identification de deux classes des systèmes hybrides très populaires, les systèmes linéaires à commutations et les systèmes affines par morceaux.

Sommaire

1.1 Identification des systèmes dynamiques hybrides

L'identification des systèmes hybrides est devenu récemment un axe de recherche actif et qui a attiré l'attention d'un nombre important de chercheurs. Cela est dû à leurs nombreuses applications et à leur capacité de modéliser certains systèmes non linéaires. En effet, la commutation d'une manière adéquate entre un ensemble de sous-systèmes linéaires évoluant chacun dans une zone de fonctionnement spécifique peut permettre de décrire la dynamique d'un système non linéaire. L'identification des systèmes hybrides a été appliquée avec succès à une grande variété de problèmes dans plusieurs domaines (plus d'exemples d'applications dans [98], [177], [68]) :

- les systèmes de contrôle du trafic aérien ([72], [161]) ;
- la segmentation vidéo et la vision par ordinateur ([192],[20], [190]) ;
- les procédés chimiques [121] ;
- la biologie ([155], [154], [41]) ;
- la robotique ([168], [30]) ;
- les réseaux de communication ([71],[119]).

Dans la littérature, on trouve une variété de classes pour décrire les systèmes hybrides. Parmi ces classes on peut citer :

- les systèmes linéaires à commutations SLS (Switched Linear Systems) [177] ;
- les systèmes affines par morceaux PWA (PieceWise Affine systems) [39] ;
- les systèmes dynamiques à logique mixte MLD (Mixed Logical Dynamical systems) [14] ;
- les systèmes linéaires à complémentarité LC (Linear Complementarity systems) [48].

En identification des systèmes hybrides, on distingue les deux classes très populaires suivantes : l'identification des systèmes linéaires à commutations SLS et l'identification des systèmes affines par morceaux PWA. La plupart des travaux d'identification des systèmes hybrides dans la littérature sont dédiés à l'identification des ces deux classes. Un système linéaire à commutations est défini comme un ensemble de nombre fini de sous-systèmes linéaires dynamiques indexés par un état discret. L'état discret indique le sous-système actif à chaque instant. La sortie du système à chaque instant est la sortie de l'un de ces sous-systèmes. Le mode de commutation entre les sous-systèmes peut être complètement arbitraire. Si le mode de commutation est déterminé par une partition polyédrique de l'espace de régression, le système est appelé système affine par morceaux.

Les méthodes d'identification des systèmes à commutations et systèmes affines par morceaux peuvent être divisées en deux catégories : les méthodes basées sur des modèles entrée-sortie et les méthodes des sous-espaces. Les méthodes des sous-espaces ([10], [17], [152], [185]) souffrent du problème lié à la pluralité des représentations d'état possibles pour un même système. De plus, ces méthodes ont besoin de supposer que les instants de commutation sont séparés par une certaine durée minimale appelée temps de séjour. À la différence, les méthodes basées sur des modèles entrée-sortie sont plus efficaces et plus adaptées quand le mode de commutation est arbitraire, elles ne nécessitent pas d'hypothèse sur la durée entre deux commutations. Nous nous intéressons dans cette partie aux méthodes d'identification basées sur des modèles entrée-sortie.

Concrètement, la plupart des méthodes existantes basées sur des modèles entrée-sortie décrivent généralement les systèmes linéaires à commutations par un modèle de régression de type ARX à commutations "Switched Auto Regressive eXogenous model (SARX)" et la plupart des travaux sur l'identification des systèmes affines par morceaux décrivent cette classe par un modèle de régression de type ARX affine par morceaux "PieceWise Affine Auto Regressive eXogenous model (PWARX)". Certaines méthodes ont été développées en décrivant les systèmes linéaires à commutations par un modèle du type erreur de sortie à commutations "Switched Output Error model (SOE)" et les systèmes affines par morceaux par un modèle du type erreur de sortie affine par morceaux "PieceWise Output Error model (PWOE)". Un excellent tutoriel sur l'identification des systèmes hybrides a été présenté dans [151]. Il donne un aperçu général sur les approches d'identification des systèmes linéaires à commutations et les systèmes affines par morceaux développés avant 2007. Un autre tutoriel a été présenté dans [68] sur les méthodes qui ont été conçues entre 2007 et 2012. Dans la suite de ce chapitre, nous présentons un état de l'art sur l'identification des systèmes hybrides décrits par les modèles SARX, PWARX, SOE et PWOE respectivement.

1.2 Identification des systèmes à commutations décrits par un modèle SARX

Dans cette section, nous présentons quelques méthodes développées dans la littérature pour l'identification des systèmes à commutations décrits par un modèle SARX. La plupart de ces méthodes sont développées en mode hors-ligne et sous l'hypothèse d'absence

de bruit ou avec l'hypothèse habituelle du bruit stochastique. Nous décrivons plus particulièrement l'approche algébrique ([193], [187], [129], [11], [191]). Nous mentionnons aussi les méthodes basées sur l'approche d'optimisation ([147], [6], [118], [149], [9], [84] et [148]). A la fin de cette section, nous parlons aussi brièvement sur les méthodes d'identification en temps réel des systèmes hybrides décrits par le modèle SARX.

Afin de présenter le principe des méthodes mentionnées précédemment, nous présentons d'abord le problème d'identification des systèmes à commutations décrits par un modèle SARX.

On considère un système discret linéaire à commutations décrit par un modèle SARX sous forme entrée-sortie comme suit :

$$y_k = -\sum_{j=1}^{n_a(\lambda_k)} a_j(\lambda_k) y_{k-j} + \sum_{i=0}^{n_b(\lambda_k)} b_i(\lambda_k) u_{k-i} + e_k \tag{1.1}$$

- $\{a_j(\lambda_k)\}_{j=1}^{n_a(\lambda_k)}$ et $\{b_i(\lambda_k)\}_{i=1}^{n_b(\lambda_k)}$ sont les paramètres associés au sous-modèle λ_k.
- $\lambda_k \in \{1, \cdots, s\}$ est l'état discret du système avec s le nombre de sous-modèles.
- $\{n_a(i), n_b(i)\}_{i=1}^{i=s}$ sont les ordres des différents sous-modèles.
- $u_k \in \mathbb{R}$ est l'entrée du système et $y_k \in \mathbb{R}$ est la sortie du système.
- $e_k \in \mathbb{R}$ représente le terme de bruit supposé en général comme un processus stochastique.

Problème d'identification : *Étant donné l'ensemble des observations* $\{u_k, y_k\}$ *générées par le modèle SARX de la forme (1.1), estimer à la fois :* $\{n_a(i), n_b(i)\}_{i=1}^{i=s}$ *les ordres de différents sous-modèles, le nombre de sous-modèles* s*, les paramètres de chaque sous-modèle* $\{a_j(\lambda_k)\}_{j=1}^{n_a(\lambda_k)}$ *et* $\{b_i(\lambda_k)\}_{i=1}^{n_b(\lambda_k)}$ *et l'état discret* $\{\lambda_k\}_{k=1}^{N}$ *avec* N *désigne le nombre de données disponibles.*

La résolution de ce problème d'identification est une tâche très complexe. Il s'agit d'estimer le nombre de sous-modèles, l'état discret en associant chaque donnée au sous-modèle le plus approprié et d'estimer les paramètres de tous les sous-modèles ainsi que leurs ordres. Dans une procédure d'identification réelle, la difficulté de ce problème d'identification dépend de la quantité d'information supposée être connue ou fixe a priori. Une hypothèse assez courante consiste à supposer que les ordres des différents sous-modèles sont fixes, égaux et connus à priori. Dans ce cas, le problème d'identification revient à distribuer les données dans s hyperplans. Cette hypothèse est justifiée par le fait

qu'un tel problème d'identification à partir d'un ensemble fini d'observations souffre de la non-unicité de la solution dans le cas où n_a et n_b ne sont pas fixés [189]. Cette hypothèse est utilisée dans la majorité des méthodes ([7], [6], [118], [8] et [148]). Certaines méthodes [193], [129] supposent la connaissance des bornes supérieurs $\{\bar{n}_a(i),\bar{n}_b(i)\}_{i=1}^{i=s}$ sur les ordres de différents sous-modèles. De plus, la majorité des méthodes requièrent la connaissance du nombre de sous-modèles s a priori ou la connaissance d'une borne supérieure $\bar{s} \geq s$ sur s afin d'estimer celui-ci à partir de données.

1.2.1 L'approche Algébrique

La méthode proposée dans [193] et [129] considère le problème d'identification des systèmes à commutations décrits par un modèle SARX comme un problème géométrique algébrique. L'idée est de reformuler le problème d'identification comme un problème d'estimation puis de différentiation d'un polynôme homogène.

Afin de présenter le principe de l'approche algébrique, considérons le modèle SARX donné par (1.1). L'objectif de l'approche algébrique est de fournir une solution analytique pour le cas déterministe ($e_k = 0$), qui se rapproche du cas stochastique pour un niveau de bruit faible. Les ordres de différents sous-modèles $\{n_a(i),n_b(i)\}_{i=1}^{i=s}$ peuvent être différents. De plus, ils peuvent être inconnus, seules des bornes supérieures $\{\bar{n}_a(i),\bar{n}_b(i)\}_{i=1}^{i=s}$ sont nécessaires. Pour des raisons de simplification, nous supposerons que tous les sous-modèles ont les mêmes ordres : $n_a(1) = \cdots = n_a(s) = n_a$ et $n_b(1) = \cdots = n_b(s) = n_b$. Par conséquent, (1.1) peut se réécrire sous la forme suivante :

$$y_k = -\sum_{j=1}^{n_a} a_j(\lambda_k) y_{k-j} + \sum_{i=0}^{n_b} b_i(\lambda_k) u_{k-i} \tag{1.2}$$

Nous définissons $\bar{\theta}_{\lambda_k} \in \mathbb{R}^n$ et $\bar{\phi}_k \in \mathbb{R}^n$ avec $n = n_a + n_b + 2$ par :

$$\bar{\theta}_{\lambda_k} = \begin{bmatrix} 1 & -a_1(\lambda_k) & \cdots & -a_{n_a}(\lambda_k) & -b_0(\lambda_k) & -b_1(\lambda_k) & \cdots & -b_{n_b}(\lambda_k) \end{bmatrix}^T$$
$$\bar{\phi}_k = \begin{bmatrix} -y_k & -y_{k-1} & \cdots & -y_{k-n_a} & u_k & u_{k-1} & \cdots & u_{k-n_b} \end{bmatrix}^T$$

En prenant en compte ces définitions, le modèle (1.2) peut se traduire comme suit : $\forall k$, il existe au moins un $\lambda_k = i \in \{1,\cdots,s\}$ tel que $\bar{\theta}_i$ et $\bar{\phi}_k$ satisfont : $\bar{\theta}_i^T \bar{\phi}_k = 0$. Il en résulte que l'équation polynomiale suivante doit être satisfaite par les paramètres du système et les données d'entrée/sortie pour n'importe quel mécanisme de commutation :

$$P_s(\bar{\phi}_k) \doteq \prod_{i=1}^{s} (\bar{\theta}_i^T \bar{\phi}_k) = 0 \tag{1.3}$$

Cette équation polynomiale est appelée la contrainte de découplage hybride (”hybrid decoupling polynomial (HDP)). Elle permet d'éliminer l'état discret en prenant le produit des équations définissant chaque sous-modèle ARX. Ceci conduit à une équation algébrique. La contrainte de découplage hybride est tout simplement un polynôme multivariable homogène de degré s en n variables :

$$P_s(z) \doteq \prod_{i=1}^{s}(\bar{\theta}_i^T z) \doteq \sum h_{s_1,\cdots,s_n} z_1^{s_1}\cdots z_n^{s_n} = h^T v_s(z) = 0 \tag{1.4}$$

$h_{s_1,\cdots,s_n}$ est le coefficient de monôme $z_1^{s_1}\cdots z_n^{s_n}$, $v_s(z) : \mathbb{R}^n \mapsto \mathbb{R}^{M_s(n)}$, $M_s(n) = \dfrac{(n+s-1)!}{s!(n-1)!}$, est l'application de Veronese de degré s (plus de détails dans [82], [191]), $h \in \mathbb{R}^{M_s(n)}$ est le vecteur de paramètres hybride qui représente le produit tensoriel symétrique des paramètres : $h = Sym(\bar{\theta}_1 \bigotimes \cdots \bigotimes \bar{\theta}_s)$.

La contrainte de découplage hybride peut être écrite en termes de polynômes en appliquant (1.4) à l'ensemble des données entrées-sorties $\{\bar{\phi}_k\}_{k=\max(n_a,n_b)}^{N-1+\max(n_a,n_b)}$ avec N le nombre de données disponibles, on obtient le système d'équations linéaires suivant :

$$L_s h = \begin{bmatrix} v_s(z_{\max(n_a,n_b)})^T \\ \vdots \\ v_s(z_{N-1+\max(n_a,n_b)})^T \end{bmatrix} h = 0_{N\times 1} \tag{1.5}$$

Si le nombre de sous-modèles est inconnu, une condition sur le rang des matrices $L_{\bar{s}}$, construites à partir des applications de Veronese de degrés $\bar{s}$, peut être utilisée pour déterminer le nombre s de sous-modèles [193]. Afin de procéder à l'estimation des vecteurs de paramètres $\{\bar{\theta}_i\}_{i=1}^{s}$, nous considérons-la dérivée de $P_s(z)$:

$$\nabla P_s(z) = \frac{\partial P_s(z)}{\partial z} = \sum_{j=1}^{s}\prod_{i\neq j}(\bar{\theta}_i^T z)\theta_j \tag{1.6}$$

Si z appartient à l'hyperplan $H_j = \{z : \bar{\theta}_j^T z = 0\}$. En prenant en compte le fait que le premier élément de $\bar{\theta}_i$ est égal à 1, on en déduit :

$$\bar{\theta}_j = \frac{\nabla P_s(z)}{e_1^T \nabla P_s(z)} \,|_{z\in H_j} \tag{1.7}$$

avec $e_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T \in \mathbb{R}^n$. Ainsi, il est possible d'estimer les paramètres du système directement à partir de la dérivée du polynôme $P_s(z)$ à partir d'une collection de s points se trouvant dans les s hyperplans respectivement.

Une technique permettant de trouver au moins un point z_j dans chaque hyperplan $\{z_j \in H_j\}_{j=1}^{s}$ est présentée dans [193], [191]. Une fois que les vecteurs de paramètres $\{\bar{\theta}_j\}_{j=1}^{s}$ sont obtenus, l'état discret peut être facilement identifié par :

$$\lambda_k = \underset{i=1,\cdots,s}{\operatorname{argmin}} (\bar{\theta}_i^T \bar{\phi}_k)^2 \tag{1.8}$$

L'approche algébrique est une méthode appropriée pour l'identification des systèmes à commutations décrits par un modèle SARX dans le cas où le bruit est nul. Elle permet de trouver une estimation exacte des vecteurs de paramètres indépendamment de l'état discret. Toutefois, l'inconvénient majeur de cette approche est sa forte sensibilité au bruit comparativement aux autres approches ([194], [196], [100]). En effet, en présence de bruit, la contrainte de découplage hybride n'est plus vérifiée et les résultats d'estimations peuvent être aberrants et éloignés des vrais vecteurs de paramètres. Un autre point faible de cette approche est que la complexité et le temps de calcul nécessaire pour résoudre le problème d'identification augmentent d'une manière rapide avec l'augmentation du nombre de sous-systèmes et leurs dimensions [118]. Cependant, il peut être utilisé pour initialiser une autre méthode d'identification comme indiqué dans [193]. Dans [11], une généralisation de l'approche algébrique pour l'identification des paramètres du modèle MIMO SARX a été proposée.

1.2.2 L'approche basée sur l'optimisation

Les méthodes développées au sein de cette approche supposent que le nombre de sous-modèles s est connu et que les ordres de différents sous-modèles sont fixes, égaux et connus à priori. Le modèle SARX sous forme entrée-sortie (1.1) peut se réécrire comme suit :

$$y_k = \phi_k^T \theta_{\lambda_k} + e_k \tag{1.9}$$

avec ϕ_k est le vecteur de régression donné par :

$$\phi_k = [-y_{k-1} \ \cdots \ -y_{k-n_a} \ \ u_k \ \ u_{k-1} \ \cdots \ u_{k-n_b}]^T \tag{1.10}$$

θ_{λ_k} est le vecteur de paramètres associé au sous-modèle désigné par λ_k donné par :

$$\theta_{\lambda_k} = [a_1(\lambda_k) \ \ \cdots \ \ a_{n_a}(\lambda_k) \ \ b_0(\lambda_k) \ b_1(\lambda_k) \ \ \cdots \ \ b_{n_b}(\lambda_k)]^T \tag{1.11}$$

En utilisant (1.9), le problème d'identification du modèle SARX décrit précédemment peut être refondu dans le problème d'optimisation suivant :

$$\begin{cases} \min\limits_{\theta_i,\beta_{k,i}} \sum\limits_{k=1}^{N} \sum\limits_{i=1}^{s} \ell(y_k - \phi_k^T \theta_i)\beta_{k,i} \\ \\ s.c \quad \sum\limits_{i=1}^{s} \beta_{k,i} = 1, \qquad \forall k \\ \beta_{k,i} \in \{0,1\}, \qquad \forall k,i \end{cases} \tag{1.12}$$

où $\ell(.)$ est une fonction non négative, généralement choisie comme étant la norme 2 ou la norme 1. La variable binaire $\beta_{k,i}$ détermine le sous-modèle auquel le point (ϕ_k, y_k) est attribué. L'état discret peut être reconstruit de la manière suivante :

$$\lambda_k = i \quad \Leftrightarrow \quad \beta_{k,i} = 1 \tag{1.13}$$

Ce problème d'optimisation peut être résolu directement en utilisant des techniques de programmation mixte en nombres entiers comme proposé dans [166]. Malheureusement, cette méthode ne peut être appliquée qu'à des petits ensembles de données à cause de la non-convexité du problème et de sa complexité algorithmique.

Plusieurs méthodes pour l'identification des systèmes à commutations proposées au cours des dernières années reposent sur des théories d'optimisation et de relaxation du problème d'optimisation (1.12) ou sur une reformulation de ce dernier. Parmi ces méthodes, on peut citer celles proposées dans ([147], [6], [118], [149], [9], [96], [84] et [148]).

Dans [84], les auteurs proposent une méthode en trois étapes basée sur la résolution d'un problème d'optimisation convexe régularisé, suivie d'une étape de classification (clustering), donnant une solution partielle au problème. Cette solution partielle est intégrée dans le problème d'origine afin de le faire apparaitre comme un problème convexe. Enfin, ce problème convexe est résolu dans la troisième étape, ce qui donne une solution approchée. Une propriété avantageuse de la méthode est qu'elle repose sur un seul paramètre de synthèse. D'autre part, on peut citer la non-adaptation en temps réel et la complexité de calcul (à cause de l'étape d'optimisation et de clustering) comme des inconvénients pour cet algorithme. De plus, la solution reste approchée à cause de la régularisation du problème d'optimisation.

L'approche d'optimisation restreinte ("sparse optimisation") présentée dans [6] pose formellement le problème d'identification de chaque sous-modèle comme un problème

d'optimisation combinatoire basé sur la minimisation de la norme l_0. Il s'agit d'un problème de minimisation NP-difficile (NP-hard). Des conditions nécessaires disponibles dans la littérature permettent de relaxer le problème d'optimisation basé sur la minimisation de la norme l_0 à un problème d'optimisation basé sur la minimisation de la norme l_1 ([6]). La procédure d'identification permet d'extraire l'ensemble des vecteurs de paramètres $\{\theta_i\}_{i=1}^{s}$ associés aux différents sous-modèles l'un après l'autre sans séparation préalable des données selon leurs sous-modèles. Par conséquent, la méthode ne peut identifier qu'un seul vecteur de paramètres à la fois. Lors de l'identification d'un vecteur de paramètres d'un sous-modèle, les données venant d'autres sous-modèles sont traitées comme des valeurs aberrantes. Une fois un modèle est identifié, l'algorithme cherche à identifier un autre vecteur de paramètres jusqu'à l'identification de s vecteurs de paramètres. Nous présentons ci-dessous brièvement le principe de cette méthode.

Considérons l'ensemble des observations $\{\phi_k, y_k\}_{k=1}^{N}$ générées par un système linéaire à commutations décrit par (1.9). Introduisons la définition du vecteur d'erreur comme suit :

$$\Phi(\theta) = y - X^T\theta - e \tag{1.14}$$

où $y = [y_1 \cdots\ y_N]^T$, $e = [e_1\ \cdots\ e_N]^T$, $X = [\phi_1\ \cdots\ \phi_N]$ et θ un vecteur de paramètres à identifier. Soit N_i le nombre de données (ϕ_k, y_k) générées exclusivement par le sous-modèle i ($i \in \{1, \cdots, s\}$). Pour $\theta = \theta_i$ alors $\Phi(\theta)$ est un vecteur restreint ("sparse vector"), c'est a dire plusieurs éléments du vecteur sont nul. Précisément le vecteur $\Phi(\theta)$ contient N_i éléments nuls et $N - N_i$ éléments non nuls.

Dans un premier temps, supposons que la séquence du bruit $\{e_k\}$ est identiquement nulle. Introduisons les notations suivantes : $\bar{x}_k = [y_k \quad -\phi_k^T]^T$ et $\bar{\theta}_i = [1 \quad \theta_i^T]^T$. À chaque instant k il existe un $i \in \{1,\ \cdots, s\}$ tel-que : $y_k - \theta_i^T\phi_k = \bar{x}_k^T\bar{\theta}_i = 0$. Les données $\{\bar{x}_k\}_{k=1}^{N}$ se trouvent donc dans l'union de s hyperplans linéaires dont les vecteurs normaux sont les vecteurs de paramètres θ_i, $i = 1, \cdots, s$.

Pour déterminer les paramètres du vecteur θ_i qui permet d'obtenir un vecteur d'erreur $\Phi(\theta)$ le moins dense, il est possible de résoudre le problème d'optimisation restreinte en se basant sur la minimisation de la norme ℓ_0 suivante :

$$\min_{\theta} \| \Phi(\theta) \|_0 \tag{1.15}$$

où $\| z \|_0$ désigne la norme ℓ_0 de z. Autrement dit, elle désigne le nombre d'éléments non nuls de z : $\| z \|_0 = | \{i : z_i \neq 0\} |$.

Résoudre le problème d'optimisation (1.15) revient à chercher un hyperplan homogène qui contient autant de données $\bar{x}_k$ que possible. Si tous les sous-modèles sont suffisamment excités dans les données $\{\bar{x}_k\}_{k=1}^{N}$ alors, la solution du problème d'optimisation (1.15) est un vecteur de paramètres de l'un des sous-modèles constituant le système.

$$\underset{\theta}{\operatorname{argmin}} \parallel \Phi(\theta) \parallel_0 = \theta_{i_o} \tag{1.16}$$

où i_o est l'un des indices de sous-modèles qui a généré le plus grand nombre de données.

Remarque 1. *Le problème (1.15) est un problème d'optimisation NP-difficile non convexe. Ce problème est en général difficile à résoudre. L'auteur présente une alternative populaire disponible dans la littérature ([27], [24]) en appliquant des conditions nécessaires afin de relaxer le problème d'optimisation basé sur la minimisation de la norme ℓ_0 à un problème d'optimisation basé sur la minimisation de la norme ℓ_1. Cette relaxation conduit au problème d'optimisation suivant :*

$$\min_{\theta} \parallel \Phi(\theta) \parallel_1 \tag{1.17}$$

Le lecteur intéressé par les conditions nécessaires afin de faire la relaxation du problème d'optimisation peut se référer à [27], [24] et [6]. Contrairement au problème d'optimisation (1.15), le problème d'optimisation (1.17) peut être résolu par des techniques d'optimisation convexe standard.

Lorsque un vecteur de paramètres est identifié, les données associées au sous-modèle identifié sont supprimées et l'objectif est de chercher à identifier un autre vecteur de paramètres sur le reste de données. La procédure est alors itérée jusqu'à l'identification de s vecteurs de paramètres.

Dans un second temps, dans le cas où les données sont corrompues par une quantité modérée de bruit $\{e_k\}$, l'auteur, présente deux voies possibles : soit supposer la connaissance d'une borne supérieure sur le bruit et résoudre le problème d'optimisation (1.17) sous la contrainte $\|e\|_\ell \leq \delta_e$ où δ_e est la borne supérieure sur le bruit et ℓ une certaine norme (par exemple ℓ_2, ou ℓ_∞), soit minimiser de manière combinatoire le problème d'optimisation (1.17) et le terme de bruit comme suit :

$$\min_{\theta,e} \gamma \parallel \Phi(\theta) \parallel_1 + \frac{1}{2}\|e\|_2^2$$

où γ est un terme de régularisation à déterminer. Pour plus de détails théoriques sur cette méthode, il est possible de se référer à ([6],[51],[27], [24] [52], [173]).

L'identification itérative des vecteurs de paramètres $\{\theta_i\}_{i=1}^{s}$ conduit à un temps de calcul important. La plupart des méthodes basées sur des approches d'optimisation souffre de cet inconvénient. De plus, ces méthodes fonctionnent en mode hors-ligne et ne peuvent pas être adaptées, pour une mise en œuvre en temps-réel.

1.2.3 Identification en temps réel des systèmes à commutations décrits par un modèle SARX

À notre connaissance, peu d'algorithmes ont été développés pour l'identification en temps réel des systèmes à commutations décrits par un modèle SARX. En effet, l'approche algébrique a été étendue pour parvenir à une identification en temps réel dans [85] et [188]. Comme nous l'avons précédemment mentionné, cette approche souffre d'une forte sensibilité au niveau de bruit. Dans [8], un algorithme d'identification des systèmes à commutations décrits par un modèle SARX a été proposé. L'algorithme alterne entre l'estimation de l'état discret et la mise à jour du vecteur de paramètres. À chaque instant, l'état discret est déterminé comme étant l'indice du sous-modèle qui a le plus probablement généré la donnée dans le sens d'un certain critère de décision basé sur le calcul de l'erreur de prédiction a priori, et le vecteur de paramètres associé à cet état discret estimé est mis à jour par l'intermédiaire de l'algorithme des moindres carrés récursifs. Les hypothèses que les ordres de différents sous-modèles sont fixes, égaux et connus à priori et la connaissance du nombre de sous-modèles ou une borne supérieur sur le nombre de sous-modèles sont considérées. De plus, la nature du terme de bruit est supposée comme un processus stochastique. L'algorithme a aussi comme point faible l'absence d'une analyse de comportement ou de convergence.

1.3 Identification des systèmes affines par morceaux décrits par un modèle PWARX

Les systèmes affines par morceaux forment une classe attractive et très populaire des systèmes hybrides. Un système affine par morceaux comprend un nombre fini de sous-systèmes indexés par une variable discrète supplémentaire appelé l'état discret. À chaque instant, la sortie du système correspond à la sortie du sous-système actif à cet instant. Le mécanisme de commutation entre les sous-systèmes est gouverné par une partition polyédrique de l'espace de régression.

L'identification des systèmes affines par morceaux est un problème délicat à résoudre qui nécessite l'estimation du nombre de sous-modèles, l'association de chaque donnée au sous-modèle le plus approprié, l'estimation des paramètres de chaque sous-modèle et l'estimation des paramètres des hyperplans qui forment une partition polyédrique de l'espace de régression.

L'identification des systèmes affines par morceaux décrits par un modèle PWARX a attiré l'attention de beaucoup de chercheurs ces dernières années. Les principales approches développées dans la littérature sont :

- l'approche algébrique [187] ;
- l'approche basée sur l'optimisation ([146], [133]) ;
- l'approche basée sur la classification (clustering) ([58], [57], [142], [155], [70], [20], [12], [116], [117], [170]) ;
- l'approche de l'erreur bornée ([13], [150]) ;
- l'approche bayésienne [102].

L'hypothèse que les ordres des différents sous-modèles sont fixes, égaux et connus à priori est considérée dans la plupart des approches sauf l'approche algébrique détaillée déjà précédemment pour le modèle SARX. Nous présentons dans ce qui suit le problème d'identification des systèmes affines par morceaux décrits par un modèle PWARX. Ensuite nous décrivons brièvement le principe de quelques approches citées ci-dessus.

On considère un système SISO discret affine par morceaux décrit par un modèle PWARX sous forme entrée-sortie comme suit :

$$y_k = f(\phi_k) + e_k \tag{1.18}$$

où

$$f(\phi_k) = \begin{cases} \bar{\phi}_k^T \theta_1 & \text{si} \quad \phi_k \in \mathscr{X}_1 \\ \bar{\phi}_k^T \theta_2 & \text{si} \quad \phi_k \in \mathscr{X}_2 \\ \quad \vdots & \\ \bar{\phi}_k^T \theta_s & \text{si} \quad \phi_k \in \mathscr{X}_s \end{cases} \tag{1.19}$$

- $\{\theta_1; \cdots \theta_s\}$ sont les vecteurs de paramètres des sous-modèles à identifier.
- $y_k \in \mathbb{R}$ est la sortie du système.
- $e_k \in \mathbb{R}$ représente le terme de bruit.

- ϕ_k est le vecteur de régression de dimension : $n = n_a + n_b + 1$, supposé appartenir à un certain polyèdre borné $\mathscr{X} \in \mathbb{R}^d$, donné par :

$$\phi_k = [-y_{k-1} \ \cdots \ -y_{k-n_a} \ \ u_k \ \ u_{k-1} \ \cdots \ u_{k-n_b}]^T \tag{1.20}$$

- $u_k \in \mathbb{R}$ est l'entrée du système.
- n_a et n_b sont les ordres du système.
- $\bar{\phi}_k$ est le vecteur de régression étendu donné par $\bar{\phi} = [\phi_k^T \quad 1]^T$.
- Les régions $\{\mathscr{X}_i\}_{i=1}^s$ définissent une partition polyédrique du domaine fermé et borné $\mathscr{X} \in \mathbb{R}^d$, c'est à dire : $\cup_{i=1}^s \ \mathscr{X}_i = \mathscr{X}$ et $\mathscr{X}_i \cap \mathscr{X}_j = \emptyset$ si $i \neq j$.

Problème d'identification : *Étant donné l'ensemble des observations $\{\phi_k, y_k\}$ générées par le modèle (1.18), estimer à la fois : le nombre s de sous-modèles, les vecteurs de paramètres $\{\theta_i\}_{i=1}^s$ et les régions $\{\mathscr{X}_i\}_{i=1}^s$.*

1.3.1 Approche basée sur le clustering

Cette approche vise à séparer les données en classes en se basant sur l'idée que des données dans l'ensemble $\{\phi_k, y_k\}_{k=1}^N$ sont susceptibles d'être générées par le même sous-modèle si elles se trouvent dans la même classe selon un critère de similarité entre les données. Elle est appliquée sur des systèmes sous les hypothèses que les ordres des sous-modèles sont finis, égaux pour tous les sous-modèles et connu a priori ainsi le nombre s de sous-modèles.

La plupart des méthodes basées sur cette approche ([58], [57], [142]) reposent sur l'utilisation de l'algorithme "K-means" pour faire le partitionnement des données dans des classes. Ensuite pour chaque classe, un vecteur de paramètres est estimé en utilisant un algorithme d'identification standard. L'étape finale de la procédure est l'estimation d'une partition polyédrique de l'espace de régression. Cette approche peut être décrite comme suit :

– Pour chaque $k = 1, \cdots, N$ un ensemble de données local C_k est construit en collectant (ϕ_k, y_k) et les $c-1$ points de données (ϕ_j, y_j) les plus proches de (ϕ_k, y_k) selon un critère de similarité (par exemple : la distance euclidienne). Le nombre entier positif c est un paramètre de synthèse. Des commentaires sur un choix adéquat de ce paramètre de synthèse se trouvent dans [58]. À des fins d'analyse, certains ensembles locaux construits ne contiennent que des points de données générés par le même sous-modèle, ils sont appelés des ensembles locaux purs. Les autres ensembles

locaux qui contiennent des données issues de différents sous-modèles sont appelés des ensembles locaux mixtes. Le nombre des ensembles locaux mixtes dépend à la fois de la distribution des points dans l'espace et du choix du paramètre de synthèse c.

- Des vecteurs de paramètres locaux $\hat{\theta}_k$ sont par la suite calculés pour chaque ensemble local de données C_k via l'algorithme des moindres carrés ou un autre algorithme d'identification standard.
- Chaque point de données (ϕ_k, y_k) est par la suite associé à un vecteur de caractéristique $\xi_k = \begin{bmatrix} \hat{\theta}_k^T & m_k^T \end{bmatrix}^T$ où $m_k = \frac{1}{c}\sum_{(\phi_k,y_k)\in C_k} \phi_k$ est le centre de l'ensemble C_k (la moyenne des régresseurs contenus dans C_k).
- Les vecteurs des caractéristiques sont ensuite partitionnées dans s classes $\{\mathrm{F_i}\}_{i=1}^{s}$ en appliquant l'algorithme K-means ([83], [50]). Il est possible d'utiliser d'autres algorithmes de classification pour partitionner les vecteurs des caractéristiques.
- Étant donné que l'application qui associe le point (ϕ_k, y_k) à un vecteur de caractéristiques ξ_k est bijective, les points de données sont classés en s classes $\{D_i\}_{i=1}^{s}$ selon la règle :

$$(\phi_k, y_k) \in D_i \quad \text{si et seulement si} \quad \xi_k \in \mathrm{F_i} \tag{1.21}$$

- Un vecteur des paramètres est estimé pour chaque classe D_i en utilisant l'algorithme des moindres carrés à nouveau.
- Finalement, il ne reste qu'à estimer les paramètres décrivant la partition polyédrique de l'espace de régression $\{\mathscr{X}_i\}_{i=1}^{s}$. Cette tache peut être accomplie en utilisant un classificateur linéaire comme le Séparateur à Vaste Marge (Support Vector Machine-"SVM").

Un autre travail qui repose sur la même idée, mais basé sur l'algorithme K-plane [22], est présenté dans [178]. Il s'agit d'utiliser l'algorithme K-plan-clustering afin de regrouper les données générées par un système décrit par le modèle (1.18) dans K-plans. Les K-plans forment une solution locale pour le problème d'optimisation non convexe en minimisant la somme des carrés des distances entre chaque point et le plan le plus proche. Dans [117], les auteurs proposent d'utiliser l'algorithme DBSCAN (the Density-Based Spatial clustering of Applications with Noise) ([167], [34]) au lieu de l'algorithme K-means pour faire le partitionnement des données.

Il est à noter que cette approche ne garantit pas la convergence vers l'optimum global. Ses performances sont liées à la procédure d'initialisation de l'algorithme de classification. La plupart des méthodes basées sur cette approche supposent que le terme bruit est un processus stochastique. De plus, le temps de calcul est important à cause de l'étape de partitionnement de données. Elles ne peuvent donc pas être appliquées pour une identification en temps réel. Plus de détails sont donnés dans ([100], [58] et [151]).

1.3.2 Approche de l'erreur bornée

Le principe de cette approche consiste à imposer à l'erreur d'identification d'avoir une amplitude bornée par une borne supérieure. Les ordres des sous-modèles sont considérés finis, égaux pour tous les sous-modèles et connus a priori. Inspirés par l'idée des algorithmes d'identification ensemblistes, *Bemporad et al.* [13] considèrent le problème d'identification des systèmes affines par morceaux décrits par un modèle PWARX comme un problème de partitionnement en un nombre minimum de sous-systèmes réalisables (MIN PFS) [4]. L'objectif est d'imposer à l'erreur d'identification d'avoir une amplitude bornée par une borne supérieure positive $\delta > 0$, soit :

$$| y_k - f(\phi_k) | \leqslant \delta, \ \forall \ k = 1, \cdots, N \tag{1.22}$$

Sous la contrainte (1.22), on peut reformuler le problème d'identification des systèmes affines par morceaux décrits par un modèle PWARX de la manière suivante :

Problème d'identification 2 : *Étant donné l'ensemble des observations* $\{\phi_k, y_k\}_{k=1}^{N}$ *et un nombre* $\delta > 0$*, estimer le plus petit entier* s*, l'ensemble des vecteurs de paramètres* $\{\theta_i\}_{i=1}^{s}$*, et une partition polyédrique* $\{\mathscr{X}_i\}_{i=1}^{s}$ *de l'espace de régression* $\mathscr{X} = \cup_{i=1}^{s} \mathscr{X}_i \in \mathbb{R}^d$ *tel que le modèle (1.18) satisfait la condition (1.22).*

Il est à noter que le paramètre de synthèse δ permet de fixer la précision du modèle recherché ainsi que le nombre de sous-modèles réalisables. Les performances de l'algorithme sont liées à un choix judicieux de δ. En effet, plus δ est grand, plus le nombre de sous-modèles est petit et le modèle estimé est moins précis. À l'inverse un δ petit conduit à un nombre important de sous-modèles. L'algorithme se déroule en 3 étapes, nous présentons ci-dessous brièvement ces étapes :

1. *Étape 1 : Classification initiale et estimation des paramètres d'une manière simultanée via (MIN PFS) :* Cette étape est basée sur une version modifiée de l'algorithme "glouton" de *Amaldi et Mattaveli.* [4]. Le but ici est de faire une classification initiale de données, estimer le nombre de sous-modèles ainsi que les paramètres correspondant à chaque sous-modèle d'une manière simultanée. Le principe est de subdiviser le problème de la partition minimum (MIN PFS) en une série de sous problèmes dits (MAX FS). Chaque sous-problème (MAX FS) permet de déterminer un vecteur θ satisfaisant un nombre maximum d'inégalités du type $\mid y_k - \theta^T \phi_k \mid \leqslant \delta$. Nous présentons les idées clés de cette première étape comme suit :
   - Étant donné l'ensemble des données $\{\phi_k, y_k\}_{k=1}^N$, on commence par déterminer un premier vecteur de paramètre θ_1 en maximisant le cardinal de l'ensemble E_1 défini par : $E_1 = \{(\phi_k, y_k) :\mid y_k - \theta_1^T \phi_k \mid \leqslant \delta$.
   - En excluant l'ensemble E_1 de l'ensemble de départ, on répète la même procédure sur le reste de données pour obtenir un deuxième vecteur de paramètres θ_2 en maximisant le cardinal de l'ensemble $E_2 = \{(\phi_k, y_k) :\mid y_k - \theta_2^T \phi_k \mid \leqslant \delta$.
   - On répète la procédure jusqu'à ce qu'on obtienne un nombre d'ensembles s qui vérifie : $E = \{E_1 \cup E_2, \cdots, \cup E_s\}$.

2. *Étape 2 "Raffinement et correction" :* Cette étape est considérée comme une procédure corrective, elle permet de corriger les erreurs de la classification et d'améliorer les valeurs des vecteurs des paramètres obtenus à la première étape. En effet, la solution trouvée à la première étape souffre de deux inconvénients. Le premier est lié à la sous-optimalité de la méthode utilisée (l'obtention du nombre minimum de sous-modèles n'est pas garantie). L'autre inconvénient est lié au problème de points de données indécidables, ce sont les points pour lesquels la contrainte $\mid y_k - f(\phi_k) \mid \leqslant \delta$ peut être vérifiée pour plusieurs sous-modèles. Cette étape permet la réaffectation des données et la remise à jour des vecteurs de paramètres définissant les sous-modèles d'une manière itérative, son principe est détaillé dans [13].

3. *Étape 3 "Estimer une partition polyédrique de l'espace de régression" :* Cette étape est dédiée à la recherche d'une partition polyédrique de l'espace de régression. Chaque sous-modèle est alors valide dans une région bien définie. L'estimation de ces régions $\{\mathcal{X}_i\}$ en utilisant les SVM constituent l'étape finale de la procédure (voir sous-section 1.3.4).

Cette approche présente l'avantage d'estimer le nombre de sous-modèles, néanmoins l'obtention du vrai nombre de sous-modèles n'est pas garantie. D'autre part, cette approche souffre d'un temps de calcul important. Ceci conduit à une limitation forte sur le volume des données qu'elle peut manipuler.

1.3.3 Approche Bayésienne

L'approche bayésienne a été développée à partir des perspectives statistiques dans [102] et[99]. Elle est appliquée sous les hypothèses que les ordres du système sont finis et connus a priori ainsi que le nombre s de sous-modèles. De plus, la fonction de densité de probabilité (fdp) du bruit $p_e(.)$ est supposée connue à priori. L'idée principale est de traiter les vecteurs de paramètres $\{\theta_i\}_{i=1}^{s}$ comme des variables aléatoires décrites par leurs fdp $P_{\theta_i}(.)$. L'algorithme fonctionne en mode hors ligne et suit un procédé itératif sous-optimal pour mettre à jour la fonction de densité de probabilité de chaque sous-modèle. À chaque instant k, la fdp d'un seul vecteur de paramètres est mise à jour. On note $P_{\theta_i}(\theta;k)$ la fdp du vecteur de paramètres θ_i à l'instant k. Soit la probabilité conditionnelle $p((y_k,\phi_k) \mid \hat{\lambda}_k = i)$ qui correspond à la probabilité que le sous-modèle i ait généré la donnée (y_k,φ_k) à l'instant k :

$$p((y_k,\phi_k) \mid \lambda_k = i) = \int_{\Theta_i} p((y_k,\phi_k) \mid \theta) \;\; P_{\theta_i}(\theta;k-1)d\theta \tag{1.23}$$

avec

$$p((y_k,\varphi_k) \mid \theta) = p_e(y_k - \theta^T \bar{\varphi}) \tag{1.24}$$

où Θ_i est l'espace des valeurs possible de θ_i.

L'état discret à l'instant k est estimé de la manière suivante :

$$\hat{\lambda}_k = \underset{i=1,\cdots,s}{\operatorname{argmax}} \;\; p((y_k,\phi_k) \mid \lambda_k = i) \tag{1.25}$$

L'estimation de l'état discret à l'instant k permet l'association de la donnée (y_k,ϕ_k) au sous-modèle $\hat{\lambda}_k$. Ensuite, la fdp qui correspond au sous-modèle $\hat{\lambda}_k$ est mise à jour comme suit :

$$P_{\theta_{\hat{\lambda}_k}}(\theta;k) = \frac{p((y_k,\phi_k) \mid \theta) \;\; P_{\theta_{\hat{\lambda}_k}}(\theta;k-1)}{\int_{\Theta_i} p((y_k,\phi_k) \mid \theta) \;\; P_{\theta_{\hat{\lambda}_k}}(\theta;k-1)d\theta} \tag{1.26}$$

Finalement, afin d'estimer les régions $\{\mathscr{X}_i\}$ qui forment une partition polyédrique de l'espace de régression, une version modifiée de l'algorithme de programmation linéaire [15] a été proposée.

Cette approche est caractérisée par la complexité qui accompagne l'idée de substituer chaque variable par sa fonction de densité de probabilité qui est une fonction continue et non linéaire. Les auteurs proposent l'approximation de celle-ci par une fonction à support fini en utilisant les théories du filtrage particulaire [5]. Cette approximation conduit à des estimées en un temps de calcul important. De plus, cette approche souffre de sa forte sensibilité à l'initialisation comme il est montré dans [100]. Notons aussi que cette approche nécessite la connaissance a priori de la fonction de densité de probabilité du bruit. Dans plusieurs procédés réels, la distribution de probabilité de bruit est inconnue et le bruit ne peut être modélisé comme un processus stochastique.

1.3.4 Estimation des régions

Dans une procédure d'identification des systèmes affines par morceaux décrits par un modèle PWARX, une fois que les étapes de classification des données et d'identification des vecteurs de paramètres sont réalisées, il faut identifier les régions de régression $\{\mathscr{X}_i\}$ qui constitue l'étape finale de la procédure. Ceci est équivalent à trouver les frontières qui délimitent chaque région dans la partition complète $\{\mathscr{X}\}$ et donc séparer les "s" régions par des séparateurs linéaires (hyperplans).

Ce problème a été largement discuté dans la littérature et la plupart des approches utilisent les SVMs ([13], [20], [146], [142], [117], [12]). L'utilisation des SVM ([184], [183]) est justifiée par des fondements théoriques solides et la capacité de traiter un grand volume de données dans un temps de calcul raisonnable (cf. chapitre 5). Les SVM sont en général utilisés pour la classification binaire. L'utilisation des SVM a été étendue pour la classification multi-classes (classification de données en s classes dans notre cas). Trois voies sont possibles ([151], [19]) pour estimer les régions $\mathscr{X}_i$:

- Construire un classificateur binaire en utilisant les SVM pour chaque paire $(\mathscr{X}_i, \mathscr{X}_j)$ "un contre un" où il est nécessaire de construire $s(s-1)/2$ classificateurs binaires, chaque classificateur sépare deux régions uniquement.
- Construire un classificateur binaire en utilisant les SVM en se basant sur le principe "un contre tous". Dans cette approche, il est nécessaire de construire s classificateurs

binaires, chaque classificateur sépare une région contre le reste.

- Construire un classificateur linéaire par morceau (PieceWise Linear Classifier) en considérant l'ensemble de toutes les données. Ceci est réalisé en utilisant "Multicatégorie SVM" (M-SVM) ([120], [13], [151]) afin de résoudre le problème de séparation en une seule étape sans le décomposer en un ensemble de problèmes de classification binaire.

Plus de détails sur les trois voies possibles se trouvent dans [79].

1.4 Identification des systèmes décrits par les modèles SOE et PWOE

Les différentes méthodes qui existent dans la littérature et que nous avons citées précédemment pour l'identification des systèmes hybrides utilisent un modèle de type ARX. En effet, les systèmes affines par morceaux ont été modélisés par un modèle de régression de type ARX affine par morceaux "PWARX" et les systèmes linéaires à commutations ont été modélisés par un modèle de régression de type ARX à commutations "SARX". Dans le cas général, les données sont bruitées et nous n'avons aucune connaissance a priori sur le bruit. Un paramétrage du problème d'identification du type erreur de sortie semble alors plus approprié [126]. Ce modèle présente l'avantage que le modèle de bruit n'est pas incorporé dans le modèle de processus puisque le bruit est ajouté directement à la sortie.

Considérons un système linéaire à commutations décrit par un modèle du type erreur de sortie à commutations "SOE", comme suit :

$$y_k = \frac{B_{\lambda_k}(q)}{A_{\lambda_k}(q)} u_k + e_k \tag{1.27}$$

- $u_k \in \mathbb{R}$ et $y_k \in \mathbb{R}$ sont respectivement l'entrée et la sortie du système.
- $e_k \in \mathbb{R}$ représente le terme de bruit.
- $\lambda_k \in \{1, \cdots, s\}$ est l'état discret du système avec s le nombre de sous-modèles.
- $A_{\lambda_k}(q)$ et $B_{\lambda_k}(q)$ sont les polynômes associés au sous-modèle λ_k dont les paramètres sont inconnus :

$$\begin{cases} A_{\lambda_k}(q) = 1 + a_1(\lambda_k) q^{-1} + \cdots + a_{n_a}(\lambda_k) q^{-n_a} \\ B_{\lambda_k}(q) = b_0(\lambda_k) + b_1(\lambda_k) q^{-1} + \cdots + b_{n_b}(\lambda_k) q^{-n_b} \end{cases} \tag{1.28}$$

- n_a et n_b sont les ordres du système ((les ordres de différents sous-modèles sont supposés fixes, égaux et connus à priori : $n_a(1) = \cdots = n_a(s) = n_a$ et $n_b(1) = \cdots = n_b(s) = n_b$).
- L'opérateur q^{-1} désigne l'opérateur de retard.

Il est à noté que le modèle PWOE est un cas particulier du modèle SOE où le mécanisme de commutation est gouverné par une partition polyédrique de l'espace de régression.

Problème d'identification : *Étant donné l'ensemble des observations* $\{u_k, y_k\}$ *générées par le modèle SOE de la forme (1.27), estimer : le nombre de sous-modèles* s*, les paramètres des polynômes* $A_{\lambda_k}(q)$ *et* $B_{\lambda_k}(q)$ *de chaque sous-modèle et l'état discret* $\{\lambda_k\}_{k=1}^{N}$.

Peu de méthodes ont été étendues pour décrire les systèmes hybrides par des modèles du type erreur de sortie. L'approche bayésienne et l'approche basée sur la classification (clustering) ont été respectivement utilisées dans [101] et [28] où les auteurs décrivent le système affine par morceaux par un modèle du type erreur de sortie affine par morceaux "PWAOE". Ces méthodes s'appliquent en mode hors-ligne et requièrent un temps de calcul important. De plus, le nombre de sous-modèles s est supposé connu et la nature du terme de bruit est supposée comme un processus stochastique.

Dans [198], les autours ont proposé une méthode pour l'identification en-ligne pour le modèle SOE. Cette méthode est une version modifiée de l'algorithme présenté dans [8] où les auteurs ont adapté l'algorithme pour le modèle du type erreur de sortie en introduisant aussi des analyses sur le critère d'estimation de l'état discret et ont proposé une stratégie de ré-initialisation en se basant sur ces analyses. À noter, aucune analyse de convergence de l'algorithme n'est fournie. Le nombre de sous-modèles s est supposé connu ici aussi et la nature du terme de bruit est supposée comme un processus stochastique.

1.5 Conclusion

Dans ce chapitre, nous avons présenté d'une manière concise les différentes méthodes qui existent dans la littérature pour l'identification des systèmes linéaires à commutations et les systèmes affines par morceaux. En effet, au cours de ces dernières années, une attention considérable a été consacrée au problème d'identification de ces deux classes des systèmes. Il est facile de constater que la plupart de ces méthodes décrivent les systèmes par des modèles ARX (un modèle SARX pour les systèmes à commutations et un modèle PWARX pour les systèmes affines par morceaux). Une hypothèse assez courante est largement utilisée dans les différentes méthodes consiste à supposer que les ordres de différents sous-modèles sont fixes, égaux et connus à priori. La quasi-totalité des méthodes souffre de l'inconvénient d'un temps de calcul important, par conséquent, la plupart des méthodes fonctionnent en mode batch. De plus, la majorité des méthodes résout ce problème sous l'hypothèse usuelle de bruit stochastique. La prise en compte de cette hypothèse nécessite une connaissance à priori de la caractéristique statistique du bruit affectant la sortie du système. Dans plusieurs procédés réels, la distribution de probabilité de bruit est inconnue. De plus, à cause d'une erreur de modélisation de bruit, la présence d'une perturbation non déterministe ou de bruit de mesure, le bruit ne peut être modélisé comme un processus stochastique. Par conséquent, une hypothèse sur l'existence uniquement d'une borne supérieure sur le bruit semble être plus appropriée. La suite de cette partie est consacrée au développement de méthodes appropriées pour l'identification des systèmes en temps réel dans le cas où la nature du bruit est inconnue, mais une borne supérieure sur son amplitude est disponible. Nous présentons dans le prochain chapitre un algorithme d'identification en temps réel des systèmes à commutations décrits par un modèle SARX en présence bruit borné.

Chapitre 2

Algorithme d'identification des systèmes à commutations décrits par un modèle SARX

Dans ce chapitre, nous considérons le problème d'identification des systèmes linéaires à commutations soumis à des perturbations bornées. La résolution de ce problème d'identification nécessite l'association de chaque donnée au sous-système le plus approprié et l'estimation de tous les vecteurs de paramètres des sous-systèmes. L'algorithme proposé dans ce chapitre est basé sur une méthode d'identification de type OBE "Outer Bounding Ellipsoid type algorithm". Cette méthode est bien appropriée pour l'identification des systèmes soumis à des perturbations bornées. L'algorithme comporte deux étapes à chaque instant : la première étape permet d'estimer l'état discret qui indique le sous-système actif et la deuxième étape permet de mettre à jour le vecteur de paramètres associé à l'état discret estimé. Une analyse de stabilité et de convergence est effectuée. Une extension pour l'identification des systèmes linéaires MIMO à commutations est également proposée. Des résultats de simulation sont donnés afin d'illustrer les performances de l'algorithme proposé.

Sommaire

2.1 Motivations et formulation du problème d'identification

Ce chapitre est consacré aux principaux développements de cette partie de thèse, notamment un algorithme d'identification des systèmes linéaires à commutations soumis à des perturbations bornées est proposé. L'algorithme peut être utilisé efficacement pour l'identification en temps réel ou en mode batch. En outre, l'algorithme est adapté à l'identification des systèmes linéaires à commutations décrits par un modèle SARX, aussi bien dans le cas SISO que le cas MIMO. Nous effectuons une analyse du comportement de l'algorithme, ce qui est relativement rare dans le contexte de l'identification des systèmes à commutations. Comme nous l'avons mentionné auparavant, la plupart des méthodes disponibles sont développées sous l'hypothèse habituelle d'un bruit stochastique. Une approche alternative est de considérer la connaissance uniquement d'une borne supérieure sur l'amplitude du bruit. Avec cette approche, le bruit est supposé être inconnu, mais borné. Le bruit pourrait être uniforme, tronqué, normal ou toute autre distribution, seule une borne supérieure sur son amplitude est supposée connue.

Nous reformulons dans la présente section, le problème d'identification des systèmes à commutations décrits par un modèle SARX présenté dans le chapitre 1 en un problème d'identification des systèmes à commutations décrits par un modèle SARX en présence d'un bruit borné. Les ordres de différents sous-modèles sont supposés fixes, égaux et connus a priori ($n_a(1) = \cdots = n_a(s) = n_a$ et $n_b(1) = \cdots = n_b(s) = n_b$). Considérons un système linéaire discret SISO à commutations décrit par un modèle SARX comme suit :

$$y_k = -\sum_{j=1}^{n_a} a_j(\lambda_k) y_{k-j} + \sum_{i=0}^{n_b} b_i(\lambda_k) u_{k-i} + e_k \tag{2.1}$$

Le modèle (2.1) peut se réécrire comme suit :

$$y_k = \phi_k^T \theta_{\lambda_k} + e_k \tag{2.2}$$

où $\phi_k \in \mathbb{R}^n$ est le vecteur de régression défini par :

$$\phi_k = [-y_{k-1} \ \cdots \ -y_{k-n_a} \ \ u_k \ \ u_{k-1} \ \cdots \ u_{k-n_b}]^T \tag{2.3}$$

- n_a et n_b sont les ordres du système et $n = n_a + n_b + 1$.
- $u_k \in \mathbb{R}$ et $y_k \in \mathbb{R}$ sont respectivement l'entrée et la sortie du système.
- $e_k \in \mathbb{R}$ est le terme inconnu de bruit. Il peut représenter le bruit de mesure ou des erreurs dans la modélisation, une borne supérieure δ_e sur son amplitude est supposée connue :

$$| e_k | \leq \delta_e \ , \quad \forall \ k \tag{2.4}$$

- θ_{λ_k} est le vecteur de paramètres associé au sous-modèle désigné par λ_k donné par :

$$\theta_{\lambda_k} = [a_1(\lambda_k) \quad \cdots \quad a_{n_a}(\lambda_k) \quad b_0(\lambda_k) \quad b_1(\lambda_k) \quad \cdots \quad b_{n_b}(\lambda_k)]^T \tag{2.5}$$

- $\lambda_k \in \{1, \cdots, s\}$ est l'état discret qui indique le sous-modèle actif à l'instant k, s est le nombre total de sous-modèles.

Notons que le modèle SARX décrit par (2.2), (2.3) et (2.4) est tel que, à chaque instant k, il existe au moins un vecteur de paramètres θ_i qui satisfait $| y_k - \phi_k^T \theta_i | \leq \delta_e$. Ce vecteur de paramètres θ_i correspond au sous-modèle actif λ_k à l'instant k. Cette caractéristique permet de définir le problème d'identification avec bruit borné d'un modèle SARX de la manière suivante :

Problème d'identification 1 : *Disposant d'un ensemble d'observations* $\{\phi_k, y_k\}_{k=1}^N$ *générés par un système de la forme (2.2), (2.3) et (2.4), l'objectif est d'estimer l'état discret* $\{\lambda_k\}_{k=1}^N$ *et les vecteurs de paramètres* $\{\theta_i\}_{i=1}^s$. *Ces estimations* $\hat{\lambda}_k$ *et* $\widehat{\theta}_i$ *doivent satisfaire :*

$$| y_k - \phi_k^T \widehat{\theta}_{\hat{\lambda}_k} | \leq \delta_e \ , \quad \forall \ k \tag{2.6}$$

Nous passons maintenant à la présentation d'une nouvelle formulation du problème d'identification. Si à l'instant k la sortie a été générée par le sous-modèle "i", c'est à dire $\lambda_k = i$, alors :

$$y_k = \phi_k^T \theta_i + e_k \tag{2.7}$$

La sortie du système y_k peut également être exprimée à partir des vecteurs de paramètres des autres sous-modèles, par exemple, en fonction du vecteur de paramètres θ_1 on a :

$$\begin{aligned} y_k &= \phi_k^T \theta_i + e_k + \phi_k^T \theta_1 - \phi_k^T \theta_1 \\ &\Downarrow \\ y_k &= \phi_k^T \theta_1 + e_k + \phi_k^T (\theta_i - \theta_1) \end{aligned} \tag{2.8}$$

Il en résulte qu'à l'instant k, nous pouvons écrire :

$$\begin{cases} y_k = \phi_k^T \theta_1 + e_k + \phi_k^T (\theta_i - \theta_1) \\ \vdots \\ y_k = \phi_k^T \theta_i + e_k \\ \vdots \\ y_k = \phi_k^T \theta_s + e_k + \phi_k^T (\theta_i - \theta_s) \end{cases} \tag{2.9}$$

Ceci, peut se réécrire sous la forme matricielle suivante :

$$\begin{bmatrix} y_k \\ \vdots \\ y_k \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} \phi_k^T \theta_1 \\ \vdots \\ \phi_k^T \theta_i \\ \vdots \\ \phi_k^T \theta_s \end{bmatrix} + \begin{bmatrix} \phi_k^T (\theta_i - \theta_1) \\ \vdots \\ e_k \\ \vdots \\ \phi_k^T (\theta_i - \theta_s) \end{bmatrix} \tag{2.10}$$

Définition 1. - *Nous définissons le vecteur global de paramètres* $\Theta^* \in \mathbb{R}^{ns}$ *par :*

$$\Theta^* = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_i \\ \vdots \\ \theta_s \end{bmatrix} \tag{2.11}$$

- *Nous définissons le vecteur de sortie étendu* $Y_k \in \mathbb{R}^s$ *comme suit :*

$$Y_k = \begin{bmatrix} y_k \\ \vdots \\ y_k \\ \vdots \\ y_k \end{bmatrix} \tag{2.12}$$

- *Nous définissons le vecteur de bruit étendu* $V_k \in \mathbb{R}^s$ *comme suit :*

$$V_k = \begin{bmatrix} e_k + \phi_k^T (\theta_i - \theta_1) \\ \vdots \\ e_k \\ \vdots \\ e_k + \phi_k^T (\theta_i - \theta_s) \end{bmatrix} \tag{2.13}$$

où ns *est le nombre total de paramètres à identifier.*

Il est intéressant de noter qu'il existe $s!$ variantes possibles de l'état discret $\{\lambda_k\}_{k=1}^N$ et du vecteur global de paramètres Θ^*. Les $s!$ variantes résultent du fait que l'état discret et le vecteur global de paramètres sont inconnus et du fait que l'ordre de rangement des vecteurs de paramètres $\{\theta_i\}_{i=1}^s$ dans le vecteur global de paramètres Θ^* est indifférent, par conséquent les différentes combinaisons entre les $\{\theta_i\}_{i=1}^s$ sont possibles. Dans ce qui suit, Θ^* fait référence à une combinaison parmi les $s!$ combinaisons possibles.

En utilisant ces définitions, la forme matricielle donnée par l'équation (2.10) peut se réécrire sous la forme compacte suivante :

$$Y_k = \varphi_k^T \Theta^* + V_k \tag{2.14}$$

avec $\varphi_k = I_s \otimes \phi_k$, où $\otimes$ désigne le produit de Kronecker. Il est facile de remarquer que, à chaque instant k, au moins une composante du vecteur V_k (la composante qui correspond au sous-modèle actif à l'instant k) a son amplitude inférieure ou égale à la borne supérieure δ_e.

En nous fondant sur ces définitions et formulations, nous redéfinissons le problème d'identification des systèmes à commutations décrits par un modèle SARX en présence de bruit borné de la manière suivante :

Problème d'identification 2 : *Étant donné un ensemble d'observations* $\{\varphi_k, Y_k\}_{k=1}^N$, *l'objectif est d'estimer l'état discret* $\{\lambda_k\}_{k=1}^N$ *et le vecteur global de paramètres* Θ^*. *Ces estimations* $\hat{\lambda}_k$ *et* $\widehat{\Theta}$ *doivent satisfaire :*

$$| \varepsilon_k(\hat{\lambda}_k) | \leq \delta_e \ , \quad \forall \ k \tag{2.15}$$

où $\varepsilon_k(\hat{\lambda}_k)$ *désigne la* $\hat{\lambda}_k^{ième}$ *composante de* $\varepsilon_k = Y_k - \varphi_k^T \widehat{\Theta}$.

Afin d'aborder le problème d'identification sous l'hypothèse du bruit borné, les méthodes d'identification ensemblistes "Set-Membership" sont considérées. Ces méthodes ne nécessitent aucune hypothèse statistique sur la nature du bruit. L'objectif est de caractériser l'ensemble de tous les vecteurs de paramètres compatibles avec les données, la structure du modèle et la borne supérieure sur le bruit. Cet ensemble est appelé l'ensemble de solutions réalisables des paramètres ("feasible solution set (FSS)"). Lorsque le modèle est linéaire, le FSS est un polytope convexe pour lequel une description exacte peut être recherchée. Cependant, cet ensemble peut être extrêmement complexe à caractériser et

les algorithmes correspondants risquent d'avoir une complexité algorithmique importante. Pour faire face à ce problème, des ensembles de formes plus simples qui contiennent le FSS sont considérés. Plusieurs algorithmes d'estimation ensembliste correspondant aux différents types d'ensembles existent dans la littérature. Parmi ces algorithmes, on peut citer : les algorithmes utilisant les zonotopes ([23], [88]), les algorithmes utilisant les parallélotopes ([37], [186], [36]), les algorithmes utilisant les orthotropes ([136], [135]) et les algorithmes utilisant les ellipsoïdes ([59], [46], [29], [179], [21], [29], [97], [158], [159], [160]). Plus de détails sur les algorithmes ensemblistes sont donnés dans : [135], [67], [69] et [31].

Nous nous intéressons d'une manière particulière aux algorithmes utilisant les ellipsoïdes. Un ellipsoïde est caractérisé par son centre et une matrice qui définit sa taille et son orientation. Les méthodes ellipsoïdales calculent une ellipsoïde contenant le FSS après chaque réception d'un nouvel échantillon en minimisant son volume, sa traçabilité [59] ou un autre critère [179]. Ce type d'algorithme présente des avantages certains. En effet, la mise en œuvre de ce type d'algorithme est simple et sa complexité algorithmique est plus faible que celle des autres types. Ceci favorise son utilisation pour l'identification en temps réel.

Afin d'illustrer le principe des méthodes ensemblistes, considérons l'identification paramétrique dans un contexte de bruit borné pour un système linéaire décrit par un modèle ARX . Soit le système suivant linéaire par rapport aux paramètres :

$$y_k = \phi_k^T \theta + e_k \tag{2.16}$$

où y_k est la sortie du système, θ est le vecteur de paramètres à identifier, ϕ_k est le vecteur de régression contenant les entrées et les sorties du système aux instants précédents, e_k représente le terme de bruit affectant la sortie. Une borne supérieure sur l'amplitude du bruit est supposée connue :

$$| e_k | \leq \delta_e \tag{2.17}$$

Génériquement le problème abordé par les méthodes ensemblistes consiste à estimer un vecteur de paramètres à l'intérieur de $\cap_k \mathscr{S}_k$ où $\mathscr{S}_k$ est l'ensemble d'observations défini par :

$$\mathscr{S}_k = \{\theta \in \mathbb{R}^n, | y_k - \phi_k^T \theta | \leq \delta_e\} \tag{2.18}$$

$\mathscr{S}_k$ est la bande de tous les vecteurs de paramètres possibles θ qui sont compatibles avec la borne supérieure sur le bruit δ_e à l'instant k. Cette bande correspond à l'espace restreint

entre les hyperplans parallèles : $y_k - \phi_k^T \theta \leq \delta_e$ et $-\delta_e \leq y_k - \phi_k^T \theta$. Disposant des mesures $\{y_k, \phi_k\}_{k=1}^N$, l'ensemble $\cap_k \mathscr{S}_k$ contient le vrai vecteur de paramètres.

Les algorithmes utilisant des ellipsoïdes (Outer Bounding Ellipsoid) "OBE" cherchent à chaque instant k, un ellipsoïde $\mathscr{E}_k$ qui englobe l'ensemble $\cap_k \mathscr{S}_k$. Dans la section suivante, un algorithme d'identification en temps réel du modèle SARX en se basant sur les algorithmes de type "OBE" est présenté afin de résoudre le problème d'identification des paramètres du modèle SARX.

2.2 L'algorithme R-SARX-OBE

2.2.1 Description de l'algorithme R-SARX-OBE

Afin de faciliter la compréhension de l'algorithme, nous résumons d'abord quelques définitions utiles. Ensuite, l'algorithme d'identification sera décrit.

Définition 2. *Nous désignons par $\hat{\lambda}_k$ l'estimation de l'état discret λ_k à l'instant k. De plus, $\widehat{\Theta}_k$ désigne l'estimation du vecteur global de paramètres Θ^* à l'instant k. $\widehat{\Theta}_k$ est structuré de la manière suivante :*

$$\widehat{\Theta}_k = \begin{bmatrix} \widehat{\theta}_{1/k} \\ \vdots \\ \widehat{\theta}_{s/k} \end{bmatrix} \tag{2.19}$$

où, pour chaque $i \in \{1, \cdots, s\}$, $\widehat{\theta}_{i/k}$ est une estimation de θ_i à l'instant k.

Définition 3. *A chaque instant k, nous définissons le prédicteur a priori et le prédicteur a posteriori respectivement comme suit :*

$$\begin{cases} \hat{Y}_{k/k-1} = \varphi_k^T \widehat{\Theta}_{k-1} = \begin{bmatrix} \phi_k^T \widehat{\theta}_{1/k-1} \\ \vdots \\ \phi_k^T \widehat{\theta}_{s/k-1} \end{bmatrix} \\ \hat{Y}_{k/k} = \varphi_k^T \widehat{\Theta}_k = \begin{bmatrix} \phi_k^T \widehat{\theta}_{1/k} \\ \vdots \\ \phi_k^T \widehat{\theta}_{s/k} \end{bmatrix} \end{cases} \tag{2.20}$$

Définition 4. *L'erreur de prédiction a priori* $\varepsilon_{k/k-1}$ *et l'erreur de prédiction a posteriori* $\varepsilon_{k/k}$ *sont définies de la manière suivante :*

$$\begin{cases} \varepsilon_{k/k-1} = Y_k - \varphi_k^T \widehat{\Theta}_{k-1} = \begin{bmatrix} y_k - \phi_k^T \widehat{\theta}_{1/k-1} \\ \vdots \\ y_k - \phi_k^T \widehat{\theta}_{s/k-1} \end{bmatrix} \\ \varepsilon_{k/k} = Y_k - \varphi_k^T \widehat{\Theta}_k = \begin{bmatrix} y_k - \phi_k^T \widehat{\theta}_{1/k} \\ \vdots \\ y_k - \phi_k^T \widehat{\theta}_{s/k} \end{bmatrix} \end{cases} \tag{2.21}$$

Nous notons que l'estimation du vecteur global de paramètres $\widehat{\Theta}_k$ doit maintenir l'amplitude d'au moins une composante du vecteur $\varepsilon_{k/k}$ inférieure ou égale à la borne supérieure sur le bruit e_k, soit inférieure ou égale à δ_e.

L'algorithme proposé ici est nommé R-SARX-OBE (Recursive-Switched AutoRegressive Exogenous-Outer Bounding Ellipsoid). A chaque instant k, l'algorithme consiste en deux étapes successives. Ces deux étapes sont décrites dans le tableau 2.1 et détaillées ci-dessous.

Étape 1 : Estimation de l'état discret λ_k

A chaque instant k, la première étape de l'algorithme R-SARX-OBE consiste à estimer l'état discret λ_k. L'état discret estimé $\hat{\lambda}_k$ correspond à l'indice de la valeur la plus petite de $\mid \varepsilon_{k/k-1}(i) \mid$ avec $i \in \{1, \cdots, s\}$.

D'un point de vue géométrique, chaque sous-modèle i paramétré par $\hat{\theta}_{i/k-1}$ génère un hyperplan $\{x \in \mathbb{R}^{(n+1)\times 1}, x^T b_i = 0\}$ avec $b_i = \begin{bmatrix} \hat{\theta}_{i/k-1} \\ 1 \end{bmatrix}$. L'état discret estimé est alors l'indice du sous-modèle qui génère le vecteur $\begin{bmatrix} -\phi_k \\ \phi_k^T \hat{\theta}_{i/k-1} \end{bmatrix}$ le plus proche de $\begin{bmatrix} -\phi_k \\ y_k \end{bmatrix}$.

Étape 2 : Mettre à jour l'estimation du vecteur global de paramètres Θ^*

- Cette deuxième étape de l'algorithme est consacrée à la mise à jour du vecteur global de paramètres estimé $\widehat{\Theta}_k$ à chaque instant k. Il s'agit d'une généralisation d'un algorithme de type OBE.

L'algorithme R-SARX-OBE

Initialisation : $P_0 = p_0 I_{ns}$, $\widehat{\Theta}_0$

Étape 1 : **Estimation de l'état discret** λ_k :

$$\begin{cases} \varepsilon_{k/k-1} = Y_k - \varphi_k^T \widehat{\Theta}_{k-1} \\ \hat{\lambda}_k = \underset{i=1,\cdots,s}{\operatorname{argmin}} \mid \varepsilon_{k/k-1}(i) \mid \\ \zeta_k = \underset{i=1,\cdots,s}{\min} \dfrac{\mid \varepsilon_{k/k-1}(i) \mid}{\delta_e} = \dfrac{\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid}{\delta_e} \end{cases} \tag{2.22}$$

Étape 2 : **Mettre à jour l'estimation du vecteur global de paramètres** Θ^* :

$$\widehat{\Theta}_k = \widehat{\Theta}_{k-1} + \Gamma_k \varepsilon_{k/k-1} \tag{2.23}$$

avec

$$\begin{cases} \Gamma_k = P_{k-1}\varphi_k\sigma_k(\gamma I_s + \varphi_k^T P_{k-1}\varphi_k\sigma_k)^{-1} \\ P_k = \dfrac{1}{\gamma}(I_{ns} - \Gamma_k\varphi_k^T)P_{k-1} \end{cases} \tag{2.24}$$

où

$$\sigma_k \quad = \quad \begin{cases} \gamma(\varphi_k^T P_{k-1}\varphi_k)^{-1}(C_k - I_s) \quad \text{Si} \quad \varphi_k^T P_{k-1}\varphi_k > 0 \quad \text{et} \quad \zeta_k > 1 \\ 0_{s\times s} \quad \text{Autrement} \end{cases} \tag{2.25}$$

et

$$C_k = \begin{bmatrix} 1 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \zeta_k & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ \hat{\lambda}_k^{\text{ième}}\text{ligne} \\ \\ \\ \end{matrix} \quad \begin{matrix} \\ \\ \hat{\lambda}_k^{\text{ième}}\text{colonne} \\ \\ \\ \end{matrix} \tag{2.26}$$

TABLE 2.1 – Algorithme d'identification avec bruit borné d'un modèle SARX

- $0 < \gamma < 1$ est le facteur d'oubli, il est utilisé dans le but de limiter l'influence d'une mauvaise initialisation et d'oublier l'influence des anciennes données. En effet, plus le nombre de données augmente, moins l'algorithme sera capable de s'adapter aux variations dans les données. Il est donc nécessaire d'introduire de l'oubli dans l'algorithme en limitant le poids des anciennes données pour maintenir la possibilité de s'adapter aux variations dans les données.
- La matrice $P_k \in \mathbb{R}^{ns \times ns}$ décrit l'orientation de l'ellipsoïde à l'instant k.
- La matrice $\Gamma_k \in \mathbb{R}^{ns \times s}$ représente le gain d'adaptation à chaque instant k afin de mettre à jour le vecteur de paramètres $\widehat{\Theta}_k$.
- Le rapport $\zeta_k = \dfrac{\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid}{\delta_e}$ est comparé à 1 comme il est décrit dans (2.25). Deux cas se produisent :
  - Si $\zeta_k \leq 1$, alors aucune mise à jour n'est faite pour $\widehat{\Theta}_k$ et l'adaptation à cet instant k est ignorée. Ceci est réalisé grâce à la matrice $\sigma_k = 0_{s \times s}$ qui garantit que $\Gamma_k = 0_{ns \times s}$.
  - Si $\zeta_k > 1$, alors dans le vecteur global de paramètres $\widehat{\Theta}_k$, uniquement le vecteur de paramètres $\widehat{\theta}_{\hat{\lambda}_k}$ associé à l'état discret estimé $\hat{\lambda}_k$ est mis à jour. Ceci est réalisé grâce à la structure particulière de $\sigma_k \in \mathbb{R}^{s \times s}$ définie par (2.25). Dans ce cas, la matrice σ_k est une matrice symétrique avec la structure suivante :

$$\sigma_k = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \neq 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix} \begin{matrix} \\ \\ \hat{\lambda}_k^{\text{ième}} \\ \\ \\ \end{matrix} \quad \begin{matrix} \\ \\ \\ \\ \hat{\lambda}_k^{\text{ième}} \end{matrix} \tag{2.27}$$

    Il n'y a qu'une valeur non nulle dans la matrice σ_k. La position de cette valeur non nulle correspond à la $\hat{\lambda}_k^{\text{ième}}$ ligne et la $\hat{\lambda}_k^{\text{ième}}$ colonne. Cette valeur est utilisée pour agir sur la matrice de gain d'adaptation Γ_k afin de mettre à jour uniquement le vecteur de paramètre associé à l'état discret estimé $\hat{\lambda}_k$.

Remarque 2. *Le nombre de sous modèles s est supposé connu à priori. Si s est inconnu, une approche itérative peut être utilisée pour l'estimer. Vu la similarité entre les systèmes linéaires à commutations et les systèmes affines par morceaux, cette approche d'estimation du nombre de sous-modèles est détaillée dans le prochain chapitre.*

2.2.2 Exécution de l'algorithme en mode batch

L'algorithme proposé peut être assimilé à un algorithme récursif des moindres carrés, sa complexité globale est $\mathscr{O}((sn)^2)$ à chaque instant. Ceci permet la mise en œuvre de l'algorithme en ligne. La complexité globale sur un horizon de longueur N est $\mathscr{O}(N(sn)^2)$, cela garantit la possibilité de la mise en œuvre de l'algorithme sur un large ensemble de données. Cependant, la mise en œuvre de l'algorithme en ligne peut souffrir de deux points faibles. Premièrement, à cause d'une mauvaise initialisation, la mise en œuvre de l'algorithme en ligne peut conduire à des mauvaises décisions sur l'estimation de l'état discret au début de la procédure d'identification. Deuxièmement, tous les sous-modèles doivent être régulièrement visités. Si un sous-modèle n'est pas sélectionné pendant long-temps (si le temps d'attente est long pour certains sous-modèles), il ne peut pas être estimé. Afin d'éviter de tels comportements pathologiques, et si la mise en œuvre de l'algorithme en mode en ligne n'est pas nécessaire, nous recommandons la mise en œuvre de l'algorithme en mode batch comme décrit dans le tableau 2.2.

Mise en œuvre hors-ligne de l'algorithme

– lancer l'algorithme d'identification sur un ensemble de données $\{\varphi_k, Y_k\}_{k=1}^N$;
iteration $\leftarrow 0$
répéter
  – appliquer un réarrangement aléatoire de l'ensemble de données $\{\varphi_k, Y_k\}_{k=1}^N$;
  – lancer l'algorithme à nouveau sur la nouvelle distribution de $\{\varphi_k, Y_k\}_{k=1}^N$ en utilisant la dernière estimation de $\widehat{\Theta}_k$ comme condition initiale.
  – *iteration* $\leftarrow$ *iteration* $+1$
jusqu'à convergence, par exemple, jusqu'à :

$$\| \widehat{\Theta}^{iteration+1} - \widehat{\Theta}^{iteration} \| \leq \varepsilon$$

TABLE 2.2 – Mise en œuvre de l'algorithme proposé en mode batch.

Remarque 3. *La borne supérieure δ_e sur l'amplitude du bruit est supposée connue à priori. Si δ_e est inconnu, la stratégie suivante est recommandée dans le cas de l'exécution de l'algorithme en mode en ligne : choisir une valeur de δ_e grande au début de la procédure d'identification, puis diminuer sa valeur au cours du temps. Si la mise en œuvre en ligne*

de l'algorithme n'est pas nécessaire, une deuxième stratégie pour faire un choix judicieux de δ_e est la suivante : itérer l'algorithme plusieurs fois en choisissant une grande valeur de δ_e à la première itération, puis diminuer sa valeur après chaque itération.

2.3 Analyse de l'algorithme R-SARX-OBE

Après avoir présenté l'algorithme R-SARX-OBE dans la section précédente, l'objectif de cette section est de faire une analyse sur son comportement et sa convergence. En effet, la convergence de l'algorithme R-SARX-OBE dépend de son initialisation et de la bonne ou mauvaise décision sur l'estimation de l'état discret. Nous présentons une analyse de convergence en montrant l'impact de l'initialisation et de l'estimation de l'état discret sur le comportement de l'algorithme.

Tout d'abord, nous considérons le comportement de l'erreur de prédiction a posteriori $\varepsilon_{k/k}$. Le lemme suivant montre que l'algorithme fournit un vecteur global de paramètres $\widehat{\Theta}_k$ estimé de sorte que l'amplitude de l'erreur de prédiction a posteriori associée à l'état discret estimé $\hat{\lambda}_k$ à chaque instant k est bornée par la borne supérieure sur le bruit δ_e.

Lemme 1. *Considérons un système à commutations avec bruit borné décrit par un modèle SARX comme défini dans la section 2.1. L'algorithme R-SARX-OBE présenté dans le tableau 2.1 est tel que, pour toutes conditions initiales, l'erreur de prédiction a posteriori associée à l'état discret estimé $\hat{\lambda}_k$ à chaque instant k satisfait :*

$$| \varepsilon_{k/k}(\hat{\lambda}_k) | \leqslant \delta_e, \quad \forall\ k \tag{2.28}$$

■

Preuve du Lemme 1.

A partir de (2.22) et (2.23), l'erreur de prédiction a posteriori $\varepsilon_{k/k}$ peut être écrite comme suit :

$$\varepsilon_{k/k} = \varepsilon_{k/k-1} - \varphi_k^T \Gamma_k \varepsilon_{k/k-1} \tag{2.29}$$

$$\varepsilon_{k/k} = (I_s - \varphi_k^T \Gamma_k)\varepsilon_{k/k-1} \tag{2.30}$$

Ceci peut aussi être réécrit, en utilisant l'expression de Γ_k comme suit :

$$\varepsilon_{k/k} = \gamma(\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k)^{-1} \varepsilon_{k/k-1} \tag{2.31}$$

Si $\varphi_k^T P_{k-1} \varphi_k > 0$, alors les deux cas suivants se produisent :

– *Si* $\zeta_k \leqslant 1$, *alors* $\sigma_k = 0$, $\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid \leq \delta_e$ *et on a :*

$$\mid \varepsilon_{k/k}(\hat{\lambda}_k) \mid = \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid \tag{2.32}$$

Ceci donne,

$$\mid \varepsilon_{k/k}(\hat{\lambda}_k) \mid \leqslant \delta_e \tag{2.33}$$

– *Si* $\zeta_k > 1$, *alors en utilisant la structure de la matrice* σ_k *(2.24) pour* $\sigma_k \neq 0$ *on trouve :*

$$\varepsilon_{k/k} = C_k^{-1} \varepsilon_{k/k-1} \tag{2.34}$$

Il en résulte :

$$\varepsilon_{k/k}(\hat{\lambda}_k) = \frac{1}{\zeta_k} \varepsilon_{k/k-1} \tag{2.35}$$

Par conséquent :

$$\mid \varepsilon_{k/k}(\hat{\lambda}_k) \mid = \delta_e \tag{2.36}$$

■

Ce lemme montre que l'algorithme proposé R-SARX-OBE réalise l'estimation d'un vecteur global de paramètres $\widehat{\Theta}_k$ de sorte que, à chaque instant k, la valeur de l'erreur de prédiction a posteriori associée à l'état discret estimé $\hat{\lambda}_k$ satisfait :

$$\mid \varepsilon_{k/k}(\hat{\lambda}_k) \mid \leqslant \delta_e \tag{2.37}$$

Ce résultat est cohérent avec l'objectif donné par l'équation (2.15). D'un point de vue géométrique, à chaque instant k, l'algorithme calcule un vecteur de paramètres $\widehat{\theta}_{\hat{\lambda}_k/k}$ tel que $\begin{bmatrix} -\phi_k \\ y_k \end{bmatrix}$ est inclus dans la bande formée par les deux hyperplans $\{x \in \mathbb{R}^{(n+1)\times 1}, x^T b_{\hat{\lambda}_k} = \delta_e\}$ et $\{x \in \mathbb{R}^{(n+1)\times 1}, x^T b_{\hat{\lambda}_k} = -\delta_e\}$ avec $b_{\hat{\lambda}_k} = \begin{bmatrix} \hat{\theta}_{\hat{\lambda}_k/k} \\ 1 \end{bmatrix}$.

Notons que ce résultat ne dépend pas de la bonne ou de la mauvaise estimation de l'état discret $\hat{\lambda}_k$.

L'objectif par la suite est d'évaluer l'impact d'une mauvaise décision sur l'estimation de l'état discret $\hat{\lambda}_k$ sur la convergence du vecteur global de paramètres estimé $\widehat{\Theta}_k$ vers le vrai vecteur global de paramètres Θ^* au cours de temps k. Deux cas se présentent :

– Si l'état discret est correctement estimé ($\hat{\lambda}_k = \lambda_k$), alors la sortie du système s'écrit comme suit :

$$y_k = \phi_k^T \theta_{\hat{\lambda}_k} + e_k \quad \text{avec} \quad \mid e_k \mid \leq \delta_e \tag{2.38}$$

– Si l'état discret est mal estimé ($\hat{\lambda}_k \neq \lambda_k$), alors la sortie du système s'écrit comme suit :

$$y_k = \phi_k^T \theta_{\hat{\lambda}_k} + e_k + \phi_k^T (\theta_{\lambda_k} - \theta_{\hat{\lambda}_k}) \tag{2.39}$$

Ceci peut être réécrit comme suit :

$$y_k = \phi_k^T \theta_{\hat{\lambda}_k} + V_k(\hat{\lambda}_k) = \phi_k^T \theta_{\hat{\lambda}_k} + e_k + d_k \tag{2.40}$$

où d_k représente l'impact d'une mauvaise décision sur l'estimation de l'état discret :

$$d_k = \begin{cases} 0, & \text{s'il n'y a pas une mauvaise décision } (\hat{\lambda}_k = \lambda_k) \\ \phi_k^T (\theta_{\lambda_k} - \theta_{\hat{\lambda}_k}), & \text{autrement} \end{cases} \tag{2.41}$$

Le terme $\phi_k^T (\theta_{\lambda_k} - \theta_{\hat{\lambda}_k})$ est fini, donc il existe une borne supérieure δ_d de telle sorte qu'à chaque instant k, nous avons $| \phi_k^T (\theta_{\lambda_k} - \theta_{\hat{\lambda}_k}) | \leq \delta_d$, d_k peut alors être interprété comme un bruit impulsif borné dont l'influence sur $\widehat{\Theta}_k$ doit être évaluée.

Avant d'introduire le prochain résultat, rappelons d'abord la définition usuelle de l'excitation persistante [126] :

Définition 5. *La séquence $\{\varphi_k\}_{k=1}^N$ est une séquence qui possède la propriété d'excitation persistante d'ordre $o_e \geq ns$ s'ils existent $\alpha > 0$ et $\beta > 0$ tel que pour tout k nous avons :*

$$\alpha I_{ns} \leq \sum_{i=0}^{o_e-1} \varphi_{k-i} \sigma_{k-i} \varphi_{k-i}^T \leq \beta I_{ns} \tag{2.42}$$

La condition d'excitation persistante joue un rôle très important dans une procédure d'identification paramétrique afin que les paramètres estimés convergent vers leurs vraies valeurs. La condition d'excitation persistante précédente exige que tous les sous-systèmes du système à commutations soient régulièrement visités et que chaque sous-système ait généré suffisamment de données. En effet, si certains sous-systèmes ne sont pas visités, alors les positions de la valeur non nulle dans la matrice σ_k reste toujours les mêmes au cours du temps et les autres positions correspondant aux sous-systèmes non visités restent égales à zéro. Il en résulte que la matrice $\sum_{i=0}^{o_e-1} \varphi_{k-i} \sigma_{k-i} \varphi_{k-i}^T$ est une matrice diagonale par blocs avec quelques blocs égaux à zéro, par conséquent la condition (2.42) ne peut pas être satisfaite dans ce contexte.

Le résultat donné ci-dessous se concentre sur l'influence des mauvaises décisions sur le comportement du vecteur d'erreur d'estimation sur les paramètres $\widetilde{\Theta}_k = \Theta^* - \widehat{\Theta}_k$.

Théorème 1. *Considérons un système à commutations soumis à un bruit borné et décrit par un modèle SARX comme défini dans la section 2.1. Si* $\{\varphi_k\}_{k=1}^{N}$ *est une séquence qui possède la propriété d'excitation persistante d'ordre* $o_e \geq ns$*, alors l'algorithme R-SARX-OBE donné dans le tableau 2.1 est tel que pour toutes conditions initiales et pour tout* $k \geq o_e + 1$ *tel que* $\sigma_k \neq 0_{s\times s}$*, le vecteur d'erreur sur les paramètres* $\widetilde{\Theta}_k = \Theta^* - \widehat{\Theta}_k$ *satisfait :*

$$\| \widetilde{\Theta}_k \|^2 \leq \eta_1 \gamma^k \| \widetilde{\Theta}_0 \|^2 + \eta_2 \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \tag{2.43}$$

avec $\eta_1 > 0$ *et* $\eta_2 > 0$*.*

En outre, si le nombre de mauvaises décisions sur l'estimation de l'état discret est négligeable, alors $\widetilde{\Theta}_k$ *est tel que :*

$$\| \widetilde{\Theta}_k \|^2 \leq \eta_1 \gamma^k \| \widetilde{\Theta}_0 \|^2 \tag{2.44}$$

et

$$\lim_{k\to\infty} \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid \leqslant \delta_e \tag{2.45}$$

■

Preuve du Théorème 1.

La preuve de ce théorème est divisée en quatre étapes :

- *Dans les étapes A.1) et A.2),nous montrons qu'ils existent* S_{sup} *et* S_{inf} *tel que :* $0 < S_{inf} I_{ns} \leq P_k^{-1} \leq S_{sup} I_{ns}$*.*
- *Dans l'étape A.3), nous montrons que la norme du vecteur d'erreur de paramètres* $\| \widetilde{\Theta}_k \|^2$ *est bornée en utilisant les résultats des étapes A.1), A.2).*
- *Dans l'étape A.4) nous concluons la preuve.*

A.1) ***Montrons qu'il existe :*** S_{sup} ***telle que*** $P_k^{-1} \leq S_{sup} I_{ns}$

A partir de l'équation (2.24) et le lemme d'inversion matricielle nous avons :

$$P_k^{-1} = \gamma P_{k-1}^{-1} + \varphi_k \sigma_k \varphi_k^T \tag{2.46}$$

(2.46) peut être réécrite sous la forme suivante :

$$P_k^{-1} = \gamma^k P_0^{-1} + \sum_{i=0}^{k-1} \gamma^i \varphi_{k-i} \sigma_{k-i} \varphi_{k-i}^T \tag{2.47}$$

Du fait que $\sum_{i=0}^{k-1}\gamma^i Z_{k-i} \leq \sum_{i=0}^{k-1}\gamma^i \sum_{i=0}^{k-1} Z_{k-i}$ *(* $\gamma > 0$ *et* $Z_{k-i} \geq 0$*) et à partir de la condition d'excitation persistante (2.42) pour* $k \geq o_e + 1$*, on trouve :*

$$P_k^{-1} \leq \gamma^k P_0^{-1} + \beta I_{ns} \sum_{i=0}^{k-1} \gamma^i \tag{2.48}$$

Il en résulte

$$P_k^{-1} \leq \gamma^k P_0^{-1} + \beta I_{ns} \frac{1-\gamma^k}{1-\gamma} \tag{2.49}$$

Nous avons $\gamma < 1$*, ceci garantit l'existence de* S_{sup} *tel que :*

$$P_k^{-1} \leq S_{sup} I_{ns} \tag{2.50}$$

avec $S_{sup} = \gamma^k p_0^{-1} + \beta \frac{1-\gamma^k}{1-\gamma}$.

A.2) ***Montrons qu'il existe*** S_{inf} ***tel que*** $0 < S_{inf} I_{ns} \leq P_k^{-1}$

A partir de (2.46), après o_e *itérations sur* P_k^{-1} *on a :*

$$P_k^{-1} = \gamma^{o_e} P_{k-o_e}^{-1} + \sum_{i=0}^{o_e-1} \gamma^i \varphi_{k-i} \sigma_{k-i} \varphi_{k-i}^T \tag{2.51}$$

Du fait que P_{k-o_e} *est une matrice définie positive et* $0 < \gamma < 1$*, on a alors :*

$$P_k^{-1} \geq \sum_{i=0}^{o_e-1} \gamma^i \varphi_{k-i} \sigma_{k-i} \varphi_{k-i}^T \tag{2.52}$$

En prenant en compte que $\gamma < 1$*, on trouve :*

$$P_k^{-1} \geq \gamma^{o_e-1} \sum_{i=0}^{o_e-1} \varphi_{k-i} \sigma_{k-i} \varphi_{k-i}^T \tag{2.53}$$

De l'hypothèse que $\{\varphi_k\}_{k=0}^N$ *est une séquence qui possède la propriété d'excitation persistante d'ordre* o_e*, il en résulte :*

$$P_k^{-1} \geq S_{inf} I_{ns} > 0 \tag{2.54}$$

avec $S_{inf} = \gamma^{o_e-1}\alpha$.

A.3) ***Montrons que*** $\| \widetilde{\Theta}_k \|^2$ ***est bornée***

On considère la fonction de Lyapunov

$$W_k = \widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k \tag{2.55}$$

A partir de (2.23), (2.24) et (2.31), on peut écrire $\widetilde{\Theta}_{k-1}$ *sous la forme suivante :*

$$\widetilde{\Theta}_{k-1} = \widetilde{\Theta}_k + \frac{1}{\gamma} P_{k-1} \varphi_k \sigma_k \varepsilon_{k/k} \tag{2.56}$$

En développant l'expression de W_{k-1} *on trouve :*

$$W_{k-1} = \widetilde{\Theta}_k^T P_{k-1}^{-1} \widetilde{\Theta}_k + \frac{1}{\gamma} \widetilde{\Theta}_k^T \varphi_k \sigma_k \varepsilon_{k/k} + \frac{1}{\gamma} \varepsilon_{k/k}^T \sigma_k \varphi_k^T \widetilde{\Theta}_k + \frac{1}{\gamma^2} \varepsilon_{k/k}^T \sigma_k \varphi_k^T P_{k-1} \varphi_k \sigma_k \varepsilon_{k/k}$$

Notons que :

$$(\widetilde{\Theta}_k^T \varphi_k \sigma_k \varepsilon_{k/k})^T = (\varepsilon_{k/k}^T \sigma_k \varphi_k^T \widetilde{\Theta}_k) = (\widetilde{\Theta}_k^T \varphi_k \sigma_k \varepsilon_{k/k}) \tag{2.57}$$

alors

$$W_{k-1} = \widetilde{\Theta}_k^T P_{k-1}^{-1} \widetilde{\Theta}_k + \frac{2}{\gamma} \widetilde{\Theta}_k^T \varphi_k \sigma_k \varepsilon_{k/k} + \frac{1}{\gamma^2} \varepsilon_{k/k}^T \sigma_k \varphi_k^T P_{k-1} \varphi_k \sigma_k \varepsilon_{k/k} \tag{2.58}$$

A partir du lemme d'inversion matricielle on a : $P_{k-1}^{-1} = \frac{1}{\gamma}(P_k^{-1} - \varphi_k \sigma_k \varphi_k^T)$. *En remplaçant cette expression dans la forme précédente de* W_{k-1}, *ceci donne :*

$$W_{k-1} = \frac{1}{\gamma} W_k - \frac{1}{\gamma} \widetilde{\Theta}_k^T \varphi_k \sigma_k \varphi_k^T \widetilde{\Theta}_k + \frac{2}{\gamma} \widetilde{\Theta}_k^T \varphi_k \sigma_k \varepsilon_{k/k} + \frac{1}{\gamma^2} \varepsilon_{k/k}^T \sigma_k \varphi_k^T P_{k-1} \varphi_k \sigma_k \varepsilon_{k/k}$$

Il en résulte

$$W_k = \gamma W_{k-1} + Q_k \tag{2.59}$$

avec

$$Q_k = \widetilde{\Theta}_k^T \varphi_k \sigma_k \varphi_k^T \widetilde{\Theta}_k - 2\widetilde{\Theta}_k^T \varphi_k \sigma_k \varepsilon_{k/k} - \frac{1}{\gamma} \varepsilon_{k/k}^T \sigma_k \varphi_k^T P_{k-1} \varphi_k \sigma_k \varepsilon_{k/k} \tag{2.60}$$

Deux cas sont possibles selon la valeur de la matrice σ_k *:*

A.3.1) $\sigma_k = 0_{s \times s}$

Dans ce cas, aucune mise à jour n'est faite pour $\widehat{\Theta}_k$ *et* $Q_k = 0$, *ce qui donne* $W_k = \gamma W_{k-1}$. *Du fait que* $\gamma < 1$, *on trouve*

$$\widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k \leq \widetilde{\Theta}_{k-1}^T P_{k-1}^{-1} \widetilde{\Theta}_{k-1} \tag{2.61}$$

A.3.2) $\sigma_k \neq 0_{s \times s}$

A l'aide de (2.14) et (2.21) on a :

$$\varphi_k^T \widetilde{\Theta}_k = \varepsilon_{k/k} - V_k \tag{2.62}$$

Q_k *peut être réécrit comme suit :*

$$\begin{aligned} Q_k = (\varepsilon_{k/k} - V_k)^T \sigma_k (\varepsilon_{k/k} - V_k) - 2(\varepsilon_{k/k} - V_k)^T \sigma_k \varepsilon_{k/k} \\ + \varepsilon_{k/k}^T \sigma_k \varepsilon_{k/k} - \frac{1}{\gamma} \varepsilon_{k/k}^T \sigma_k \varphi_k^T P_{k-1} \varphi_k \sigma_k \varepsilon_{k/k} - \frac{\gamma}{\gamma} \varepsilon_{k/k}^T \sigma_k \varepsilon_{k/k} \end{aligned} \tag{2.63}$$

Par conséquent

$$Q_k = V_k^T \sigma_k V_k - \frac{1}{\gamma} \varepsilon_{k/k}^T \sigma_k (\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k) \varepsilon_{k/k} \tag{2.64}$$

En utilisant (2.31) il en résulte :

$$Q_k = V_k^T \sigma_k V_k - \gamma \varepsilon_{k/k-1}^T \sigma_k (\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k)^{-1} \varepsilon_{k/k-1} \tag{2.65}$$

A partir de (2.25), on a :

$$\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k = \gamma C_k \tag{2.66}$$

A la suite, Q_k est écrit comme suit :

$$Q_k = V_k^T \sigma_k V_k - \varepsilon_{k/k-1}^T \sigma_k C_k^{-1} \varepsilon_{k/k-1} \tag{2.67}$$

Il n'y a qu'une valeur non nulle dans la matrice σ_k. La position de cette valeur non nulle correspond à la $\hat{\lambda}_k^{ième}$ ligne et le $\hat{\lambda}_k^{ième}$ colonne. Nous désignons cette valeur par a_k à chaque instant k. En utilisant aussi la forme de la matrice C_k donnée par (2.26), l'expression de de Q_k devient :

$$Q_k = a_k V_k^2(\hat{\lambda}_k) - \varepsilon_{k/k-1}^T(\hat{\lambda}_k) a_k \frac{1}{\zeta_k} \varepsilon_{k/k-1}(\hat{\lambda}_k) \tag{2.68}$$

A ce stade, nous rappelons que $\zeta_k = \dfrac{\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid}{\delta_e}$, du fait que $V_k(\hat{\lambda}_k) = e_k + d_k$, par conséquent :

$$Q_k = a_k (e_k + d_k)^2 - a_k \delta_e \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid \tag{2.69}$$

Q_k peut être réécrit de la manière suivante :

$$Q_k = a_k (e_k^2 - \delta_e \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid) + a_k (d_k^2 + 2 d_k e_k) \tag{2.70}$$

$\sigma_k \neq 0_{s \times s}$, alors $\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid > \delta_e$ et on a : $\mid e_k \mid \leq \delta_e$, d'où $(e_k^2 - \delta_e \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid) < 0$. D'autre part, a_k est nombre réel fini, par conséquent il existe $\rho > 0$ tel que $\mid a_k \mid \leq \rho$, d'où :

$$Q_k \leq \rho (d_k^2 + 2 d_k e_k) \tag{2.71}$$

A partir de (2.59) et (2.71), W_k devient :

$$W_k \leq \gamma W_{k-1} + q_k \tag{2.72}$$

avec $q_k = \rho (d_k^2 + 2 d_k e_k)$. Cette inégalité donne :

$$\widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k \leqslant \gamma \widetilde{\Theta}_{k-1}^T P_{k-1}^{-1} \widetilde{\Theta}_{k-1} + \rho (d_k^2 + 2 d_k e_k) \tag{2.73}$$

Il en résulte :

$$\widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k \leqslant \gamma^k \widetilde{\Theta}_0^T P_0^{-1} \widetilde{\Theta}_0 + \rho \sum_{i=0}^{k-1} \gamma^i (d_{k-i}^2 + 2 d_{k-i} e_{k-i}) \tag{2.74}$$

On a : d_{k-i} *est tel que* $\mid d_{k-i} \mid \leq \delta_d$, e_{k-i} *est tel que* $\mid e_{k-i} \mid \leq \delta_e$. *Alors nous obtenons :*

$$\widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k \leq \gamma^k \widetilde{\Theta}_0^T P_0^{-1} \widetilde{\Theta}_0 + \rho \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid (\delta_b + 2\delta_e) \tag{2.75}$$

Nous avons déjà montré que $0 < S_{inf} I_{ns} \leq P_k^{-1}$, *on obtient alors :*

$$\widetilde{\Theta}_k^T S_{inf} \widetilde{\Theta}_k \leq \gamma^k \widetilde{\Theta}_0^T P_0^{-1} \widetilde{\Theta}_0 + \rho(\delta_b + 2\delta_e) \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \tag{2.76}$$

avec $P_0 = p_o I_{ns}$. *Cette inégalité donne (2.43) :*

$$\parallel \widetilde{\Theta}_k \parallel^2 \leq \eta_1 \gamma^k \parallel \widetilde{\Theta}_0 \parallel^2 + \eta_2 \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \tag{2.77}$$

avec $\eta_1 = \dfrac{1}{p_o S_{inf}} > 0$ *et* $\eta_2 = \dfrac{1}{S_{inf}} \rho(\delta_b + 2\delta_e) > 0$.

A.4) Conclusion de la preuve

- *Si le nombre de mauvaises décisions sur l'estimation de l'état discret est négligeable, alors* $\sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \sim 0$ *et par conséquent on trouve (2.44).*
- $\lim_{k\to\infty} \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid \leqslant \delta_e$ *résulte du fait que* $\gamma < 1$.

■

Ce théorème exprime le comportement de l'erreur d'estimation sur les paramètres entre $\widehat{\Theta}_k$ et Θ^*. L'équation (2.43) montre que la présence des mauvaises décisions sur l'estimation de l'état discret peut conduire à une erreur dans l'estimation du vecteur global de paramètres. Néanmoins l'impact de ces mauvaises décisions est limité. En effet, le terme $\eta_2 \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid$ représente l'erreur due à des mauvaises décisions sur l'estimation de l'état discret. L'effet des mauvaises décisions sur l'estimation de l'état discret est atténué, filtré et ignoré au cours de temps grâce au facteur d'oubli γ. Il est intéressant de noter que la proportion des mauvaises décisions sur l'estimation de l'état discret diminue au cours de temps et par conséquent, $\parallel \widetilde{\Theta}_k \parallel^2$ décroît d'une manière exponentielle. $\parallel \widetilde{\Theta}_k \parallel^2$ décroît tant que $\sigma_k \neq 0_{s\times s}$ et que la propriété de l'excitation persistante est satisfaite.

2.4 Interprétation géométrique de l'algorithme R-SARX-OBE

Dans cette section, nous présentons une interprétation géométrique de l'algorithme. Tout d'abord, nous considérons les définitions de l'ensemble d'observations $\mathscr{S}_k(i)$ et l'ellipsoïde $\mathscr{E}_{k-1}$ pour chaque instant k de la manière suivante :

Définition 6. *L'ensemble d'observations $\mathscr{S}_k(i)$ est défini à chaque instant k comme suit :*

$$\mathscr{S}_k(i) = \{\Theta \in \mathbb{R}^{ns}, \mid \varepsilon_k(i) \mid \leq \delta_e\} \tag{2.78}$$

avec $\varepsilon_k = Y_k - \varphi_k^T \Theta$. $\mathscr{S}_k(i)$ est la bande qui contient tous les Θ possibles pour lesquels, la $i^{ième}$ composante (θ_i) est compatible avec la borne supérieure sur le bruit δ_e. Cette bande correspond à l'intersection entre les deux demi-espaces $\varepsilon_k(i) \leq \delta_e$ et $-\delta_e \leq \varepsilon_k(i)$. On notera que $\Theta^* \in \mathscr{S}_k(\lambda_k)$.

Définition 7. *L'ellipsoïde $\mathscr{E}_{k-1}$ est défini à chaque instant k comme suit :*

$$\mathscr{E}_{k-1} = \{\Theta \in \mathbb{R}^{ns}, (\Theta - \widehat{\Theta}_{k-1})^T P_{k-1}^{-1} (\Theta - \widehat{\Theta}_{k-1}) \leq \mu_{k-1}^2\} \tag{2.79}$$

où P_{k-1}^{-1} et μ_{k-1}^2 caractérisent l'orientation et la taille de l'ellipsoïde, $\widehat{\Theta}_{k-1}$ représente le centre de l'ellipsoïde.

Le résultat suivant donne une interprétation géométrique sur le fonctionnement de l'algorithme à chaque instant k.

Théorème 2. *Considérons un système à commutations soumis à un bruit borné et décrit par un modèle SARX comme défini dans la section 2.1. L'algorithme d'identification donné dans le tableau 2.1 permet l'estimation de l'ellipsoïde $\mathscr{E}_k$, un contour externe de l'ellipsoïde $(\mathscr{S}_k(\hat{\lambda}_k) \cap \mathscr{E}_{k-1})$ tel que*

$$\mu_k^2 = \gamma \mu_{k-1}^2 + \varepsilon_{k/k-1}^T \sigma_k \left(\zeta_k^{-2} I_s - C_k^{-1}\right) \varepsilon_{k/k-1} \tag{2.80}$$

et

$$\mu_k^2 \leq \gamma \mu_{k-1}^2 \tag{2.81}$$

De plus, si $\{\varphi_k\}_{k=1}^N$ est une séquence qui possède la propriété d'excitation persistante d'ordre $o_e \geq ns$ alors $\mathscr{E}_k$ est borné.

Finalement, si on suppose que $\Theta^* \in \mathscr{E}_{k-1}$ et s'il n'y a pas de mauvaise décision c'est à dire $\hat{\lambda}_k = \lambda_k$, alors

$$\Theta^* \in \mathscr{E}_k \tag{2.82}$$

■

Preuve du Théorème 2.

1. *Tout d'abord, prenons le cas* $\sigma_k \neq 0_{s\times s}$.

   On considère $\Theta \in \mathscr{E}_{k-1}$ *et* $\Theta \in \mathscr{S}_k(\hat{\lambda}_k)$, *on a d'après les définitions 6 et 7 :*

$$\gamma(\Theta - \widehat{\Theta}_{k-1})^T P_{k-1}^{-1}(\Theta - \widehat{\Theta}_{k-1}) \leq \gamma\mu_{k-1}^2 \tag{2.83}$$

   et

$$|\, \varepsilon_k(\hat{\lambda}_k)\,| \leq \delta_e \tag{2.84}$$

   On a $\dfrac{|\, \varepsilon_k(\hat{\lambda}_k)\,|}{\delta_e} \leq 1$, *par conséquent on peut écrire :*

$$\varepsilon_k^T M_{\delta_e} \varepsilon_k - 1 \leq 0 \tag{2.85}$$

   avec $M_{\delta_e} \in \mathbb{R}^{s\times s}$ *donnée par :*

$$M_{\delta_e} = \begin{array}{c} \\ \\ \\ \\ \\ \hat{\lambda}_k \end{array} \begin{array}{c} \overset{\hat{\lambda}_k}{\overbrace{\phantom{\begin{bmatrix}0 & \cdots & \cdots & \cdots & 0\end{bmatrix}}}} \\ \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \delta_e^{-2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix} \end{array} \tag{2.86}$$

   Ceci donne pour chaque ε_k

$$\varepsilon_k^T \left(M_{\delta_e} \varepsilon_k^T \sigma_k \varepsilon_k - \sigma_k\right) \varepsilon_k \leq 0 \tag{2.87}$$

   Ceci donne

$$\left(M_{\delta_e} \varepsilon_k^T \sigma_k \varepsilon_k - \sigma_k\right) \leq 0 \tag{2.88}$$

   Par conséquent

$$\varepsilon_{k/k-1}^T \left(M_{\delta_e} \varepsilon_k^T \sigma_k \varepsilon_k - \sigma_k\right) \varepsilon_{k/k-1} \leq 0 \tag{2.89}$$

   d'où

$$\varepsilon_{k/k-1}^T M_{\delta_e} \varepsilon_{k/k-1} (\varepsilon_k^T \sigma_k \varepsilon_k) - \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} \leq 0 \tag{2.90}$$

   En utilisant la structure de la matrice M_{δ_e} *donnée par (2.86), on trouve*

$$\frac{(\varepsilon_{k/k-1}(\hat{\lambda}_k))^2}{\delta_e^2} (\varepsilon_k^T \sigma_k \varepsilon_k) - \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} \leq 0 \tag{2.91}$$

Alors,

$$\varepsilon_k^T \sigma_k \varepsilon_k \leq \zeta_k^{-2} \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} \tag{2.92}$$

En prenant en compte (2.83), si $\Theta \in (\mathscr{S}_k(\hat{\lambda}_k) \cap \mathscr{E}_{k-1})$, *on obtient :*

$$\gamma(\Theta - \widehat{\Theta}_{k-1})^T P_{k-1}^{-1}(\Theta - \widehat{\Theta}_{k-1}) + \varepsilon_k^T \sigma_k \varepsilon_k \leq \gamma \mu_{k-1}^2 + \zeta_k^{-2} \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} \tag{2.93}$$

Rappelons que $\widehat{\Theta}_{k-1}$ *et* P_{k-1}^{-1} *sont tels que :*

$$\begin{cases} \widehat{\Theta}_{k-1} = \widehat{\Theta}_k - \gamma P_{k-1} \varphi_k \sigma_k \varepsilon_{k/k} \\ P_{k-1}^{-1} = \dfrac{1}{\gamma}(P_k^{-1} - \varphi_k \sigma_k \varphi_k^T) \end{cases} \tag{2.94}$$

En utilisant (2.94) dans (2.93), on a :

$$(\Theta - \widehat{\Theta}_k)^T P_k^{-1}(\Theta - \widehat{\Theta}_k) \leq \gamma \mu_{k-1}^2 + \zeta_k^{-2} \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} - \varepsilon_{k/k}^T \frac{\sigma_k}{\gamma} \left(\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k\right) \varepsilon_{k/k} \tag{2.95}$$

On dénote μ_k^2 *comme suit :*

$$\mu_k^2 = \gamma \mu_{k-1}^2 + \zeta_k^{-2} \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} - \varepsilon_{k/k}^T \frac{\sigma_k}{\gamma} \left(\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k\right) \varepsilon_{k/k} \tag{2.96}$$

Compte tenu de (2.31), nous pouvons écrire :

$$\mu_k^2 = \gamma \mu_{k-1}^2 + \zeta_k^{-2} \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} - \varepsilon_{k/k-1}^T \gamma \sigma_k \left(\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k\right)^{-1} \varepsilon_{k/k-1} \tag{2.97}$$

Soit

$$\mu_k^2 = \gamma \mu_{k-1}^2 + \zeta_k^{-2} \varepsilon_{k/k-1}^T \sigma_k \varepsilon_{k/k-1} - \varepsilon_{k/k-1}^T \sigma_k C_k^{-1} \varepsilon_{k/k-1} \tag{2.98}$$

Ceci correspond à μ_k^2 *donné par (2.80), d'où*

$$(\Theta - \widehat{\Theta}_k)^T P_k^{-1}(\Theta - \widehat{\Theta}_k) \leq \mu_k^2 \tag{2.99}$$

2. *Pour le cas* $\sigma_k = 0_{s \times s}$ *le résultat est le même du fait que* $P_k^{-1} = \gamma P_{k-1}^{-1}$ *et* $\widehat{\Theta}_k = \widehat{\Theta}_{k-1}$.
3. *Si* $\{\varphi_k\}_{k=1}^N$ *est une séquence qui possède la propriété d'excitation persistante d'ordre* $o_e \geq ns$, *alors il a été prouvé que* P_k *est borné (voir la démonstration du théorème 1). À propos de* μ_k^2, *si* $\sigma_k = 0_{s \times s}$ *alors* $\mu_k^2 = \gamma \mu_{k-1}^2$. *Si* $\sigma_k \neq 0_{s \times s}$, *on sait que* $\zeta_k > 1$ *alors* $\varepsilon_{k/k-1}^T \sigma_k \left(\zeta_k^{-2} I_s - C_k^{-1}\right) \varepsilon_{k/k-1} < 0$ *par conséquent* μ_k^2 *est borné et* $\mu_k^2 \leq \gamma \mu_{k-1}^2$. *Ceci montre que* $\mathscr{E}_k$ *est borné.*
4. *Si* $\hat{\lambda}_k = \lambda_k$ *alors* $\Theta^* \in \mathscr{S}_k(\hat{\lambda}_k)$. *Si de plus nous avons* $\Theta^* \in \mathscr{E}_{k-1}$, *il en résulte que* $\Theta^* \in (\mathscr{S}_k \cap \mathscr{E}_{k-1}) \subset \mathscr{E}_k$.

■

Ce théorème montre qu'à chaque instant k, l'algorithme R-SARX-OBE calcule un vecteur global de paramètres $\widehat{\Theta}_k$ appartenant à l'intersection entre l'ellipsoïde $\mathscr{E}_{k-1}$ et la bande $\mathscr{S}_k(\hat{\lambda}_k)$. $\widehat{\Theta}_k$ est le centre de l'ellipsoïde $\mathscr{E}_k$. Si $\widehat{\Theta}_{k-1}$ appartient déjà à $\mathscr{S}_k(\hat{\lambda}_k)$ alors l'adaptation du vecteur de paramètre est ignorée. Ceci est illustré sur la figure 2.1.

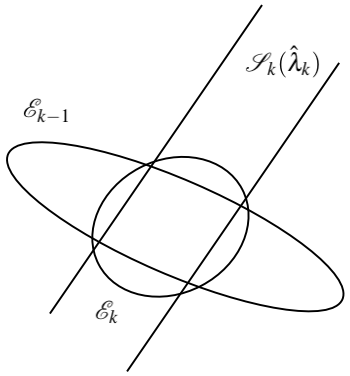

FIGURE 2.1 – Exemple illustratif : $\mathscr{S}_k(\hat{\lambda}_k) \cap \mathscr{E}_{k-1} \subset \mathscr{E}_k$.

$\mathscr{E}_k$ est un ellipsoïde externe qui englobe l'ensemble de $\left(\bigcap_{i=0}^{k} \mathscr{S}_i(\hat{\lambda}_i)\right) \bigcap \mathscr{E}_0$ et il a été prouvé que $\mathscr{E}_k$ est borné. En outre, il a été démontré aussi que si le nombre de mauvaises décisions sur l'estimation de l'état discret est négligeable et si Θ^* appartient à $\mathscr{E}_0$, alors Θ^* appartient à $\mathscr{E}_k$. $\mathscr{E}_k$ décrit un ensemble d'incertitude sur les paramètres : il correspond au voisinage de $\widehat{\Theta}_k$ qui contient Θ^*.

2.5 Identification des systèmes MIMO à commutations décrits par un modèles MIMO-SARX

Nous avons présenté dans la section 2.2 un algorithme d'identification des systèmes à commutations soumis à un bruit borné et décrits par un modèle SISO-SARX. Dans cette section, nous proposons une extension de l'algorithme au cas des systèmes MIMO à commutations décrits par un modèle MIMO-SARX. Considérons un système linéaire MIMO à commutations décrit par un modèle MIMO-SARX de la manière suivante :

$$y_k = -\sum_{j=1}^{n_a} A_j(\lambda_k) y_{k-j} + \sum_{i=0}^{n_b} B_i(\lambda_k) u_{k-i} + e_k \tag{2.100}$$

- n_a et n_b sont les ordres du système
- $u_k \in \mathbb{R}^{n_u}$ et $y_k \in \mathbb{R}^{n_y}$.
- $e_k \in \mathbb{R}^{n_y}$ est le terme de bruit supposé inconnu mais borné :

$$e_k^T \Delta_e^{-2} e_k \leq 1 \ , \quad \forall \ k \tag{2.101}$$

  où Δ_e est une matrice symétrique positive connue qui reflète la borne supérieure connue sur le bruit e_k.
- $\{A_j(\lambda_k)\} \in \mathbb{R}^{n_y \times n_y}$ diagonal et $\{B_i(\lambda_k)\} \in \mathbb{R}^{n_y \times n_u}$ sont les matrices de paramètres inconnus à identifier associés au sous-modèle désigné par λ_k.
- $\lambda_k \in \{1, \cdots, s\}$ est l'état discret qui indique le sous-modèle actif à l'instant k, s est le nombre total de sous-modèles.

La sortie y_k peut être exprimée de la manière suivante :

$$y_k = \phi_k^T \theta_{\lambda_k} + e_k \tag{2.102}$$

avec

$$\begin{cases} \theta_{\lambda_k} = vect(\Sigma(\lambda_k)) \\ \phi_k = r_k \otimes I_{n_y} \end{cases} \tag{2.103}$$

où

$$\Sigma(\lambda_k) = (A_1(\lambda_k) \cdots A_{n_a}(\lambda_k) \quad B_0(\lambda_k) \quad B_1(\lambda_k) \cdots B_{n_b}(\lambda_k)) \tag{2.104}$$

et

$$r_k^T = \begin{bmatrix} -y_{k-1}^T & \cdots & -y_{k-n_a}^T & u_k^T & u_{k-1}^T & \cdots & u_{k-n_b}^T \end{bmatrix} \tag{2.105}$$

La fonction *vect* permet de transformer la matrice $\Sigma(\lambda_k)$ en arrangeant ses colonnes sous forme d'un vecteur colonne. L'objectif ici est d'estimer l'état discret $\{\lambda_k\}_{k=1}^N$ et les vecteurs de paramètres $\{\theta_i\}_{i=1}^s$. Le nombre de paramètres à identifier est $n = (n_y n_a + n_u(n_b+1))n_y$. De la même façon que pour le cas SISO (cf. section 2.1), la forme compacte pour le cas MIMO s'écrit comme suit :

$$Y_k = \varphi_k^T \Theta^* + V_k \tag{2.106}$$

avec $Y_k \in \mathbb{R}^{sn_y}$ est le vecteur de sortie étendu, $\Theta^* \in \mathbb{R}^n$ est le vecteur global de paramètres, $V_k \in \mathbb{R}^{sn_y}$ est le vecteur de bruit étendu et $\varphi_k = I_s \otimes \phi_k = I_s \otimes (r_k \otimes I_{n_y})$.

L'algorithme d'identification dans le cas MIMO (nommé "algorithme MIMO-SARX-OBE") est organisé de la même façon que l'algorithme R-SARX-OBE dans le cas SISO. Une différence principale entre les deux formulations concerne l'objectif d'identification.

En effet, l'objectif dans le cas MIMO est de fournir une estimation de l'état discret $\hat{\lambda}_k$ et une estimation du vecteur global de paramètres $\widehat{\Theta}$ tels que à chaque instant k la condition suivante est satisfaite :

$$\varepsilon_k(\hat{\lambda}_k)^T \Delta_e^{-2} \varepsilon_k(\hat{\lambda}_k) \leq 1 \quad \forall\ k \tag{2.107}$$

où $\varepsilon_k = Y_k - \varphi^T \widehat{\Theta}$ désigne le vecteur d'erreur, il est structuré comme suit :

$$\varepsilon_k^T = \begin{bmatrix} \varepsilon_k(1)^T & \cdots & \varepsilon_k(s)^T \end{bmatrix} \tag{2.108}$$

où $\varepsilon_k(i) \in \mathbb{R}^{n_y \times 1}$ désigne la i^{me} composante de ε_k. Les deux étapes de l'algorithme MIMO-SARX-OBE sont récapitulées dans le tableau 2.3.

Lemme 2. *Considérons un système linéaire MIMO à commutations soumis à un bruit borné et décrit par un modèle MIMO SARX. L'algorithme d'identification donné dans le tableau 2.3 est tel que pour toutes conditions initiales, l'erreur de prédiction a posteriori $\varepsilon_{k/k}$ satisfait :*

$$\varepsilon_{k/k}(\hat{\lambda}_k)^T \Delta_e^{-2} \varepsilon_{k/k}(\hat{\lambda}_k) \leqslant 1, \quad \forall\ k \tag{2.109}$$

■

Preuve du Lemme 2.

A partir de la définition de $\varepsilon_{k/k}$, $\varepsilon_{k/k-1}$, (2.115) et de l'expression de Γ_k, l'erreur de prédiction a posteriori $\varepsilon_{k/k}$ peut être écrite comme suit :

$$\varepsilon_{k/k} = \gamma(\gamma I_s + \varphi_k^T P_{k-1} \varphi_k \sigma_k)^{-1} \varepsilon_{k/k-1} \tag{2.110}$$

Si $\varphi_k^T P_{k-1} \varphi_k > 0$, alors les deux cas suivants se produisent :

- *Si $\zeta_k \leqslant 1$, alors $\sigma_k = 0_{sn_y \times sn_y}$, par conséquent $\varepsilon_{k/k}^{(\hat{\lambda}_k)} = \varepsilon_{k/k-1}^{(\hat{\lambda}_k)}$, ce qui donne :*

$$\varepsilon_{k/k}^{(\hat{\lambda}_k)^T} \Delta_e^{-2} \varepsilon_{k/k}^{(\hat{\lambda}_k)} \leq 1 \tag{2.111}$$

- *Si $\zeta_k > 1$, alors en utilisant la structure de la matrice σ_k (2.24) pour $\sigma_k \neq 0_{sn_y \times sn_y}$ on trouve : $\varepsilon_{k/k} = C_k^{-1} \varepsilon_{k/k-1}$. Il en résulte : $\varepsilon_{k/k}^{(\hat{\lambda}_k)} = {\zeta_k}^{-1} \varepsilon_{k/k-1}^{(\hat{\lambda}_k)}$ Par conséquent :*

$$\varepsilon_{k/k}^{(\hat{\lambda}_k)^T} \Delta_e^{-2} \varepsilon_{k/k}^{(\hat{\lambda}_k)} = \varepsilon_{k/k-1}^{(\hat{\lambda}_k)^T} \Delta_e^{-2} \varepsilon_{k/k-1}^{(\hat{\lambda}_k)} {\zeta_k}^{-2} \tag{2.112}$$

En utilisant la valeur de ζ_k, on trouve :

$$\varepsilon_{k/k}^{(\hat{\lambda}_k)^T} \Delta_e^{-2} \varepsilon_{k/k}^{(\hat{\lambda}_k)} = 1 \tag{2.113}$$

Ce lemme montre que l'algorithme fournit un vecteur global de paramètres $\widehat{\Theta}_k$ tel que l'erreur de prédiction a posteriori à chaque instant k satisfait l'objectif d'identification donné par (2.107). Concernant le comportement de $\widehat{\Theta}_k$, les résultats des théorèmes 1 et 2 restent toujours valable pour le cas MIMO-SARX.

L'algorithme MIMO-SARX-OBE.

Initialisation : $\widehat{\Theta}_0$, $P_0 = p_0 I_{ns}$

Étape 1 : **Estimation de l'état discret** λ_k :

$$\begin{cases} \hat{\lambda}_k = \underset{i=1,\cdots,s}{\operatorname{argmin}} \sqrt{\varepsilon_{k/k-1}(i)^T \Delta_e^{-2} \varepsilon_{k/k-1}(i)} \\ \\ \zeta_k = \sqrt{\varepsilon_{k/k-1}(\hat{\lambda}_k)^T \Delta_e^{-2} \varepsilon_{k/k-1}(\hat{\lambda}_k)} \end{cases} \tag{2.114}$$

Step 2 : **Mettre à jour l'estimation du vecteur global de paramètres** $\widehat{\Theta}_k$:

$$\widehat{\Theta}_k = \widehat{\Theta}_{k-1} + \Gamma_k \varepsilon_{k/k-1} \tag{2.115}$$

avec

$$\begin{cases} \Gamma_k = P_{k-1}\varphi_k \sigma_k (\gamma I_{sn_y} + \varphi_k^T P_{k-1} \varphi_k \sigma_k)^{-1} \\ P_k = \dfrac{1}{\gamma}(I_{sn} - \Gamma_k \varphi_k^T) P_{k-1} \end{cases} \tag{2.116}$$

où

$$\sigma_k \;=\; \begin{cases} \gamma(\varphi_k^T P_{k-1}\varphi_k)^{-1}(C_k - I_{sn_y}) & \text{si} \quad \varphi_k^T P_{k-1}\varphi_k > 0 \quad \text{et} \quad \zeta_k > 1 \\ 0_{sn_y \times sn_y} & \text{autrement} \end{cases} \tag{2.117}$$

et

$$C_k = \begin{bmatrix} 1 & \cdots & & \cdots & & \cdots & 0 \\ \vdots & \ddots & & \ddots & & \ddots & \vdots \\ & & \zeta_k & \cdots & 0 & & \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ & & 0 & \cdots & \zeta_k & & \\ \vdots & \ddots & & \ddots & & \ddots & \vdots \\ 0 & \cdots & & \cdots & & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ 1 \\ \vdots \\ n_y \\ \\ \\ \end{matrix} \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \hat{\lambda}_k \tag{2.118}$$

$$\overbrace{1 \quad \cdots \quad n_y}^{\hat{\lambda}_k}$$

TABLE 2.3 – Algorithme d'identification du système linéaire MIMO à commutations avec bruit borné.

2.6 Exemples numériques

Afin d'illustrer les performances de l'algorithme, nous présentons dans cette section 3 expériences de simulations. Les deux premières expériences sont dédiées à l'identification des systèmes à commutations décrits par un modèle SISO-SARX. Dans la première expérience, nous présentons des résultats numériques illustrant les performances de l'algorithme proposé implémenté en ligne. Ces performances sont comparées à celles obtenues avec l'algorithme récursif proposé dans [8]. Dans la deuxième expérience, nous exécutons l'algorithme en mode batch et nous comparons ses performances à celles obtenues avec l'algorithme proposé dans [148]. La troisième expérience est dédiée à l'identification des systèmes MIMO à commutations décrits par un modèle MIMO-SARX.

Toutes les simulations réalisées ci-dessous sont obtenues en exécutant "Matlab 8.3 (R2014a)- sur processeur 2,7 GHz Intel Core i7". Afin de quantifier l'efficacité de l'algorithme proposé, les indices statistiques suivants sont utilisés dans toutes les simulations :

– FIT

$$\text{FIT} = 100\left(1 - \frac{\| y - \hat{y} \|}{\| y - \bar{y}\ 1_N \|}\right) \tag{2.119}$$

– RMSE ("The Root Mean Squared Error")

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N} \| y_k - \hat{y}_k \|^2} \tag{2.120}$$

– PEE ("The Parameter Estimation Error")

$$\text{PEE} = \| \Theta^* - \widehat{\Theta} \| \tag{2.121}$$

où $\hat{y}$ est la séquence de sortie estimée construite à partir du vecteur de paramètres estimé $\hat{\boldsymbol{\Theta}}$ et la séquences $\{\hat{\lambda}_k\}_{k=1}^{N}$, y est la vraie séquence de sortie, $\bar{y}$ représente la moyenne de la vraie séquence de sortie, 1_N est un vecteur de longueur N avec toutes ses composantes égales à 1, N est le nombre de données disponibles.

2.6.1 Première expérience - Exécution de l'algorithme R-SARX-OBE en mode en-ligne

Nous présentons ici, une expérience de simulation afin d'illustrer les performances de l'algorithme proposé R-SARX-OBE à identifier en temps réel les paramètres d'un modèle SISO-SARX. L'exemple numérique utilisé dans [8] est considéré. Les données ont été générées selon (2.2) et (2.4). Le système considéré est le suivant :

$$y_k = \begin{bmatrix} -y_{k-1} & -y_{k-2} & u_k & u_{k-1} \end{bmatrix} \theta_{\lambda_k} + e_k \tag{2.122}$$

avec :

$$\begin{cases} \theta_1 = \begin{bmatrix} -0.0322 & 0.8017 & -1.2878 & -1.1252 \end{bmatrix}^T \\ \theta_2 = \begin{bmatrix} -0.1921 & 0.5917 & 1.1050 & 0.0316 \end{bmatrix}^T \\ \theta_3 = \begin{bmatrix} 1.4746 & -0.5286 & -0.4055 & 0.2547 \end{bmatrix}^T \end{cases} \tag{2.123}$$

Le nombre de sous-modèles est $s = 3$. Les données ont été obtenues avec une commutation arbitraire de l'état discret $\lambda_k \in \{1,2,3\}$. La séquence d'entrée u_k est une séquence aléatoire de moyenne nulle avec une distribution uniforme sur $[-2\ ;\ 2]$. Le terme de bruit est une séquence aléatoire de moyenne nulle uniformément répartie dans $[-\delta_e\ ;\ \delta_e]$. La borne supérieure sur le bruit δ_e a été ajustée de façon à obtenir une valeur souhaitée du rapport signal sur bruit (SNR).

Dans un premier temps, l'algorithme R-SARX-OBE a été appliqué sur un horizon de longueur $N = 1000$ et pour $SNR = 20dB$. La figure 2.2 montre le comportement de l'erreur de prédiction a priori $\varepsilon_{k/k-1}(\hat{\lambda}_k)$ et l'erreur de prédiction a posteriori $\varepsilon_{k/k}(\hat{\lambda}_k)$ associées à l'état discret estimé en fonction du temps k. Il est clair que l'algorithme proposé réalise l'estimation de l'état discret et l'estimation du vecteur de paramètres tel que l'erreur a posteriori associée à l'état discret estimé soit bornée par la borne supérieure sur le bruit. De plus, l'erreur à priori associée à l'état discret estimé diminue et sa valeur tend vers l'intervalle $[-\delta_e\ ;\ \delta_e]$.

La convergence du vecteur global de paramètres estimé $\widehat{\Theta}_k$ vers le vrai vecteur global de paramètres Θ^* est illustrée sur la figure 2.3 en fonction du temps. Il apparaît clairement que l'algorithme permet une convergence assez rapide de $\widehat{\Theta}_k$ vers Θ^*.

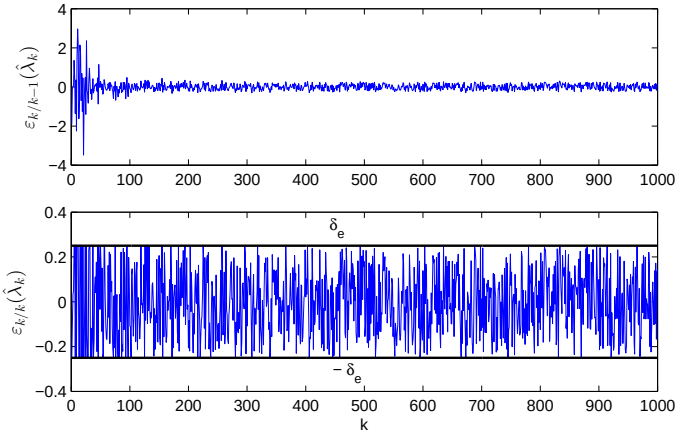

FIGURE 2.2 – $\varepsilon_{k/k}(\hat{\lambda}_k)$ et $\varepsilon_{k/k-1}(\hat{\lambda}_k)$ en fonction du temps k pour SNR$=20$dB.

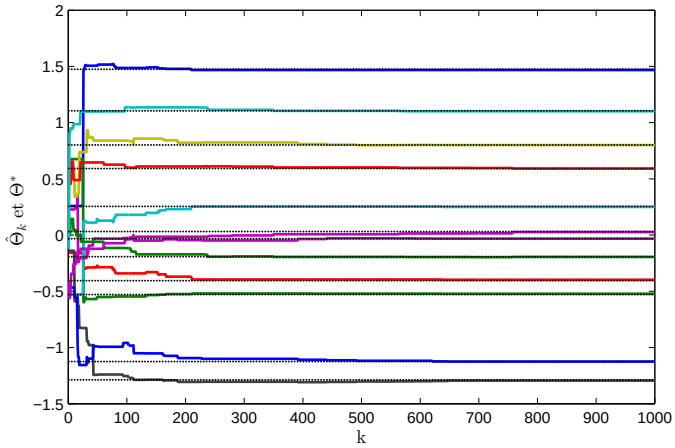

FIGURE 2.3 – Convergence de paramètres estimés (trait plein) vers les vraies paramètres (trait pointillé).

Cette convergence assure la bonne estimation de l'état discret comme l'illustre la figure 2.4. Pour des raisons de lisibilité, la représentation de l'état discret réel λ_k du système (cercle) et de l'état discret estimé $\hat{\lambda}_k$ (point) a été illustrée sur un horizon restreint à $\begin{bmatrix}200 & 300\end{bmatrix}$. Il apparaît clairement que l'état discret est correctement estimé, $\hat{\lambda}_k$ correspond à λ_k sur la quasi-totalité des points de l'horizon de temps.

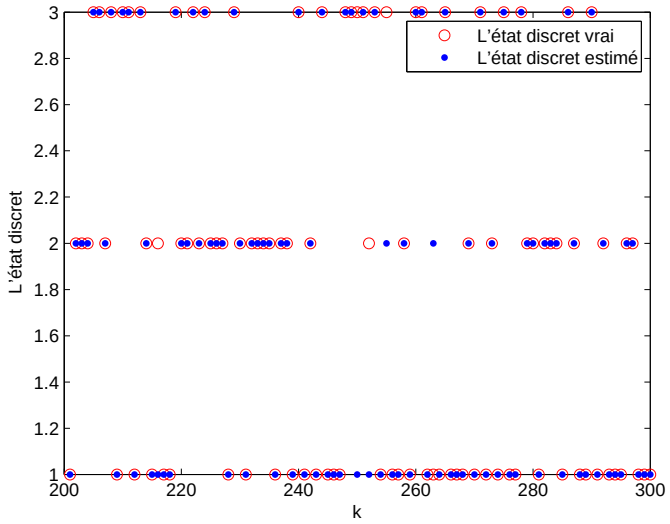

FIGURE 2.4 – Comparaison de l'état discret λ_k et de son estimé $\hat{\lambda}_k$ en fonction du temps k.

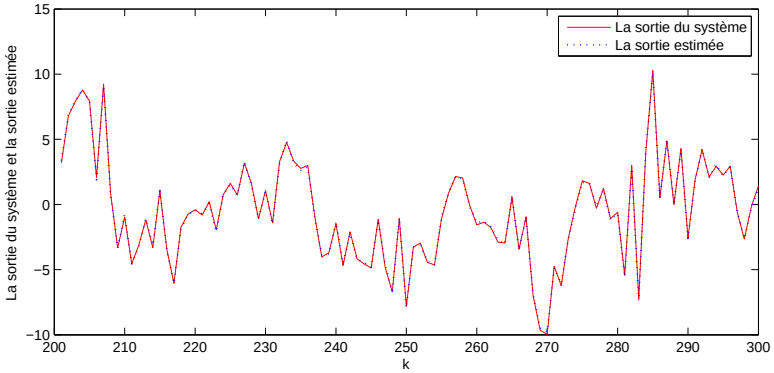

FIGURE 2.5 – La sortie du système et la sortie estimée en fonction du temps k.

Sur la figure 2.5 nous présentons la sortie réelle du système et la sorte estimée du modèle sur le même horizon de temps $\begin{bmatrix} 200 & 300 \end{bmatrix}$. La sortie estimée correspond bien à la sortie réelle du système.

Dans un second temps, comme il est important de comparer les performances de l'algorithme proposé par rapport à d'autres méthodes, nous comparons l'algorithme R-SARX-OBE avec la méthode récursive proposée dans [8].

(Temps de calcul sur 100 jeux de données : 14.584s , 1 jeux de données : 0.1458s)

$\hat{\theta}_1 \pm$ écart-type	$\hat{\theta}_2 \pm$ écart-type	$\hat{\theta}_3 \pm$ écart-type
-0.0306 ± 0.0400	-0.1874 ± 0.0799	1.4667 ± 0.0411
0.7913 ± 0.0803	0.5891 ± 0.0663	-0.5199 ± 0.0462
-1.2584 ± 0.0956	1.0948 ± 0.0806	-0.4112 ± 0.0645
-1.1106 ± 0.0911	0.0361 ± 0.0922	0.2497 ± 0.0445

FIT	**RMSE**	**PEE**
89.2 %	0.4512 ± 0.0694	0.0862 ± 0.0265

TABLE 2.4 – Première expérience - Résultats et mesures statistiques avec l'algorithme proposé dans [8]

(Temps de calcul sur 100 jeux de données : 12.8128 s , 1 jeux de données : 0.1281 s)

$\hat{\theta}_1 \pm$ écart-type	$\hat{\theta}_2 \pm$ écart-type	$\hat{\theta}_3 \pm$ écart-type
-0.0319 ± 0.0019	-0.1922 ± 0.0023	1.4736 ± 0.0025
0.8014 ± 0.0016	0.5915 ± 0.0016	-0.5275 ± 0.0032
-1.2867 ± 0.0032	1.1041 ± 0.0032	-0.4055 ± 0.0034
-1.1243 ± 0.0046	0.0317 ± 0.0044	0.2544 ± 0.0037

FIT	**RMSE**	**PEE**
92 %	0.4332 ± 0.0030	0.0090 ± 0.0038

TABLE 2.5 – Première expérience - Résultats et mesures statistiques avec l'algorithme proposé

Une simulation Monte-Carlo avec 100 jeux de données a été réalisée (sur des séquences de longueur $N = 1000$) avec les deux algorithmes. L'initialisation aléatoire a été utilisée pour les deux algorithmes.

Les valeurs moyennes de chaque paramètre et les écarts-type sur chaque paramètre pour un rapport signal sur bruit $SNR = 25$dB sont présentés dans les tableaux 2.4 et 2.5 pour chaque algorithme. Ces résultats illustrent la supériorité des performances de l'algorithme proposé par rapport à l'algorithme présenté dans [8] : les valeurs moyennes des estimées sont plus proches des valeurs réelles que celles obtenues avec l'algorithme présenté dans [8], l'écart-type est assez faible sur chaque paramètre par rapport aux résultats de l'algorithme [8].

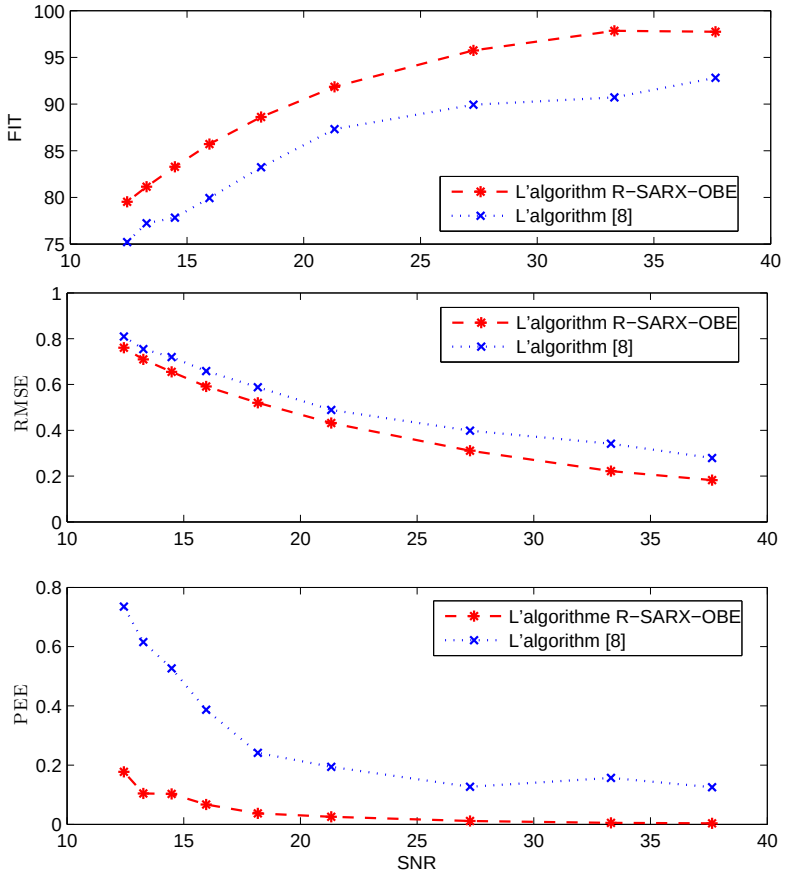

FIGURE 2.6 – Première expérience - mesure statistiques en fonction du SNR.

La figure 2.6 montre l'évolution des mesures statistiques en fonction du rapport signal sur bruit (SNR) pour les deux algorithmes. Les résultats confirment clairement la bonne robustesse de l'algorithme vis-à-vis du niveau de bruit par rapport à l'algorithme [8]. Ceci prouve l'intérêt de l'algorithme proposé.

Remarque 4. *Nous ne comparons pas avec l'approche algébrique proposée dans [85] et [188] car, il est précisé dans la littérature ([118]) que cette approche est appropriée pour l'identification des systèmes à commutations décrits par un modèle SARX dans le cas où le niveau bruit est faible ou nul.*

2.6.2 Deuxième expérience - Exécution de l'algorithme R-SARX-OBE en mode hors-ligne

L'algorithme proposé peut être mis en œuvre en mode hors-ligne. Pour cette raison, cette deuxième expérience de simulation est dédiée à la comparaison de performances de l'algorithme avec des méthodes hors ligne. La comparaison est faite avec la méthode proposée dans [148]. L'exemple numérique utilisé dans [148] est considéré. Le système est caractérisé par :

$$y_k = \begin{bmatrix} -y_{k-1} & -y_{k-2} & u_k \end{bmatrix} \theta_{\lambda_k} + e_k \tag{2.124}$$

avec :

$$\begin{cases} \theta_1^T = \begin{bmatrix} 0.2 & 0.24 & 2 \end{bmatrix} \\ \theta_2^T = \begin{bmatrix} -1.4 & -0.53 & 1 \end{bmatrix} \\ \theta_3^T = \begin{bmatrix} 1.7 & -0.72 & 0.5 \end{bmatrix} \end{cases} \tag{2.125}$$

Le nombre de sous-modèles est $s = 3$. Les données ont été obtenues avec une commutation arbitraire de l'état discret $\lambda_k \in \{1,2,3\}$. La séquence d'entrée u_k est une séquence aléatoire de moyenne nulle avec une distribution uniforme sur $[-1\ ;\ 1]$. Le terme de bruit est une séquence aléatoire de moyenne nulle uniformément répartie dans $[-\delta_e\ ;\ \delta_e]$. La borne supérieure sur le bruit δ_e a été ajustée de façon à obtenir une valeur désirée du rapport signal sur bruit (SNR).

Une simulation Monte-Carlo avec 25 jeux de données a été réalisée (sur des séquences de longueur $N = 120$) avec les deux algorithmes pour un rapport signal sur bruit $SNR = 25$dB. La valeur moyenne de chaque paramètre, l'écarts-type sur chaque paramètre et la mesure des indices statistiques sont présentés dans le tableau 2.6 et le tableau 2.7 pour chaque algorithme respectivement.

Les résultats montrent que les performances des deux algorithmes sont presque similaires avec une légère supériorité pour l'algorithme proposé. L'observation la plus intéressante concerne le temps de calcul (CPU time) qui est nettement plus faible pour l'algorithme proposé que pour l'algorithme présenté dans [148] (plus de 800s pour l'algorithme [148] et moins de 0.05s pour l'algorithme proposé).

(Temps de calcul sur 25 jeux de données : 19400s , 1 jeux de données : 807.511s)

$\hat{\theta}_1$ ± écart-type	$\hat{\theta}_2$ ± écart-type	$\hat{\theta}_3$ ± écart-type
0.2008 ± 0.058	-1.3804 ± 0.0533	1.6897 ± 0.0146
0.2420 ± 0.0064	-0.5147 ± 0.0518	-0.7083 ± 0.0166
2.0720 ± 0.0166	0.9923 ± 0.0119	0.5014 ± 0.01

FIT	**RMSE**	**PEE**
97.14 %	0.2844 ± 0.0163	0.013 ± 0.0079

TABLE 2.6 – Deuxième expérience - Résultats et mesures statistiques avec l'algorithme proposé dans [148].

(Temps de calcul sur 25 jeux de données : 0.746s , 1 jeux de données : 0.03s)

$\hat{\theta}_1$ ± écart-type	$\hat{\theta}_2$ ± écart-type	$\hat{\theta}_3$ ± écart-type
0.2001 ± 0.0125	-1.3948 ± 0.0111	1.7015 ± 0.0117
0.2416 ± 0.0085	-0.5262 ± 0.0115	-0.7119 ± 0.0155
1.9864 ± 0.0558	0.9988 ± 0.0533	0.5093 ± 0.0213

FIT	**RMSE**	**PEE**
97.35 %	0.2712 ± 0.009	0.0093± 0.0058

TABLE 2.7 – Deuxième expérience - Résultats et mesures statistiques avec l'algorithme proposé.

Remarque 5. *La simulation Monte-Carlo a été réalisée sur des séquences de longueur* $N = 120$ *et avec seulement* 25 *jeux de données parce que l'algorithme présenté dans [148] a une complexité de calcul importante. De plus, cette complexité augmente d'une manière exponentielle par rapport au nombre de sous-modèles et la dimension du vecteur de régression.*

Remarque 6. *Nous n'avons pas comparé avec l'approche algébrique proposée dans [193] et l'approche d'optimisation proposée dans [118] car, les performances de ces approches sont faibles par rapport aux performances de l'algorithme [148]. Ceci a été illustré dans les exemples numériques présentés dans [118] et [148].*

2.6.3 Troisième expérience - Exécution de l'algorithme MIMO-SARX-OBE

Nous présentons ici des résultats numériques afin d'illustrer les performances de l'algorithme MIMO-SARX-OBE à identifier les paramètres d'un modèle linéaire MIMO SARX. Les données de simulation sont générées avec un nombre d'entrées $n_u = 2$ et un nombre de sorties $n_y = 2$. Le système est caractérisé par :

$$y_k = -A_1(\lambda_k)y_{k-1} + B_0(\lambda_k)u_k + B_1(\lambda_k)u_{k-1} + e_k \tag{2.126}$$

où les matrices de paramètres sont données comme suit :

$$\begin{cases} A_1(1) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \; B_1(1) = \begin{bmatrix} 0 & 0.25 \\ 0.8 & 0 \end{bmatrix} \\ B_2(1) = \begin{bmatrix} 0.85 & 0 \\ 0 & 0 \end{bmatrix} \end{cases}$$

$$\begin{cases} A_1(2) = \begin{bmatrix} 0.4 & 0 \\ 0 & 0.6 \end{bmatrix}, B_1(2) = \begin{bmatrix} -0.2 & 0.1 \\ 1.2 & -1.2 \end{bmatrix} \\ B_2(2) = \begin{bmatrix} -0.45 & 0 \\ 0.08 & 0 \end{bmatrix} \end{cases}$$

$$\begin{cases} A_1(3) = \begin{bmatrix} -0.9 & 0 \\ 0 & -0.9 \end{bmatrix}, B_1(3) = \begin{bmatrix} 1.1 & -0.3 \\ 0.55 & -0.8 \end{bmatrix} \\ B_2(3) = \begin{bmatrix} 1 & 1.5 \\ 0 & 0 \end{bmatrix} \end{cases}$$

Le nombre de sous-modèles est $s = 3$. Les données ont été obtenues avec une commutation arbitraire de l'état discret $\lambda_k \in \{1,2,3\}$. Les deux séquences d'entrées sont deux séquences aléatoires de moyennes nulles avec une distribution uniforme sur $[-2\ ;\ 2]$. Les deux composantes e_k^1 et e_k^2 du terme de bruit e_k ont été générées de la même façon que dans [163] et [158] de la manière suivante :

$$e_k = \Delta_e \begin{bmatrix} \frac{1}{2}(v_k^1 + sin(\pi k/10)) \\ \\ \frac{1}{2}(v_k^2 + cos(\pi k/15)) \end{bmatrix} \tag{2.127}$$

où v_k^1 et v_k^2 sont des bruits blancs uniformément répartis dans $[-1\ ;\ 1]$ et la matrice Δ_e a été choisie comme suit :

$$\Delta_e = \begin{bmatrix} 0.15 & 0 \\ 0 & 0.15 \end{bmatrix}. \tag{2.128}$$

Les trois vecteurs de paramètres construits à partir des matrices de paramètres sont :

$$\theta_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0.8 \\ 0.25 \\ 0 \\ 0.85 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \theta_2 = \begin{bmatrix} 0.4 \\ 0 \\ 0 \\ 0.6 \\ -0.2 \\ 1.2 \\ 0.1 \\ -1.2 \\ -0.45 \\ 0.08 \\ 0 \\ 0 \end{bmatrix} \quad \theta_3 = \begin{bmatrix} -0.9 \\ 0 \\ 0 \\ -0.9 \\ 1.1 \\ 0.55 \\ -0.3 \\ -0.8 \\ 1 \\ 0 \\ 1.5 \\ 0 \end{bmatrix} \tag{2.129}$$

Dans un premier temps, l'algorithme MIMO-SARX-OBE a été appliqué sur une séquence de longueur $N = 1000$. La figure 2.7 illustre le comportement des deux composantes de l'erreur de prédiction a posteriori associée à l'état discret estimé en fonction du temps k. Il est clair que l'algorithme proposé réalise l'estimation de l'état discret et l'estimation du vecteur de paramètres tel que l'erreur a posteriori associée à l'état discret estimé soit bornée. De plus, le comportement de l'erreur d'estimation $\| \Theta^* - \widehat{\Theta}_k \|$ en fonction du temps est illustré sur la figure 2.8. Il apparaît que l'erreur d'estimation tend vers zéro. Ceci montre que les paramètres estimés tendent au voisinage de leur vraie valeur.

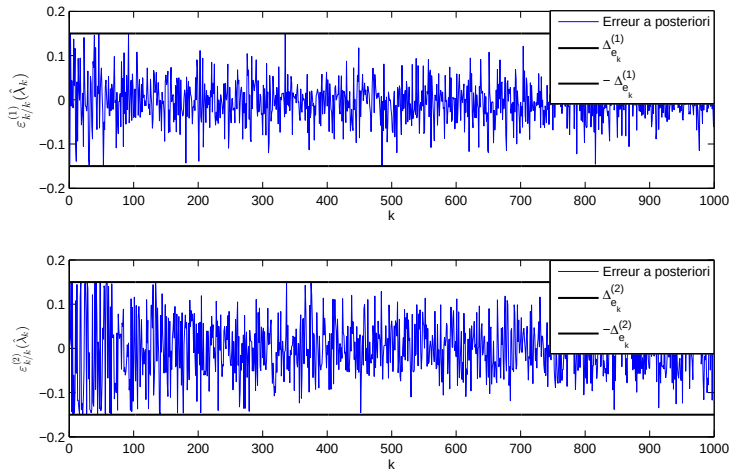

FIGURE 2.7 – Troisième expérience - $\varepsilon_{k/k}(\hat{\lambda}_k)$ en fonction du temps k.

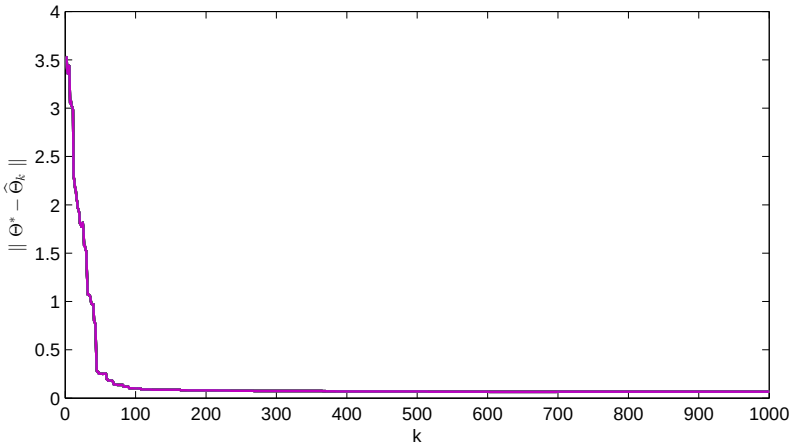

FIGURE 2.8 – Troisième expérience - $\| \Theta^* - \widehat{\Theta}_k \|$ en fonction du temps k.

(Temps de calcul sur 100 jeux de données : 54.94s , 1 jeux de données : 0.549s)

$\hat{\theta}_1$ ± écart-type	$\hat{\theta}_2$ ± écart-type	$\hat{\theta}_3$ ± écart-type
0.9949 ± 0.0045	0.4026 ± 0.0048	-0.8989 ± 0.0041
-0.0003 ± 0.0046	0.0004 ± 0.0046	0.0009 ± 0.0039
-0.0001 ± 0.0070	-0.0001 ± 0.0068	0.0005 ± 0.0045
0.9971 ± 0.0070	0.6018 ± 0.0067	-0.8973 ± 0.0051
-0.0009 ± 0.0179	-0.1995 ± 0.0181	1.0963 ± 0.0179
0.8058 ± 0.0169	1.1949 ± 0.0197	0.5504 ± 0.0189
0.2484 ± 0.0163	0.1037 ± 0.0138	-0.2985 ± 0.0196
-0.0121 ± 0.0144	-1.1837 ± 0.0150	-0.7958 ± 0.0197
0.8395 ± 0.0150	-0.4377 ± 0.0138	0.9947 ± 0.0229
0.0005 ± 0.0148	0.0810 ± 0.0149	0.0014 ± 0.0211
0.0023 ± 0.0175	0.0028 ± 0.0165	1.4919 ± 0.0189
-0.0012 ± 0.0144	-0.0017 ± 0.0185	0.0007 ± 0.0150

RMSE	**PEE**
0.3522 ±0.0097	0.0254 ± 0.0030

TABLE 2.8 – Troisième expérience - MIMO SARX - Résultats et mesures statistiques.

Dans un second temps, une simulation Monte-Carlo avec 100 jeux de données a été réalisée sur des séquences de longueur $N = 1000$. Les valeurs moyennes des paramètres, les écarts-types sur les paramètres et la mesure des indices statistiques sont présentés dans le tableau 2.8.

Les résultats montrent que l'algorithme proposé peut être utilisé avec succès pour l'identification des paramètres du modèle MIMO SARX. Les valeurs moyennes des estimations sont proches des valeurs réelles, l'écart-type est assez faible et la complexité de calcul est très faible.

2.7 Conclusion

Dans ce chapitre, nous avons proposé un algorithme d'identification en temps réel pour les systèmes linéaires à commutations soumis à un bruit borné. L'algorithme se distingue par une alternance entre l'estimation de l'état discret et la mise à jour du vecteur de paramètres associé à l'état discret estimé. Les propriétés de convergence et de stabilité ont été établies. Le comportement de l'algorithme dépend de l'estimation de l'état discret et de son initialisation. Une attention particulière a été accordée à l'impact d'une mauvaise décision sur l'estimation de l'état discret et la sensibilité à l'initialisation qui diminue au cours de temps. Un des avantages de la méthode proposée est sa faible complexité de calcul. Notons que l'utilisation d'une version en mode batch de l'algorithme via un schéma itératif a été également recommandée. Nous avons montré l'efficacité et l'intérêt de l'algorithme par des résultats de simulation qui ont montré que la version en-ligne et la version batch sont compétitives par rapport aux solutions disponibles dans la littérature. Une extension pour l'identification des systèmes MIMO à commutations a également été présentée. D'autres extensions de l'algorithme sont possibles. Un algorithme pour l'identification des systèmes affine par morceaux est notamment présenté dans le prochain chapitre.

Chapitre 3

Algorithme d'identification des systèmes affines par morceaux décrits par un modèle PWARX

Dans ce chapitre, nous considérons le problème d'identification des systèmes affines par morceaux décrits par un modèle PWARX en présence d'un bruit borné. La résolution de ce problème passe par l'estimation du nombre de sous-modèles, l'association de chaque donnée au sous-modèle le plus approprié, l'estimation de tous les vecteurs de paramètres des sous-modèles et l'estimation des paramètres des hyperplans qui forment une partition polyédrique de l'espace de régression. L'algorithme proposé dans ce chapitre procède en deux étapes. Dans la première étape, il associe chaque point de données au sous-modèle le plus approprié et réalise l'identification des paramètres de chaque sous-modèle. Cette étape est une extension de l'algorithme R-SARX-OBE présenté dans le chapitre précédent. Dans la deuxième étape, l'objectif est d'estimer les vecteurs de paramètres définissant la partition polyédrique. Un exemple numérique est donné à la fin de ce chapitre afin d'illustrer les performances de l'algorithme.

Sommaire

3.1 Formulation du problème d'identification

Dans cette section, nous formulons le problème d'identification des systèmes affines par morceaux décrits par un modèle PWARX en présence d'un bruit borné. Considérons un système discret affine par morceaux décrit par un modèle PWARX comme suit :

$$y_k = \begin{cases} \bar{\phi}_k^T \theta_1 + e_k & \text{si} \quad \phi_k \in \mathscr{X}_1 \\ \bar{\phi}_k^T \theta_2 + e_k & \text{si} \quad \phi_k \in \mathscr{X}_2 \\ \quad \vdots & \\ \bar{\phi}_k^T \theta_s + e_k & \text{si} \quad \phi_k \in \mathscr{X}_s \end{cases} \tag{3.1}$$

- s est le nombre de sous-modèles.
- $\{\theta_1, \cdots, \theta_s\}$ sont les vecteurs de paramètres des différents sous-modèles.
- $y_k \in \mathbb{R}$ est la sortie du système.
- $\bar{\phi}_k$ est le vecteur de régression étendu défini par $\bar{\phi}_k = [\phi_k^T \quad 1]^T$, où $\phi_k \in \mathbb{R}^n$ est le vecteur de régression donné par :

$$\phi_k = [-y_{k-1} \ \cdots \ -y_{k-n_a} \ \ u_k \ \ u_{k-1} \ \cdots \ u_{k-n_b}]^T \tag{3.2}$$

- n_a et n_b sont les ordres du système et $n = n_a + n_b + 1$.
- $u_k \in \mathbb{R}$ est l'entrée du système.
- $e_k \in \mathbb{R}$ est le terme de bruit. Une borne supérieure δ_e sur son amplitude est supposée connue :

$$| e_k | \leq \delta_e \ , \quad \forall \ k \tag{3.3}$$

- Les régions $\{\mathscr{X}_i\}_{i=1}^s$ forment une partition polyédrique du domaine borné et fermé $\mathscr{X} \in \mathbb{R}^n$, c'est à dire $\cup_{i=1}^s \ \mathscr{X}_i = \mathscr{X}$ et $\mathscr{X}_i \cap \mathscr{X}_j = \emptyset$ si $i \neq j$.

À chaque instant k, un seul sous-modèle est actif et sa sortie correspond à la sortie du système y_k. Donc, à chaque instant k il existe au moins un vecteur de paramètres θ_i tel que : $| y_k - \bar{\phi}_k^T \theta_i | \leq \delta_e$. Ainsi, le problème d'identification d'un système affine par morceaux décrit par un modèle PWARX en présence d'un bruit borné est le suivant :

Problème d'identification : *Étant donné un ensemble d'observations* $\{\bar{\phi}_k, y_k\}_{k=1}^N$ *générées par un système de la forme (3.1), (3.2) et (3.3), l'objectif est d'estimer le nombre*

de sous-modèles s, les vecteurs de paramètres $\{\theta_i\}_{i=1}^{s}$ *et les régions* $\{\mathscr{X}_i\}_{i=1}^{s}$. *Les vecteurs de paramètres estimés* $\{\widehat{\theta}_i\}_{i=1}^{s}$ *doivent satisfaire :*

$$| y_k - \bar{\phi}_k^T \widehat{\theta}_{\hat{i}} | \leq \delta_e \ , \quad \forall \ k \tag{3.4}$$

où $\hat{i}$ *désigne le sous-modèle actif estimé à l'instant* k *et* $\widehat{\theta}_i$ *est le vecteur de paramètres estimés associé à* $\hat{i}$.

Concernant l'estimation de la partition polyédrique, il est nécessaire de classer d'abord les points de données disponibles en groupes $\{D_i\}_{i=1}^{s}$ tel que $(y_k, \bar{\phi}_k) \in D_i$ si et seulement si $(y_k, \bar{\phi}_k)$ est attribué au $i^{\text{ième}}$ sous-modèle.

Maintenant, nous procédons à une reformulation du problème d'identification comme nous l'avons fait avec les systèmes linéaires à commutations dans le chapitre 2. Pour ceci, le sous-modèle actif à chaque instant k est désigné par λ_k avec $\lambda_k \in \{1, \cdots, s\}$. Si $\lambda_k = i$ à l'instant k, la sortie du système est écrite en fonction du vecteur de paramètres θ_i :

$$y_k = \bar{\phi}_k^T \theta_i + e_k \tag{3.5}$$

On peut exprimer la sortie du système y_k à l'instant k en fonction des autres vecteurs de paramètres comme suit :

$$\begin{cases} y_k = \bar{\phi}_k^T \theta_1 + e_k + \bar{\phi}_k^T (\theta_i - \theta_1) \\ y_k = \bar{\phi}_k^T \theta_2 + e_k + \bar{\phi}_k^T (\theta_i - \theta_2) \\ \qquad \vdots \\ y_k = \bar{\phi}_k^T \theta_s + e_k + \bar{\phi}_k^T (\theta_i - \theta_s) \end{cases} \tag{3.6}$$

L'équation (3.6) peut être réécrite sous une forme matricielle comme suit :

$$\begin{bmatrix} y_k \\ \vdots \\ y_k \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} \bar{\phi}_k^T \theta_1 \\ \vdots \\ \bar{\phi}_k^T \theta_i \\ \vdots \\ \bar{\phi}_k^T \theta_s \end{bmatrix} + \begin{bmatrix} e_k + \bar{\phi}_k^T (\theta_i - \theta_1) \\ \vdots \\ e_k \\ \vdots \\ e_k + \bar{\phi}_k^T (\theta_i - \theta_s) \end{bmatrix} \tag{3.7}$$

À ce stade, et comme nous l'avons fait avec les systèmes linéaires à commutations, la forme matricielle donnée par (3.7) peut s'écrire de la manière suivante :

$$Y_k = \varphi_k^T \Theta^* + V_k \tag{3.8}$$

avec

$$V_k = \begin{bmatrix} e_k + \bar{\phi}_k^T(\theta_i - \theta_1) \\ \vdots \\ e_k \\ \vdots \\ e_k + \bar{\phi}_k^T(\theta_i - \theta_s) \end{bmatrix} \qquad Y_k = \begin{bmatrix} y_k \\ \vdots \\ y_k \end{bmatrix} \qquad \Theta^* = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_s \end{bmatrix} \tag{3.9}$$

où $Y_k \in \mathbb{R}^s$, $\Theta^* \in \mathbb{R}^{(n+1)s}$, $V_k \in \mathbb{R}^s$ et $\varphi_k = I_s \otimes \bar{\phi}_k$ avec $\otimes$ désigne le produit de Kronecker. Compte tenu de ces formulations, le problème d'identification considéré est redéfini comme suit :

Problème d'identification 2 : *Étant donné un ensemble d'observations* $\{\varphi_k, Y_k\}_{k=1}^N$, *estimer le nombre de sous-modèles* s, *le vecteur global de paramètres* Θ^* *et les régions* $\{\mathscr{X}_i\}_{i=1}^s$. *Le vecteur global de paramètres estimés* $\widehat{\Theta}$ *doit satisfaire*

$$| \varepsilon_k(\hat{\lambda}_k) | \leq \delta_e \ , \quad \forall \ k \tag{3.10}$$

où $\varepsilon_k(\hat{\lambda}_k)$ *désigne la* $\hat{\lambda}_k^{ième}$ *composante de* $\varepsilon_k = Y_k - \varphi_k^T \widehat{\Theta}$ *et* $\hat{\lambda}_k$ *désigne l'estimation du sous-modèle actif.*

3.2 Algorithme d'identification des systèmes affines par morceaux

L'algorithme proposé dans ce chapitre pour l'identification des systèmes affines par morceaux consiste en deux étapes indépendantes. La première étape de l'algorithme permet d'associer chaque donnée au sous-modèle le plus approprié, réaliser l'estimation du vecteur global de paramètres et de classifier les données dans des groupes. La deuxième étape de l'algorithme permet d'estimer les régions de la partition polyédrique en utilisant les SVM.

3.2.1 Étape 1 : Estimation du sous-modèle actif et le vecteur global de paramètres

Cette première étape de l'algorithme est similaire à l'algorithme d'identification des systèmes linéaires à commutations présenté dans le chapitre 2. En suivant la même procédure que dans le chapitre 2, nous supposons que le nombre de sous-modèles s est

connu et nous concentrons sur l'estimation du sous-modèle actif à chaque instant k et sur l'identification du vecteur global de paramètres. $\widehat{\Theta}_k$ désigne une estimation de Θ^* à l'instant k. Nous considérons la définition suivante de l'erreur de prédiction a priori et l'erreur de prédiction a posteriori :

Définition 8. *L'erreur de prédiction a priori $\varepsilon_{k/k-1}$ et l'erreur de prédiction a posteriori $\varepsilon_{k/k}$ sont définies par :*

$$\begin{cases} \varepsilon_{k/k-1} = Y_k - \varphi_k^T \widehat{\Theta}_{k-1} = \begin{bmatrix} y_k - \bar{\phi}_k^T \hat{\theta}_{1/k-1} \\ \vdots \\ y_k - \bar{\phi}_k^T \hat{\theta}_{s/k-1} \end{bmatrix} \\ \varepsilon_{k/k} = Y_k - \varphi_k^T \widehat{\Theta}_k = \begin{bmatrix} y_k - \bar{\phi}_k^T \hat{\theta}_{1/k} \\ \vdots \\ y_k - \bar{\phi}_k^T \hat{\theta}_{s/k} \end{bmatrix} \end{cases} \tag{3.11}$$

où pour chaque $i \in \{1, \cdots, s\}$, $\widehat{\theta}_{i/k}$ désigne l'estimation de θ_i à l'instant k.

La première étape de l'algorithme est décrite dans le tableau 3.1. Pour chaque paire de données $\{\varphi_k, Y_k\}_{k=1}^N$ cette étape de l'algorithme réalise :

Estimation du sous-modèle actif :
L'estimation du sous-modèle actif à chaque instant k peut être interprété comme le sous-modèle qui correspond à l'indice i de la plus petite composante de $\mid \varepsilon_{k/k-1}(i) \mid$ avec $i \in \{1, \cdots, s\}$.

Estimation du vecteur global de paramètres $\widehat{\Theta}_k$:
Après l'estimation du sous-modèle actif, l'objectif est de mettre à jour dans le vecteur global de paramètres $\widehat{\Theta}_k$ seulement le vecteur de paramètre $\theta_{\hat{\lambda}_k}$ qui correspond au sous-modèle estimé comme étant actif à l'instant k (le sous-modèle $\hat{\lambda}_k$).

A chaque instant k, le rapport $\zeta_k = \dfrac{\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid}{\delta_e}$ est comparé à 1 et deux cas se produisent :

- Si $\zeta_k \leq 1$, alors l'adaptation à cet instant k est gelée, soit $\widehat{\Theta}_k = \widehat{\Theta}_{k-1}$. Ceci est réalisé grâce à la matrice $\sigma_k = 0_{s \times s}$ qui garantie que $\Gamma_k = 0_{ns \times s}$.
- Si $\zeta_k > 1$, alors dans le vecteur global de paramètres $\widehat{\Theta}_k$, uniquement le vecteur de paramètres $\widehat{\theta}_{\hat{\lambda}_k}$ associé au sous-modèle actif estimé $\hat{\lambda}_k$ est mis à jour. Ceci est réalisé grâce à la structure particulière de la matrice symétrique $\sigma_k \in \mathbb{R}^{s \text{ x } s}$.

Algorithme d'identification - Étape 1.

Entrées $\{\varphi_k, Y_k\}_{k=1}^{N}, s, \Theta_0,\ P_0, \delta_e, \{D_i\}_{i=1}^{s} = \emptyset$

Pour $k = max(n_a, n_b) + 1$ *à* N *fait*

- Estimation du sous-modèle actif :

$$\begin{cases} \varepsilon_{k/k-1} = Y_k - \varphi_k^T \widehat{\Theta}_{k-1} \\ \hat{\lambda}_k = \underset{i=1,\cdots,s}{\text{argmin}} \mid \varepsilon_{k/k-1}(i) \mid \\ \zeta_k = \underset{i=1,\cdots,s}{\min} \dfrac{\mid \varepsilon_{k/k-1}(i) \mid}{\delta_e} = \dfrac{\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid}{\delta_e} \end{cases} \tag{3.12}$$

- mise à jour du vecteur global de paramètres estimé $\widehat{\Theta}_k$:

$$\widehat{\Theta}_k = \widehat{\Theta}_{k-1} + \Gamma_k \varepsilon_{k/k-1} \tag{3.13}$$

avec

$$\begin{cases} \Gamma_k = P_{k-1}\varphi_k\sigma_k(I_s + \varphi_k^T P_{k-1}\varphi_k\sigma_k)^{-1} \\ P_k = (I_{(n+1)s} - \Gamma_k\varphi_k^T)P_{k-1} \end{cases} \tag{3.14}$$

où

$$\sigma_k \quad = \quad \begin{cases} (\varphi_k^T P_{k-1}\varphi_k)^{-1}(C_k - I_s) & \text{si} \quad \varphi_k^T P_{k-1}\varphi_k > 0 \quad \text{et} \quad \zeta_k > 1 \\ 0_{s\times s} \quad \text{Autrement} \end{cases} \tag{3.15}$$

et

$$C_k = \begin{bmatrix} 1 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \zeta_k & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \end{matrix} \hat{\lambda}_k^{\text{ième}} \tag{3.16}$$

$$\hat{\lambda}_k^{\text{ième}}$$

- Classification de données :

$$(y_k, \bar{\phi}_k) \in D_{\hat{\lambda}_k} \tag{3.17}$$

Fin pour

Sorties $\widehat{\Theta}_k, \{D_i\}_{i=1}^{s}$

TABLE 3.1 – Étape 1 : Estimation du sous-modèle actif, $\widehat{\Theta}_k$ et classification de données

Classification de données :

La classification des données dans des groupes est une tâche essentielle pour estimer la partition polyédrique par la suite. Les groupes de données $\{D_i\}_{i=1}^{s}$ sont formés en associant chaque donnée au groupe le plus approprié $(y_k, \bar{\phi}_k) \in D_i$ avec i correspond à l'indice de la plus petite composante $\mid \varepsilon_{k/k-1}(i) \mid$ avec $i \in \{1, \cdots, s\}$. Afin d'avoir une classification fiable de données, il est recommandé d'itérer l'algorithme décrit dans le tableau 3.1 plusieurs fois, ceci afin d'utiliser les vecteurs de paramètres estimés après leurs convergence vers les vrais valeurs des paramètres.

Remarque 7. *Cette première étape est similaire à l'algorithme R-SARX-OBE présenté dans le chapitre 2. Par conséquent, tous les théorèmes, les remarques et les commentaires décrits dans le chapitre 2 sur le fonctionnement, le comportement, l'excitation persistante et la convergence de l'algorithme R-SARX-OBE, restent toujours valides pour la première étape de l'algorithme présenté ici.*

3.2.2 Étape 2 : Estimation de la partition polyédrique en utilisant les SVM

La première étape de l'algorithme décrite dans le tableau 3.1 fournit la classification des données dans les groupes $\{D_i\}_{i=1}^{s}$. Il ne reste plus qu'à identifier les régions $\{\mathscr{X}_i\}_{i=1}^{s}$ définissant la partition polyédrique tel que $\phi_k \in \mathscr{X}_i$ si $(y_k, \bar{\phi}_k) \in D_i$. Ceci est équivalent à trouver les frontières qui délimitent chaque région et donc séparer les s régions par des séparateurs linéaires (hyperplans). Ce problème est largement discuté dans la littérature et la plupart des méthodes utilisent les SVMs ([13], [20], [146], [142], [117] et [12]). De notre côté, nous utilisons aussi les SVMs pour l'estimation de la partition polyédrique. En général, les SVMs sont utilisés pour le problème de classification binaire ([184], [183]). Dans le cas de plusieurs régions, nous pouvons former un classificateur linéaire pour chaque paire de régions ou utiliser Multi-classes SVMs (M-SVMs) pour séparer plusieurs régions de manière directe ([120], [13], [151]). Nous présentons le principe des SVMs dans le chapitre 5.

3.3 Stratégie pour estimer le nombre de sous-modèles

Dans cette section, nous présentons une stratégie pour estimer le nombre de sous-modèles. En effet, dans la première étape de l'algorithme, le nombre de sous-modèles s

a été supposé connu. Une approche itérative est présentée ici pour estimer le nombre de sous-modèles s s'il est inconnu. Pour ceci, nous utilisons l'indice "FIT" défini dans le chapitre 2. Dans un premier temps, nous choisissons $s = 1$ et $FIT_0 = 0$. Une stratégie pour déterminer le nombre de sous-modèles est d'incrémenter s et calculer le FIT correspondant pour chaque valeur de s. Le FIT augmentera en incrémentant s jusqu'à une certain valeur de s où le FIT se stabilise autour d'une valeur fixe. Le nombre le plus approprié de sous-modèles s est la première valeur de s où la variation du FIT devient insignifiante. Cette approche itérative est illustrée dans le tableau 3.2.

Remarque 8. *Une autre stratégie pour estimer le nombre de sous-modèles consiste à supposer la connaissance d'un majorant $\bar{s}$ sur le nombre de sous-modèles s tel que $\bar{s} \geq s$ et de lancer l'algorithme dans un premier temps avec $\bar{s}$ comme nombre de sous-modèles. Après la convergence, les données seront associées principalement aux s vrais sous-modèles, tandis que le nombre de points associés aux $\bar{s} - s$ sous-modèles sera très faible ou même nulle. Ceci permet d'estimer le vrai nombre de sous-modèles et éliminer les sous-modèles inutiles. Il est recommandé d'appliquer ou de combiner les deux stratégies afin de valider le vrai nombre de sous-modèles.*

Remarque 9. *Les deux stratégies peuvent être aussi appliquées pour estimer le nombre de sous-modèles dans le cas des systèmes linéaires à commutations que nous avions présenté dans le chapitre 2.*

Algorithme - Estimation du nombre de sous-modèles
– **1- Initialisation :** $s = 1$ $FIT_0 = 0$
– **2- Exécuter l'algorithme décrit dans le tableau 3.1 en utilisant** s **comme nombre de sous-modèles.**
– **3- Calculer le** FIT **correspondant à** s
– **4- Si** $(\mid FIT_s - FIT_{s-1} \mid \leq \varepsilon)$
retourner s
sinon
$s = s + 1$;
aller à 2 ;
Fin si

TABLE 3.2 – Une approche itérative pour estimer le nombre de sous-modèles

3.4 Exemple numérique

Nous illustrons dans cette section l'efficacité et les performances de l'algorithme proposé. Nous considérons le modèle PWARX caractérisé par :

$$y_k = \begin{cases} 0.6y_{k-1} + 0.4u_k + 0.2 + e_k \text{ si } \; y_{k-1} + 0.2u_k \geq 0 \\ \qquad\qquad\qquad\qquad\qquad \text{et } \; 0.2y_{k-1} + 0.5u_k > 0 \\ -0.8y_{k-1} + 0.6u_k - 0.5 + e_k \text{ si } \; y_{k-1} + 0.2u_k < 0 \\ \qquad\qquad\qquad\qquad\qquad \text{et } \; y_{k-1} - 0.3u_k < 0 \\ 0.4y_{k-1} - 0.35u_k - 0.25 + e_k \text{si } 0.2y_{k-1} + 0.5u_k \leq 0 \\ \qquad\qquad\qquad\qquad\qquad \text{et } \; y_{k-1} - 0.3u_k \geq 0 \end{cases}$$

Le nombre de sous-modèles est $s = 3$. La séquence d'entrée u_k de longueur $N = 2000$ est une séquence aléatoire de moyenne nulle avec une distribution uniforme sur $[-1 \; ; \; 1]$. Le terme de bruit e_k est une séquence aléatoire de moyenne nulle uniformément répartie dans $[-\delta_e \; ; \; \delta_e]$, δ_e a été ajusté de façon à obtenir une valeur souhaitée du rapport signal sur bruit (SNR). La partition des points de données pour un rapport signal sur bruit $\text{SNR} = 25dB$ est présentée sur la figure 3.1. L'objectif ici est d'estimer le nombre de sous-modèles, les vecteurs de paramètres et la partition polyédrique.

Estimer le nombre de sous-modèles

Afin d'estimer le nombre de sous-modèles, l'algorithme décrit dans le tableau 3.2 a été appliqué. Le résultat est présenté sur la figure 3.2 pour différents valeurs de SNR. Il est clair que le nombre de sous-modèles est $s = 3$, ce qui correspond à la valeur réelle du nombre de sous-modèles. Nous conclons que la stratégie proposée permet d'estimer le nombre de sous-modèles même avec un faible SNR.

Estimer les vecteurs de paramètres

Une simulation Monte-Carlo avec 100 jeux de données a été réalisée sur des séquences de longueur $N = 2000$ en appliquant l'algorithme décrit dans le tableau 2.1. L'initialisation aléatoire a été utilisée et l'algorithme a été itéré 3 fois afin d'améliorer les résultats. Les valeurs moyennes des paramètres estimés et les écarts types sur les paramètres estimés ($SNR = 25$dB) sont rapportés dans le tableau 3.3.

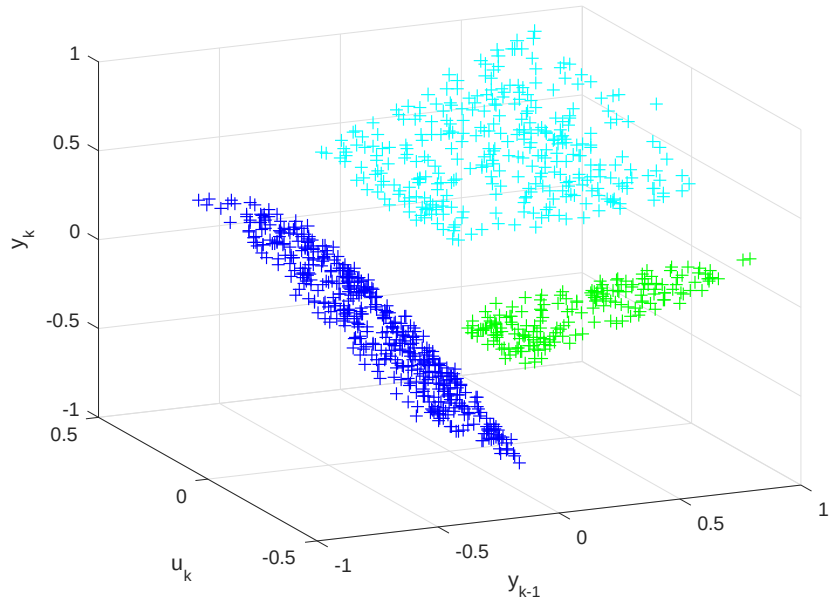

FIGURE 3.1 – Partition des points de données disponibles.

Ces résultats montrent l'efficacité et les bonnes performances de l'algorithme proposé : les valeurs moyennes estimées sont proches des valeurs réelles et l'écart-type est très faible sur chaque paramètre estimé. De plus, une observation très intéressante est que le temps de calcul nécessaire pour l'exécution de l'algorithme est très faible, la solution est obtenue après seulement 0.1272s.

On se propose maintenant d'évaluer l'influence du niveau de bruit sur le comportement de l'algorithme. Pour ceci, nous réalisons différentes simulations avec différents rapports signal sur bruit SNR en ajustant la valeur de δ_e. La variation du FIT en fonction du SNR est représentée sur la figure 3.3 pour les différentes valeurs du SNR. Les résultats montrent la robustesse de l'algorithme vis-à-vis du niveau de bruit.

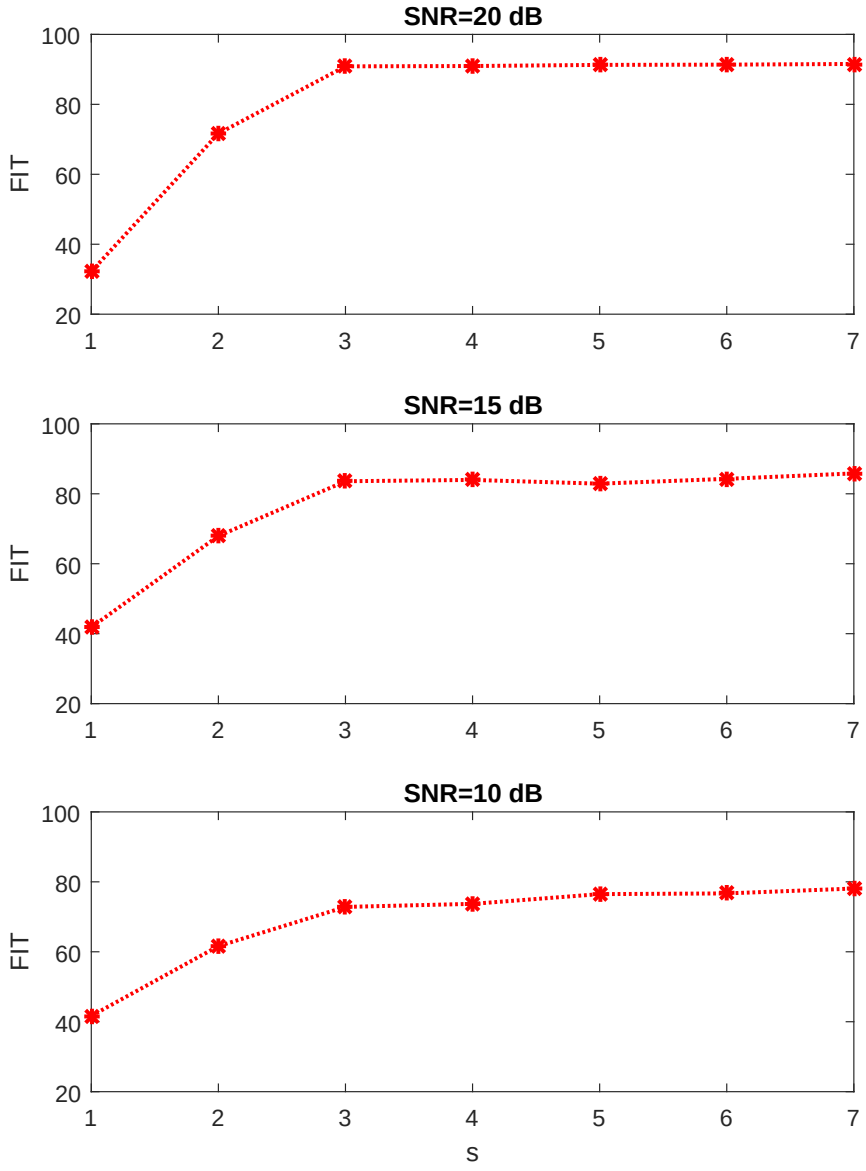

FIGURE 3.2 – Le FIT en fonction du nombre de sous-modèles s pour différents valeurs de SNR.

(Temps de calcul sur 100 jeux de données : 12.72s , 1 jeux de données : 0.1272s)

$\widehat{\theta}_1 \pm$ écart-type	$\widehat{\theta}_2 \pm$ écart-type	$\widehat{\theta}_3 \pm$ écart-type
0.5997 ± 0.0009	-0.7956 ± 0.0058	0.3998 ± 0.0012
0.4000 ± 0.0008	0.5994 ± 0.0024	-0.3494 ± 0.0014
0.2002 ± 0.0006	-0.4987 ± 0.0020	-0.2496 ± 0.0011

TABLE 3.3 – Résultats pour un SNR=25dB

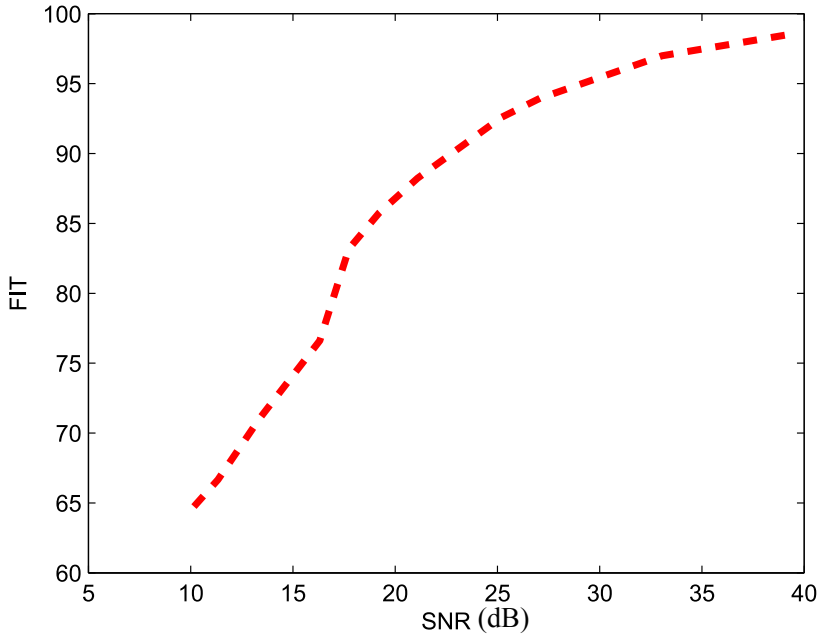

FIGURE 3.3 – FIT en fonction du SNR.

Estimer la partition polyédrique

Comme la plupart des autres méthodes qu'on trouve dans la littérature, la partition polyédrique est estimée en utilisant les SVM. Une simulation Monte-Carlo avec 100 jeux de données a été réalisée. Les valeurs moyennes estimées de paramètres des hyperplans qui forment la partition polyédrique pour un SNR= $25dB$ sont données dans le tableau 3.4. Il est clair que la partition estimée est très proche de la vraie partition.

$h_1 \pm$ écart-type	$h_2 \pm$ écart-type	$h_3 \pm$ écart-type
1.0085 ± 0.0662	0.1921 ± 0.0156	1.0019 ± 0.0672
0.2252 ± 0.0251	0.5079 ± 0.0124	-0.3019 ± 0.0228
0	0	0

TABLE 3.4 – Estimation de la partition polyédrique

3.5 Conclusion

Dans ce chapitre, nous avons présenté un algorithme pour l'identification des systèmes affines par morceaux en présence d'un bruit borné. L'algorithme consiste en deux étapes séparées : la première étape est inspirée de l'algorithme R-SARX-OBE présenté dans le chapitre précédent où chaque point de données est associé au sous-modèle le plus approprié et l'identification des paramètres de tous les sous-modèles est réalisée, la deuxième étape réalise l'estimation des vecteurs de paramètres définissant la partition polyédrique. Les résultats de simulation montrent l'efficacité et l'intérêt de l'algorithme présenté. D'autres extensions de l'algorithme R-SARX-OBE sont encore possibles, à savoir, un algorithme pour l'identification des systèmes à commutations décrits par un modèle du type erreur de sortie est présenté dans le prochain chapitre.

Chapitre 4

Algorithme d'identification des systèmes à commutations décrits par un modèle SOE

Dans ce chapitre, nous considérons le problème d'identification à partir des données d'entrée/sortie d'un système linéaire à commutations décrit par un modèle du type erreur de sortie à commutations (Switched Output Error "SOE") en présence d'un bruit borné. L'algorithme proposé est issu d'une adaptation de l'algorithme R-SARX-OBE proposé dans le chapitre 2. On établit les propriétés de stabilité et de convergence de l'algorithme proposé. Des résultats de simulation sont donnés à la fin du chapitre pour illustrer les performances de l'algorithme.

Sommaire

4.1 Formulation du problème d'identification

Dans cette section, nous reformulons le problème d'identification des systèmes linéaires à commutations décrits par un modèle SOE en présence de bruit borné. Les ordres des différents sous-modèles sont supposés fixes, égaux et connus a priori ($n_a(1) = \cdots = n_a(s) = n_a$ et $n_b(1) = \cdots = n_b(s) = n_b$). Considérons un système linéaire discret SISO à commutations décrit par un modèle SOE comme suit :

$$y_k = \frac{B_{\lambda_k}(q)}{A_{\lambda_k}(q)} u_k + e_k \tag{4.1}$$

- $u_k \in \mathbb{R}$ et $y_k \in \mathbb{R}$ sont respectivement l'entrée et la sortie du système.
- $e_k \in \mathbb{R}$ est le terme inconnu de bruit. Une borne supérieure δ_e sur son amplitude est supposée connue :

$$| e_k | \leq \delta_e \ , \quad \forall \ k \tag{4.2}$$

- $\lambda_k \in \{1, \cdots, s\}$ est l'état discret qui indique le sous-modèle actif à l'instant k, s est le nombre total de sous-modèles.
- $A_{\lambda_k}(q)$, $B_{\lambda_k}(q)$ sont les polynômes de paramètres inconnus du sous-modèle λ_k

$$\begin{cases} A_{\lambda_k}(q) = 1 + a_1(\lambda_k) q^{-1} + \cdots + a_{n_a}(\lambda_k) q^{-n_a} \\ B_{\lambda_k}(q) = b_0(\lambda_k) + b_1(\lambda_k) q^{-1} + \cdots + b_{n_b}(\lambda_k) q^{-n_b} \end{cases} \tag{4.3}$$

- n_a et n_b sont les ordres du système, q^{-1} désigne l'opérateur de retard.

La sortie prédicteur est définie en tenant compte de la structure de prédicteur optimal ([126]), l'équation du système (4.1) implique ([198] ,[211]) :

$$y_k = \hat{y}_k + e_k = \frac{B_{\lambda_k}(q)}{A_{\lambda_k}(q)} u_k + e_k \tag{4.4}$$

$\hat{y}_k$ est la partie sans bruit (non-mesurable) de la sortie. Les vecteurs de paramètres à identifier du modèle erreur de sortie à commutations sont les suivants :

$$\theta_{\lambda_k} = \begin{bmatrix} a_1(\lambda_k) & \cdots & a_{n_a}(\lambda_k) & b_0(\lambda_k) & b_1(\lambda_k) & \cdots & b_{\lambda_k}^{(n_b)} \end{bmatrix} \tag{4.5}$$

En utilisant cette définition des vecteurs de paramètres et à partir de la définition du vecteur de régression $\phi_k \in \mathbb{R}^n$ comme :

$$\phi_k^T = \begin{bmatrix} -\hat{y}_{k-1} & \cdots & -\hat{y}_{k-n_a} & u_k & u_{k-1} & \cdots & u_{k-n_b} \end{bmatrix} \tag{4.6}$$

avec $n = n_a + n_b + 1$ est le nombre de paramètres de chaque sous-modèle, $\hat{y}_k$ peut être écrit comme suit :

$$\hat{y}_k = \phi_k^T \theta_{\lambda_k} = y_k - e_k \tag{4.7}$$

Ainsi :

$$y_k = \phi_k^T \theta_{\lambda_k} + e_k \tag{4.8}$$

Nous notons que les prédicteurs optimaux $\hat{y}_{k-i}$ introduits dans le vecteur de régression ϕ_k sont inconnus et non mesurables. Dans l'algorithme d'identification, ces prédicteurs seront remplacés par leurs estimés. De plus, à chaque instant k, au moins un vecteur de paramètres θ_i est tel que $| y_k - \phi_k^T \theta_i | \leq \delta_e$. Ce θ_i correspond au sous-modèle actif à l'instant k $(i = \lambda_k)$.

Si à l'instant k la sortie du système a été générée par le sous-modèle "i", c'est à dire $\lambda_k = i$, alors :

$$y_k = \phi_k^T \theta_i + e_k$$

En suivant la même procédure que dans le chapitre 2, la forme compacte de l'équation (4.1) est la suivante :

$$Y_k = \varphi_k^T \Theta^* + V_k \tag{4.9}$$

avec $\varphi_k = I_s \otimes \phi_k$, $V_k \in \mathbb{R}^s$ est le vecteur de bruit étendu, $\Theta^* \in \mathbb{R}^{ns \times 1}$ est le vecteur global de paramètres et $Y_k \in \mathbb{R}^{s \times 1}$ est le vecteur de sortie étendu. Cette écriture compacte sera utile pour l'étude de la convergence de l'algorithme.

Problème d'identification : *Étant donné un ensemble d'observations* $\{u_k, Y_k\}_{k=1}^N$, *le nombre de sous-modèles* s *et leurs ordres* n_a *et* n_b. *L'objectif est d'estimer l'état discret* $\{\lambda_k\}_{k=1}^N$ *et le vecteur global de paramètres* Θ^*. *Ces estimations* $\hat{\lambda}_k$ *et* $\widehat{\Theta}$ *doivent satisfaire :*

$$| \varepsilon_k(\hat{\lambda}_k) | \leq \delta_e \ , \quad \forall \ k \tag{4.10}$$

où $\varepsilon_k(\hat{\lambda}_k)$ *désigne la* $\hat{\lambda}_k^{ième}$ *composante de* $\varepsilon_k = Y_k - \varphi_k^T \widehat{\Theta}$.

4.2 L'algorithme R-SOE-OBE

Avant de présenter l'algorithme d'identification, nous introduisons quelques définitions afin de faciliter la compréhension de l'algorithme, en l'occurrence le prédicteur a priori et

le prédicteur a posteriori sont respectivement donnés comme suit :

$$\begin{cases} \hat{Y}_{k/k-1} = \widehat{\varphi}_k^T \widehat{\Theta}_{k-1} = \begin{bmatrix} \hat{\phi}_k^T \widehat{\theta}_{1/k-1} \\ \vdots \\ \hat{\phi}_k^T \widehat{\theta}_{s/k-1} \end{bmatrix} \\ \hat{Y}_{k/k} = \widehat{\varphi}_k^T \widehat{\Theta}_k = \begin{bmatrix} \hat{\phi}_k^T \widehat{\theta}_{1/k} \\ \vdots \\ \hat{\phi}_k^T \widehat{\theta}_{s/k} \end{bmatrix} \end{cases}$$

avec $\widehat{\varphi}_k$ est une estimation de la matrice φ_k obtenu en remplaçant la partie inconnue non-mesurable $\hat{y}_{k-i}$ par leurs prédicteurs a posteriori $\hat{Y}_{k-i/k-i}(\hat{\lambda}_{k-i})$:

$$\widehat{\varphi}_k = I_s \otimes \hat{\phi}_k$$

avec

$$\hat{\phi}_k^T = \begin{bmatrix} -\hat{Y}_{k-1/k-1}(\hat{\lambda}_{k-1}) & \cdots & -\hat{Y}_{k-n_a/k-n_a}(\hat{\lambda}_{k-n_a}) & u_{k-1} & \cdots & u_{k-n_b} \end{bmatrix}$$

L'erreur de prédiction a priori $\varepsilon_{k/k-1}$ et l'erreur de prédiction a posteriori $\varepsilon_{k/k}$ sont définies de la manière suivante :

$$\begin{cases} \varepsilon_{k/k-1} = Y_k - \hat{Y}_{k/k-1} = \begin{bmatrix} y_k - \hat{\phi}_k^T \widehat{\theta}_{1/k-1} \\ \vdots \\ y_k - \hat{\phi}_k^T \widehat{\theta}_{s/k-1} \end{bmatrix} \\ \varepsilon_{k/k} = Y_k - \hat{Y}_{k/k} = \begin{bmatrix} y_k - \hat{\phi}_k^T \widehat{\theta}_{1/k} \\ \vdots \\ y_k - \hat{\phi}_k^T \widehat{\theta}_{s/k} \end{bmatrix} \end{cases} \tag{4.11}$$

L'algorithme R-SOE-OBE ('Recursive Switched Output Error (R-SOE-OBE)') proposé ici est une adaptation de l'algorithme d'identification des systèmes linéaires à commutations décrit par un modèle SARX présenté dans le chapitre 2. L'algorithme R-SOE-OBE est décrit dans le tableau 4.1. A chaque instant k, l'algorithme se déroule en deux étapes successives que nous détaillons ci-dessous.

Étape 1 : Estimation de l'état discret λ_k.

L'estimation du sous-modèle actif $\hat{\lambda}_k$ correspond à l'indice i de la plus petite composante de $\mid \varepsilon_{k/k-1}(i) \mid$ avec $i \in \{1, \cdots, s\}$.

L'algorithme R-SOE-OBE

Initialisation :

- $\widehat{\Theta}_0$.
- $P_0 = p_0 I_{ns}$ avec $p_0 \gg 1$.
- $\{\hat{Y}_{k/k-1}\}_{k=1}^{\max\{n_a,n_b\}} = \{\hat{Y}_{k/k}\}_{k=1}^{\max\{n_a,n_b\}} = 0_s$
- (0_s est un vecteur de longueur s avec toutes les composantes $= 0$)

Étape 1 : **Estimation de l'état discret λ_k**

$$\begin{cases} \varepsilon_{k/k-1} = Y_k - \widehat{\varphi}_k^T \widehat{\Theta}_{k-1} \\ \hat{\lambda}_k = \underset{i=1,\cdots,s}{\operatorname{argmin}} \mid \varepsilon_{k/k-1}(i) \mid \\ \zeta_k = \underset{i=1,\cdots,s}{\min} \dfrac{\mid \varepsilon_{k/k-1}(i) \mid}{\delta} = \dfrac{\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid}{\delta} \end{cases} \tag{4.12}$$

Étape 2 : **Mise à jour de $\widehat{\Theta}_k$**

$$\widehat{\Theta}_k = \widehat{\Theta}_{k-1} + \Gamma_k \varepsilon_{k/k-1} \tag{4.13}$$

avec

$$\begin{cases} \Gamma_k = P_{k-1}\widehat{\varphi}_k \sigma_k (\gamma I_s + \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k)^{-1} \\ P_k = \dfrac{1}{\gamma}(I_{ns} - \Gamma_k \widehat{\varphi}_k^T) P_{k-1} \end{cases} \tag{4.14}$$

où

$$\sigma_k \quad = \quad \begin{cases} \gamma(\widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k)^{-1}(C_k - I_s) & \text{si} \quad \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k > 0 \quad \text{et} \quad \zeta_k > 1 \\ 0_{s\times s} \quad \text{Autrement} \end{cases} \tag{4.15}$$

et

$$C_k = \begin{bmatrix} 1 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \zeta_k & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ \hat{\lambda}_k^{\text{ième}} \\ \\ \\ \end{matrix} \qquad \hat{\lambda}_k^{\text{ième}} \tag{4.16}$$

– δ est un paramètre introduit par l'utilisateur.

– $0 < \gamma < 1$ est le facteur d'oubli.

TABLE 4.1 – Algorithme d'identification avec bruit borné d'un modèle SOE

Étape 2 : Mise à jour de $\widehat{\Theta}_k$

Cette étape de l'algorithme est dérivée d'un algorithme de type OBE. Comme dans le chapitre 2, le rapport $\zeta_k = \dfrac{|\,\varepsilon_{k/k-1}(\hat{\lambda}_k)\,|}{\delta}$ est comparé à 1 comme le montre l'équation (4.15), deux cas se produisent :

- Si $\zeta_k \leq 1$, alors aucune mise à jour n'est faite pour $\widehat{\Theta}_k$ grâce à la matrice $\sigma_k = 0$ qui donne $\Gamma_k = 0$.
- Si $\zeta_k > 1$, alors, dans le vecteur global de paramètres $\widehat{\Theta}_k$, seul le vecteur de paramètres $\widehat{\theta}_{\hat{\lambda}_k}$ est mis à jour. Ceci est établi grâce à la structure particulière de la matrice $\sigma_k \in \mathbb{R}^{s\ x\ s}$.

A partir du tableau 4.1, $\varepsilon_{k/k}$ peut être écrit comme :

$$\varepsilon_{k/k} = \varepsilon_{k/k-1} - \widehat{\varphi}_k^T \Gamma_k \varepsilon_{k/k-1}$$

$$\varepsilon_{k/k} = (I_s - \widehat{\varphi}_k^T \Gamma_k)\varepsilon_{k/k-1}$$

En utilisant l'expression de Γ_k on trouve :

$$\varepsilon_{k/k} = \gamma(\gamma I_s + \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k)^{-1} \varepsilon_{k/k-1} \tag{4.17}$$

Si $\widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k > 0$, alors les deux cas suivants se produisent :

- Si $\zeta_k \leq 1$, alors $\sigma_k = 0$ et on a

  $$|\,\varepsilon_{k/k}(\hat{\lambda}_k)\,| = |\,\varepsilon_{k/k-1}(\hat{\lambda}_k)\,|$$

  Ceci donne :

  $$|\,\varepsilon_{k/k}(\hat{\lambda}_k)\,| \leqslant \delta \tag{4.18}$$

- Si $\zeta_k > 1$, en utilisant la définition de σ_k dans (4.15) pour $\sigma_k \neq 0$ on trouve :

  $$\varepsilon_{k/k} = C_k^{-1} \varepsilon_{k/k-1}$$

  Il en résulte

  $$\varepsilon_{k/k}(\hat{\lambda}_k) = \frac{\delta}{|\,\zeta_k\,|}\ \zeta_k$$

  Ainsi, à partir (4.12)

  $$|\,\varepsilon_{k/k}(\hat{\lambda}_k)\,| = \delta \tag{4.19}$$

Ce résultat est cohérent avec l'objectif de l'étude. L'algorithme proposé réalise l'estimation d'un vecteur global de paramètres $\widehat{\Theta}_k$ de sorte que, à chaque instant k : $|\,\varepsilon_{k/k}(\hat{\lambda}_k)\,| \leq \delta$. Ainsi, δ est une borne sur l'erreur a posteriori associée au sous-modèle actif estimé. La valeur de δ est spécifiée en tenant compte de la borne supérieure sur le bruit δ_e (cf. section 4.4).

4.3 Analyse de l'algorithme R-SOE-OBE

L'objectif de cette section est de fournir une analyse de stabilité et de convergence de l'algorithme "R-SOE-OBE". Comme nous l'avons dit dans le chapitre 2, la convergence de l'algorithme dépend de son initialisation et de la bonne ou mauvaise décision sur l'estimation de l'état discret. Nous présentons une analyse de convergence en montrant l'impact de l'initialisation et l'estimation de l'état discret sur le comportement de l'algorithme "R-SOE-OBE".

Nous rappelons la définition de l'impact d'une mauvaise décision sur l'estimation de l'état discret introduite dans le chapitre 2 :

$$d_k = \begin{cases} 0, & \text{si } \hat{\lambda}_k = \lambda_k \\ \phi_k^T(\theta_{\lambda_k} - \theta_{\hat{\lambda}_k}), & \text{si } \hat{\lambda}_k \neq \lambda_k \end{cases} \tag{4.20}$$

avec $\mid \phi_k^T(\theta_{\lambda_k} - \theta_{\hat{\lambda}_k}) \mid \leq \delta_d$. Nous rappelons aussi la définition de l'excitation persistante, $\{\widehat{\varphi}_k\}_{k=1}^N$ est une séquence qui possède la propriété d'excitation persistante d'ordre $o_e \geq ns$ s'il existe $\alpha > 0$ et $\beta > 0$ tel que pour chaque k :

$$\alpha I_{ns} \leq \sum_{i=0}^{o_e-1} \widehat{\varphi}_{k-i}\sigma_{k-i}\widehat{\varphi}_{k-i}^T \leq \beta I_{ns} \tag{4.21}$$

Hypothèse 1. *Considérons les hypothèses suivantes :*

A.1 $\{A_i(q)\}_{i=1}^{i=s}$ *est tel que :*

$$\| 1 - A_i(q) \|_1 < 1$$

A.2 δ *est tel que :*

$$\delta \geq \frac{\| A_i(q) \|_1}{1 - \| 1 - A_i(q) \|_1}\delta_e$$

Compte tenu des hypothèses (A.1) et (A.2) et la condition d'excitation persistante, nous présentons le résultat de convergence suivant :

Théorème 3. *Considérons la classe de système définie dans la section 4.1. Supposons que les conditions (A.1) et (A.2) sont satisfaites. Si* $\{\widehat{\varphi}_k\}_{k=1}^N$ *est une séquence qui possède la propriété de l'excitation persistante d'ordre* $o_e \geq ns$*, alors l'algorithme "R-SOE-OBE" décrit dans le tableau 4.1 est tel que :*

– *Pour chaque* $k \geq o_e + 1$*, le vecteur d'erreur sur les paramètres* $\widetilde{\Theta}_k = \Theta^* - \widehat{\Theta}_k$ *satisfait :*

$$\| \widetilde{\Theta}_k \|^2 \leq \eta_1\gamma^k \| \widetilde{\Theta}_0 \|^2 + \eta_2 \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \tag{4.22}$$

avec $\eta_1 > 0$ *et* $\eta_2 > 0$

– *Si le nombre de mauvaises décisions sur l'estimation de l'état discret est négligeable, alors* $\widetilde{\Theta}_k$ *est tel que :*

$$\| \widetilde{\Theta}_k \|^2 \leq \eta_1 \gamma^k \| \widetilde{\Theta}_0 \|^2 \tag{4.23}$$

■

Remarque 10. *Les hypothèses (A.1) et (A.2) ne sont que des conditions suffisantes et l'algorithme peut bien fonctionner dans certains cas où ces hypothèses ne sont pas satisfaites. Des hypothèses similaires ont été utilisées dans d'autres algorithmes de type OBE pour l'identification des systèmes décrits par des modèles du type erreur de sortie ([158], [159] [160]).*

Preuve du Théorème 3. *La preuve est réalisée en plusieurs étapes :*

– *Dans l'étape IV.1) nous montrons qu'il existe* S_{sup} *et* S_{inf} *tel que :* $0 < S_{inf} I_{ns} \leq P_k^{-1} \leq S_{sup} I_{ns}$.
– *Dans l'étape IV.2), en utilisant les résultats de l'item IV.1) et la fonction de Lyapunov, nous montrons que l'erreur de paramètres* $\widetilde{\Theta}_k$ *est bornée.*
– *Dans l'item IV.3) nous concluons la preuve.*

IV.1) Montrons qu'il existe S_{sup} *et* S_{inf} *tel que :* $0 < S_{inf} I_{ns} \leq P_k^{-1} \leq S_{sup} I_{ns}$

A partir de l'équation (4.14) et le lemme d'inversion matricielle on a :

$$P_k^{-1} = \gamma P_{k-1}^{-1} + \widehat{\varphi}_k \sigma_k \widehat{\varphi}_k^T \tag{4.24}$$

L'équation (4.24) peut être réécrite comme :

$$P_k^{-1} = \gamma^k P_0^{-1} + \sum_{i=0}^{k-1} \gamma^i \widehat{\varphi}_{k-i} \sigma_{k-i} \widehat{\varphi}_{k-i}^T \tag{4.25}$$

Du fait que $\sum_{i=0}^{k-1} \gamma^i Z_{k-i} \leq \sum_{i=0}^{k-1} \gamma^i \sum_{i=0}^{k-1} Z_{k-i}$ *(* $\gamma > 0$ *et* $Z_{k-1} \geq 0$*) et de la condition d'excitation persistante (4.21) on a :*

$$P_k^{-1} \leq \gamma^k P_0^{-1} + \beta I_{ns} \sum_{i=0}^{k-1} \gamma^i \tag{4.26}$$

Du fait que $\gamma < 1$*, ceci garantit l'existence de* S_{sup} *tel que :*

$$P_k^{-1} \leq S_{sup} I_{ns} \tag{4.27}$$

avec $S_{sup} = \gamma^k p_o + \beta \dfrac{1-\gamma^k}{1-\gamma}$.

De (4.24), après oe itérations sur P_k^{-1} *on trouve :*

$$P_k^{-1} = \gamma^{o_e} P_{k-o_e}^{-1} + \sum_{i=0}^{o_e-1} \gamma^i \widehat{\varphi}_{k-i} \sigma_{k-i} \widehat{\varphi}_{k-i}^T \tag{4.28}$$

Ainsi

$$P_k^{-1} \geq \sum_{i=0}^{o_e-1} \gamma^i \widehat{\varphi}_{k-i} \sigma_{k-i} \widehat{\varphi}_{k-i}^T \tag{4.29}$$

Prenant en compte que $\gamma < 1$*, il en résulte :*

$$P_k^{-1} \geq \gamma^{o_e-1} \sum_{i=0}^{o_e-1} \widehat{\varphi}_{k-i} \sigma_{k-i} \widehat{\varphi}_{k-i}^T \tag{4.30}$$

Du fait que : $\{\widehat{\varphi}_k\}_{k=1}^N$ *est une séquence qui possède la propriété de l'excitation persistante d'ordre* o_e*, il en résulte que :*

$$P_k^{-1} \geq S_{inf} I_{ns} > 0 \tag{4.31}$$

avec $S_{inf} = \gamma^{o_e-1}\alpha$.

IV.2) Montrons que $\| \widetilde{\Theta}_k \|^2 \leq \eta_1 \gamma^k \| \widetilde{\Theta}_0 \|^2 + + \eta_2 \dfrac{1-\gamma^{k\mathscr{L}_k}}{1-\gamma}$

On considère la fonction de Lyapunov

$$W_k = \widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k$$

avec $\widetilde{\Theta}_k = \Theta^* - \widehat{\Theta}_k$.

A partir de (4.13), (4.14) et (4.17), $\widetilde{\Theta}_{k-1}$ *s'écrit comme suit :*

$$\widetilde{\Theta}_{k-1} = \widetilde{\Theta}_k + \frac{1}{\gamma} P_{k-1} \widehat{\varphi}_k \sigma_k \varepsilon_{k/k} \tag{4.32}$$

En développant l'expression de W_{k-1} *on trouve :*

$$W_{k-1} = \widetilde{\Theta}_k^T P_{k-1}^{-1} \widetilde{\Theta}_k + \frac{1}{\gamma} \widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \varepsilon_{k/k} + \frac{1}{\gamma} \varepsilon_{k/k}^T \sigma_k \widehat{\varphi}_k^T \widetilde{\Theta}_k + \frac{1}{\gamma^2} \varepsilon_{k/k}^T \sigma_k \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k \varepsilon_{k/k}$$

Notons que :

$$(\widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \varepsilon_{k/k})^T = (\varepsilon_{k/k}^T \sigma_k \widehat{\varphi}_k^T \widetilde{\Theta}_k) = (\widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \varepsilon_{k/k})$$

Ainsi

$$W_{k-1} = \widetilde{\Theta}_k^T P_{k-1}^{-1} \widetilde{\Theta}_k + \frac{2}{\gamma} \widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \varepsilon_{k/k} + \frac{1}{\gamma^2} \varepsilon_{k/k}^T \sigma_k \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k \varepsilon_{k/k}$$

Le lemme d'inversion matricielle donne : $P_{k-1}^{-1} = \frac{1}{\gamma}(P_k^{-1} - \widehat{\varphi}_k \sigma_k \widehat{\varphi}_k^T)$. *En utilisant cette expression dans l'expression précédente de* W_{k-1}, *on trouve :*

$$W_{k-1} = \frac{1}{\gamma}W_k - \frac{1}{\gamma}\widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \widehat{\varphi}_k^T \widetilde{\Theta}_k + \frac{2}{\gamma}\widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \varepsilon_{k/k} + \frac{1}{\gamma^2}\varepsilon_{k/k}^T \sigma_k \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k \varepsilon_{k/k}$$

Ainsi

$$W_k = \gamma W_{k-1} + Q_k \tag{4.33}$$

avec

$$Q_k = \widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \widehat{\varphi}_k^T \widetilde{\Theta}_k - 2\widetilde{\Theta}_k^T \widehat{\varphi}_k \sigma_k \varepsilon_{k/k} - \frac{1}{\gamma}\varepsilon_{k/k}^T \sigma_k \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k \varepsilon_{k/k}$$

Deux cas sont possibles selon la valeur de la matrice σ_k *:*

IV.2.1) $\sigma_k = 0_{s\times s}$

Dans ce cas, aucune mise à jour n'est faite pour $\widehat{\Theta}_k$ *et* $Q_k = 0$, *ce qui donne* $W_k = \gamma W_{k-1}$. *Du fait que* $\gamma < 1$, *on trouve*

$$\widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k \leq \widetilde{\Theta}_{k-1}^T P_{k-1}^{-1} \widetilde{\Theta}_{k-1} \tag{4.34}$$

IV.2.1) $\sigma_k \neq 0_{s\times s}$

Dans ce cas, on définit : $S_k = \widehat{\varphi}_k^T \widetilde{\Theta}_k$, *alors* Q_k *peut être réécrit comme suit :*

$$Q_k = S_k^T \sigma_k S_k - 2S_k^T \sigma_k \varepsilon_{k/k} + \varepsilon_{k/k}^T \sigma_k \varepsilon_{k/k} - \frac{1}{\gamma}\varepsilon_{k/k}^T \sigma_k \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k \varepsilon_{k/k} - \frac{\gamma}{\gamma}\varepsilon_{k/k}^T \sigma_k \varepsilon_{k/k}$$

Ainsi

$$Q_k = (\varepsilon_{k/k} - S_k)^T \sigma_k (\varepsilon_{k/k} - S_k) - \frac{1}{\gamma}\varepsilon_{k/k}^T \sigma_k (\gamma I_s + \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k) \varepsilon_{k/k}$$

A partir de (4.17) on trouve :

$$Q_k = (\varepsilon_{k/k} - S_k)^T \sigma_k (\varepsilon_{k/k} - S_k) - \gamma \varepsilon_{k/k-1}^T \sigma_k (\gamma I_s + \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k)^{-1} \varepsilon_{k/k-1}$$

De (4.15), on a :

$$\gamma I_s + \widehat{\varphi}_k^T P_{k-1} \widehat{\varphi}_k \sigma_k = \gamma C_k$$

À la suite, Q_k *s'écrit comme suit :*

$$Q_k = (\varepsilon_{k/k} - S_k)^T \sigma_k (\varepsilon_{k/k} - S_k) - \varepsilon_{k/k-1}^T \sigma_k C_k^{-1} \varepsilon_{k/k-1} \tag{4.35}$$

En utilisant la structure particulière de σ_k*, la valeur différente de zéro dans la matrice* σ_k *est notée* $a_k \geq 0$*. En utilisant aussi la forme de la matrice* C_k*, la forme précédente de* Q_k *devient :*

$$Q_k = a_k[\varepsilon_{k/k}(\hat{\lambda}_k) - S_k(\hat{\lambda}_k)]^2 - \varepsilon^2_{k/k-1}(\hat{\lambda}_k)a_k\frac{1}{\zeta_k} \tag{4.36}$$

nous rappelons que $\zeta_k = \dfrac{\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid}{\delta}$*, ainsi*

$$Q_k = a_k[\varepsilon_{k/k}(\hat{\lambda}_k) - S_k(\hat{\lambda}_k)]^2 - a_k\delta \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid \tag{4.37}$$

La relation entre $\varepsilon_{k/k}(\hat{\lambda}_k)$ *et* $S_k(\hat{\lambda}_k)$ *peut être aisément établie :*

$$\begin{aligned}\varepsilon_{k/k}(\hat{\lambda}_k) = y_k - \hat{\phi}_k^T\hat{\theta}_{\hat{\lambda}_k/k} &= \phi_k^T\theta_{\lambda_k} + e_k - \hat{\phi}_k^T\hat{\theta}_{\hat{\lambda}_k/k} + \hat{\phi}_k^T\theta_{\hat{\lambda}_k} - \hat{\phi}_k^T\theta_{\hat{\lambda}_k} + \phi_k^T\theta_{\hat{\lambda}_k} - \phi_k^T\theta_{\hat{\lambda}_k} \\ &= \hat{\phi}_k^T\tilde{\theta}_{\hat{\lambda}_k/k} + d_k + e_k + (\phi_k^T - \hat{\phi}_k^T)\theta_{\hat{\lambda}_k}\end{aligned}$$

En gardant à l'esprit les structures de ϕ_k^T *et* $\hat{\phi}_k^T$*, ceci donne :*

$$\varepsilon_{k/k}(\hat{\lambda}_k) = \frac{1}{A_{\hat{\lambda}_k}(q)}\hat{\phi}_k^T\tilde{\theta}_{\hat{\lambda}_k/k} + \frac{1}{A_{\hat{\lambda}_k}(q)}d_k + e_k = \frac{1}{A_{\hat{\lambda}_k}(q)}S_k(\hat{\lambda}_k) + \frac{1}{A_{\hat{\lambda}_k}(q)}d_k + e_k$$

Ainsi,

$$S_k(\hat{\lambda}_k) = A_{\hat{\lambda}_k}(q)\varepsilon_{k/k}(\hat{\lambda}_k) - A_{\hat{\lambda}_k}(q)e_k - d_k \tag{4.38}$$

En remplaçant l'expression de $S_k(\hat{\lambda}_k)$ *dans (4.37),* Q_k *se réécrit comme suit :*

$$Q_k = a_k[(1 - A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k) + A_{\hat{\lambda}_k}(q)e_k + d_k]^2 - a_k\delta \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid$$

Cela peut être réécrit comme suit :

$$\begin{aligned}Q_k = a_k[((1 - A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k) + A_{\hat{\lambda}_k}(q)e_k)^2 - \delta \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid] \\ + a_k[d_k^2 + 2((1 - A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k) + A_{\hat{\lambda}_k}(q)e_k)d_k]\end{aligned}$$

Nous pouvons écrire $Q_k = q_k + L_k$ *avec*

$$\begin{aligned}q_k = a_k[((1 - A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k) + A_{\hat{\lambda}_k}(q)e_k)^2 - \delta \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid] \\ L_k = a_k d_k[d_k + 2((1 - A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k) + A_{\hat{\lambda}_k}(q)e_k)]\end{aligned}$$

Maintenant, nous allons prouver que $q_k \leq 0$. $q_k \leq 0$ *cela signifie que :*

$$((1-A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k)+A_{\hat{\lambda}_k}(q)e_k)^2 \leq \delta \mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid \tag{4.39}$$

Du fait que $\mid \varepsilon_{k/k-1}(\hat{\lambda}_k) \mid > \delta$, *alors (4.39) est satisfaite si :*

$$((1-A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k)+A_{\hat{\lambda}_k}(q)e_k)^2 \leq \delta^2 \tag{4.40}$$

De l'inégalité triangulaire, (4.40) est satisfaite si la condition suivante est satisfaite :

$$((1-A_{\hat{\lambda}_k}(q))^2\varepsilon^2_{k/k}(\hat{\lambda}_k))^{(\frac{1}{2})}+(A^2_{\hat{\lambda}_k}(q)e^2_k)^{(\frac{1}{2})} \leq \delta \tag{4.41}$$

A ce stade, nous introduisons l_1 *la norme induite* $\|\cdot\|_1$ *et supposons l'hypothèse (A.1) d'être vraie. Alors (4.41) est vérifiée si :*

$$\| 1-A_{\hat{\lambda}_k}(q) \|_1 \mid \varepsilon_{k/k}(\hat{\lambda}_k) \mid + \| A_{\hat{\lambda}_k}(q) \|_1 \mid e_k \mid \leq \delta \tag{4.42}$$

On sait que : $\mid \varepsilon_{k/k}(\hat{\lambda}_k) \mid \leq \delta$, $\mid e_k \mid \leq \delta_e$. *Ainsi, (4.42) est vérifiée si :*

$$\| 1-A_{\hat{\lambda}_k}(q) \|_1 \, \delta + \| A_{\hat{\lambda}_k}(q) \|_1 \, \delta_e \leq \delta \tag{4.43}$$

Ceci conduit à la condition (A.2) :

$$\delta \geq \frac{\| A_{\hat{\lambda}_k}(q) \|_1}{1-\| 1-A_{\hat{\lambda}_k}(q) \|_1}\delta_e$$

qui assure $q_k \leq 0$. *Ceci donne :* $Q_k \leq L_k$, *par conséquent :*

$$W_k \leq \gamma W_{k-1}+L_k \tag{4.44}$$

Maintenant, il ne reste plus qu'à exprimer la borne supérieure sur $\| \widetilde{\Theta}_k \|^2$. *En utilisant la norme induite* l_1, *on a :*

$$\mid (1-A_{\hat{\lambda}_k}(q))\varepsilon_{k/k}(\hat{\lambda}_k) + A_{\hat{\lambda}_k}(q)e_k \mid \leq \| 1-A_{\hat{\lambda}_k}(q) \|_1 \, \delta + \| A_{\hat{\lambda}_k}(q) \|_1 \, \delta_e$$

Ainsi, en utilisant (4.43), on trouve :

$$\mid L_k \mid \leq a_k \mid d_k \mid (\mid d_k \mid +2\delta) \tag{4.45}$$

A partir de (4.45) et (4.44), on trouve :

$$\widetilde{\Theta}^T_k P^{-1}_k \widetilde{\Theta}_k \leqslant \gamma^k \widetilde{\Theta}^T_0 P^{-1}_0 \widetilde{\Theta}_0 + \sum_{i=0}^{k-1} \gamma^i a_k (d^2_{k-i} + 2 \mid d_{k-i} \mid \delta)$$

On a $a_k \geq 0$ *est fini, alors il existe* $\rho > 0$ *tel que* $a_k \leq \rho$. d_k *est tel que* $\mid d_k \mid \leq \delta_d$. *On obtient :*

$$\widetilde{\Theta}_k^T P_k^{-1} \widetilde{\Theta}_k \leqslant \gamma^k \widetilde{\Theta}_0^T P_0^{-1} \widetilde{\Theta}_0 + \rho \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid (\delta_b + 2\delta_e) \tag{4.46}$$

Il a été prouvé que $0 < S_{inf} I_{ns} \leq P_k^{-1}$, *ainsi, on obtient :*

$$\widetilde{\Theta}_k^T S_{inf} \widetilde{\Theta}_k \leqslant \gamma^k \widetilde{\Theta}_0^T P_0^{-1} \widetilde{\Theta}_0 + \rho(\delta_b + 2\delta_e) \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \tag{4.47}$$

Avec $P_0 = p_o I_{ns}$, *cette inégalité donne (4.22) :*

$$\| \widetilde{\Theta}_k \|^2 \leq \eta_1 \gamma^k \| \widetilde{\Theta}_0 \|^2 + \eta_2 \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \tag{4.48}$$

avec $\eta_1 = \dfrac{1}{p_o S_{inf}} > 0$ *et* $\eta_2 = \dfrac{1}{S_{inf}} \rho(\delta_b + 2\delta_e) > 0$.

IV.3) Conclusion de la preuve

Si le nombre de mauvaises décisions sur l'estimation de l'état discret est négligeable, alors $\sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid \sim 0$ *et par conséquent on trouve (4.23).*

■

Ce théorème a la même interprétation que le théorème 1 donné dans le chapitre 2. En effet, le théorème énonce que l'erreur d'estimation sur les paramètres est bornée, même en présence des mauvaises décisions sur l'estimation de l'état discret ou dans le cas d'une mauvaise initialisation. Le terme $\eta_2 \sum_{i=0}^{k-1} \gamma^i \mid d_{k-i} \mid$ représente l'erreur due à des mauvaises décisions sur l'estimation de l'état discret. Il est évident qu'une mauvaise initialisation peut conduire à des mauvaises décisions sur l'estimation de l'état discret et à une erreur sur l'estimation du vecteur global de paramètres. Cependant, l'effet d'une mauvaise initialisation peut être lissé et filtré grâce au facteur d'oubli (si l'initialisation n'est pas trop mauvaise). En l'absence des mauvaises décisions sur l'estimation de l'état discret, l'erreur sur l'estimation de Θ^* diminue aussi longtemps que $\sigma_k \neq 0$ et que la condition d'excitation persistante est satisfaite.

4.4 Discussion

Discussion sur le choix de δ

L'algorithme R-SOE-OBE a un paramètre de synthèse $\boldsymbol{\delta}$ que l'utilisateur doit spécifier convenablement sa valeur en tenant compte de la borne supérieure connue sur le bruit δ_e. Le choix de $\boldsymbol{\delta}$ est vitale pour la stabilité de l'algorithme et la convergence du vecteur global de paramètres estimé $\widehat{\Theta}_k$ vers le vrai vecteur de paramètres Θ^*. Pour une valeur de $\boldsymbol{\delta}$ très grande par rapport à δ_e, l'algorithme est stable, mais la mise à jour de $\widehat{\Theta}_k$ est gelée à plusieurs instants k dès qu'une grande erreur de prédiction a priori est tolérée. Pour une valeur de $\boldsymbol{\delta}$ trop faible, les conditions de stabilité ne sont pas satisfaites et la convergence de l'algorithme n'est pas garantie. Comme le choix de $\boldsymbol{\delta}$ dépend du système à travers $A_{\lambda_k}(q)$ (hypothèse (A.2)) qui est inconnu et dépend de la borne supérieure sur le bruit δ_e, la stratégie suivante est recommandée : choisir une valeur de $\boldsymbol{\delta}$ grande au début de la procédure d'identification, puis diminuer sa valeur au cours du temps. Cette stratégie est illustré dans la prochaine section.
A noter qu'il serait souhaitable de relâcher les conditions (A.1) et (A.2) via un filtrage d'adaptation des données comme il a été proposé dans [158], [159] et [160]. Ceci constitue une perspective de ces travaux.

Extension de l'algorithme pour l'identification des systèmes MIMO à commutations décrits par un modèle MIMO-SOE

L'extension de l'algorithme R-SOE-OBE pour l'identification des systèmes MIMO à commutations décrit par un modèle du type erreur de sortie (MIMO-SOE) est très similaire à la procédure d'extension de l'algorithme R-SARX-OBE pour l'identification du modèle MIMO-SARX présenté dans le chapitre 2. De plus, le résultat du théorème 3 sur le comportement de $\widehat{\Theta}_k$ reste valable pour le cas MIMO-SOE.

Interprétation géométrique

L'interprétation géométrique fournie dans le chapitre 2 sur le fonctionnement de l'algorithme R-SARX-OBE peut être adaptée pour l'algorithme R-SOE-OBE en utilisant la matrice de donnée estimée $\widehat{\varphi}_k$ et en prenant en compte la structure du modèle erreur de sortie.

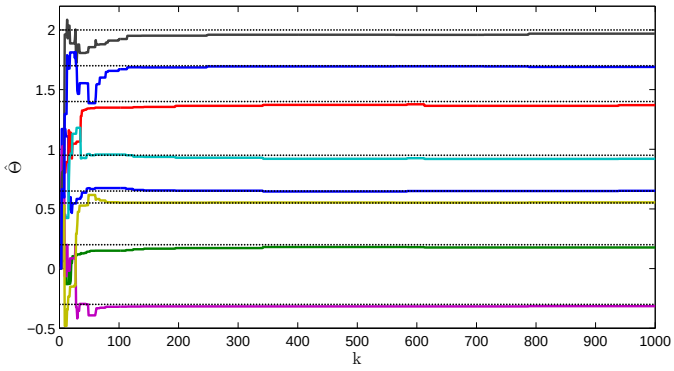

FIGURE 4.1 – Convergence de $\widehat{\Theta}_k$ (trait plein) vers Θ^* (trait pointillé).

4.5 Exemple numérique

Afin d'illustrer les performances de l'algorithme R-SOE-OBE proposé, le modèle SOE suivant composé de deux sous-modèles est considéré :

$$y_k = \frac{B_{\lambda_k}}{A_{\lambda_k}} + e_k \tag{4.49}$$

avec

$$\text{sous-modèle 1} \begin{cases} A_1(q) = 1 + 0.65q^{-1} + 0.2q^{-2} \\ B_1(q) = 1.4 + 0.95q^{-1} \end{cases}$$

$$\text{sous-modèle 2} \begin{cases} A_2(q) = 1 - 0.3q^{-1} + 0.55q^{-2} \\ B_2(q) = 2 + 1.7q^{-1} \end{cases}$$

Le nombre de sous-modèles est $s = 2$. Les données ont été obtenues avec une commutation arbitraire de l'état discret $\lambda_k \in \{1,2\}$. La séquence d'entrée u_k est une séquence aléatoire de moyenne nulle avec une distribution uniforme sur $[-2\ ;\ 2]$. Le terme de bruit est une séquence aléatoire de moyenne nulle uniformément répartie dans $[-\delta_e\ ;\ \delta_e]$. δ_e a été ajusté de façon à obtenir une valeur souhaitée du rapport signal sur bruit (SNR).

Dans un premier temps, l'algorithme R-SOE-OBE a été appliqué sur un horizon de longueur $N = 1000$ et pour $SNR = 25dB$. Le paramètre δ a été choisi $\delta = 1.5\delta_e$. La convergence de $\widehat{\Theta}_k$ vers Θ^* est illustrée sur la figure 4.1 en fonction du temps. Il apparaît clairement que l'algorithme R-SOE-OBE permet une convergence assez rapide de $\widehat{\Theta}_k$ au voisinage de Θ^*.

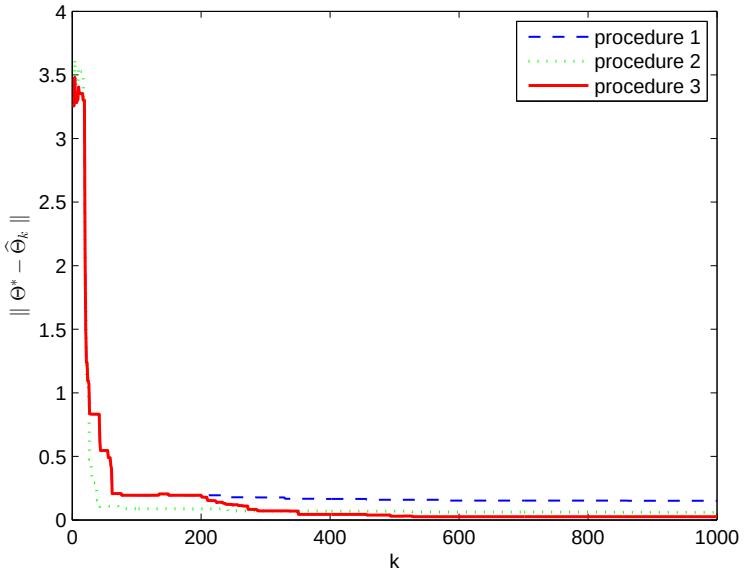

FIGURE 4.2 – $\| \Theta^* - \widehat{\Theta}_k \|$ en fonction du temps k.

Afin d'évaluer l'influence du choix de δ sur les résultats, 3 procédures d'identification ont été réalisées comme suit :

- Procédure 1 : $\delta = 5\delta_e$.
- Procédure 2 : $\delta = 2\delta_e$.
- Procédure 3 : Nous choisissons une valeur de δ grande au début de la procédure d'identification, puis nous diminuons cette valeur au cours du temps. δ est choisi comme suit :

$$\begin{cases} \delta = 5\delta_e & \text{pour } k \in [1 \quad 200] \\ \delta = 2\delta_e & \text{pour } k \in [201 \quad 400] \\ \delta = 1.5\delta_e & \text{pour } k \in [401 \quad 1000] \end{cases}$$

Pour chaque procédure, nous calculons à chaque instant k l'erreur d'estimation sur les paramètres $\| \Theta^* - \widehat{\Theta}_k \|$. Les résultats de simulations pour les trois procédures sont illustrés sur la figure 4.2. Il est clair que la troisième procédure donne les meilleurs résultats. Elle permet d'améliorer la qualité d'estimation au cours du temps.

$\hat{\theta}_1$	$\hat{\theta}_2$	Indices statistiques
0.6512 ± 0.0083	-0.2990 ± 0.0100	
0.1999 ± 0.0081	0.5500 ± 0.0079	**FIT** = 93.6%
1.3954 ± 0.0272	2.0026 ± 0.0220	**PPE** = 0.0143
0.9525 ± 0.0335	1.7026 ± 0.0294	

TABLE 4.2 – Résultats et mesures statistiques

Si la mise en œuvre en ligne de l'algorithme R-SOE-OBE n'est pas nécessaire, l'algorithme R-SOE-OBE peut être mis en œuvre en mode hors-ligne. Une deuxième stratégie pour faire un choix judicieux sur δ dans le cas de l'exécution de l'algorithme en mode hors-ligne est la suivante :

- Itérer l'algorithme plusieurs fois.
- Choisir une valeur de δ grande à la première itération, puis diminuer sa valeur après chaque itération.

Dans un second temps, l'algorithme R-SOE-OBE a été appliqué sur la base d'une simulation Monte-Carlo avec 100 jeux de données (des séquences de longueur $N = 1000$, $\delta = 1.5\delta_e$ et un SNR $= 25$dB). La valeur moyenne de chaque paramètre et l'écart-type sur chaque paramètre sont présentés dans le tableau 4.2. Les résultats montrent les performances fournies par l'algorithme R-SOE-OBE : les valeurs moyennes des estimées sont proches des valeurs réelles et l'écart-type est assez faible sur chaque paramètre.

Afin d'évaluer les performances de l'algorithme R-SOE-OBE vis-à-vis du niveau de bruit, nous utilisons les indices "*FIT*" et "*PEE*" définis dans le chapitre 2. La figure 4.3 illustre l'évolution de ces deux indices en fonction de la variation de SNR. Les résultats confirment l'importance de l'algorithme R-SOE-OBE, en l'occurrence les bonnes performances et la bonne robustesse vis-à-vis du niveau bruit.

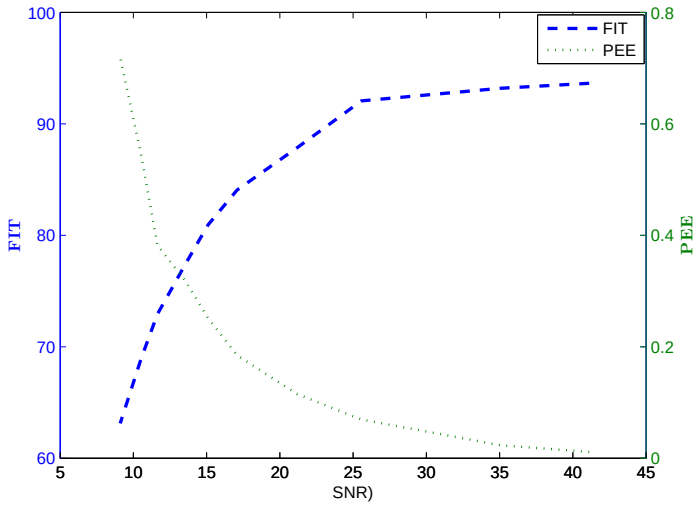

FIGURE 4.3 – FIT et PEE en fonction de SNR.

4.6 Conclusion

Dans ce chapitre, nous avons proposé l'algorithme R-SOE-OBE. Il s'agit d'une adaptation de l'algorithme R-SARX-OBE afin de réaliser l'identification en temps réel des systèmes linéaires à commutations avec un bruit borné décrits par un modèle du type erreur de sortie. Les propriétés de convergence et de stabilité ont été établies. Des résultats de simulation ont montré l'efficacité et l'intérêt de l'algorithme R-SOE-OBE.

Deuxième partie

De la classification à l'identification des systèmes ayant une sortie binaire

Chapitre 5

Classification supervisée par Support Vector Machines (SVMs)

Dans ce chapitre, nous présentons le principe des SVMs. Les SVMs classent les données d'apprentissage en deux classes en cherchant l'hyperplan séparateur optimal qui maximise la marge entre les données de deux classes. Les cas des données linéairement séparables et non linéairement séparables sont envisagés.

Sommaire

5.1 Introduction

Les SVMs sont des outils communément utilisés dans la classification supervisée au cours de ces dernières années. Ils ont été utilisés avec succès dans plusieurs domaines, notamment en traitement d'image, la reconnaissance des formes et le diagnostic médical. La popularité des SVMs est justifiée par le développement fondamentale sous-jacent et une capacité de gérer des grands ensembles de données avec une charge de calcul raisonnable.

Les SVMs reposent sur deux idées clés. La première idée est la notion de marge maximale. La marge est la distance entre l'hyperplan séparateur et les données les plus proches de cet hyperplan. Dans un problème de classification binaire, l'objectif est de trouver les paramètres d'un hyperplan qui permet de séparer les données en deux classes en maximisant la marge entre les données de ces deux classes. Cet hyperplan est l'hyperplan optimal. Dans certains cas, lorsque les données ne sont pas linéairement séparables dans l'espace considéré, il n'est pas possible de trouver une frontière linéaire qui sépare les données en deux classes. La deuxième idée clé des SVMs est la possibilité de transformer l'ensemble de données d'apprentissage non linéairement séparables dans l'espace de départ en un ensemble de données d'apprentissage linéairement séparables dans un espace caractéristique de dimension plus grande que la dimension de l'espace de départ. Cette transformation de données est réalisée grâce à des fonctions noyaux. Nous détaillons le principe de ces deux idées dans les prochaines sections.

Une première version de l'algorithme de classification de données en utilisant les SVMs permet de trouver l'hyperplan séparateur optimal dans le cas où les données sont linéairement séparables a été présentée dans [182]. Une deuxième version de l'algorithme permet de trouver l'hyperplan séparateur optimal dans le cas où les données ne sont pas linéairement séparables a été présentée dans [18]. Des limitations pratiques importantes comme la présence de bruit sur les données d'apprentissage ou la présence de valeurs aberrantes ont été étudiées en utilisant le concept des variables de relâchement dans [184] qui ont permis d'introduire la notion de marge souple par rapport aux erreurs admissibles. Ces variables permettent d'autoriser certaines erreurs et de trouver une marge souple. Dans ce qui suit, nous détaillons d'une manière concise le principe des SVMs dans les deux cas (données linéairement séparables et données non linéairement séparables). Pour plus de détails sur les SVMs, le lecteur est invité à se référer à ([184], [26], [44], [169] et [171]).

5.2 Données linéairement séparables

Ce cas est relativement simple par rapport au cas de données non linéairement séparables. Afin d'illustrer la manière dont l'hyperplan optimal est estimé, considérons $\{x_k\}_{k=1}^{N} \subset \mathbb{R}^d$ l'ensemble des données d'apprentissage avec leurs labels $\{s_k = \pm 1\}_{k=1}^{N}$, N est le nombre de données d'apprentissage disponibles. L'objectif des SVMs est d'estimer les paramètres de l'hyperplan séparateur entre les deux classes $+1$ et -1. Cet hyperplan est considéré comme une frontière de décision. Il existe plusieurs hyperplans qui peuvent séparer les données en deux classes comme l'illustre la figure 5.1. Les performances de ces hyperplans sont identiques en apprentissage, par contre, les performances en prédiction sur de nouvelles données peuvent être très différentes.

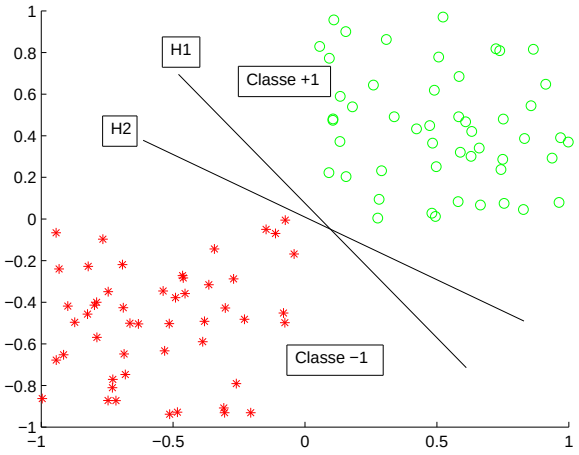

FIGURE 5.1 – Plusieurs hyperplans peuvent séparer les données en deux classes

L'objectif de l'algorithme de classification utilisant les SVMs est de trouver les paramètres de l'hyperplan qui sépare les données en deux classes en commettant un minimum d'erreur et en maximisant la marge entre les deux classes. Ceci constitue une condition d'optimalité sur la position de l'hyperplan séparateur comme l'illustre la figure 5.2. Soit H l'hyperplan séparateur, w son vecteur normal et b son décalage par rapport à l'origine. L'équation de H a la forme d'un modèle linéaire comme suit :

$$f(x) = w^T x + b = 0 \tag{5.1}$$

Pour chaque point de données x_k comme entrée, la fonction f retourne la sortie $f(x_k)$

donnée comme suit :

$$\begin{cases} \text{Si} \quad f(x_k) > 0 \quad \text{alors} \quad x_k \in \text{classe} \quad +1 \\ \text{Si} \quad f(x_k) < 0 \quad \text{alors} \quad x_k \in \text{classe} \quad -1 \end{cases} \tag{5.2}$$

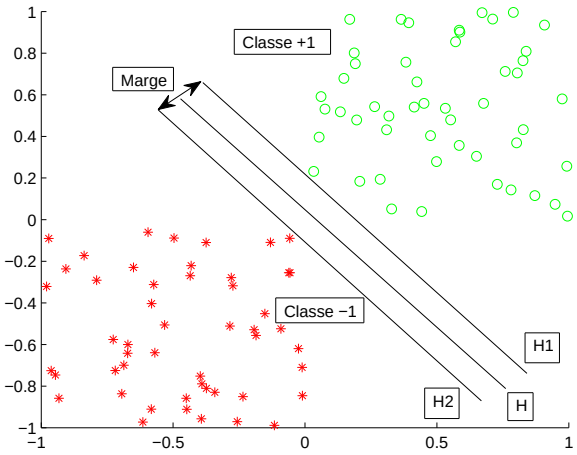

FIGURE 5.2 – Principe de l'hyperplan séparateur optimal

La distance d'un point x à l'hyperplan séparateur est donnée par :

$$d(x) = \frac{|\, wx + b \,|}{\| w \|}$$

La maximisation de la marge revient alors à minimiser $\| w \|$ sous la contrainte $s_k(w^T x_k + b) > 0$. Le problème d'optimisation à résoudre afin de trouver les paramètres de l'hyperplan séparateur optimal est formalisé de la manière suivante : trouver w et b qui permettent de :

$$\begin{cases} \min\limits_{w,b} \ \frac{1}{2} \| w \|^2 \\ \text{s.c} \quad s_k(w^T x_k + b) > 0 \ ; \ \forall \ k \end{cases} \tag{5.3}$$

Les vecteurs de support sont les points les plus proches de l'hyperplan, mais appartenant à des classes différentes. Puisque les paramètres w et b sont définis dans (5.1) à un facteur de de proportionnalité près, les vecteurs support sont les points pour lesquels $[s_k(w^T x_k + b) = 1]$. Ceci signifie que les SVMs prennent en compte la position des points par rapport à l'hyperplan $f(x) = 0$ et de leurs positions par rapport aux hyperplans $f(x) = \pm 1$, ce qui permet de remplacer la contrainte $s_k(w^T x_k + b) > 0$ par la contrainte

$s_k(w^T x_k + b) \geq 1$. Cette contrainte est imposée afin de trouver une solution unique au problème d'optimisation. La formulation du problème d'optimisation primal est exprimée alors sous la forme suivante :

$$\begin{cases} \min\limits_{w,b} \frac{1}{2} \parallel w \parallel^2 \\ \text{s.c} \quad s_k(w^T x_k + b) \geq 1 \ ; \ \forall \ k \end{cases} \tag{5.4}$$

En terme de complexité, le problème d'optimisation (5.4) est délicat à résoudre et il est plus facile de résoudre son problème dual qui est construit à partir du lagrangien donné par :

$$L(w,b) = \frac{1}{2} \parallel w \parallel^2 - \sum_{k=1}^{N} \alpha_k [s_k(w^T x_k + b) - 1] \tag{5.5}$$

où $\alpha_k \in \mathbb{R}^+$ sont les coefficients de Lagrange. Le lagrangien doit être minimisé par rapport à w et b, et maximisé par rapport à α. La résolution du problème d'optimisation passe par la satisfaction des conditions de KKT (Karush-Kuhn-Tucker) qui consistent à annuler les dérivées partielles du lagrangien donné par (5.5) par rapport aux paramètres w et b :

$$\begin{cases} \dfrac{\partial L}{w} = 0 \Rightarrow \quad w = \sum\limits_{k=1}^{N} \alpha_k s_k x_k \\ \dfrac{\partial L}{b} = 0 \Rightarrow \quad \sum\limits_{k=1}^{N} \alpha_k s_k = 0 \end{cases} \tag{5.6}$$

La dérivée partielle du lagrangien par rapport à w conduit au résultat $w = \sum_{k=1}^{N} \alpha_k s_k x_k$. Cette relation signifie que l'hyperplan séparateur est exprimé à partir des réalisations dont le coefficient de Lagrange n'est pas nul. Les points x_k correspondant à des α_k positifs non nuls sont les vecteurs de support. En substituant (5.6) dans (5.5), nous obtenons le problème dual suivant :

$$\begin{cases} \max\limits_{\alpha} \sum\limits_{k=1}^{N} \alpha_k - \frac{1}{2} \sum\limits_{k=1}^{N} \sum\limits_{j=1}^{N} \alpha_k \alpha_j s_k s_j x_k^T x_j \\ \text{s.c} \quad \sum\limits_{k=1}^{N} \alpha_k s_k = 0 \end{cases} \tag{5.7}$$

C'est un problème quadratique de taille N. Sa résolution donne les multiplicateurs de Lagrange optimaux α_k^* (les α_k positifs non nuls). Pour chaque nouveau point de données x à classifier, la fonction de décision est donnée par :

$$f(x) = \sum_{k=1}^{N} \alpha_k^* s_k^* (x_k^* x) + b \tag{5.8}$$

où $\sum_{k=1}^{N} \alpha_k^* s_k^* x_k^* = w$. La détermination de la valeur de b peut se faire à partir de la relation $\alpha^*(s^*(wx^* + b) - 1) = 0$ en utilisant n'importe quel vecteur de support x^*. Afin de limiter l'influence du bruit et de trouver une valeur plus fiable pour b, un traitement en moyenne est possible en utilisant tous les vecteurs de support.

5.3 Données non linéairement séparables

En général, dans des applications réelles, les données d'apprentissage ne sont pas linéairement séparables dans l'espace de départ. Un exemple de données d'apprentissage non linéairement séparables est illustré sur la figure 5.3. Cette non-linéarité peut être transformée en linéarité en projetant les données d'apprentissage dans un espace caractéristique de dimension plus grande que l'espace de départ. Cette projection permet d'augmenter la séparabilité des données. Ainsi, une fois que les données sont linéairement séparables dans l'espace caractéristique augmenté, nous pouvons appliquer les SVMs pour trouver la frontière de décision. Notons que cette frontière de décision est non-linéaire dans l'espace de départ.

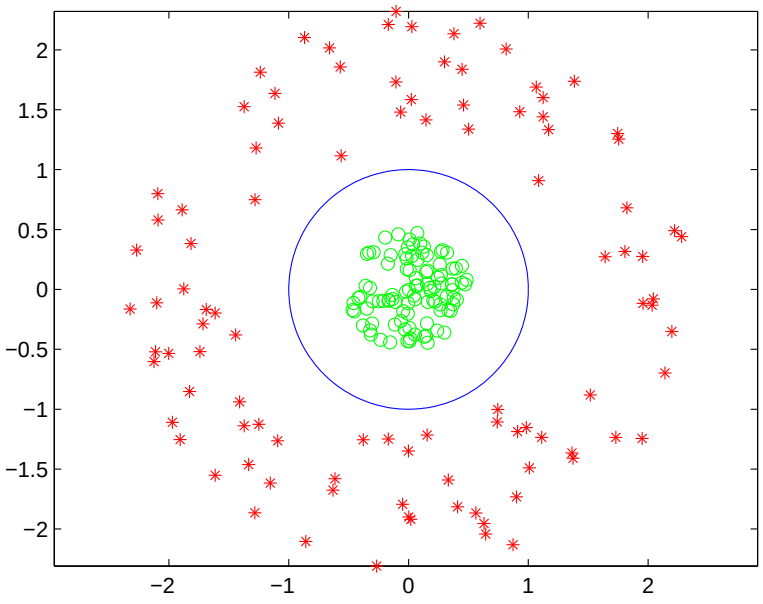

FIGURE 5.3 – Données non linéairement séparable dans l'espace de départ avec une frontière non linéaire.

Afin d'illustrer le principe de projection de données dans un espace caractéristique augmenté, nous considérons l'application non linéaire suivante :

$$\begin{cases} \varphi : \mathscr{X} \mapsto \mathscr{F} \\ \quad x \mapsto \varphi(x) \end{cases} \tag{5.9}$$

où $\mathscr{X}$ est l'espace de départ de dimension n_h et $\mathscr{F}$ l'espace caractéristique de dimension $n_f > n_h$. Il s'agit d'appliquer l'algorithme d'apprentissage dans $\mathscr{F}$ au lieu de $\mathscr{X}$ en considérant l'ensemble $\{\varphi(x_k)\}_{k=1}^N$, $\varphi(x_k) \in \mathscr{F}$ comme ensemble d'apprentissage avec leurs catégories $\{s_k = \pm 1\}_{k=1}^N$. La formulation du problème d'optimisation primal est exprimée alors sous la forme suivante :

$$\begin{cases} \min\limits_{w,b} \frac{1}{2} \parallel w \parallel^2 \\ \text{s.c} \quad s_k(w^T \varphi(x) + b) \geq 1 \ ; \ \forall \ t \end{cases} \tag{5.10}$$

Dans certains cas, la dimension de l'espace caractéristique augmenté peut être très grande (peut aller jusqu'à l'infini), la construction de l'application φ est relativement compliquée. Néanmoins, le passage au problème dual permet d'utiliser une fonction positive, nommée fonction noyau, telle que la connaissance explicite de la fonction φ devient inutile. En effet, le lagrangien est donné par :

$$L(w,b) = \tfrac{1}{2} \parallel w \parallel^2 - \sum_{k=1}^{N} \alpha_k [s_k(w^T \varphi(x) + b) - 1] \tag{5.11}$$

En appliquant les conditions de KKT (Karush-Kuhn-Tucker) on trouve :

$$\begin{cases} \dfrac{\partial L}{w} = 0 \Rightarrow \quad w = \sum\limits_{k=1}^{N} \alpha_k s_k \varphi(x) \\ \dfrac{\partial L}{b} = 0 \Rightarrow \qquad \sum\limits_{k=1}^{N} \alpha_k s_k = 0 \end{cases} \tag{5.12}$$

En substituant (5.12) dans (5.11), nous obtenons le problème dual suivant :

$$\begin{cases} \max\limits_{\alpha} \sum\limits_{k=1}^{N} \alpha_k - \frac{1}{2} \sum\limits_{k=1}^{N} \sum\limits_{j=1}^{N} \alpha_k \alpha_j s_k s_j K(x_k, x_j) \\ \qquad\qquad \text{s.c} \qquad \sum\limits_{k=1}^{N} \alpha_k s_k = 0 \end{cases} \tag{5.13}$$

$K(x_k, x_j) = \varphi(x_k)^T \varphi(x_j)$ désigne la fonction noyau positive utilisée dans l'algorithme d'apprentissage dans le cas des données non linéairement séparables.

On note que l'algorithme de base n'a pas été modifié, seul le produit scalaire $x_k^T x_j$ a été remplacé par une fonction noyau. Chaque fonction noyau correspond à une application de projection φ dans un espace augmenté $\mathscr{F}$. Seule la fonction noyau interviendra dans les calculs, la connaissance de φ n'est pas nécessaire. La question qui se pose à ce stade est : quelles sont les fonctions noyaux qui permettent de trouver un tel espace caractéristique $\mathscr{F}$ et une telle application de projection φ ?
On distingue trois fonctions noyaux typiques : la fonction noyau linéaire, $K(x_k,x_j) = x_k^T x_j$, la fonction noyau polynomiale de degré d, $K(x_k,x_j) = (x_k^T x_j + \tau)^d$ et la fonction noyau gaussienne, $K(x_k,x_j) = exp(- \parallel x_k - x_j \parallel^2 / \sigma^2$ (voir [169]). Le choix d'une fonction noyau et la détermination de ses paramètres (tels que les paramètres d et τ pour la fonction polynomiale ou le paramètre σ pour la fonction gaussienne) sont fait à partir de la validation croisée et des tests statistiques sur les données d'apprentissage.

En pratique, les données d'apprentissage sont bruitées et ne peuvent alors pas être séparables même dans l'espace caractéristique augmenté. Dans ce cas, on utilise la technique de marge souple (soft margin) qui a été présentée dans [184]. Pour ce faire, des variables de relâchement v_k sont ajoutés pour relâcher la contrainte sur la marge. Le problème d'optimisation primal devient alors :

$$\begin{cases} \min\limits_{w,b,v} \frac{1}{2} \parallel w \parallel^2 + \gamma \sum\limits_{k=1}^{N} v_k \\ \text{s.c} \quad s_k(w^T \varphi(x_k) + b) \geq 1 - v_k \quad \text{et} \quad v_k \geq 0 \quad ; \forall\, k \end{cases} \tag{5.14}$$

où $\gamma \in \mathbb{R}_0^+$ est la constante de régularisation (un paramètre de pénalité). Ce paramètre détermine la tolérance des SVMs vis a vis du nombre de points mal classés. Il est impossible de connaître a priori la valeur optimal de γ. Il existe plusieurs méthodes pour déterminer cette valeur de γ, la plus utilisée est la validation croisée. Afin de résoudre le problème primal (5.14), le lagrangien suivant est construit :

$$L(w,b,v,\alpha,\beta) = \frac{1}{2} \parallel w \parallel^2 - \gamma \sum_{k=1}^{N} v_k - \sum_{k=1}^{N} \alpha_k [s_k(w^T \varphi(x_k) + b) - 1 + v_k] - \sum_{k=1}^{N} \beta_k v_k \tag{5.15}$$

où $\alpha_k \in \mathbb{R}^+$ et $\beta_k \in \mathbb{R}^+$ sont les coefficients de Lagrange.

Les conditions d'optimalités sont données par :

$$\begin{cases} \dfrac{\partial L}{w} = 0 \Rightarrow \quad w = \displaystyle\sum_{k=1}^{N} \alpha_k s_k \varphi(x_k) \\ \dfrac{\partial L}{b} = 0 \Rightarrow \qquad \displaystyle\sum_{k=1}^{N} \alpha_k s_k = 0 \\ \dfrac{\partial L}{v} = 0 \Rightarrow \qquad \beta_k = \gamma - \alpha_k \end{cases} \tag{5.16}$$

En substituant (5.16) dans (5.15), nous trouvons le problème dual suivant :

$$\begin{cases} \max\limits_{\alpha} \displaystyle\sum_{k=1}^{N} \alpha_k - \frac{1}{2} \sum_{k=1}^{N} \sum_{j=1}^{N} \alpha_k \alpha_j s_k s_j K(x_k, x_j) \\ \qquad\qquad \text{s.c} \qquad \displaystyle\sum_{k=1}^{N} \alpha_k s_k = 0 \\ 0 \leq \alpha_k \leq \gamma \; ; \; k = 1, \cdots N \end{cases} \tag{5.17}$$

Ce problème est le même que (5.13) avec une contrainte supplémentaire sur α_k. Pour chaque nouvelle donnée x à classifier, la fonction de décision est donnée par :

$$f(x) = \sum_{i=1}^{N} \alpha_i^* s_i^* K(x_i, x) + b \tag{5.18}$$

5.4 Conclusion

Dans ce chapitre, nous avons présenté d'une manière concise le principe des SVMs issu de la théorie de l'apprentissage statistique. Ce principe est basé sur deux idées : la marge maximale et la fonction noyau. L'idée de la marge maximale consiste à chercher l'hyperplan qui donne une marge maximale entre deux classes de données. L'idée de fonction noyau permet de généraliser les SVMs pour le cas où les données sont non linéairement séparables en projetant les données sur un espace de dimension supérieure. Les problèmes d'optimisation primal et dual dans le cas où les données sont linéairement séparables et dans le cas où les données sont non linéairement séparables ont été présentés. Dans les deux prochains chapitres, nous proposons d'appliquer les SVMs pour résoudre le problème d'identification des systèmes non linéaires à temps discret et à temps continu dans le cas où la non-linéarité est produite par un capteur binaire.

Chapitre 6

Identification des systèmes ayant une sortie binaire via les SVMs

Dans ce chapitre, nous considérons le problème d'identification des systèmes linéaires ayant une sortie binaire. Ce problème est délicat à résoudre à cause de la binarisation du signal de sortie que l'on peut interpréter comme une non-linéarité produite par un capteur binaire, la partie linéaire du système est modélisée par un filtre FIR (Finite Impulse Respense). Pour résout ce problème, nous adaptons une approche qui consiste à reformuler le problème d'identification comme un problème de classification binaire. Cette reformulation permet alors d'utiliser des algorithmes d'apprentissage supervisé tel que les SVMs. Des résultats de simulations sont donnés à la fin du chapitre afin d'illustrer les performances de l'algorithme proposé.

Sommaire

6.1 Contexte et motivations

Pour certains systèmes, la seule information disponible sur le signal de sortie à chaque instant est une mesure binaire. Ces mesures binaires sont fournies par un capteur binaire placé en cascade avec le système linéaire. L'utilisation de ces capteurs binaires provoque une perte d'information sur le signal de sortie. La cause de l'utilisation d'un tel capteur peut être économique ou technique. Il existe de nombreux exemples de systèmes utilisant des capteurs binaires, notamment dans les processus chimiques pour la surveillance de la pression ou du niveau de liquide, les systèmes de communications pour indiquer les conditions de transmission dans les réseaux ATM ("Asynchronous Transfer Mode") ([201], [203]).

Dans le contexte des systèmes à sortie binaire, l'identification du comportement dynamique est un problème délicat à résoudre. Plusieurs méthodes d'identification ont été proposées ([203], [200], [202], [42], [91], [45], [92] et [80]). Une synthèse des méthodes développées avant l'année 2010 a été faite dans [201]. Ces méthodes se distinguent par des contraintes et des limitations restrictives. Dans [203], il est nécessaire de connaître la fonction de distribution du bruit et de connaître la valeur du seuil caractéristique de commutation du capteur. Malheureusement, dans la pratique, la fonction de distribution du bruit est en général inconnue. Dans ([42], [91], [45], [92], [80] et [157]), des conditions sur le seuil caractéristique de commutation sont imposées (l'utilisation d'un seuil égal à zéro, un seuil différent de zéro, mais connu ou un seuil réglable). Les méthodes présentées dans ([200], [202] et [134]) proposent d'identifier à la fois le seuil caractéristique et les paramètres du système. Cependant, ces méthodes nécessitent l'utilisation d'un signal d'entrée spécifique : un signal périodique dont la période dépend de l'ordre du système.

On se propose dans ce chapitre d'aborder ce problème d'identification. Ainsi, on surmonte certaines difficultés des méthodes disponibles dans la littérature. Nous introduisons un point de vue original pour l'identification des systèmes ayant une sortie binaire. Nous reformulons le problème d'identification comme un problème de classification binaire : le vecteur d'observation est interprété comme un point dans l'espace et la sortie binaire du système est interprétée comme l'étiquette d'une classe. Cette reformulation repose sur un résultat d'identifiabilité et permet d'utiliser des algorithmes de classification supervisée tels que les SVMs.

6.2 Formulation du problème d'identification

Considérons un système linéaire discret couplé avec une non-linéarité statique en sortie comme l'illustre la figure 6.1. Le système est modélisé par :

$$\begin{cases} y_k = G(q)u_k + e_k \\ s_k = \mathbf{Q}_C(y_k) \end{cases} \tag{6.1}$$

où $u_k \in \mathbb{R}$ et $s_k \in \mathbb{R}$ sont l'entrée et la sortie du système respectivement. $y_k \in \mathbb{R}$ est un signal interne inconnu et $e_k \in \mathbb{R}$ est le terme de bruit.

$G(q)$ est la fonction de transfert du type filtre FIR (Finite Impulse Response)d'ordre n donnée par :

$$G(q) = \sum_{i=0}^{n} g_i q^{-i} \tag{6.2}$$

où $\{g_i\}$ sont les paramètres de la réponse impulsionnelle.La fonction non linéaire $\mathbf{Q_C}(.)$ correspond à un capteur binaire défini par :

$$\mathbf{Q}_C(x) = \begin{cases} s_{haut} & \text{si} \quad x - C \geq 0 \\ s_{bas} & \text{si} \quad x - C < 0 \end{cases} \tag{6.3}$$

C est un seuil constant inconnu.

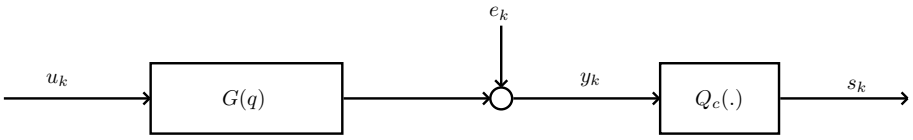

FIGURE 6.1 – Le système considéré

Cette modélisation nous permet de définir le problème d'identification de système linéaire couplé avec un capteur binaire de la manière suivante :

Problème d'identification : *Étant donné l'ensemble des observations* $\{(u_k, s_k)\}_{k=1}^{N}$, *avec* N *le nombre de données disponibles, proposer un algorithme d'identification qui permet d'estimer les paramètres de la fonction de transfert* $G(q)$ *et le seuil du capteur binaire* C.

Ce problème d'identification est bien posé si les hypothèses données ci-dessous sont satisfaites :

B.1 La sortie du système est centrée : $s_{bas} = s_{haut}$. Par conséquent, la sortie du système s'écrit comme suit :

$$s_k = s_{haut} sign(y_k - C) \tag{6.4}$$

où la fonction *sign* est la fonction mathématique qui donne le signe d'un nombre réel.

B.2 Le gain statique de $G(q)$ est connu.

B.3 La séquence d'entrée $\{u_k\}$ est telle que pour chaque instant k : $y_k \neq C$.

L'hypothèse (B.1) est considérée pour des raisons de facilité d'écriture et sans aucune perte de généralité. L'hypothèse (B.2) est nécessaire afin de satisfaire la condition d'identifiabilité. Sans cette hypothèse, il est impossible de distinguer le couple $(G(q), C)$ du couple $a(G(q), C)$ avec $a > 0$. Cette hypothèse est posée dans d'autres méthodes (par exemple [91], [92]). Si la valeur du seuil de capteur C est connue et différente de zéro, cette hypothèse devient non nécessaire. En effet, certaines méthodes ([80], [35]) remplacent la connaissance du gain statique par la connaissance du seuil de capteur binaire. Finalement, l'hypothèse (B.3) est nécessaire pour éviter les points singuliers. Si la séquence $\{u_k\}$ possède la propriété de l'excitation persistante [126], alors le cas $y_k = C$ est marginal et ceci n'affectera pas la convergence de l'algorithme d'identification.

6.3 Algorithme d'identification

Dans cette section, dans un premier temps, nous présentons une reformulation du problème d'identification comme un problème de classification binaire. Dans un second temps, nous présentons un algorithme d'identification qui permet de résoudre ce problème.

Nous définissons $\theta_o \in \mathbb{R}^{n+1}$ comme étant le vecteur de paramètres et $\phi_k \in \mathbb{R}^{n+1}$ comme étant le vecteur de régression par :

$$\theta_o = \begin{pmatrix} g_0 \\ \vdots \\ g_n \end{pmatrix} \qquad \phi_k = \begin{pmatrix} u_k \\ \vdots \\ u_{k-n} \end{pmatrix} \tag{6.5}$$

En utilisant les définitions de θ_o et ϕ_k, la sortie binaire du système s_k peut être réécrite comme suit :

$$s_k = s_{haut} sign(\phi_k^T \theta_o - C + e_k) \tag{6.6}$$

L'objectif maintenant est d'estimer le couple (θ_o, C). En utilisant le Théorème 4 donné ci-dessous, nous nous assurons que le problème d'identification est solvable.

Théorème 4. *Pour $N \geq n+3$, considérons deux filtres FIR de même ordre $G_1(q)$ et $G_2(q)$ et deux seuils C_1 et C_2.*

Si, pour toute séquence d'entrée $\{u_k\}_{k=1}^N$ et toute séquence de bruits $\{e_k\}_{k=1}^N$: "$G_1(q)u_k - C_1 + e_k$ et $G_2(q)u_k - C_2 + e_k$" ont toujours le même signe, alors il existe un scalaire positif unique μ tel que :

$$\begin{cases} G_1(q) = \mu G_2(q) \\ C_1 = \mu C_2 \end{cases} \tag{6.7}$$

■

Preuve du Théorème 4. *Les notations suivantes sont utilisées*

$$G_1(q) = \sum_{i=0}^{n} g_i^{(1)} q^{-i} \tag{6.8}$$

$$G_2(q) = \sum_{i=0}^{n} g_i^{(2)} q^{-i} \tag{6.9}$$

Il en résulte que

$$\begin{cases} y_k^{(1)} = G_1(q)u_k - C_1 + e_k = \overline{\phi}_k^T \overline{\theta}_1 \\ y_k^{(2)} = G_2(q)u_k - C_2 + e_k = \overline{\phi}_k^T \overline{\theta}_2 \end{cases} \tag{6.10}$$

avec

$$\overline{\phi}_k = \begin{pmatrix} -1 \\ u_k \\ \vdots \\ u_{k-n} \\ d_k \end{pmatrix} \quad \overline{\theta}_1 = \begin{pmatrix} C_1 \\ g_0^{(1)} \\ \vdots \\ g_n^{(1)} \\ 1 \end{pmatrix} \quad \overline{\theta}_2 = \begin{pmatrix} C_2 \\ g_0^{(2)} \\ \vdots \\ g_n^{(2)} \\ 1 \end{pmatrix} \tag{6.11}$$

Si $y_k^{(1)}$ et $y_k^{(2)}$ ont toujours le même signe pour chaque séquence $\{u_k\}_{k=1}^N$ et $\{e_k\}_{k=1}^N$, alors quel que soit le vecteur $\overline{\phi}_k$ nous avons :

$$y_k^{(1)} y_k^{(2)} = \overline{\theta}_1^T \overline{\phi}_k \overline{\phi}_k^T \overline{\theta}_2 > 0 \tag{6.12}$$

(En excluant les cas marginaux $y_k^{(1)} = 0$ *et* $y_k^{(2)} = 0$*), alors :*

$$\overline{\theta}_1^T \left(\sum_{k=1}^{N} \overline{\phi}_k \overline{\phi}_k^T \right) \overline{\theta}_2 > 0 \tag{6.13}$$

$N \geq n+3$*, il en résulte que pour toute matrice symétrique définie positive* M*, nous avons :*

$$\overline{\theta}_1^T M \overline{\theta}_2 > 0 \tag{6.14}$$

Cette inégalité est vraie si et seulement s'il existe un scalaire unique $\mu > 0$ *tel que* $\overline{\theta}_1 = \mu \overline{\theta}_2$.

■

Ce résultat peut être réécrit comme suit : Si deux couples $(G_1(q), C_1)$ et $(G_2(q), C_2)$ sont tels que le produit $(G_1(q)u_k - C_1 + e_k)(G_2(q)u_k - C_2 + e_k)$ a toujours le même signe pour chaque instant k, alors $(G_1(q), C_1)$ et $(G_2(q), C_2)$ sont proportionnels.

Ce résultat d'identifiabilité a une grande importance dans la mesure où il indique qu'il est possible d'identifier θ_o et C à partir de la connaissance du signe de $(\phi_k^T \theta_o - C + e_k)$. De ce résultat, l'objectif maintenant est d'estimer un couple $(\widehat{\theta}, \widehat{C})$ tel que $\forall k \in [1; N]$:

$$s_k(\phi_k^T \widehat{\theta} - \widehat{C} + e_k) > 0 \tag{6.15}$$

Une fois ce couple $(\widehat{\theta}, \widehat{C})$ obtenu, il est possible de revenir au couple (θ_o, C) par une normalisation en utilisant le gain statique connu du système (hypothèse B.2).

Avec une sortie binaire, le bruit additif e_k peut selon son amplitude introduire une commutation parasite de la sortie. Deux cas se produisent :

1. e_k introduit une mauvaise commutation du capteur :

$$\begin{cases} \text{si} \quad (\phi_k^T \widehat{\theta} - \widehat{C}) < 0 \quad \text{et} \quad (\phi_k^T \widehat{\theta} - \widehat{C} + e_k) > 0 \Rightarrow e_k > 0 \quad \text{et} \quad s_k > 0 \\ \text{si} \quad (\phi_k^T \widehat{\theta} - \widehat{C}) > 0 \quad \text{et} \quad (\phi_k^T \widehat{\theta} - \widehat{C} + e_k) < 0 \Rightarrow e_k < 0 \quad \text{et} \quad s_k < 0 \end{cases} \tag{6.16}$$

   Dans ce cas le bruit a une influence sur la sortie du système donnée par le capteur binaire (une décision erronée prise par le capteur). On peut remarquer que la sortie s_k et le terme de bruit e_k sont du même signe.

2. e_k n'introduit pas une mauvaise commutation du capteur :

$$\begin{cases} \text{si} \quad (\phi_k^T \widehat{\theta} - \widehat{C}) < 0 \quad \text{et} \quad (\phi_k^T \widehat{\theta} - \widehat{C} + e_k) < 0 \Rightarrow \quad s_k < 0 \\ \text{si} \quad (\phi_k^T \widehat{\theta} - \widehat{C}) > 0 \quad \text{et} \quad (\phi_k^T \widehat{\theta} - \widehat{C} + e_k) > 0 \Rightarrow \quad s_k > 0 \end{cases} \tag{6.17}$$

Dans ce cas, le bruit n'a aucune influence sur la sortie du système, sa présence n'a aucun effet et il peut être remplacé par zéro ($e_k = 0$).

Ainsi, la condition (6.15) peut être réécrite comme suit :

$$s_k(\phi_k^T \widehat{\theta} - \widehat{C} + \widehat{e}_k) > 0 \tag{6.18}$$

avec $\widehat{e}_k$ représente l'impact de bruit sur la sortie du système. $\widehat{e}_k$ est interprété de la manière suivante : si le terme de bruit e_k introduit une mauvaise commutation du capteur, alors $\widehat{e}_k$ correspond à e_k, tandis que, si le terme de bruit e_k n'introduit pas une mauvaise commutation du capteur, alors $\widehat{e}_k$ correspond à zéro. Nous notons que $\widehat{e}_k$ et s_k ont le même signe.

Le nombre de données N est fini, par conséquent, il existe un certain $\varepsilon>0$ tel que pour chaque $k \in [1;N]$, (6.18) s'écrit :

$$s_k(\phi_k^T \widehat{\theta} - \widehat{C} + \widehat{e}_k) \geq \varepsilon \tag{6.19}$$

Du fait que le couple $(\widehat{\theta},\widehat{C})$ est estimé à un coefficient près (Théorème 4), on peut réécrire (6.19) comme suit :

$$s_k(\phi_k^T \widehat{\theta} - \widehat{C} + \widehat{e}_k) \geq 1 \tag{6.20}$$

En définissant $v_k = s_k \widehat{e}_k$, ceci produit la contrainte suivante sur le couple $(\widehat{\theta},\widehat{C})$ pour chaque $k \in [1;N]$:

$$\begin{cases} s_k(\widehat{\theta}^T \phi_k - \widehat{C}) \geq 1 - v_k \\ v_k \geq 0 \end{cases} \tag{6.21}$$

La contrainte imposée sur l'estimation du couple $(\widehat{\theta},\widehat{C})$ est similaire à la contrainte imposée dans l'algorithme de classification en utilisant les SVM (cf. chapitre 5). Par conséquent, nous pouvons utiliser les SVM pour estimer le couple $(\widehat{\theta},\widehat{C})$.

Le problème d'identification considéré peut s'écrire comme un problème de classification binaire : étant donné un ensemble de points d'apprentissage $\{\phi_k\}_{k=1}^N$ dans $\mathbb{R}^{n+1}$ avec leurs labels $\{s_k = \pm s_{haut}\}_{k=1}^N$, il s'agit d'estimer l'hyperplan qui sépare les données de la classe s_{haut} de données de la classe $-s_{haut}$. Dans notre problème d'identification, cet hyperplan est paramétré par le couple (θ_o, C). Afin d'illustrer ce point de vue, nous considérons un exemple à deux dimensions avec $G_o(q) = 0.8 + 0.2q^{-1}$ et $C = 0.5$. La figure 6.2 illustre la partition de points dans l'espace et l'hyperplan séparateur.

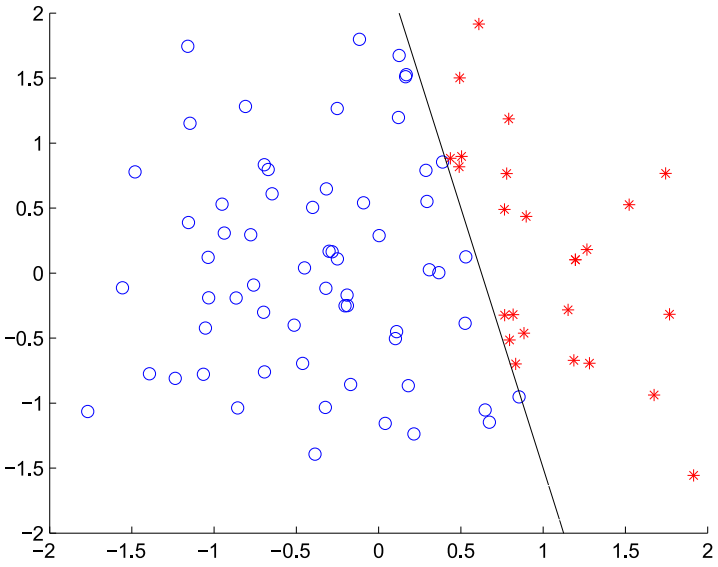

FIGURE 6.2 – Exemple illustratif : $G_o(q) = 0.8 + 0.2q^{-1}$ et $C = 0.5$

Algorithme d'identification

Étape 1 trouver $\widehat{\theta}$ et $\widehat{C}$ solution de

$$\begin{cases} \min\limits_{\theta,C,v} \frac{1}{2}\|\theta\|^2 + \gamma \sum\limits_{k=1}^{N} v_k \\ \text{s.c} \quad s_k(\theta^T \phi_k - C) \geq 1 - v_k \quad \text{et} \quad v_k \geq 0 \quad ; \quad \forall k \in [1;N] \end{cases}$$

Étape 2 Normaliser par la valeur du gain statique.

TABLE 6.1 – Les deux étapes de l'algorithme d'identification

L'algorithme d'identification proposé est basé sur l'utilisation des SVMs pour trouver l'hyperplan séparateur entre les deux classes. L'algorithme est composé de deux étapes. La première étape utilise les SVMs afin d'estimer un couple $(\widehat{\theta},\widehat{C})$, la deuxième étape est une étape de normalisation pour trouver (θ_o, C) sous l'hypothèse B.2. L'algorithme proposé est illustré dans le tableau 6.1. γ est un facteur de pénalisation (cf. chapitre 5).

Remarque 11. *L'algorithme proposé interprète les données disponibles comme un ensemble de points dans l'espace. Il est ainsi possible de construire une base d'apprentissage avec une ou plusieurs expériences.*

6.4 Simulations et résultats

Nous présentons dans cette section un exemple numérique afin d'illustrer les performances de l'algorithme proposé. Les données numériques ont été générées selon (6.1), (6.2) et (6.3) avec $s_{haut} = 1$. Le système considéré est le suivant :

$$G(s) = \frac{1}{\left(\frac{s}{\tau}+1\right)\left(\frac{s^2}{\omega_o^2}+2\zeta\frac{s}{\omega_o}+1\right)}$$

où $\tau = 4\pi$, $\omega_o = 20\pi$, $\zeta = 0.2$ et la fréquence d'échantillonnage est de $100Hz$. La partie statique non linéaire $\mathbf{Q}_C(.)$ possède un seuil $C = 0.1$.

L'algorithme a été appliqué dans les différentes situations. Pour chaque situation, l'ordre du modèle FIR estimé est $n = 40$. La séquence d'entrée $\{u_k\}$ est une séquence aléatoire de moyenne nulle avec une distribution uniforme sur $[-2\ ;\ 2]$. Un bruit gaussien de moyenne nulle est ajouté à la sortie. L'amplitude du bruit est ajustée de façon à obtenir une valeur souhaitée du rapport signal sur bruit (SNR). L'algorithme d'identification est appliqué sur la base d'une simulation Monte-Carlo avec 100 jeux de données.
Pour mesurer la qualité de l'estimation, l'indice statistique "FIT" suivant a été utilisé :

$$FIT = 100\left(1 - \frac{\| y - \hat{y} \|}{\| y - \bar{y} \|}\right) \tag{6.22}$$

avec $\hat{y}_k = \widehat{G}(q)u_k$.

Détermination de la valeur de γ sur les performances de l'algorithme

Dans un premier temps, nous avons choisi un nombre de données $N = 2000$ et un rapport signal sur bruit $SNR = 5dB$. Une possibilité de voir l'influence du choix de γ sur les résultats d'estimation est de faire varier sa valeur et calculer le "FIT" correspondant pour chaque valeur. La meilleure valeur de γ est celle qui maximise le "FIT. La figure 6.3 illustre la variation du "FIT" en fonction de γ. Il apparaît que le choix optimal de γ est près de 10^{-1} pour ce SNR.

Pour ce choix de γ, la figure 6.4 illustre une comparaison entre la réponse impulsionnelle estimée et la réponse impulsionnelle réelle. On voit clairement que malgré la binarisation du signal de sortie du système, on arrive à identifier correctement le modèle du système.

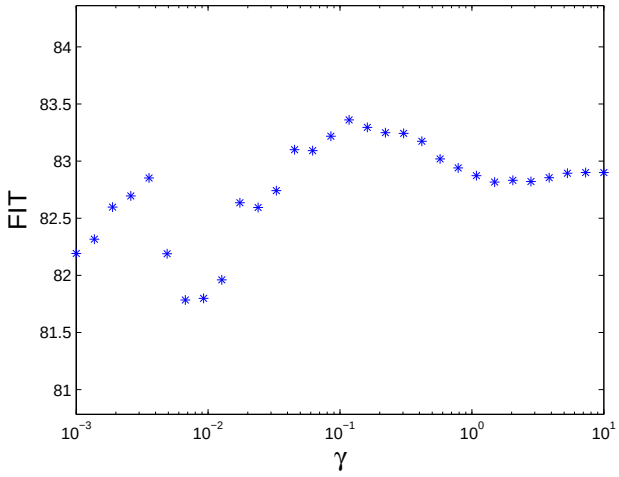

FIGURE 6.3 – Le FIT en fonction de γ pour $N = 2000$ et $SNR = 5dB$

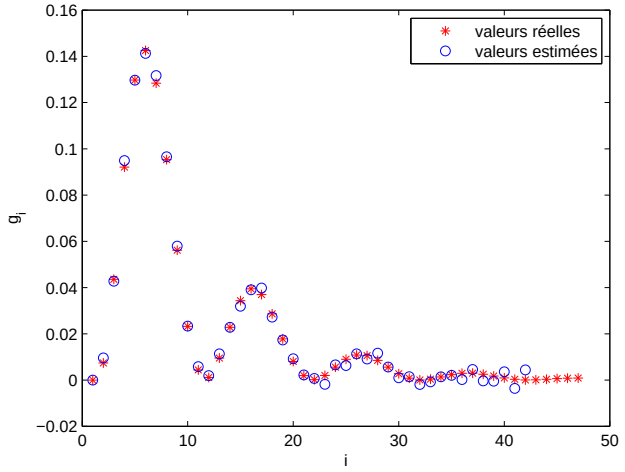

FIGURE 6.4 – L'estimation moyenne des paramètres g_i de la réponse impulsionnelle pour $N = 2000$ et $SNR = 5dB$

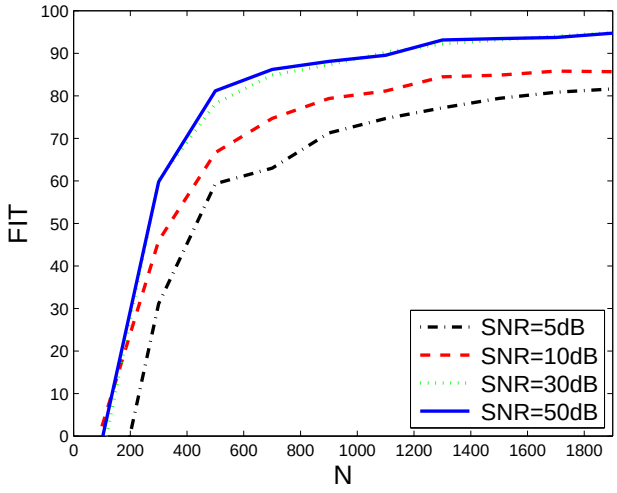

FIGURE 6.5 – Le FIT en fonction de N et SNR

Influence du niveau bruit et le nombre de données disponibles sur les résultats d'estimation

Afin d'évaluer l'influence du nombre de données disponibles et le niveau du bruit sur les résultats d'estimations, nous avons appliqué l'algorithme d'identification pour différentes valeurs de N et SNR : N varie de 100 jusqu'à 2000 pour les SNR suivants : $5dB$, $10dB$, $30dB$ et $50dB$.

Les variation du "FIT" en fonction de différentes valeurs de N et SNR sont illustrés sur la figure 6.5. Comme on pouvait s'y attendre, les performances s'améliorent avec l'augmentation du nombre de données N et du SNR. Ces bonnes performances sont confirmées par la figure 6.6 qui illustre la variation de l'erreur entre le vrai vecteur de paramètres et le vecteur de paramètres estimé $\| \theta_o - \widehat{\theta} \|$ pour les différentes valeurs de N et SNR.

Maintenant, il ne reste qu'à montrer la bonne estimation de seuil de capteur C. La figure 6.7 illustre l'effet du niveau de bruit et le nombre de données disponible sur l'estimation de C en utilisant les différentes valeurs de N et SNR. Les résultats montrent que la méthode proposée estime correctement le seuil du capteur dans le cas où N est suffisamment grand et pour des SNR les plus élevés ($\widehat{C} \simeq 0.1$).

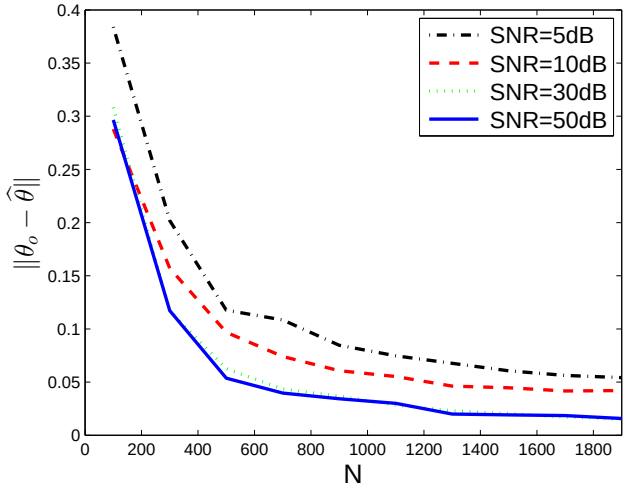

FIGURE 6.6 – $\|\theta_o - \widehat{\theta}\|$ en fonction de N et SNR

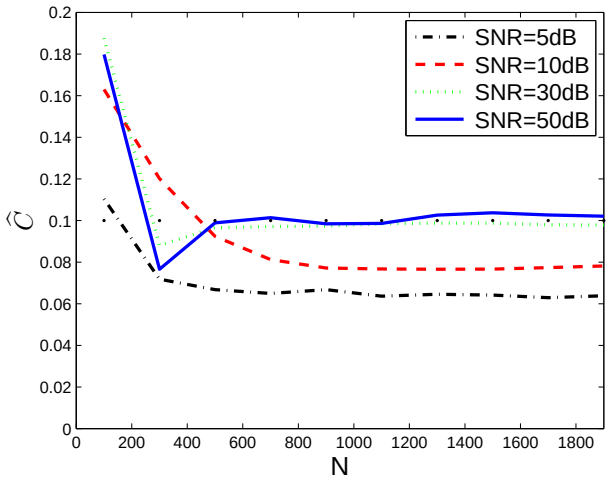

FIGURE 6.7 – L'estimation de C en fonction de N et SNR

6.5 Conclusion

Dans ce chapitre, nous avons reformulé le problème d'identification des systèmes ayant une sortie binaire comme un problème de classification. Un algorithme de classification basé sur les SVMs a été proposé pour l'identification du système modulo un résultat d'identifiabilité approprié. Les résultats de simulation montrent que les SVMs sont appropriés pour l'identification des systèmes ayant une sortie binaire et sont utilisable en présence de bruit de mesures inéluctable. Dans le prochain chapitre, nous proposons un algorithme d'identification de système à temps continu ayant une sortie binaire.

Chapitre 7

Identification des systèmes continus ayant une sortie binaire

Ce chapitre est consacré au problème d'identification des systèmes continus ayant une sortie binaire. C'est un problème ouvert pour lequel aucune solution n'a été proposée en dépit de la diversité des méthodes d'identification des systèmes continus disponibles dans la littérature. En effet, ces méthodes ne sont pas bien appropriées dans le cas d'observation binaire. Nous proposons une première solution basée sur une reconstruction consistante de la séquence de sortie du système modulo une séquence d'entrée appropriée et une spécification adéquate de la fonction de binarisation. L'estimation des paramètres est ensuite réalisée par l'une des méthodes d'identification des systèmes continus disponibles. Des résultats de simulation sont donnés à la fin du chapitre pour illustrer la validité de la méthode proposée.

Sommaire

7.1 Contexte et motivations

L'identification des systèmes linéaires discrets ayant une sortie binaire a reçu une attention particulière au cours de ces dernières années (cf. chapitre 6). Même si ces modèles discrets peuvent être ensuite convertis en modèles continus, cela ne peut pas être suffisant pour résoudre le problème d'identification dans le cas où les données ne sont pas uniformément échantillonnées ou que l'on veuille retrouver des paramètres physiques. L'identification d'un modèle à temps continu est motivé par des considération de simplicité du processus de validation du modèle [66] et par l'estimation des paramètres physiques, ce qui donne des possibilités d'analyse des systèmes à des non automaticiens.

Deux approches d'identification des systèmes continus ont été proposées [181], l'approche directe et l'approche indirecte. Les deux approches nécessitent des données échantillonnées. L'approche indirecte consiste à estimer un modèle discret dans un premier temps et convertir ce modèle en un modèle continu dans un second temps. En général, cette approche indirecte fait appel à des algorithmes d'optimisation itératifs très coûteux en termes de complexité de calcul et elle ne garantit pas la convergence vers l'optimum global [63]. À propos de l'approche directe, un grand nombre de techniques ont été proposées dans la littérature. Cette approche peut supporter un échantillonnage non uniforme des données. Certaines méthodes peuvent être trouvées dans ([206], [65], [164], [66] et [62]). Cette approche a atteint un haut degré de maturité, comme en témoigne l'intégration de ces méthodes dans la boîte à outils de "IDENT" de "Matlab", le développement de la boîte à outils de "CONTSID" [61], [64] et la boîte à outils "CAPTAIN" [208], [207] pour Matlab. Les deux approches nécessitent un capteur de résolution suffisamment élevé sur la sortie du processus afin de réaliser l'identification. Pour l'identification des systèmes continus ayant une sortie binaire, du fait de la faible résolution sur le signal de sortie, il apparaît que l'application de ces deux approches est inappropriée.

On se propose dans ce chapitre d'aborder ce problème d'identification des systèmes continus ayant une sortie binaire. Pour ce faire, nous proposons un algorithme en deux étapes. La première étape de l'algorithme est dédiée à la reconstruction du signal de sortie afin d'avoir un signal à haute résolution. Elle est basée sur deux idées clés : tout d'abord, l'utilisation d'une séquence d'entrée spécifique permettant le paramétrage du signal de sortie. Ensuite, l'utilisation des SVMs permettant l'estimation de ce paramétrage. La deuxième étape de l'algorithme est consacrée à l'estimation des paramètres du

modèle continu. L'estimation est réalisée via l'une des méthodes d'identification des systèmes continus classiques (disponibles dans la littérature). Avant d'entamer l'explication de ces deux étapes, nous formalisons d'abord le problème d'identification dans la prochaine section.

7.2 Formulation du problème d'identification

Considérons un système linéaire continu couplé avec un capteur binaire comme l'illustre la figure 7.1.

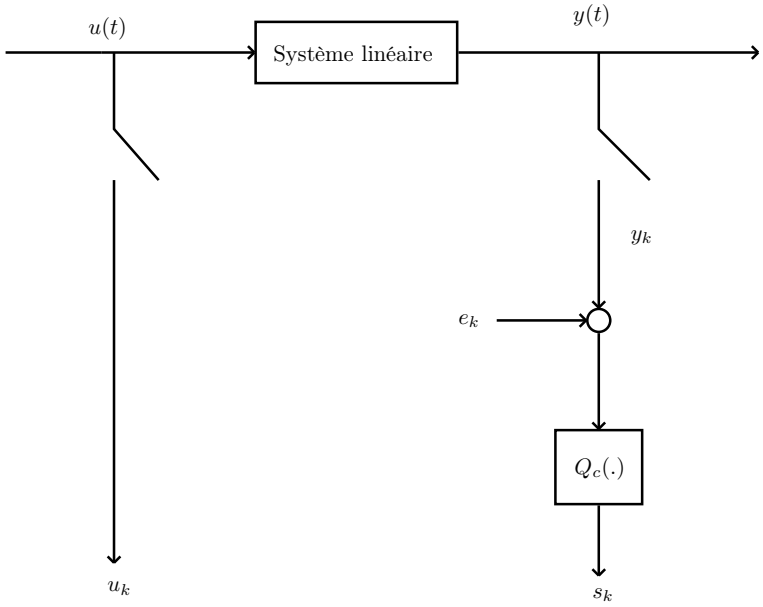

FIGURE 7.1 – Le modèle à temps continu considéré

La partie linéaire du système est décrit par le modèle suivant :

$$Y(s) = G(s)U(s) \tag{7.1}$$

où s est l'opérateur différentiel, $Y(s)$ et $U(s)$ sont les transformées de Laplace de $u(t)$ et $y(t)$ qui sont l'entrée et la sortie de la partie linéaire du système respectivement. $G(s)$ est l'opérateur de transfert modélisé par

$$G(s) = \frac{B(s)}{A(s)} = \frac{b_0 s^{n_b} + b_1 s^{n_b-1} + \cdots b_{n_b}}{s^{n_a} + a_1 s^{n_a-1} + \cdots a_{n_a}} \ , \ n_a \geq n_b \tag{7.2}$$

Comme dans le chapitre précédent, la sortie du système $s_k \in \mathbb{R}$ est supposée être centrée et normalisée : $-s_{bas} = s_{haut} = 1$. La non-linéarité statique $\mathbf{Q}_C(.)$ est alors caractérisée par :

$$\mathbf{Q}_C(x) = \begin{cases} 1 & \text{si} \quad x - C \geq 0 \\ -1 & \text{si} \quad x - C < 0 \end{cases} \tag{7.3}$$

L'entrée et la sortie du système sont échantillonnées à des instants t_k. Il s'ensuit que les signaux échantillonnés d'entrée et de sortie disponibles sont donnés par :

$$\begin{cases} u_k = u(t = t_k) \\ s_k = \mathbf{Q}_C\left(y_k + e_k\right) \end{cases} \tag{7.4}$$

où $y_k = y(t = t_k)$ et e_k représente le terme de bruit. Cette modélisation nous permet de définir le problème d'identification des systèmes linéaires continus couplé avec un capteur binaire de la manière suivante :

Problème d'identification : *Étant donné un ensemble de N observations échantillonnées $\{(u_k, s_k)\}_{k=1}^{N}$ et les ordres du système n_a et n_b, proposer un algorithme d'identification permettant d'estimer les paramètres de la fonction de transfert $G(s)$ qui décrit le système.*

Afin de compléter la description du problème d'identification considéré, nous supposons que le seuil du capteur binaire est connu et différent de zéro. Ceci afin de garantir l'unicité de la solution (cf. chapitre 6). Le terme de bruit $\{e_k\}$ est supposé être un bruit blanc de moyenne nulle, non corrélée avec la séquence d'entrée.

7.3 Algorithme d'identification

L'algorithme d'identification proposé est composé de deux étapes : la première étape permet de reconstruire un signal de sortie à haute résolution, la seconde étape est consacrée à l'estimation des paramètres de la fonction de transfert.

-Reconstruction du signal de sortie haute résolution

L'objectif de cette première étape est de reconstruire un signal de sortie à haute résolution. L'idée ici consiste à utiliser une séquence d'entrée $u(t)$ spécifique, soit la somme de p sinusoïdes comme suit :

$$u(t) = \sum_{i=1}^{p} \mu_i sin(\omega_i t) \tag{7.5}$$

où $\{\omega_i\}_{i=1}^{p}$ sont les pulsations angulaires et μ_i sont les amplitudes choisies par l'utilisateur. Une phase peut être ajoutée si cela est jugé souhaitable. D'une manière générale, nous recommandons une distribution logarithmique des pulsations angulaires dans la bande passante du système dans le cas où cette dernière est connue. Cela devrait assurer une richesse suffisante de la base de données d'apprentissage et par conséquent satisfaire la condition d'excitation persistante.

Du fait que le système est linéaire, $y(t)$ peut également être écrit comme une somme de p sinusoïdes comme suit :

$$y(t) = \sum_{i=1}^{p} \alpha_i sin(\omega_i t) + \sum_{i=1}^{p} \beta_i cos(\omega_i t) \tag{7.6}$$

Ceci permet d'obtenir la forme suivante de régression linéaire :

$$y(t) = \varphi(t)^T \eta \tag{7.7}$$

avec

$$\eta = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_p \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} \qquad \varphi(t) = \begin{pmatrix} sin(\omega_1 t) \\ \vdots \\ sin(\omega_p t) \\ cos(\omega_1 t) \\ \vdots \\ cos(\omega_p t) \end{pmatrix} \tag{7.8}$$

Ensuite, il résulte de (7.7) que :

$$y_k = \varphi(t_k)^T \eta \tag{7.9}$$

En utilisant (7.4) et (7.9), s_k peut être réécrit comme suit :

$$s_k = \mathbf{Q}_C \left(\varphi(t_k)^T \eta + e_k \right) \tag{7.10}$$

A partir de (7.3), il en résulte que η est tel que $\forall k \in \{1, 2, \ldots, N\}$

$$s_k \left(\varphi(t_k)^T \eta + e_k - C \right) > 0 \tag{7.11}$$

Nous avons déjà vu ce type de contrainte dans le chapitre 6. En suivant le même raisonnement que celui utilisé dans le chapitre 6, le problème d'estimation de η correspond à un problème de classification supervisée de données en deux classes. Ce problème peut se reformuler de la manière suivante : étant donné un ensemble de points d'apprentissage

$\{\varphi(t_k)\}_{k=1}^{k=N}$ avec leurs labels correspondants $\{s_k\}_{k=1}^{N}$, trouver l'hyperplan paramétré par (η, C) qui sépare les points correspondant à la classe $s_k = s_{haut}$ de points correspondant à la classe $s_k = -s_{haut}$. Plus formellement, le problème d'estimation de η correspond au problème primal des SVMs :

$$\begin{cases} \min\limits_{\widehat{\eta},\widehat{C},v} \frac{1}{2} \parallel \widehat{\eta} \parallel^2 + \gamma \sum\limits_{k=1}^{N} v_k \\ \text{S.c} \quad s_k \left(\varphi(t_k)^T \widehat{\eta} - \widehat{C} \right) \geq 1 - v_k \text{ et } v_k \geq 0 \ ; \ \forall \ k \end{cases} \tag{7.12}$$

où $v_k = s_k \widehat{e_k}$ avec $\widehat{e_k}$ représente l'effet du bruit sur la sortie du système (cf. chapitre 6). À noter, la résolution de ce problème ne nécessite pas d'hypothèse sur la régularité de l'échantillonnage. Le cas de données irrégulièrement échantillonnées peut être traité de la même manière que le cas de données régulièrement échantillonnées.

L'utilisation des SVMs (de la même manière que dans le chapitre 6) permet de fournir une estimation $(\widehat{\eta}, \widehat{C})$ du couple (η, C) avec un facteur multiplicatif près. Ce facteur multiplicatif peut être compensé par une normalisation utilisant la valeur de C supposée connue et différente de zéro. Une fois qu'une estimation $\widehat{\eta}$ est disponible, il ne reste plus qu'à reconstruire le signal de sortie haute résolution échantillonné comme suit :

$$\widehat{y_k} = \varphi(t_k)^T \widehat{\eta} \tag{7.13}$$

$\widehat{y_k}$ désigne l'estimation de y_k.

- Identification du système à temps continu

L'objectif de la deuxième étape est de réaliser l'identification de $G(s)$. Tout d'abord, il est à noter que la séquence d'entrée générée selon (7.5) possède la propriété d'excitation persistante d'ordre q si les pulsations angulaires $\{\omega_i\}_{i=1}^{p}$ sont distinctes ([126]).
Il en résulte que cette séquence d'excitation est adaptée pour l'identification du système si $p \geq n_a + n_b + 1$.

Si le signal y_k était connu, il serait possible d'identifier directement $G(s)$ en utilisant des approches d'identification à temps continu disponible dans la littérature. Dans notre problème d'identification actuel, y_k est inconnu, la seule sortie du système disponible est s_k et la connaissance de s_k n'est pas suffisante pour réaliser l'identification du système.

La première étape de l'algorithme nous permet de surmonter cette difficulté en remplaçant le signal échantillonné inconnu y_k par le signal reconstruit $\widehat{y}_k$. Il suffit alors, d'appliquer l'un des algorithmes usuels d'identification des systèmes à temps continu disponibles dans la littérature en utilisant les N données échantillonnées $\{u_k, \widehat{y}_k\}$ afin d'identifier $G(s)$.

7.4 Simulations et résultats

Afin d'illustrer les performances de la procédure proposée, nous appliquons l'algorithme sur le système test de Rao-Garnier ([62]) suivant :

$$G(s) = \frac{-6400s + 1600}{s^4 + 5s^3 + 408s^2 + 416s + 1600} \tag{7.14}$$

Les données numériques ont été générées selon (7.1), (7.2), (7.4) et (7.3) où le seuil est choisi $C = 0.1$ et $s_{haut} = 1$. Le système est échantillonné avec une fréquence d'échantillonnage de $100Hz$. Le terme de bruit e_k est un bruit blanc gaussien de moyenne nulle et son amplitude est ajustée pour avoir différents niveaux de SNR. À noter que pour la mise en œuvre de l'algorithme, nous avons utilisé les fonctions disponibles dans la boîte à outils statistiques et apprentissage ("Statistics and Machine Learning toolbox") et la boîte à outils système Identification ("System Identification toolbox") fournies par Matlab.

La séquence d'entrée a été générée selon (7.5) avec $\{\omega_i\}_{i=1}^{p}$ un ensemble de $p = 10$ pulsations angulaires uniformément réparties sur $[0.1\ rad/s; 100\ rad/s]$. La première étape de l'algorithme permet de reconstruire un signal de sortie à haute résolution. La figure 7.2 illustre l'évolution temporelle de la sortie réelle du système $y_k + e_k$ et la sortie reconstruite $\widehat{y}_k$ pour une seule expérience, un nombre de données $N = 5000$ et un $SNR = 20dB$. Cette figure montre l'efficacité de la première étape dans la reconstruction du signal à haute résolution.

Afin d'étudier l'influence du nombre de données N sur les performances de l'algorithme, une simulation Monte-Carlo avec 100 jeux de données pour un $SNR = 20dB$ a été réalisée. L'algorithme a été appliqué pour différentes valeurs de N (N varie de 1000 jusqu'à 5000).

La figure 7.3 présente la moyenne (ligne complète) et l'écart type (lignes pointillées) de l'erreur normalisée d'estimation sur les paramètres ($NPEE$: Normalized Parameter Estimation Error) définie par :

$$NPEE = \frac{\|\theta - \widehat{\theta}\|}{\|\theta\|} \tag{7.15}$$

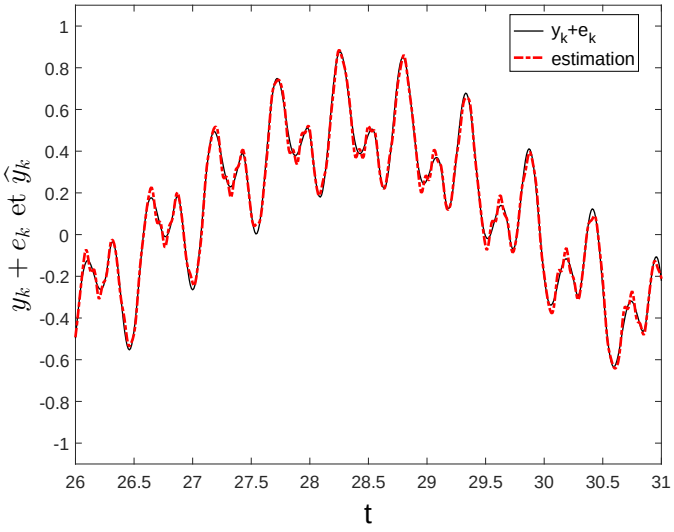

FIGURE 7.2 – y_k et son estimation pour un $SNR = 20dB$.

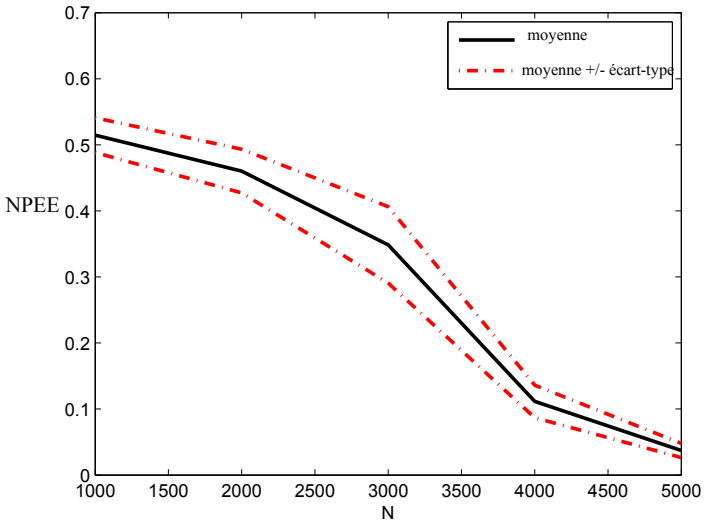

FIGURE 7.3 – La moyenne et l'écart type de $NPEE$ en fonction de N pour $SNR = 20dB$.

Les résultats montrent les bonnes performances de l'algorithme proposé, plus N est grand, plus la moyenne et l'écart type de $NPEE$ sont faibles.

Dans un second temps, afin d'observer l'impact du niveau de bruit sur la qualité d'estimation, une simulation Monte-Carlo avec 100 jeux de données ($N = 5000$) a été réalisée pour différentes valeurs de SNR. L'algorithme a été appliqué pour les niveaux de SNR suivants : $5dB$, $10dB$, $15dB$ et $20dB$.

La figure 7.4 présente la moyenne (ligne complète) et l'écart type (lignes pointillées) de $NPEE$. Il apparaît clairement qu'une valeur suffisamment élevée de SNR permet d'avoir une moyenne et un écart type de $NPEE$ faibles. Les diagrammes de Bode de la fonction de transfert réelle et les 100 modèles estimés pour un $SNR = 5dB$ sont illustrés sur la figure 7.5. L'algorithme proposé conduit à des résultats non biaisés avec des écarts-types raisonnables.

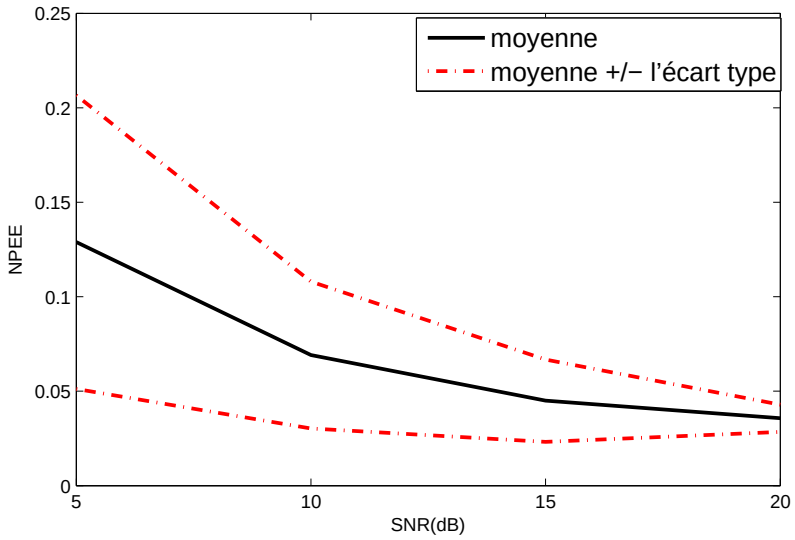

FIGURE 7.4 – La moyenne et l'ecart type de $NPEE$ en fonction de SNR pour $N = 5000$.

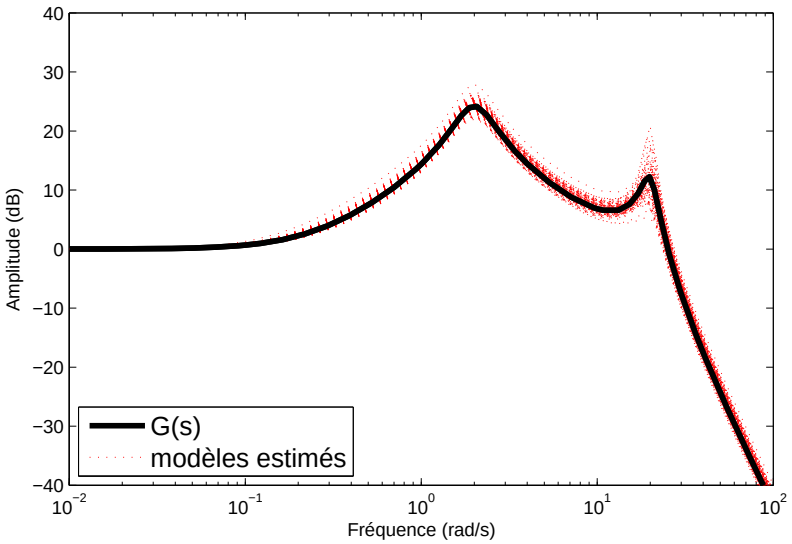

FIGURE 7.5 – La fonction de transfert (ligne complète) et ses estimations sur 100 expériences (lignes pointillées) pour $N = 5000$ et $SNR = 5dB$.

7.5 Conclusion

Dans ce chapitre, nous avons proposé une méthode pour l 'identification des systèmes continus ayant une sortie binaire. La méthode est composée de deux étapes. La première étape assure une reconstruction consistante du signal de sortie du système modulo une séquence d'entrée spécifique (somme de sinusoïdes). La seconde étape réalise l'estimation des paramètres du système via l'un des algorithmes usuels d'identification des systèmes continus disponibles dans la littérature. Des résultats de simulation ont montré que la solution proposée permet d'estimer les paramètres du modèle avec précision même en présence de bruit.

Troisième partie

De la classification à l'extraction de motifs

Chapitre 8

Un aperçu sur les algorithmes d'extraction de motifs

Le problème de la découverte de motifs à partir d'une série de données temporelles est un problème classique de la fouille des données. Il s'agit d'un processus d'exploration et d'analyse dont le but est de trouver des structures originales et d'extraire des informations significatives. Dans ce chapitre, nous présentons d'une manière concise un état de l'art sur les algorithmes d'extraction de motifs disponibles dans la littérature.

Sommaire

8.1 Introduction et définitions

Le problème de l'extraction de motifs a reçu une attention remarquable ces dernières années et suscite actuellement une importante activité de recherche. L'extraction de motifs peut être réalisée sur des séries temporelles ([205], [128]), des documents texte ([210], [110]), des graphes [114] ou des séquences vidéo [144]. Ses domaines d'application sont très variés, e.g., la médecine, la robotique, le traitement d'image, l'analyse de données ou encore la détection d'anomalies ([56], [128], [132], [137], [140], [111] [139], [40], [145]). Cette partie de la thèse s'inscrit dans le cadre du projet CASSIE dont l'objectif, qui a été défini conjointement avec les différents partenaires, est l'extraction de motifs à partir d'une série de données temporelles. Ainsi, les méthodes d'extraction de motifs à partir de document texte, image, vidéo ou autre ne sont pas détaillées ici.

Dans le cadre du projet CASSIE, une série temporelle représente un phénomène ou une série d'évènements ; elle résulte d'un processus de collecte d'informations ou de mesures. Cette collecte peut être réalisée d'une manière régulière selon une fréquence d'échantillonnage donnée [54]. Nous nous focalisons sur les séries temporelles monovariables à valeur réelle pour une question de simplicité d'écriture et de présentation. Nous ne considérons pas les cas des séries temporelles à valeur symbolique, des séries temporelles multivariables, de plusieurs séries couvrant simultanément plusieurs dimensions dans la même plage de temps. Dans la suite nous utilisons la définition suivante :

Définition 9. *Une série temporelle X est une suite ordonnée de N variables réelles :*

$$X = \{x_1, \cdots, x_N\} \qquad x_k \in \mathbb{R}$$

Une série temporelle est donc définie comme un ensemble de mesures effectuées à des instants de temps contigus. Des données météorologiques, financières, de consommation domestique, de suivi de populations sont des exemples des séries temporelles considérées ici. Dans le cadre de la recherche de motifs au sein d'une série temporelle nous ne nous intéressons pas aux propriétés globales de la série mais plutôt aux relations entre les sous-séquences de la série.

Définition 10. *Soit X une série temporelle de taille N. Une sous-séquence S de X est une suite de taille $m \leq N$ constituée d'éléments de X à des instants de temps contigus*

$$S = \{x_{k-m+1}, \cdots, x_k\}$$

avec $1 + m \leq k \leq N$.

L'extraction de motifs à partir d'une série temporelle consiste à trouver toutes les sous-séquences qui apparaissent de manières récurrente à des instants distincts sur la série temporelle. Plus formellement, un motif peut être défini comme suit :

Définition 11. *Soit une série temporelle X. Toute sous-séquence de X qui se produit à plusieurs reprises dans la série temporelle correspond à un motif.*

La figure 8.1 illustre cette définition et donne un exemple de la présence d'un motif dans une série temporelle.

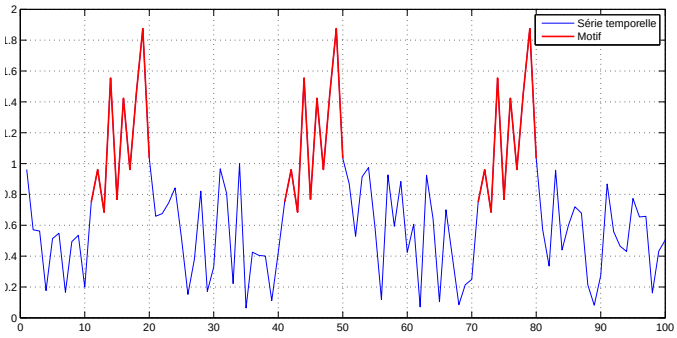

FIGURE 8.1 – Illustration de la présence d'un motif dans une série temporelle

Un même motif peut apparaître à différents instants sur la série temporelle. L'ensemble des sous-séquences correspondant à ce motif est appelée une classe de motif.

Définition 12. *Soit $S^{(1)}$, $S^{(2)}$, ...$S^{(p)}$, p sous-séquences de X correspond à un même motif. La classe de motif est désignée par*

$$C = \{S^{(1)}, \cdots, S^{(p)}\}$$

Du fait de la présence de bruit lors de la collecte d'informations, ou tout simplement parce que le phénomène observé présente certaines irrégularités, il est probable que des sous-séquences correspondant à un même motif ne soient pas identiques. Afin de prendre en compte ces imperfections lors de l'extraction de motifs il est nécessaire d'utiliser les notions de distance entre sous-séquences et de distance entre classes de motif.

- La distance entre deux sous-séquences $S^{(1)}$ et $S^{(2)}$ est notée $d_{S^{(1)},S^{(2)}}$. Le calcul de la distance entre deux sous-séquences permet d'estimer le degré de similarité entre les deux sous-séquences. Si la distance $d_{S^{(1)},S^{(2)}}$ est faible alors les deux sous-séquences $S^{(1)}$ et $S^{(2)}$ sont considérées comme similaires. Elles représentent alors le même motif et peuvent être agrégées au sein de la même classe.
- La distance entre deux classes $C^{(1)}$ et $C^{(2)}$ est notée $D_{C^{(1)},C^{(2)}}$. Le calcul de la distance entre deux classes permet d'estimer le degré de similarité entre les deux classes. Si la distance $D_{C^{(1)},C^{(2)}}$ est faible alors les deux classes représentent le même motif et peuvent être agrégées au sein de la même classe.

Différentes distances, entre sous-séquences et entre classes, sont proposées dans la littérature. Le choix de telle ou telle distance conditionne le résultat de la recherche de motif. Le choix de la distance entre classe oriente par exemple la morphologie des classes résultantes de l'analyse. Différentes distances seront discutées dans le chapitre suivant.

A partir de ce qui précède, nous caractérisons à présent un motif de la manière suivante :

- Un motif est caractérisé par une classe de motifs ;
- Cette classe est constituée de sous-séquences apparaissant à des instants distincts sur la série temporelle ;
- Chaque sous-séquence de cette classe est similaire à toute autre sous-séquence de la classe.

Nous définissons enfin les notions d'inertie intra-classes et d'inertie inter-classes permettant de caractériser les classes. Dans les définitions suivantes, $G_{c^{(j)}}$ est le centre de gravité de la classe $C^{(j)}$.

Définition 13. *L'inertie d'une classe $C^{(j)}$ (inertie intra-classe) correspond à la somme des carrées des distances au centre de la classe :*

$$I_{intra}(C^{(j)}) = \sum_{S^{(i)} \in C^{(j)}} d^2_{S^{(i)},G_{c^{(j)}}} \tag{8.1}$$

Définition 14. *Étant donné un ensemble de classes $\mathscr{C} = \{C^{(1)}, \cdots, C^{(q)}\}$, l'inertie inter-classes correspond à la somme des carrées des distances entre les centres des classes :*

$$I_{inter}(\mathscr{C}) = \sum_{k=2}^{q} \sum_{i<k} d^2_{G_{c^{(k)}},G_{c^{(i)}}} \tag{8.2}$$

8.2 Aperçu sur les algorithmes d'extraction de motifs disponibles

Différents algorithmes sont disponibles dans la littérature pour l'extraction de motifs. En examinant ces algorithmes, on peut constater que la plupart des travaux liés aux séries temporelles sont classés en deux catégories [212], [2] : les méthodes de recherche de motifs à partir d'un ensemble de séries temporelles (Whole time series clustering) et les méthodes de recherche de motifs à partir d'un ensemble de sous-séquences extraites via une fenêtre glissante à partir d'une seule série temporelle (Subsequence time series clustering).

Les méthodes d'extraction de motifs sur un ensemble de séries temporelles se trouvent vite limitées à cause du nombre important de données dans chaque série et des dimensions des séries temporelles. Une solution proposée par ces méthodes est de représenter les données des séries temporelles dans un autre espace de fonctionnalité de dimension inférieure. L'objectif de cette réduction de dimension est de réduire les besoins en mémoire et réduire d'une manière significative la complexité de calcul. Ainsi, dans [195], les données des séries temporelles sont représentées à l'aide d'une transformée en ondelettes de Haar ([176], [174]). Dans [130] les données des séries temporelles sont représentées à travers un ensemble de coefficients cepstrales ([127]) afin de faire le regroupement dans le domaine fréquentiel. Dans [138], les données des séries temporelles sont représentées par des fonctions linéaires par morceaux. Dans [123], [106], [124] et [125] une représentation symbolique des données a été utilisée. Une telle représentation permet de profiter de la richesse des structures de données et des algorithmes d'extraction de motifs à partir de document texte disponibles dans la littérature. Certains algorithmes, tels que ceux présentés dans [25] et [38], utilisent une projection aléatoire des données après avoir symbolisé les séries temporelles. A noter, en raison de la nature probabiliste de ces algorithmes, qu'il n'est pas garanti de trouver l'ensemble des motifs ([54]). Plusieurs autres représentations sont disponibles dans la littérature ([123], [54] et [125]). Nous ne détaillons pas ces représentations ici, parce que notre objectif dans cette partie est l'extraction de motifs à partir d'un ensemble de sous-séquences d'une seule série temporelle.

Les méthodes d'extraction de motifs sur un ensemble de sous-séquences sont effectuées ici sur une seule série temporelle [105]. Trois aspects permettent de caractériser ces méthodes comme l'illustre la figure 8.2 : le choix d'une mesure de similarité, le choix d'un modèle de classe et le choix d'un algorithme de classification.

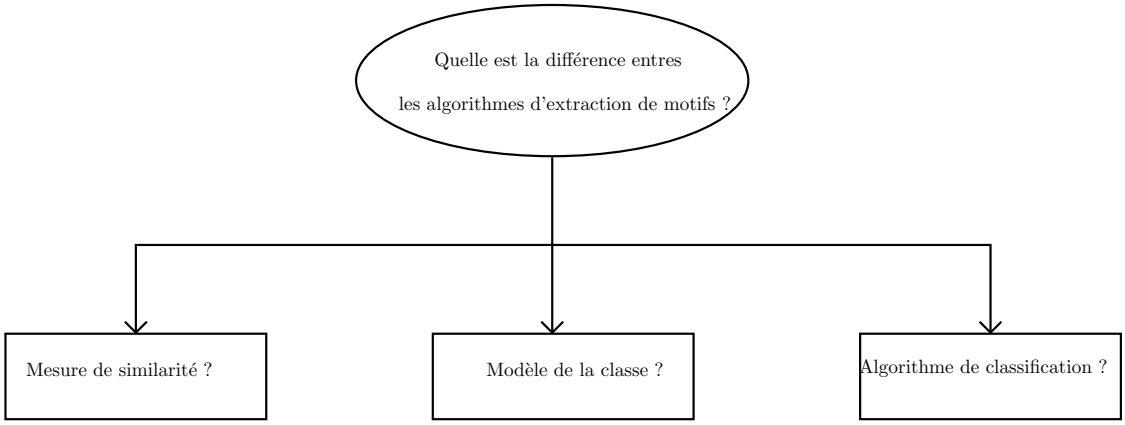

FIGURE 8.2 – Trois aspects caractérisent les algorithmes d'extraction de motifs

Les algorithmes d'extraction de motifs reposent sur une mesure de similarité entre les sous-séquences. Le choix d'une mesure de similarité conditionne d'une manière importante les performances des algorithmes d'extraction de motifs. Un choix approprié dépend de la caractéristique de la série temporelle, la longueur de la série et bien sûr de l'objectif attendu. Différentes mesures de similarité peuvent être utilisées [3],[55]. Les plus utilisées sont : la distance euclidienne et la distance DTW (Dynamic Time Warping). La distance Euclidienne est largement utilisée comme un outil de mesure de similarité entre des sous-séquences de même longueur dans les algorithmes d'extraction de motifs ([3],[55], [195], [49] [131], [143]). La distance DTW est utilisée dans plusieurs algorithmes comme un outil de mesure de similarité entre des sous-séquences de longueurs différentes ([209], [199], [90], [33]). Ces mesures de similarité et d'autres mesures sont détaillées dans le prochain chapitre.

Un autre aspect qui permet de différencier les algorithmes d'extraction de motifs entre eux est le choix d'un modèle pour une classe de sous-séquences. Trouver un modèle de classe ou un représentant de classe est une étape essentielle dans les algorithmes d'extraction de motifs. En effet, la qualité des classes dépend de la qualité des modèles de classes. Plusieurs approches sont utilisées pour définir les modèles des classes : certains algorithmes utilisent la moyenne de toutes les sous-séquences de la classe comme étant le modèle de la classe ([107], [43] et [153]). D'autres méthodes considèrent la medoïde de la classe comme étant le modèle de la classe ([86], [103] et [197]). La medoïde d'une classe est la sous-séquence de la classe présentant la distance minimale avec l'ensemble des autres

sous-séquences de la classe. La figure 8.3 illustre la medoïde et la moyenne d'un ensemble de sous-séquences.

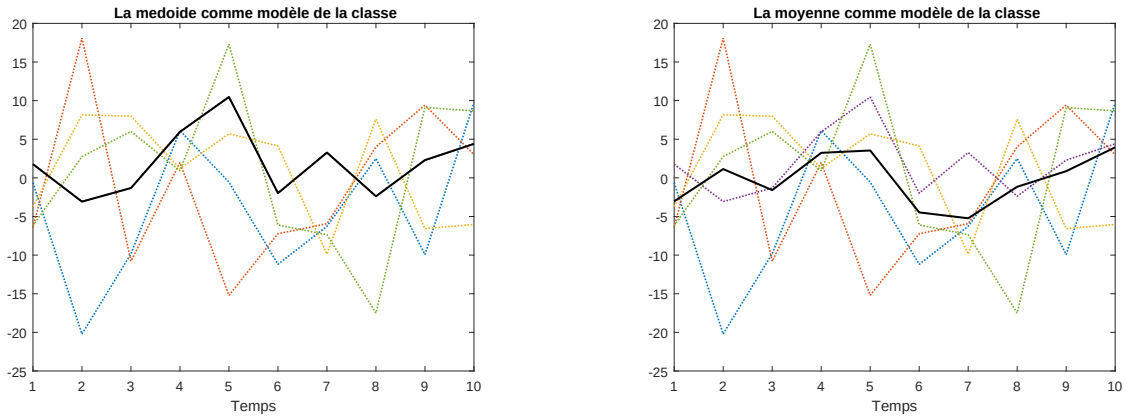

FIGURE 8.3 – Exemples de modèles des classes

Outre la mesure de similarité et le modèle des classes, la majorité des algorithmes d'extraction de motifs reposent sur des méthodes de classification afin de regrouper des sous-séquences (ou des séries) similaires au sein des classes de telle manière que les sous-séquences d'une même classe soient aussi proches les unes des autres que possible. Deux types de classification sont disponibles : la classification supervisée et la classification non supervisée. Il ne s'agit pas ici d'être exhaustif mais de présenter différents éléments permettant de nous orienter dans les choix techniques.

- L'objectif de la classification supervisée est de trouver des règles permettant de classer des sous-séquences dans des classes déjà identifiées et connues. Par la suite, ces règles permettent de répartir de nouvelles sous-séquences au sein des classes. Plusieurs méthodes de classification supervisée sont disponibles dans la littérature. Les plus utilisées sont : les réseaux de neurones [87], les arbres de décision [162] et les SVMs présentés dans le chapitre 5. Parmi les algorithmes d'extraction de motifs basés sur des méthodes de classification supervisée nous pouvons citer : [113], [115], [112]. Ces algorithmes supposent la connaissance a priori du nombre de motifs et de sous-séquences appartenant à ces motifs.

- La classification non supervisée a pour objectif de structurer et regrouper au sein d'une même classe des sous-séquences qui présentent des caractéristiques communes. Ces méthodes de classification non supervisée reposent sur une mesure de similarité qui reflète la ressemblance entre les sous-séquences. L'idée est de grouper les

sous-séquences dans des classes homogènes selon certains critères tout en favorisant l'hétérogénéité entre ces différentes classes. Plusieurs méthodes de classification non supervisée sont disponibles dans la littérature. Les plus utilisées sont : les méthodes de partitionnement [94] et les méthodes hiérarchiques [103]. Parmi les algorithmes d'extraction de motifs reposant sur des techniques de classification non supervisée, nous citons : [53], [122], [204], [165], [90]. Certains de ces algorithmes supposent la connaissance a priori du nombre de classes, d'autres non.

En ce qui concerne le comportement des algorithmes d'extraction de motifs, la majorité des algorithmes sont entravés par les problèmes suivants [212] :

- Complexité de calcul : trois étapes sont nécessaires pour la majorité des algorithmes d'extraction de motifs rendant la complexité de calcul de ces algorithmes importante. Il s'agit de l'utilisation d'une fenêtre glissante pour extraire des sous-séquences à partir d'une série temporelle qui génère un nombre important de sous-séquences, la mesure de similarité entre chaque paire de sous-séquences et la nécessité d'appliquer une méthode de classification afin de regrouper les sous-séquences similaires dans des classes.
- Haute utilisation de la mémoire : la mémoire consommée lors de l'exécution d'un algorithme d'extraction de motifs est en général importante, ce qui réduit l'efficacité de l'algorithme.
- Résultat incertain : les algorithmes d'extraction de motifs basé sur la classification non supervisé ne sont généralement pas efficaces à cause de l'absence des connaissances à priori.
- Paramétrage des algorithmes : les algorithmes ont plusieurs paramètres de synthèse, e.g., le seuil sur le degré de similarité, le nombre de motifs et la longueur des motifs.

Pour un état de l'art détaillé sur les algorithmes d'extraction de motifs à partir des séries temporelles, le lecteur est invité se référer à ([212], [122], [60] et [2]).

8.3 Interprétation de résultats

Une étape essentielle dans un processus d'extraction de motifs est l'analyse et l'interprétation des résultats obtenus par les algorithmes d'extraction de motifs. L'intérêt de cette interprétation est de fournir des connaissances à l'utilisateur. L'interprétation des résultats peut être réalisée en utilisant des techniques de visualisation et de sémantisation

qui réalisent une combinaison entre les résultats de l'extraction de motifs et des interactactions avec l'utilisateur. Ceci permet de considérer ces motifs identifiés comme des connaissances utiles [104]. Dans certains cas, un traitement a posteriori est nécessaire afin d'affiner les résultats et les rendre plus interprétables. Dans le cadre du projet CASSIE, cette étape d'interprétation et de sémantisation de résultats est à la charge des autres partenaires du projet. De ce fait, nous ne détaillons pas cette étape dans ce manuscrit.

8.4 Conclusion

Dans ce chapitre nous avons présenté un bref aperçu sur les algorithmes d'extraction de motifs à partir d'une série temporelle. Cette extraction de motifs est génériquement réalisée via la formation de classes de sous-séquences, chaque classe étant constituée de sous-séquences similaires et chaque classe peut caractériser un motif présent sur la série temporelle. Les algorithmes d'extraction de motifs sont basés sur un choix de mesure de similarité, un choix de modèle pour chaque classe et un choix d'une méthode de classification afin de regrouper les sous-séquences similaires. Ces choix sont conditionnés par les contraintes du problème pour lequel l'extraction de motifs est requise.

Notre rôle dans le cadre du projet CASSIE consiste à développer des algorithmes de traitement de données temporelles d'une manière non supervisée. Ceci nous oriente vers la proposition d'un algorithme d'extraction de motifs basé sur des techniques de la classification non supervisée. Le chapitre suivant présente différentes options possibles en terme de classification non supervisée, en terme de mesure de similarité et en terme de méthode d'agrégation.

Chapitre 9

Notions de classification non supervisée

La classification non supervisée est une technique d'analyse de données qui permet de structurer des données présentant des caractéristiques communes dans des classes homogènes sans connaissances a priori, en favorisant l'hétérogénéité entre ces différentes classes. Dans ce chapitre, nous présentons quelques algorithmes de classification non supervisée parmi les plus répandus dans la littérature. Ce tour d'horizon va nous permettre de faire les choix techniques dans la perspective de l'extraction de motifs.

Sommaire

9.1 Introduction

La classification non supervisée est une procédure d'analyse de données. Dans le cadre de l'extraction de motifs, le contexte est le suivant : on suppose disposer des sous-séquences issues d'une série temporelle et certaines de ces sous-séquences correspondent à un même motif. Puisque les classes d'appartenances des sous-séquences ne sont pas connues a priori, on utilisera la classification non supervisée pour former ces classes de motif.

Les méthodes de classification non supervisée sont généralement réparties en trois catégories : les méthodes de partitionnement [94], les méthodes hiérarchique [103] et les méthodes probabilistes [47]. Étant donné que les méthodes probabilistes supposent que les sous-séquences sont issues d'une loi normale, nous ne nous intéressons pas à ces méthodes dans ce manuscrit. Nous utiliserons les méthodes de partitionnement et les méthodes hiérarchiques. Au sein de ces deux catégories, on peut considérer différentes variantes selon les choix de la distance entre sous-séquences et la distance entre classes. En dépit de cette diversité de méthodes, un objectif commun est de garantir une grande similarité intra-classes (minimiser l'inertie intra-classe)et une faible similarité inter-classes (maximiser l'inertie inter-classes).

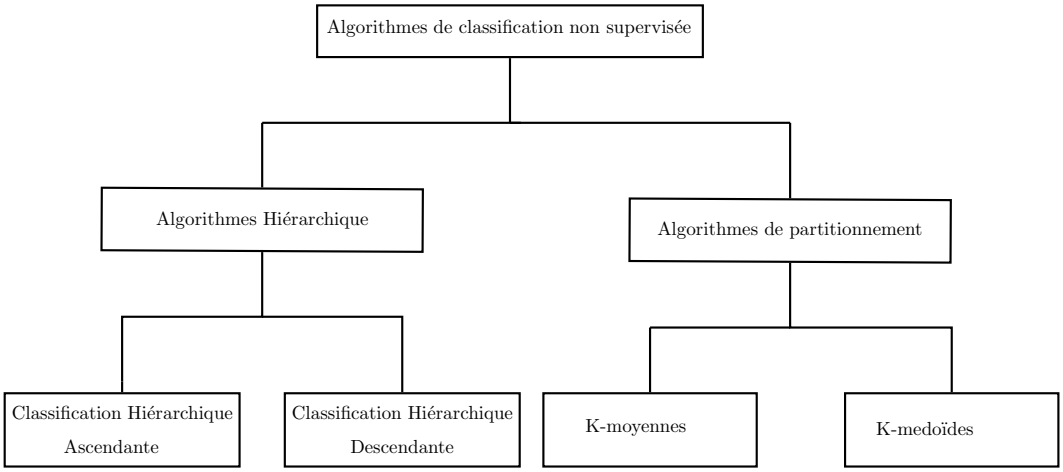

FIGURE 9.1 – Deux grandes familles de la classification non supervisée

L'intérêt de telle ou telle méthode dépend de la nature du problème considéré et des contraintes de mise en oeuvre. Nous adopterons les critères de sélection suivants :

- La connaissance ou non a priori du nombre de classes ;
- Le volume et la nature des données à traiter ;

– La possibilité donnée à l'utilisateur de fixer un seuil de similarité entre sous-séquences d'une classe ;
– Les capacités de calcul.

9.2 Classification par partitionnement

Les méthodes de classification par partitionnement proposent une répartition de l'ensemble des sous-séquences en K classes, le nombre de classes K doit ainsi être fixé a priori. Chaque classe doit contenir au moins une sous-séquence, et chaque sous-séquence doit appartenir à une classe unique.

Une première approche pourrait consister à construire toutes les répartitions possibles et ne garder que la meilleure répartition à l'issue d'une phase d'estimation de la qualité de la classification. Le problème est que le nombre de répartitions possibles croit exponentiellement en fonction de K et du nombre de sous-séquences, il n'est donc pas possible de mettre en oeuvre une telle approche. Les solutions enumérées dans la suite ne sont que des solutions sous-optimales qui ont le mérite de pouvoir être implémentées.

Parmi les méthodes de classification par partitionnement communément utilisées dans la littérature, nous trouvons l'algorithme K-moyennes (appelé aussi algorithme des centres mobiles ou algorithme K-means) constitué des étapes suivantes :

1- Initialisation : désignation de K sous-séquences comme centres de classes d'une manière aléatoire.

2 - Pour chaque sous-séquence
  – Calcul de la distance avec les K centres de classes.
  – Affectation à la classe pour laquelle la distance au centre est la plus faible.

3 - Une fois toutes les sous-séquences réparties dans K classes, calcul d'un nouveau centre pour chaque classe.

  Retour à l'étape 2 pour plusieurs itérations du processus de répartition. Le cycle peut être répété jusqu'à la convergence ou pour un nombre d'itérations maximum fixé.

La figure 9.2 illustre le processus de partitionnement de 11 points en 3 classes en utilisant l'algorithme K-moyennes.

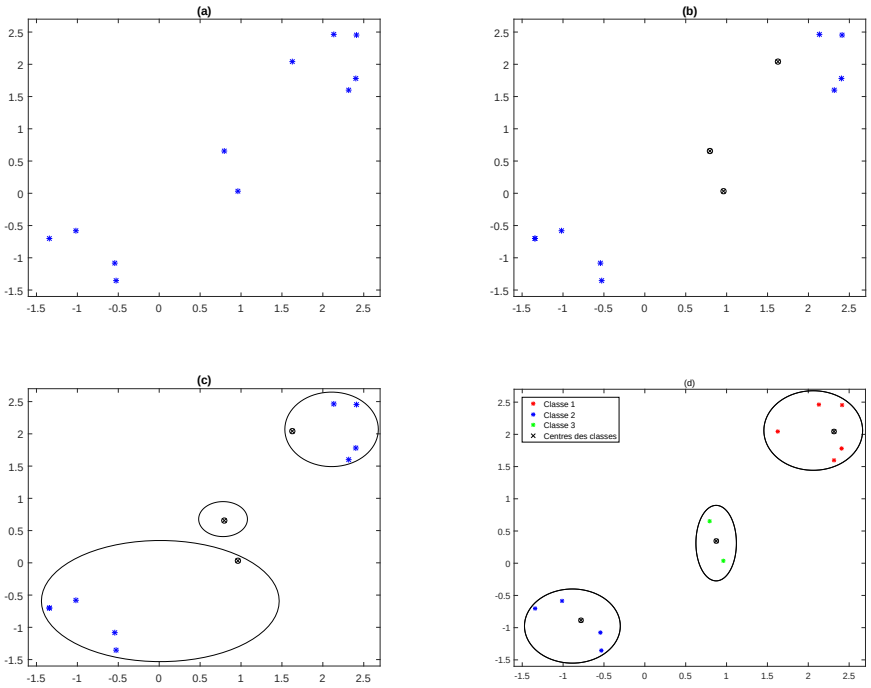

FIGURE 9.2 – (a) partition des points dans l'espace (b) désignation de 3 points comme centres de classes d'une manière aléatoire (c) calcul de la distance avec les 3 centres de classes et affectation des points à la classe pour laquelle la distance au centre est la plus faible. (4) classification finale après plusieurs itérations du processus de répartition.

Différentes variantes sont possibles selon le mode de calcul de la distance entre sous-séquences et le mode de calcul du centre d'une classe. La distance utilisée usuellement est la distance euclidienne et le centre d'une classe est le barycentre de la classe. Une variante de l'algorithme K-moyennes est l'algorithme K-medoïdes qui consiste à affecter comme centre d'une classe la médoïde de la classe. L'intérêt étant que la médoïde d'une classe soit la sous-séquence possédant la dissimilarité moyenne la plus faible avec les autres sous-séquences de la classe. Une autre variante de l'algorithme K-moyenne est l'algorithme K-moyenne flou. Le principe ici est qu'une même sous-séquence peut appartenir à plusieurs classes. Ceci est réalisé via l'association à chaque sous-séquences de K pondérations comprises entre 0 et 1 et caractérisant le degré d'appartenance (pondération proche de 1) ou de non-appartenance (pondération proche de 0) à une classe. Plus de détails et des exemples sur les algorithmes de partitionnement se trouvent dans les publications : [95], [132], [93], [81], [1].

La charge de calcul de ces algorithmes est relativement faible (complexité en $\mathcal{O}(lKs)$ où l est le nombre d'itération et s le nombre de sous-séquences). Néanmoins, on distingue deux faits qui constituent les principales limitations de ces algorithmes dans le contexte de l'extraction de motifs dans une série temporelle. Le premier fait concerne la convergence globale qui n'est pas garantie au point où des initialisations différentes conduisent à des classes différentes. Le second fait relève de la connaissance requise a priori sur le nombre de classes.

9.3 Classification hiérarchique

Les méthodes de classification hiérarchique proposent la construction d'un arbre hiérarchique (appelé dendrogramme) établissant les liens de proximité, de proche en proche, entre les sous-séquences. On distingue deux approches de classification pour parvenir à un tel arbre hiérarchique : la Classification Ascendante Hiérarchique (CAH) et la Classification Descendante Hiérarchique (CDH).

La CAH permet de construire une hiérarchie des sous-séquences sous la forme d'un arbre dans un ordre ascendant comme suit :

1 - Initialisation : chaque sous-séquence est considérée comme une classe.

2 - Pour chaque classe calcul de la distance avec les autres classes.

3 - Agrégation des deux classes les plus proches, donc formation d'une nouvelle classe intégrant ces 2 classes les plus proches.
Retour à l'étape 2 jusqu'à aggrégation de toutes les sous-séquences dans une seule et même classe.

La CDH permet de construire une hiérarchie des sous-séquences sous forme d'un arbre dans un ordre descendant comme suit :

1 - Initialisation : création d'une classe unique regroupant toutes les sous-séquences.

2 - Pour chaque classe calcul de l'inertie intra-classe.

3 - Subdivision de la classe en 2 classes avec la plus petite inertie intra-classe.
Retour à l'étape 2 jusqu'à ce que chaque sous-séquence soit considérée comme une classe.

Pour chacune de ces deux approches, à chaque itération il est possible de connaître le nombre de classes et leur inertie intra-classe. Il est ainsi possible d'arrêter le processus si le nombre de classes souhaité est atteint ou si un seuil fixé par avance sur la hauteur de l'arbre hiérarchique est dépassé (la hauteur traduisant la distance entre les classes). L'arbre hiérarchique n'est pas nécessairement construit totalement.

La figure 9.3 illustre la manière de construire l'arbre hiérarchique dans les deux sens (CAH et CDH). Elle représente le dendrogramme qui établit le lien entre les sous-séquences (numérotées ici de 1 à 11 en abscisse), leur proximité relative (échelle en ordonnée) et la fusion des classes. Sur cet exemple un seuil permet de couper l'arbre hiérarchique pour obtenir des classes homogènes.

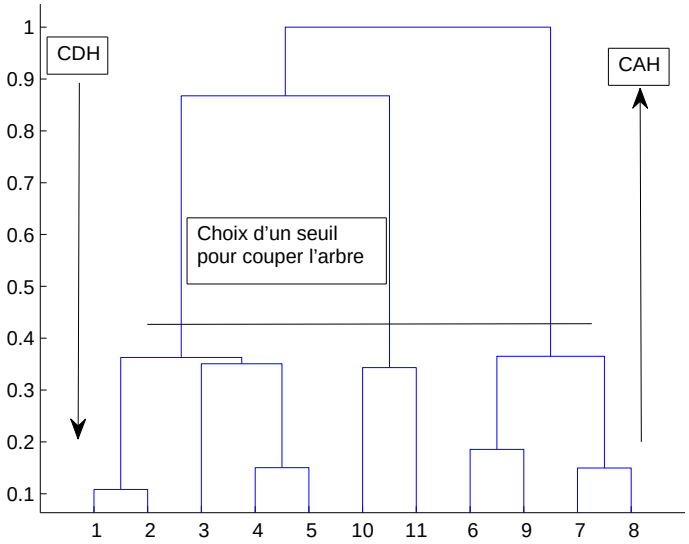

FIGURE 9.3 – Principe de la classification hiérarchique (les données sont les mêmes que celles utilisées pour la figure 9.2)

Les méthodes de classification hiérarchique présentent les caractéristiques suivantes :

- La possibilité de fixer un seuil pour couper l'arbre hiérarchique d'une manière automatique en assurant une similarité entre les sous-séquences d'une même classe.
- La possibilité de visualiser l'arbre hiérarchique afin de fixer le nombre optimal de classes.
- L'implémentation aisée de l'algorithme.

Ici aussi différentes variantes sont possibles selon le mode de calcul de la distance entre sous-séquences et la distance entre classes. Différentes possibilités sont listées dans les sections suivantes.

Le principal avantage des méthodes de classification hiérarchique par rapport aux méthodes de classification par partitionnement concerne sa convergence. En effet, l'initialisation n'est pas aléatoire et l'exécution d'une méthode de classification hiérarchique sur le même jeu de données donnera toujours les mêmes résultats. Par ailleurs, la connaissance a priori du nombre de classes n'est plus indispensable. On notera toutefois que la charge de calcul augmente en fonction du nombre de données (complexité en $\mathcal{O}(s^3)$ où s est le nombre de sous-séquences).

Il a été proposé dans la littérature de combiner des méthodes de classification hiérarchique et des méthodes de classification par partitionnement. L'idée est de déterminer dans un premier temps une partition pour un nombre élevé préalablement fixé de classes (classification par partitionnement), et dans un second temps de permettre une éventuelle agrégation entre certaines classes conformément à un critère de proximité. En itérant cette procédure, il est possible de former un nombre réduit de classes homogènes [141].

9.4 Distance entre sous-séquences

Comme cela a été vu auparavant, la mise en oeuvre des méthodes de classification précédentes nécessite la définition d'une distance entre sous-séquences pour d'évaluer le degré de similarité entre les sous-séquences. Le choix d'une mesure de similarité a un impact significatif sur les résultats des méthodes de classification. En fonction du domaine d'application et de la nature des sous-séquences comparées, différentes mesures de similarité peuvent être utilisées [3],[55].

Les distances les plus utilisées pour des sous-séquences issues de séries temporelles peuvent être réparties en trois catégories [89] :

- Les distances basées sur les normes L_p : la distance de Minkowski (cas particulier : la distance euclidienne, distance de City block et la distance de Chebychev) et la distance de Mahalanobis ;
- Les distances statistiques : la distance par cosinus, la distance par corrélation et la distance de Spearman ;

– Les distances élastiques : la distance d'alignement temporel dynamique DTW (Dynamic Time Warping distance).

Une comparaison entre certaines mesures de similarité est rapportée dans [49]. Certaines de ces distances sont rappelées ci-dessous pour deux sous-séquences $S^{(1)}$ et $S^{(2)}$ de même longueur m. Ces sous-séquences sont représentées par des vecteurs colonne de même nom et de dimension m.

– **Distance de Minkowski**

$$d^{Mink}_{S^{(1)},S^{(2)}} = (\sum_{k=1}^{m} |S^{(1)}_k - S^{(2)}_k|^p)^{1/p} \tag{9.1}$$

Pour $p = 1$ la distance de Minkowski correspond à la distance city bloc, pour $p = 2$ elle correspond à la distance euclidienne, et pour $p = \infty$ elle correspond à la distance de Chebychev.
La distance Euclidienne est largement utilisée dans la littérature. Néanmoins, il convient de noter que ce type de distance est peu pratique dans la perspective de réaliser de la classification non-supervisée de manière automatique. En effet, cette distance n'est pas normalisée et son intervalle de valeur possible est conditionné par l'amplitude des échantillons de la série temporelle. Il n'est donc pas aisé de fixer par avance un seuil de similarité entre sous-séquences pour la formation des classes.

– **Distance par cosinus, par corrélation et de Spearman**

- Distance par cosinus :

$$d^{cosi}_{S^{(1)},S^{(2)}} = 1 - Cos_{S^{(1)},S^{(2)}} \tag{9.2}$$

où $Cos_{S^{(1)},S^{(2)}}$ est le cosinus

$$Cos_{S^{(1)},S^{(2)}} = \frac{\left(S^{(1)}\right)^T S^{(2)}}{\sqrt{\left(\left(S^{(1)}\right)^T S^{(1)}\right)\left(\left(S^{(2)}\right)^T S^{(2)}\right)}} \tag{9.3}$$

- Distance par corrélation :

$$d^{corr}_{S^{(1)},S^{(2)}} = d^{cosi}_{\overline{S^{(1)}},\overline{S^{(2)}}} \tag{9.4}$$

avec $\overline{S^{(i)}} = S^{(i)} - \frac{1}{m}\sum_{k=1}^{m} S^{(i)}_k$

- Distance de Spearman :

$$d^{spea}_{S^{(1)},S^{(2)}} = d^{corr}_{r_{S^{(1)}},r_{S^{(2)}}} \tag{9.5}$$

où $r_{S^{(i)}}$ est le vecteur des rangs des éléments de $S^{(i)}$ ordonnés.

A la différence de la distance de Minkowski, les 3 distances précédentes sont des distances normalisées. Le coefficient de corrélation normalisé $Cos_{S^{(1)},S^{(2)}}$ a en effet une valeur comprise entre -1 et 1. Une valeur comprise entre 0 et 1 caractérise une relation proportionnelle, une valeur comprise entre -1 et 0 caractérise une relation inversement proportionnelle. Les distances $d^{cosi}_{S^{(1)},S^{(2)}}$, $d^{corr}_{S^{(1)},S^{(2)}}$ et $d^{spea}_{S^{(1)},S^{(2)}}$ ont chacune leur valeur comprise entre 0 et 2. Cette normalisation permettra ultérieurement le choix d'un seuil de similarité entre sous-séquences. Selon [175], les performances de la distance par cosinus et par corrélation sont proches et elles sont significativement meilleures que celles de la distance euclidienne dans certaines applications.
Remarquons que la distance de Spearman consiste à déceler une corrélation, non pas entre les valeurs prises par les deux sous-séquences, mais entre les rangs de ces sous-séquences. Il est intéressant de l'utiliser lorsque les deux sous-séquences semblent déjà corrélées afin d'affiner la relation de corrélation.

– **Distance d'alignement temporel dynamique DTW "Dynamic Time Warping distance**
La particularité de la distance DTW est sa capacité à supporter un échantillonnage irrégulier ou à comparer des sous-séquences de longueurs différentes en cherchant l'alignement optimal entre les sous-séquences [108]. Considérons $S^{(1)}$ et $S^{(2)}$ deux sous-séquences de longueur m et n respectivement. Une matrice de dimension $(m \times n)$ est construite, à chaque entrée (i, j) de cette matrice, une distance locale $d_{S^{(1)}(i),S^{(2)}(j)}$ est calculée. Pour calculer la distance $d_{S^{(1)},S^{(2)}}$, il suffit alors de chercher le chemin dans cette matrice pour aller du point initial $(1,1)$ au point final (m,n) en progressant par voisinage de cellule en cellule de façon à minimiser la somme des distances locales rencontrées. La distance DTW peut se calculer en utilisant la récurrence suivante :

$$d_{S^{(1)},S^{(2)}}(i,j) = d_{S^{(1)}(i),S^{(2)}(j)} + min \begin{cases} d_{S^{(1)},S^{(2)}}(i-1,j-1) \\ d_{S^{(1)},S^{(2)}}(i,j-1) \\ d_{S^{(1)},S^{(2)}}(i-1,j) \end{cases} \tag{9.6}$$

où la distance globale entre les deux sous-séquences est $d^{DTW}_{S^{(1)},S^{(2)}} = d_{S^{(1)},S^{(2)}}(m,n)$, alors que $d_{S^{(1)}(i),S^{(2)}(j)}$ est la distance locale. Le calcul de la distance DTW est plus lent

que le calcul des distances L_p normes et les distances statistiques. Sa complexité de calcul est d'ordre quadratique $\mathscr{O}(m \times n)$ [32].

9.5 Distance entre classes

Il s'agit à présent de définir une distance entre classes. Ceci va permettre l'agrégation de classes afin d'assurer la formation de classes homogènes. La définition d'une distance entre classe revient à définir une mesure de proximité entre deux classes. Dans ce sens, différents critères ont été proposés dans la littérature ([16], [132], [81]). Les plus communs sont les suivants :

- **Le critère de saut minimum ("single linkage")**
  La distance entre deux classes $C^{(1)}$ et $C^{(2)}$ est déterminée par la plus courte distance séparant une sous-séquence de $C^{(1)}$ et une sous-séquence de $C^{(2)}$.

$$D_{C^{(1)},C^{(2)}} = \min_{S^{(1)} \in C^{(1)}, S^{(2)} \in C^{(2)}} d_{S^{(1)},S^{(2)}} \tag{9.7}$$

  Ce critère génère un effet de chaînage et permet l'agrégation dans une même classe de sous-séquences apparaissant sous forme de chaîne. A noter qu'il est possible (probable) que certaines sous-séquences des classes formées soient éloignées les unes des autres.

- **Le critère de saut maximum ("complete linkage")**
  La distance entre deux classes $C^{(1)}$ et $C^{(2)}$ est déterminée par la plus grande distance séparant une sous-séquence de $C^{(1)}$ et une sous-séquence de $C^{(2)}$.

$$D_{C^{(1)},C^{(2)}} = \max_{S^{(1)} \in C^{(1)}, S^{(2)} \in C^{(2)}} d_{S^{(1)},S^{(2)}} \tag{9.8}$$

  Ce critère génère des classes compactes et donne des résultats satisfaisants lorsque les sous-séquences forment déjà naturellement des "classes" bien distinctes. Si les classes ont plutôt une forme allongée, ou sont en forme de "chaîne", cette distance n'est pas adaptée.

- **Le critère de la moyenne ("average linkage")**
  La distance entre deux classes $C^{(1)}$ et $C^{(2)}$ est calculée comme la distance moyenne

entre toutes les sous-séquences deux à deux dans les deux classes différentes.

$$D_{C^{(1)},C^{(2)}} = \frac{1}{n_{C^{(1)}} n_{C^{(2)}}} \sum_{S^{(1)} \in C^{(1)}} \sum_{S^{(2)} \in C^{(2)}} d_{S^{(1)},S^{(2)}} \tag{9.9}$$

où $n_{C^{(i)}}$ est le cardinal de $C^{(i)}$. Ce critère est efficace lorsque les sous-séquences forment déjà naturellement des "classes" bien distinctes ; il s'est révélé également bien adaptée dans le cas de classes allongées, de type "chaîne".

– **Le critère de la moyenne pondérée ("weighted linkage")**
À la différence du critère précédent de la moyenne, dans ce critère le cardinal des classes respectives est utilisé ici comme pondération. Soit $C^{(1)}$ une classe créée par la combinaison de deux classes $C^{(11)}$ et $C^{(12)}$, la distance entre les classes $C^{(1)}$ et $C^{(2)}$ est définie comme étant la somme de la distance moyenne entre $C^{(11)}$ et $C^{(2)}$ et la distance moyenne entre $C^{(12)}$ et $C^{(2)}$ divisée par deux.

$$D_{C^{(1)},C^{(2)}} = D_{C^{(11)} \cup C^{(12)},C^{(2)}} = \frac{D_{C^{(11)},C^{(2)}} + D_{C^{(12)},C^{(2)}}}{2} \tag{9.10}$$

Ce critère est préféré au critère de la moyenne lorsque les cardinaux des classes sont assez inégaux.

– **Le critère de Ward ("ward linkage")**
Le critère de Ward utilise la distance euclidienne, il consiste à choisir à chaque étape le regroupement de classes tel que l'augmentation de l'inertie intra-classe soit minimale.

$$D_{C^{(1)},C^{(2)}} = \sqrt{\frac{n_{C^{(1)}} n_{C^{(2)}}}{n_{C^{(1)}} + n_{C^{(2)}}}} d^{Mink/2}_{G_{C^{(1)}},G_{C^{(2)}}} \tag{9.11}$$

où $G_{C^{(i)}}$ est le centre de gravité de $C^{(i)}$ et $d^{Mink/2}_{G_{C^{(1)}},G_{C^{(2)}}}$ la distance euclidienne.

– **Le critère de centre de gravité ("centroid linkage")**
Suivant ce critère la distance entre les classes $C^{(1)}$ et $C^{(2)}$ est définie par la distance euclidienne entre les centres de gravité.

$$D_{C^{(1)},C^{(2)}} = d^{Mink/2}_{G_{C^{(1)}},G_{C^{(2)}}} \tag{9.12}$$

– **Le critère de médiane ("median linkage")**
Ce critère calcule la distance euclidienne entre les centres de gravité pondérés de deux classes afin de prendre en compte le cardinal des classes.

$$D_{C^{(1)},C^{(2)}} = d^{Mink/2}_{\widetilde{G}_{C^{(1)}},\widetilde{G}_{C^{(2)}}} \tag{9.13}$$

où $\widetilde{G}_{C^{(i)}}$ est le centre de gravité pondéré de la classe $C^{(i)}$. Si $C^{(i)}$ est l'agrégation de deux classes $C^{(i1)}$ et $C^{(i2)}$ alors $\widetilde{G}_{C^{(i)}}$ est défini par

$$\widetilde{G}_{C^{(i)}} = \frac{1}{2}(\widetilde{G}_{C^{(i1)}} + \widetilde{G}_{C^{(i2)}}) \tag{9.14}$$

L'efficacité des différents critères d'agrégation pour construire l'arbre hiérarchique peut être évaluée par le coefficient de corrélation cophenetic introduit dans [172]. Ce coefficient de corrélation est défini comme étant le coefficient de corrélation linéaire entre les distances cophénétiques obtenues à partir de l'arbre (distance entre deux sous-séquences représentées dans un arbre par la hauteur des branches qui finissent par les réunir dans une même classe) et les distances initiales rangées dans une matrice de similarité. Ainsi, il s'agit d'une mesure de la corrélation linéaire entre les valeurs de la matrice de distances initiales et celle des distances cophénétiques. Plus sa valeur est proche de 1, meilleure est la classification. L'expression de ce coefficient est

$$c = \frac{\sum_{i<j}(d_{S^{(i)},S^{(j)}} - \bar{d})(t_{S^{(i)},S^{(j)}} - \bar{t})}{\sqrt{(\sum_{i<j}(d_{S^{(i)},S^{(j)}} - \bar{d})^2)(\sum_{i<j}(t_{S^{(i)},S^{(j)}} - \bar{t})^2)}} \tag{9.15}$$

où $\bar{d}$ est la valeur moyenne des distances $d_{S^{(i)},S^{(j)}}$, $t_{S^{(i)},S^{(j)}}$ est la hauteur de la branche à laquelle les deux sous-séquences $S^{(i)}$ et $S^{(j)}$ sont reliées entre elles et $\bar{t}$ est la valeur moyenne des distances $t_{S^{(i)},S^{(j)}}$.

9.6 Conclusion

Dans ce chapitre, nous avons présenté quelques algorithmes de classification non supervisée, notamment les algorithmes de partitionnement et les algorithmes hiérarchiques. Les principaux avantages des algorithmes hiérarchiques par rapport aux algorithmes de partitionnement résident dans leur convergence et la non-nécessité de la connaissance du nombre de classes a priori. Par la suite, nous avons présenté les différentes mesures de similarité entre sous-séquences et les différents critères d'agrégations entre classes. Les performances des algorithmes de classification non supervisée dépendent significativement de choix appropriés d'une mesure de similarité et d'un critère d'agrégation. Le choix d'une mesure de similarité ou d'un critère d'agrégation dépend de la nature des données, des objectifs souhaités et des connaissances disponibles a priori. Dans le prochain chapitre, nous présentons un algorithme d'extraction de motifs basé sur l'utilisation des techniques de classification non supervisée.

Chapitre 10

Algorithme d'extraction de motifs

Dans ce chapitre, nous présentons un algorithme d'extraction de motifs à partir de données temporelles basé sur des techniques de classification non supervisée en justifiant certains choix techniques. L'algorithme permet d'extraire des motifs et de construire une bibliothèque de motifs sans connaissance a priori sur la nature et le nombre de motifs. Cet algorithme a été développé dans le cadre du projet CASSIE.

Sommaire

10.1 Objectifs et contraintes

Nos objectifs en terme d'extraction de motifs au sein du projet CASSIE sont les suivants :

- déceler la présence éventuelle de motifs sur une série temporelle ;
- réaliser la construction des classes de motif ;
- modéliser le motif pour chaque classe ;
- regrouper ces modèles dans une bibliothèque de motifs ;

Les difficultés posées par ces objectifs sont multiples. Tout d'abord, le nombre de motifs présent sur la série temporelle est inconnu, il est même possible qu'aucun motif n'apparaisse. Il va donc être nécessaire d'estimer le nombre de motifs. De même, toutes les sous-séquences de la série temporelle ne correspondent pas forcément à un motif ou ne peuvent être inclues dans une classe de motifs. Il est probable que de nombreuses sous-séquences ne ressemblent à aucune autre sous-séquence. Ensuite, si au moins un motif est présent sur la série temporelle, ses instants d'occurence et sa longueur sont inconnus. Enfin, les séries temporelles doivent être analysées sans connaissance a priori sur la nature des informations qu'elles représentent. Il n'est donc pas possible d'utiliser de connaissance a priori (fréquence d'apparition, régularité des occurences, etc.) pour la réalisation des tâches précédentes. Rajoutons que ces différentes tâches doivent être mise en œuvre sous forme de briques logicielles écrites en C/C++ afin de pouvoir assurer un maximum de portabilités et de pouvoir les embarquer sur la plateforme de développement du partenaire Digital Airways.

Nous proposons dans ce chapitre un algorithme, reposant sur des techniques de classification non supervisée, pour la recherche, l'identification et la localisation des motifs apparaissant sur une série de données temporelles. Il s'agit aussi de regrouper ces motifs identifiés dans une bibliothèque de motifs.

10.2 Description de l'algorithme

10.2.1 Structure de l'algorithme

La structure de l'algorithme d'extraction de motifs est présenté sur la figure 10.1. La recherche et la localisation des motifs se réalisent en mode batch. Il est nécessaire de récupérer l'ensemble des données avant de réaliser le traitement.

L'algorithme nécessite en outre la disposition de données échantillonnées de manière régulière.

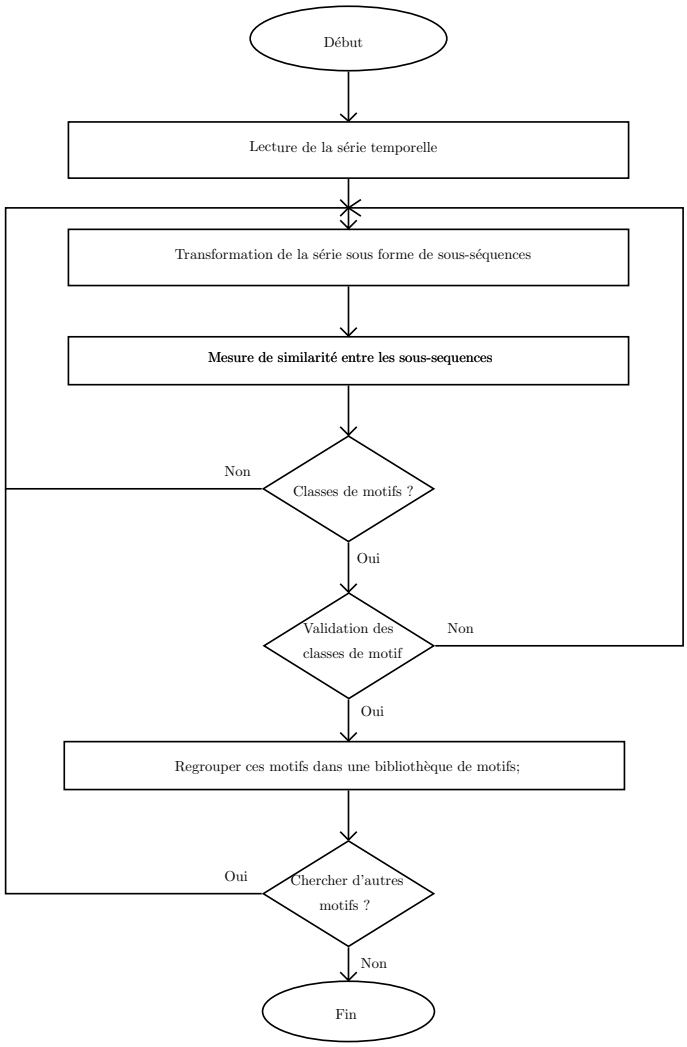

FIGURE 10.1 – Structure de l'algorithme

Les principaux points saillants de l'algorithme sont :

- une transformation de la série de données sous forme de sous-séquences de longueur m ;
- une répartition des sous-séquences en classes de motifs ;
- une validation des classes de motifs.

Les paragraphes ci-dessous détaillent les choix réalisés sur ces différents points.

10.2.1.1 Transformation de la série temporelle sous forme de sous-séquences de longueur m

Les instants d'occurrence des motifs sont inconnus, par conséquent toute sous-séquence peut a priori faire partie d'une classe de motif. La première étape de l'algorithme consiste donc à récupérer toutes les sous-séquences possibles, pour une longueur m fixée, en utilisant une fenêtre glissante. Ceci est illustré sur la figure 10.2. Il convient de noter qu'entre deux sous-séquences consécutives il y a chevauchement de $m-1$ échantillons, ceci devra être pris en compte lors de la formation des classes.

La longueur des motifs est inconnu a priori. Il est par conséquent nécessaire de réaliser l'analyse pour une gamme de longueur $m \in [m_{min}; m_{max}]$. Le choix de la longueur maximale m_{max} et la longueur minimale m_{min} dépend de la nature des données mais aussi des résultats attendus. Il est conseillé de chercher dans un premier temps les motifs de longue durée, c'est à dire de réaliser l'extraction de motif pour $m = m_{max}$ et ensuite de décrémenter la valeur de m jusqu'à $m = m_{min}$.

Pour chaque longueur m, une fois les classes de motif créées, il peut être nécessaire de supprimer de la série temporelle les sous-séquences appartenant à ces classes.

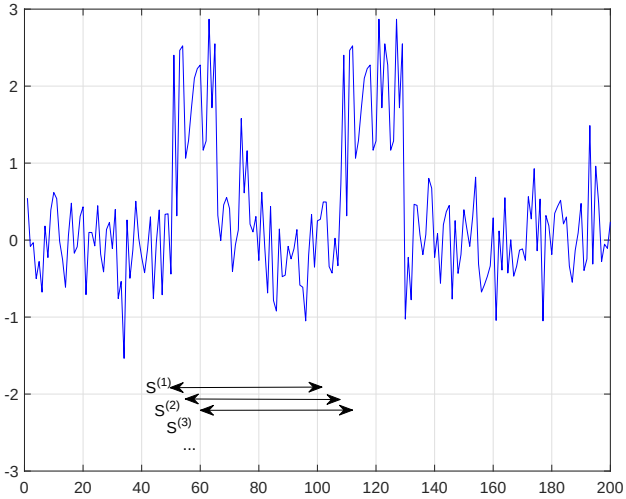

FIGURE 10.2 – Transformation de la série temporelle sous forme de sous-séquences

10.2.1.2 Répartition des sous-séquences en classes de motif

La répartition des sous-séquences en classes de motif est réalisée via une méthode de classification non-supervisée. Les méthodes de classification par partitionnement ne sont pas appropriées ici du fait de la nécessité pour ces méthodes de connaître a priori le nombre de classes de motifs.

L'utilisation des méthodes de classification hiérarchiques semble ici plus adaptée. L'idée est d'estimer dans un premier temps la similitude entre les différentes sous-séquences et de former l'arbre hiérarchique (figure 10.3). Cet arbre hiérarchique va nous permettre d'une part de localiser les "paquets" de sous-séquences similaires et d'autre part d'isoler les sous-séquences ne ressemblant à aucune autre sous-séquence.

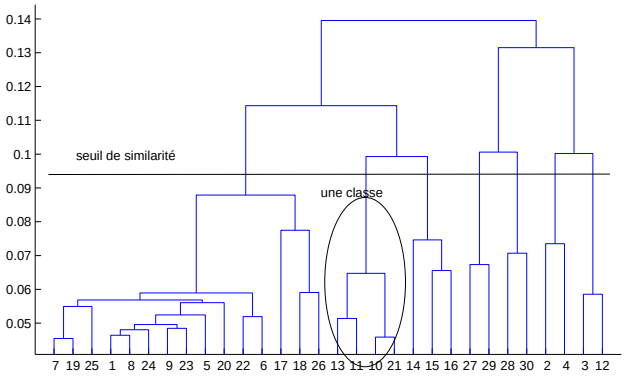

FIGURE 10.3 – Construction de l'arbre hiérarchique

Nous proposons de fixer un seuil de similarité afin de couper l'arbre hiérarchique d'une manière automatique. Ceci va assurer la création de classes constituées de sous-séquences similaires. Ceci va aussi permettre de ne pas avoir à construire nécessairement l'intégralité de l'arbre, d'où un gain de temps en terme de mise en oeuvre.

Le choix de la classification hiérarchique nous contraint à deux choix pour la mise en œuvre : le premier sur le choix de la distance entre sous-séquences, le second sur le choix de la distance entre classes.

Choix de la distance entre sous-séquences

Étant donné qu'une des contraintes pour l'extraction de motifs est une mise en œuvre automatique, nous avons choisi une distance normalisée. Ceci nous permet de fixer par

avance le seuil de similarité et non d'avoir à choisir de seuil au cas par cas, suivant l'amplitude des échantillons de la série temporelle. Ce choix de la distance normalisée exclu les distances basées sur la norme L_p et les distances élastiques.

Pour ce qui est du choix de la distance normalisée, nous proposons de favoriser l'usage d'une distance pour laquelle la charge de calcul est réduite. Ceci exlut la distance de Spearman. Les distances par cosinus et par corrélation permettent d'analyser des relations linéaires entre les sous-séquences et présentent des performances proches. La différence entre les deux réside dans le fait que la distance de corrélation est insensible à la présence d'un offset (centrage des données).

Choix de la distance entre classes

Le choix de la distance entre classes doit permettre de former classes naturelles et homogènes. Étant donné que le critère de saut minimum ("single linkage") peut générer un effet de chaînage et qu'il est probable que certaines sous-séquences des classes formées soient éloignées les unes des autres, nous excluons ce critère des choix possibles. D'autre part, comme l'efficacité de différents critères d'agrégation dépend du volume et de la nature des données, les différents critères d'agrégation pour construire l'arbre hiérarchique doivent être évalués par le coefficient de corrélation cophenetic via des simulations numériques sur les données de la série temporelle. Le critère qui donne un coefficient de corrélation le plus proche de 1 sera sélectionné. Ceci est illustré dans la prochaine section.

10.2.1.3 Validation des classes de motif

Chaque classe formée à l'étape précédente contient probablement des sous-séquences avec chevauchement. Il est aussi possible que certaines classes ne soient qu'artificielles dans le sens que les sous-séquences les constituants ne soient similaires les unes des autres que par coïncidence. Ainsi, il s'agit ici de prendre en compte uniquement les sous-séquences sans chevauchement dans une classe et d'exclure les classes dont le cardinal n'est pas significatif. La procédure est la suivante :

- Création d'un modèle pour chaque classe obtenue à l'étape précédente.
- Pour chaque classe, recherche sur la série temporelle des sous-séquences qui ont une forte similarité avec le modèle. Sont considérées comme similaires au modèle les sous-séquences dont la distance avec le modèle est inférieure au seuil utilisé dans l'étape de classification.

– Si deux sous-séquences se chevauchent, on ne garde que la sous-séquence qui présente la distance minimale avec le modèle.
– Élimination des classes dont le cardinal est inférieur à un seuil préfixé.

Le modèle d'une classe est une sous-séquence représentant au mieux la classe. Plusieurs options sont possibles pour la construction de ce modèle. Il peut être la moyenne de toutes les sous-séquences de la classe, la médiane ou encore la sous-séquences de la classe présentant la distance minimale avec l'ensemble des autres sous-séquences de la classe.

Le cardinal de la classe représente le nombre d'occurence du motif sur la longueur de la série temporelle. Le seuil sur le cardinal d'une classe peut être lié à une fréquence minimale d'apparition du motif sur un intervalle de temps. Si le nombre de sous-séquences constituant une classe est supérieur à la fréquence d'apparition minimale f_{min}, on sauvegarde le motif dans la bibliothèque de motifs. Une fois qu'on a sauvegardé le motif, on supprime toutes les sous-séquences correspondantes à ce motif de la série de données. Si le nombre de sous-séquences sauvegardées pour une classe est inférieur à la fréquence d'apparition minimale f_{min}, le motif sera ignoré.

10.2.2 Résumé de l'algorithme

Le résumé descriptif de l'algorithme est présenté dans le tableau 10.1. On note que l'algorithme a des paramètres de synthèses qui doivent être spécifiés convenablement par l'utilisateur. Le choix de ces paramètres conditionne d'une manière importante les performances de l'algorithme. Ces paramètres de synthèses sont :

- La longueur maximale m_{max} et la longueur minimale m_{min} des motifs recherchés : Le choix de la longueur maximale m_{max} et la longueur minimale m_{min} dépend essentiellement de la nature et de l'amplitude des données, et dépend aussi des objectifs souhaités. L'algorithme cherche à identifier des motifs de longueur m tel que $m_{min} \leq m \leq m_{max}$.

- La fréquence d'apparition minimale f_{min}. La fréquence d'apparition minimale f_{min} permet d'élaguer les motifs non fréquents et par conséquent, éviter la création de motifs correspondant à des coïncidences.

- Le seuil de distance $seuil_{distance}$. Si le seuil distance fixé par l'utilisateur est élevé, des sous-séquences ne se ressemblant pas risque d'être inclues dans une même classe, c'est-à -dire assignées au même motif qui ne sera pas, représentatif d'un évènement. Si ce seuil est fixé trop proche de zéro, des sous-séquences représentant le même évènement, à un bruit de mesure prêt, risque d'être assignées à deux motifs différents. Ceci va générer un nombre de classes important avec un cardinal faible.

Remarque 12. *L'algorithme proposé est dédié à l'extraction de motifs à partir de séries de données temporelles réelles. Des extensions de l'algorithme pour l'extraction de motifs à partir d'autres types de données (données symboliques, données binaires, données multivariables de même type ou mixtes) ont été développées. De plus, des détails techniques définis par le GREYC et DigitalAirways lors de réunions techniques spécifiques ont permis le développement d'une procédure d'identification de motifs en temps réel. Ces travaux ont été fournis à DigitalAirways sous forme de briques logicielles écrites en C/C++ dans le cadre du projet CASSIE.*

- ***Entrées : série de données*** T, $seuil_{distance}$, f_{min}, m_{max} ***et*** m_{min}
- ***Sorties : Une bibliothèque de motifs***
- ***Début***
- ***(1) : pour*** m ***décroissant*** $m_{min} \leq m \leq m_{max}$
- ***(2) : Pour chaque longueur*** m ***:***
  - ***(2-1) : Transformer la série de données sous forme de sous-séquences en utilisant une fenêtre glissante.***
  - ***(2-2) : Mesurer la similarité entre les sous-séquences en utilisant une distance normalisée".***
  - ***(2-3) : Former les classes en utilisant la classification hiérarchique.***
  - ***(2-4) : Les sous-séquences qui ont une corrélation très élevée entre elles (distance inférieure au*** $seuil_{distance}$***) sont réunies dans la même classe. Les classes constituées de peu d'individus ne sont pas prises en compte.***
  - ***(2-5) : Un modèle de chaque classe est calculé.***
  - ***(2-6) : Pour chaque classe, garder uniquement les sous-séquences sans chevauchement.***
  - ***(2-7) : -Si le nombre de sous-séquences détectées d'un motif*** $\geq f_{min}$ ***:***
    - ***Sauvegarder le motif(modèle-longueur*** m ***- fréquence d'apparition - instant d'apparition)***
    - ***Supprimer toutes les sous-séquences du motif***
    - ***Aller chercher d'autres motifs avec d'autres fréquences d'apparition (s'il y en a) ou d'autres longueurs*** m***.***
  - ***(2-7) : -Si le nombre de séquences détectées d'un motif*** $< f_{min}$
    - ***Ignorer la classe formée et sa modèle***
    - ***Aller chercher d'autres motifs avec d'autres fréquences d'apparition ou d'autres longueurs*** m
- ***Fin***

TABLE 10.1 – Le pseudo code de l'algorithme proposé

10.3 Exemples de mise en œuvre

Afin d'illustrer certains aspects de la solution proposée, nous présentons dans cette section, deux exemples de mise en œuvre de notre algorithme d'extraction de motifs. Dans le premier exemple, nous analysons les données de consommation d'eau dans une maison. Le deuxième exemple porte sur l'analyse des données de consommation d'électricité au sein d'un local à usage professionnel. Les deux séries de données sont fournies par notre partenaire DigitalAirways. Toutes les analyses réalisées dans ce chapitre sont obtenues en exécutant "Matlab 8.3 (R2014a)- sur processeur 2,7 GHz Intel Core i7".

10.3.1 Consommation d'eau

Un premier exemple d'analyse porte sur la consommation en eau d'un foyer sur plusieurs semaines. La figure 10.4 présente les données analysées sur une période de 10 semaines avec une période d'échantillonnage égale à 1 minute. Le nombre de données disponibles est alors $N = 100800$. Cette série de données correspond au cumul de différentes activités régulières (douche, bain, chasse d'eau, lave-linge, lave-vaisselle, etc). L'objectif est de permettre l'extraction automatique de ces activités.

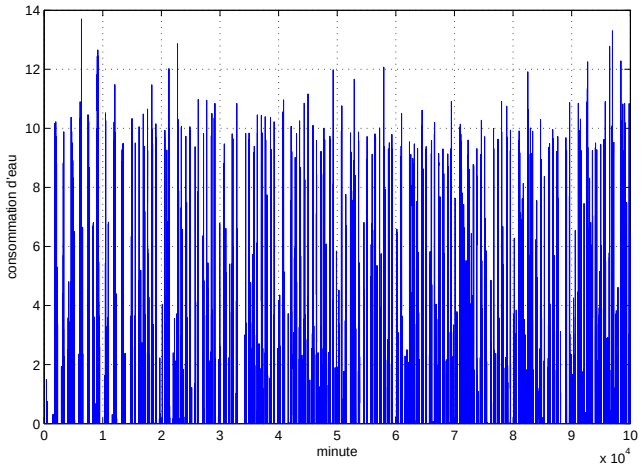

FIGURE 10.4 – Consommation en eau d'un foyer sur plusieurs semaines

Dans un premier temps, l'objectif est de choisir la meilleure distance entre sous-séquences, la meilleure distance entre classes et d'étudier l'évolution du temps de calcul

(CPU time) de l'algorithme en fonction de nombre de données disponibles N. Deux distances entre sous-séquences peuvent être utilisées (distance par cosinus et distance par corrélation) et plusieurs distances entre classes peuvent être utilisées. Afin de sélectionner la meilleure distance entre sous-séquences et la meilleure distance entre classes, nous procédons à une simulation numérique en utilisant les différentes distances possibles. La qualité de ces distances est évaluée par le coefficient de corrélation cophénétique. Nous appliquons notre algorithme sur une période de 3 semaines ($N = 30240$) avec le paramétrage suivant :

- Fréquence d'apparition minimale $f_{min} = 3$.
- $m_{min} = 10$ minutes, $m_{max} = 10$ minutes
- $seuil_{distance} = 0.05$.

La longueur maximale m_{max} est choisi volontairement égale à la longueur minimale m_{min} car l'objectif de cette première simulation n'est pas la recherche de motifs, mais plutôt la sélection de la meilleure distance entre sous-séquences et la meilleure distance entre classes. La fréquence d'apparition minimale est choisie égale à $f_{min} = 3$, ceci signifie qu'on cherche des motifs qui apparaissent au moins une fois par semaine. Le calcul du coefficient de corrélation cophénétique et le temps nécessaire à l'exécution de l'algorithme pour les différentes distances possibles sont données dans le tableau 10.2.

Distances entre sous-séquences	Distances entre classes	Corrélation cophénétique	Temps de calcul (sec)
Distance par cosinus	Average	0.7322	4.6251
	Centroid	0.6946	4.5788
	Complete	0.6029	4.4951
	Median	0.6273	4.5807
	Ward	0.6919	4.5212
	Weighted	0.6921	4.4819
Distance par corrélation	Average	0.7295	4.6962
	Centroid	0.6939	4.5436
	Complete	0.7022	4.5313
	Median	0.6130	4.5925
	Ward	0.7006	4.7127
	Weighted	0.6433	4.5116

TABLE 10.2 – Consommation d'eau- Coefficient de corrélation "cophenetic" c et le temps de calcul pour les différents choix possibles

Il apparaît qu'avec les deux distances entres sous-séquences (distance par cosinus et distance par corrélation), la distance entre classes "average" donne les meilleurs résultats. La qualité des résultats avec la distance par corrélation n'est pas trop dégradée par rapport à la qualité des résultats avec la distance par cosinus. Par contre, le temps nécessaire à l'exécution de l'algorithme avec la distance par corrélation est plus grand que le temps nécessaire à l'exécution de l'algorithme avec la distance par cosinus. Étant donné que les algorithmes hiérarchiques souffrent déjà de l'augmentation de la complexité de calcul d'une manière cubique en fonction du nombre de données, un choix d'une mesure de distance par corrélation est un handicap lors du traitement de séries de données de grand volume. Ceci nous oriente à choisir la distance par cosinus comme distance entre sous-séquences et le critère d'agrégation "average" comme distance entre classes.

Afin d'illustrer l'évolution du temps de calcul nécessaire à l'exécution de l'algorithme en fonction du nombre de données, nous procédons à plusieurs expériences pour différentes valeurs de N. Le temps d'exécution de l'algorithme pour différentes valeurs de N est donné dans le tableau 10.3.

N	$N = 20160$	$N = 50400$	$N = 70560$	$N = 100800$
Temps d'exécution (sec)	2.1744	6.7369	11.2510	24.7646

TABLE 10.3 – Evolution du temps d'exécution de l'algorithme en fonction de N

Selon les résultats donnés dans le tableau 10.3, on voit clairement que le temps de calcul nécessaire à l'exécution de l'algorithme augmente d'une manière exponentielle avec l'augmentation du nombre de données. Ceci est dû à l'utilisation de la classification hiérarchique comme cœur de l'algorithme. A noter que le temps d'exécution de l'algorithme proposé dépend aussi de l'étendue de l'intervalle $[m_{min}, m_{max}]$. Plus l'étendue de l'intervalle est large, plus le temps d'exécution de l'algorithme est important.

Dans un second temps, l'objectif est d'extraire les motifs présents sur la série de données. Nous appliquons notre algorithme sur la série de données ($N = 100800$) avec le paramétrage suivant :

- Fréquence d'apparition minimale $f_{min} = 10$.
- $m_{min} = 10$ minutes, $m_{max} = 50$ minutes.
- $seuil_{distance} = 0.05$.

Nous cherchons des motifs de longueurs entre [10 , 50] minutes avec une apparition au moins une fois par semaine. Figure 10.5 illustre deux motifs identifiés par l'algorithme.

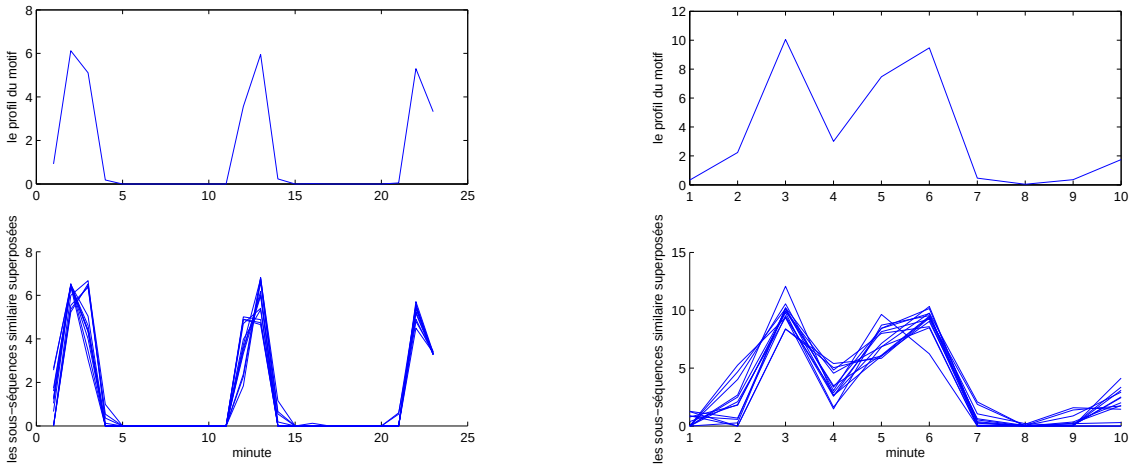

FIGURE 10.5 – Le modèle et les sous-séquences superposées de deux motifs identifiés

L'observation a posteriori des instants d'apparition, le volume et la longueur de ces deux motifs révèlent qu'elles correspondent probablement à un lave-linge (motif de 25 minutes) et une douche (motif de 10 minutes). D'autres motifs identifiés par l'algorithme sont illustrés sur la figure 10.6. L'interprétation des résultats afin de les considérer comme des connaissances utiles si c'est possible est à la charge des autres partenaires du projet.

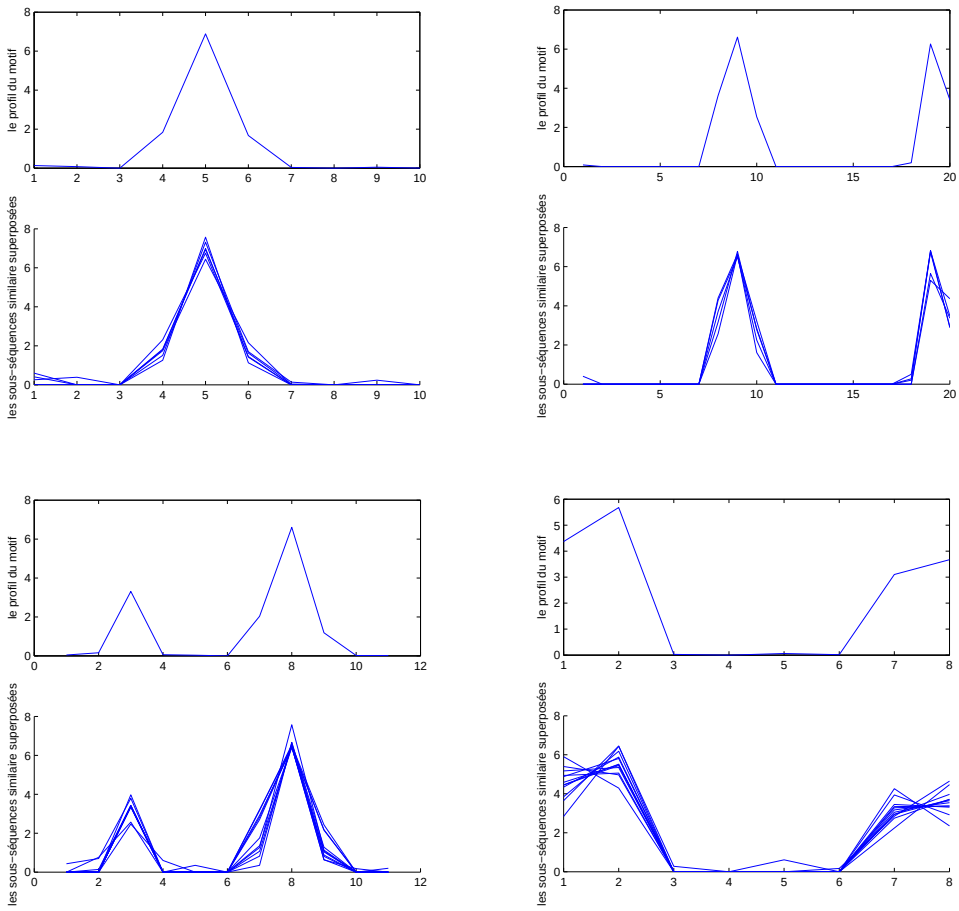

FIGURE 10.6 – Consommation en eau - des motifs identifiés par l'algorithme

10.3.2 Consommation d'électricité

Cette analyse a été réalisée sur une consommation d'électricité. La figure 10.7 présente la série de données traitées. Cette série de données représente 7 jours successifs de consommation en électricité au sein d'un local à usage professionnel.

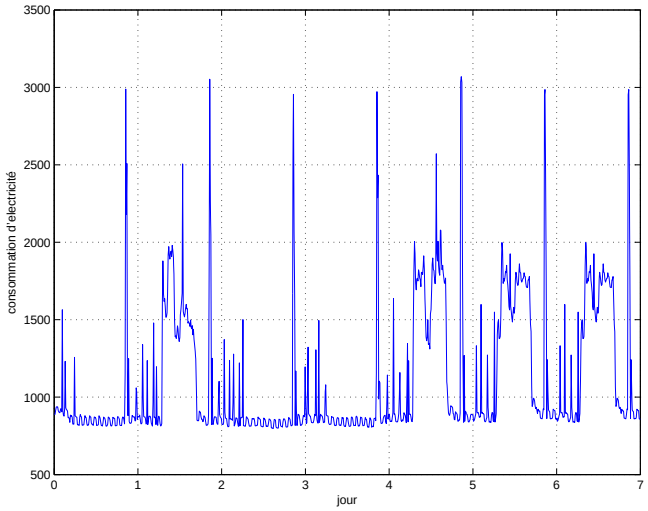

FIGURE 10.7 – Consommation en électricité sur une période de 7 jours

A priori ces données correspondent à des mesures effectuées du jeudi 14 mai (jour 1) au mercredi 20 mai 2015 (jour 7). L'objectif est de chercher des motifs qui représente un comportement type. Par exemple, distinguer les jours fériés et les jours travaillés. Une simulation numérique comme celle effectuée dans l'exemple de consommation d'eau a été effectuée afin de sélectionner la meilleure distance entre sous-séquences et la meilleure distance entre classes pour cette série de données avec le paramétrage suivant :

- Fréquence d'apparition minimale $f_{min} = 2$.
- $m_{min} = m_{max} = 24$ heures (12×24 échantillons).
- $seuil_{distance} = 0.05$.

Les résultats de simulation pour les différentes distances sont donnés dans le tableau 10.4.

Distances entre sous-séquences	Distances entre classes	Corrélation cophénétique	Temps de calcul
Distance par cosinus	Average	0.6375	0.6600
	Centroid	0.5725	1.3458
	Complete	0.4232	0.4423
	Median	0.5461	1.4807
	Ward	0.5475	1.7675
	Weighted	0.5774	0.5060
Distance par corrélation	Average	0.6181	1.3887
	Centroid	0.5564	2.0704
	Complete	0.5242	1.3184
	Median	0.3502	2.4210
	Ward	0.5375	2.2292
	Weighted	0.5640	1.3222

TABLE 10.4 – Consommation d'électricité- Coefficient de corrélation "cophenetic" c pour différents choix possibles

Il apparaît que la distance par cosinus comme distance entre sous-séquences et le critère "average" comme distance entre classes donnent les meilleurs résultats. Nous utilisons alors la distance par cosinus comme distance entre sous-séquences et le critère "average" comme distance entre classes pour cette série de données.

Nous appliquons à présent notre algorithme sur la série de données en utilisant les distances choisies. Le résultat retourné par l'algorithme est illustré sur la figure 10.8. Deux motifs sont estimés. Il est aisé de distinguer les jours fériés (jours 1, 3 et 4) où le jour 1 correspond au jeudi 14 mai qui représente le jeudi de l'ascension et les jours 3 et 4 correspondent aux jours de fin de semaine. Il est aussi possible de distinguer les jours travaillés (jours 2, 5, 6 et 7). Figure 10.9 illustre les modèles de ces deux motifs et les sous-séquences superposées de chaque motif, ils correspondent bien aux deux types de jours : férié et travaillé. Pour les jours travaillés, on distingue notamment clairement la période d'activité.

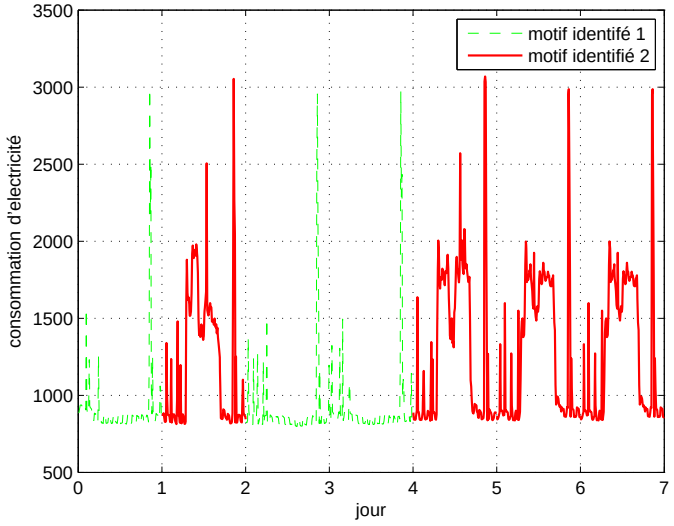

FIGURE 10.8 – Deux motifs identifiés : les jours fériés et les jours travaillés

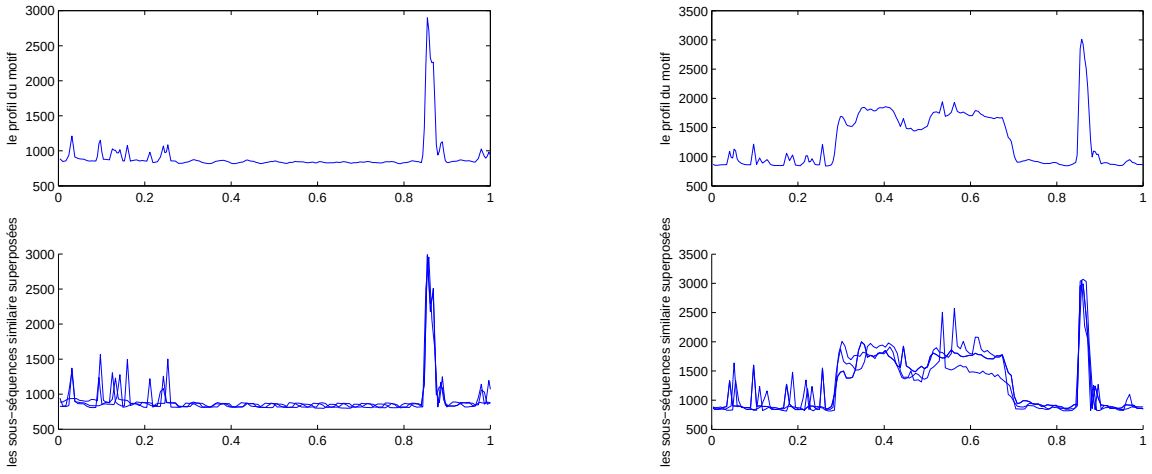

FIGURE 10.9 – Profil de deuxième motif identifié

10.4 Conclusion

Dans ce chapitre, nous avons présenté un algorithme d'extraction de motifs à partir de série de données temporelles qui permet de rechercher, localiser des motifs et les retourner sous forme d'une bibliothèque de motifs. L'algorithme repose sur l'utilisation des techniques de classification non supervisée. Des briques logiciels ont été constituées sur la base de l'algorithme d'extraction de motifs et plusieurs versions de ces briques logiciels ont été livrées à Digitil Airways. Le passage d'une version à une autre a permis d'affiner les propriétés de l'algorithme, notamment une réduction de la charge de calcul, une simplification de la mise en œuvre et une amélioration des performances en terme d'extraction de motifs et de capacité de traitement quant au volume de la série de données. Nous avons testé notre algorithme d'extraction de motifs sur des données réelles fournies par notre partenaire dans le projet DigitalAirways. Les résultats illustrent que l'algorithme permet bien d'extraire des motifs interprétables présents sur les séries de données.

Conclusion et Perspectives

Conclusion

Dans cette thèse, nous avons présenté des contributions aux thématiques de l'identification des systèmes hybrides, de l'identification des systèmes ayant une sortie binaire et de l'extraction des motifs à partir des données. Cette présentation a été structurée en trois parties. Deux parties ont été consacrées au développement des travaux de recherches au sein du laboratoire d'automatique de Caen dans le cadre de l'identification des systèmes. La troisième partie a été réservée à la réalisation des objectifs définis dans le cadre du projet CASSIE. Le facteur commun entre ces parties est l'utilisation des techniques de classification de données. Ci-dessous, nous résumons brièvement les principales contributions de cette thèse :

- *Identification des systèmes hybrides.* Nous avons proposé un algorithme d'identification en temps réel des systèmes à commutations en présence d'un bruit borné en considérant une hypothèse sur l'existence uniquement d'une borne supérieure sur l'amplitude du bruit. La plupart des méthodes disponibles dans la littérature résolvent ce problème en supposant que le bruit est un processus stochastique. Nous avons considéré dans un premier temps le modèle SARX pour décrire les systèmes linéaires à commutations, l'algorithme proposé pour l'identification des paramètres de ce modèle consiste en deux étapes. À chaque instant, la première étape permet de classifier un point de données en l'associant au sous-système actif à l'instant considéré et la deuxième étape permet d'actualiser le vecteur de paramètres associé au sous-système actif. L'algorithme proposé se distingue par sa faible complexité de calcul, un fonctionnement en temps réel ou en mode batch et une indépendance du comportement de l'algorithme vis-à-vis des commutations (arbi-

traire ou non). Une analyse de comportement et de convergence de l'algorithme a été effectuée ce qui est relativement rare dans le contexte de l'identification des systèmes à commutations. Dans un second temps, trois extensions de l'algorithme ont été proposées en considérant d'autres modèles (les modèles PWARX, MIMO-SARX et SOE) pour décrire les systèmes à commutations. L'efficacité et l'intérêt des algorithmes proposés ont été validés par des résultats de simulations qui ont montré la compétitivité des algorithmes proposés par rapport aux solutions disponibles dans la littérature. Ces contributions font l'objet des publications suivantes : [73], [74], [75], [77] et [76].

- *Identification des systèmes ayant une sortie binaire.* Dans cette partie, nous avons exploité des techniques de classification supervisée afin de proposer un algorithme d'identification des systèmes ayant une sortie binaire. L'algorithme proposé repose sur une formulation du problème d'identification comme un problème de classification binaire et sur l'utilisation des SVMs. L'algorithme permet de s'affranchir des limitations et des contraintes imposées par les algorithmes d'identification des systèmes ayant une sortie binaire disponibles dans la littérature. L'algorithme a été étendu par la suite au cas des systèmes continus ayant une sortie binaire modulo l'utilisation d'une séquence d'excitations particulière. Ces contributions font l'objet des publications suivantes : [78] et [156].

- *Extraction de motif à partir de données.* Dans cette partie, nous avons développé un algorithme d'extraction de motifs qui repose sur l'utilisation des techniques de classification non supervisée, en l'occurrence, des méthodes de la classification hiérarchique, afin de pouvoir rechercher, localiser des motifs et les retourner sous forme d'une bibliothèque de motifs. La stratégie proposée pour la recherche et la localisation de motifs répond aux différentes contraintes imposées par les différents partenaires du projet, à savoir, le nombre de motifs est inconnu, la nature des motifs est inconnue, les instants d'apparition des motifs sont inconnus et les fréquences d'apparition sont inconnues. L'interprétation des résultats de l'algorithme, retournés sous forme d'une bibliothèque de motifs, peut conduire potentiellement à des connaissances utiles et interprétables. Les contributions de cette partie font l'objet des briques logicielles écrites en C/C++ livrés à Digital Airways dans le cadre du projet CASSIE.

Perspectives

Plusieurs points restent encore à améliorer ou à explorer dans le futur. Certains points sont donnés ci-dessous :

- *Identification des systèmes hybrides.* Deux points sont à développer. Premièrement, d'autres extensions de l'algorithme R-SARX-OBE sont encore possibles. À titre d'exemple, l'algorithme peut être adapté pour l'identification des systèmes hybrides décrits par un modèle DHA ("Discret Hybrid Automata). Le modèle DHA décrit dans [180], résulte de la connexion d'une machine à états finis FSM ("Finite State Machine"), qui régit l'état logique du système, avec un modèle linéaire à commutation qui décrit l'état dynamique continu du système. Il serait intéressant également d'adapter l'algorithme pour l'identification des systèmes hybrides non linéaires et de l'adapter aussi pour réaliser l'identification en boucle fermé. Deuxièmement, les algorithmes d'identification des systèmes hybrides disponibles dans la littérature sont essentiellement dédiés à l'identification des systèmes hybrides décrits par des modèles discrets. Cependant, l'utilisation d'un modèle à temps continu pour la modélisation des processus est souvent appréciée pour l'estimation de certains paramètres physiques et l'interprétation physique directe. Ainsi, il semble intéressant d'envisager l'identification des systèmes hybrides décrits par des modèles continus.

- *Identification des systèmes discrets ayant une sortie binaire.* L'algorithme d'identification proposé est dédié à l'identification des systèmes ayant une sortie binaire où la partie linéaire du système est décrit par un filtre à une réponse impulsionnelle FIR. De futures recherches pourraient être focaliser sur des structures moins restrictives, par exemple : le modèle OE, le modèle non linéaire de Wiener ou le modèle non linéaire de Hammerstein/Wiener. Une autre piste de recherche possible est d'utiliser les SVM multi-classes pour l'identification des systèmes ayant une sortie quantifiée (la sortie du système a plus de deux niveaux).

- *Identification des systèmes continus ayant une sortie binaire.* Certaines questions sur le fonctionnement de l'algorithme d'identification demeurent et font l'objectif de futures recherches. Parmi ces questions, comment choisir l'ensemble des fréquences angulaires $\{\omega_i\}$? Comment choisir la fréquence d'échantillonnage par rapport au choix de l'ensemble des fréquences angulaires ? Comment satisfaire la condition de l'excitation persistante avec le nombre minimum de fréquences angulaires ?

- *Extraction de motif à partir de données.* L'algorithme proposé nécessite la disposition des données échantillonnées d'une manière régulière. Les données aberrantes ou manquantes ne sont pas prises en compte lors du traitement. Un affinement de l'algorithme proposé est encore nécessaire. Il est possible d'envisager de réaliser un traitement des données échantillonnées d'une manière irrégulière tout en optimisant la complexité de calcul de l'algorithme et de prendre en compte les données aberrantes ou manquantes. L'utilisation de la distance DTW peut être considérée pour réaliser un traitement sur des données échantillonnées d'une manière irrégulière. Malheureusement, son utilisation conduit à l'augmentation de la complexité de calcul de l'algorithme, ce qui constitue une limitation de l'algorithme. D'autres travaux peuvent se concentrer sur l'extension de l'algorithme pour extraire des motifs constitués de sous-séquences discontinues (continues par morceaux). Une application de l'algorithme proposé pour la détection d'anomalie serait également intéressante à étudier. Dans certains cas, l'extraction de comportements atypiques inhabituels pourrait remplacer l'extraction des comportements habituels les plus fréquents (les motifs récurrents). L'algorithme proposé peut être utilisé pour la détection automatique d'anomalies.

Bibliographie personnelle

ARTICLE SOUMIS

A1 - A. Goudjil, M. Pouliquen, E. Pigeon, O. Gehan. *Identification Scheme for Switched Linear Systems in presence of Bounded Noise.* submitted to Asian Journal of Control. Novembre 2017

PUBLICATIONS DANS DES CONFÉRENCES INTERNATIONALES

C1 - A. Goudjil, M. Pouliquen, E. Pigeon, O. Gehan. *Identification Algorithm for MIMO Switched Output Error model in presence of Bounded Noise.* IEEE Conference on Decision and Control, Melbourne, Australia. Décembre 2017

C2 - A. Goudjil, M. Pouliquen, E. Pigeon, O. Gehan. *Recursive Output Error Identification Algorithm for Switched Linear systems with Bounded Noise.* The 20th IFAC World Congress, Toulouse, France. Juillet 2017

C3 - A. Goudjil, M. Pouliquen, E. Pigeon, O. Gehan. *Convergence Analysis of a Real-Time Identification Algorithm for Switched Linear Systems with Bounded Noise.* IEEE Conference on Decision and Control, Las Vegas, USA. Décembre 2016

C4 - M. Pouliquen, A. Goudjil, E. Pigeon, O. Gehan. *Continuous-Time System Identification using Binary Measurements.* IEEE Conference on Decision and Control, Las Vegas, USA. Décembre 2016

C5 - A. Goudjil, M. Pouliquen, E. Pigeon, O. Gehan. *A Real-Time Identification Algorithm for Switched linear Systems with Bounded Noise.* European Control Conference, Aalborg, Denmark. Juillet 2016

C6 - M. Pouliquen, T. Menard, E. Pigeon, O. Gehan, A. Goudjil. *Recursive System Identification Algorithm using Binary Measurements.* European Control Conference, Aalborg, Denmark. Juillet 2016

C7 - A. Goudjil, M. Pouliquen, E. Pigeon, O. Gehan. *Identification Algorithm for Piecewise Affine Systems with bounded disturbances.* The 24th Mediterranean Conference on Control and Automation, Athens, Greece. Juin 2016

C8 - E. Pigeon, J.B. Fabin, M. Pouliquen, B. Mauvieux, O. Gehan, T. Menard, A. Goudjil, S. Moussay. *Identification of circadian rhythm.* The 24th Mediterranean Conference on Control and Automation, Athens, Greece. Juin 2016

C9 - A. Goudjil, M. Pouliquen, E. Pigeon and O. Gehan. *Identification of systems using binary sensors via support vector machines.* IEEE Conference on Decision and Control, Osaka, Japon. Décembre 2015

RAPPORTS INTERNES

R1 - M. Pouliquen, A. Goudjil, E. Pigeon and O. Gehan. Projet CASSIE - rapport revue 1. Septembre 2015

R2 - M. Pouliquen, A. Goudjil, E. Pigeon and O. Gehan. Projet CASSIE - rapport revue 2. Septembre 2016

R3 - A. GOUDJIL. Génération du code C à partir du code Matlab. Septembre 2016

R4 - M. Pouliquen, A. Goudjil, E. Pigeon and O. Gehan. Projet CASSIE - rapport revue 3. Juillet 2017

AUTRES ACTIVITÉS

A. Goudjil. *Identification des systèmes à commutations en présence d'un bruit borné.* Présentation orale, journée Groupe de Travail Identification, Paris. 18 Mai 2017.

A. Goudjil. *Génération du code C à partir du code Matlab.* Présentation orale, séminaire Automatique, Caen. 30 Novembre 2016.

A. Goudjil. *Identification des systèmes ayant une sortie binaire via SVM.* Poster, journée annuelle du laboratoire GREYC, Caen. Juin 2015.

Bibliographie

[1] C. Aggarwal and C. Reddy. *Data clustering : algorithms and applications*. CRC Press, 2013.

[2] S. Aghabozorgi, A. Shirkhorshidi, and T. Wah. Time-series clustering–a decade review. *Information Systems*, 53 :16–38, 2015.

[3] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In *International Conference on Foundations of Data Organization and Algorithms, Chicago, USA*, 1993.

[4] E. Amaldi and M. Mattavelli. The MIN PFS problem and piecewise linear model estimation. *Discrete Applied Mathematics*, 118 :115–143, 2002.

[5] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2) :174–188, 2002.

[6] L. Bako. Identification of switched linear systems via sparse optimization. *Automatica*, 47(4) :668–677, 2011.

[7] L. Bako, K. Boukharouba, E. Duviella, and S. Lecoeuche. Switched affine models for describing nonlinear systems. In *IFAC Conference on Analysis and Design of Hybrid Systems, Zaragoza, Spain*, 2009.

[8] L. Bako, K. Boukharouba, E. Duviella, and S. Lecoeuche. A recursive identification algorithm for switched linear/affine models. *Nonlinear Analysis : Hybrid Systems*, 5(2) :242–253, 2011.

[9] L. Bako and S. Lecoeuche. A sparse optimization approach to state observer design for switched linear systems. *Systems and Control Letters*, 62(2) :143–151, 2013.

[10] L. Bako, G. Mercere, and S. Lecoeuche. On-line structured subspace identification with application to switched linear systems. *International Journal of Control*, 82(8) :1496–1515, 2009.

[11] L. Bako and R. Vidal. Algebraic identification of switched MIMO ARX models. In *Hybrid Systems : Control and Computation, St Louis, USA*, 2008.

[12] R. Baptista, J. Ishihara, and G. Borges. Split and merge algorithm for identification of piecewise affine systems. In *IEEE American Control Conference, San Francisco, USA*, 2011.

[13] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino. A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, 50(10) :1567–1580, 2005.

[14] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3) :407–427, 1999.

[15] K. Bennett and O. Mangasarian. Multicategory discrimination via linear programming. *Optimization Methods and Software*, 3 :27–39, 1994.

[16] P. Berkhin. A survey of clustering data mining techniques. *Grouping multidimensional data*, pages 25–71, 2006.

[17] J. Borges, V. Verdult, M. Verhaegen, and M. Botto. A switching detection method based on projected subspace classification. In *IEEE Conference on Decision and Control and European Control Conference, Seville, Spain*, 2005.

[18] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory, Pittsburgh*, 1992.

[19] K. Boukharouba. *Modélisation et classification de comportements dynamiques des systèmes hybrides*. Phd thesis, Université des Sciences et Technologie de Lille, France, 2011.

[20] K. Boukharouba, L. Bako, and S. Lecoeuche. Identification of piecewise affine systems based on Dempster-Shafer theory. In *IFAC Symposium on System Identification, Saint-Malo, France*, 2009.

[21] M. Boutayeb, Y. Becis, and M. Darouach. Recursive identification of linear multivariable systems with bounded disturbances. In *IFAC World Congress, Barcelona*, 2002.

[22] P. Bradley and O. Mangasarian. K-plane clustering. *Journal of Global optimization*, 16 :23–32, 2000.

[23] J. Bravo, T. Alamo, and E. Camacho. Bounded error identification of systems with time-varying parameters. *IEEE Transactions on Automatic Control*, 51(7) :1144–1150, 2006.

[24] A. Bruckstein, D. L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51(1) :34–81, 2009.

[25] J. Buhler and M. Tompa. Finding motifs using random projections. *Journal of computational biology*, 9(2) :225–242, 2002.

[26] C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2) :121–167, 1998.

[27] E. Candes, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted l1 minimization. *Journal Fourier analysis and Applications*, 14 :877–905, 2008.

[28] N. Canty, T. O'Mahony, and M. Cychowski. An output error algorithm for piecewise affine system identification. *Control Engineering Practice*, 20(4) :444–452, 2012.

[29] C. Canudas-De-Wit and J. Carrillo. A modified EW-RLS algorithm for systems with bounded noise. *Automatica*, 26(3) :599–606, 1990.

[30] R. Carloni, R. Sanfelice, A. Teel, and C. Melchiorri. A hybrid control strategy for robust contact detection and force regulation. In *American Control Conference, New York City, USA*, 2007.

[31] M. Casini, A. Garulli, and A. Vicino. On input design in l1 conditional set membership identification. *Automatica*, 42(5) :815–823, 2006.

[32] N. Castro and P. Azevedo. Multiresolution motif discovery in time series. In *SDM, Ohio, USA*, 2010.

[33] J. Catalano, T. Armstrong, and T. Oates. Discovering patterns in real-valued time series. In *European Conference on Principles of Data Mining and Knowledge Discovery, Berlin, Germany*, 2006.

[34] C. Chaitali. Optimizing clustering technique based on partitioning DBSCAN and ant clustering algorithm. *International Journal of Engineering and Advanced Technology*, 2(2) :212–215, 2012.

[35] T. Chen, Y. Zhao, and L. Ljung. Impulse response estimation with binary measurements : a regularized FIR model approach. In *IFAC Symposium on System Identification, Brussels, Belgium*, 2012.

[36] L. Chisci, A. Garulli, A. Vicino, and G. Zappa. Block recursive parallelotopic bounding in set membership identification. *Automatica*, 34(1) :15–22, 1998.

[37] L. Chisci, A. Garulli, and G. Zappa. Recursive state bounding by parallelotopes. *Automatica*, 32(7) :1049–1055, 1996.

[38] B. Chiu, E. Keogh, and S. Lonardi. Probabilistic discovery of time series motifs. In *Proceedings of the 9th International Conference on Knowledge discovery and data mining, Washington, USA*, 2003.

[39] F. Christophersen. *Optimal Control of Constrained Piecewise Affine Systems*, volume 359 of Lecture Notes in Control and Information Sciences. Springer Berlin Heidelberg, 2007.

[40] M. Chuah and F. Fu. ECG anomaly detection via time series analysis. In *International Symposium on Parallel and Distributed Processing and Applications, Niagara Falls, Canada*, 2007.

[41] E. Cinquemani, A. Milias-Argeitis, and J. Lygeros. Identification of genetic regulatory networks : A stochastic hybrid approach. In *IFAC World Congress, Seoul, Korea,*, 2008.

[42] E. Colinet and J. Juillard. A weighted least-squares approach to parameter estimation problems based on binary measurements. *IEEE Transactions on Automatic Control*, 55(1) :148–152, 2010.

[43] A. Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Proceedings IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, Vancouver, Canada*, 2001.

[44] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[45] C. Csaji and E. Weyer. Recursive estimation of ARX systems using binary sensors with adjustable thresholds. In *IFAC Symposium on System Identification, Brussels, Belgium*, 2012.

[46] S. Dasgupta and Y.-F. Huang. Asymptotically convergent modified recursive least square with data-dependent updating and forgetting factor for systems with bounded noise. *IEEE Transactions on Information Theory*, 33(3) :383–392, 1987.

[47] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[48] A. V. der Schaft and J. Schumacher. Complementarity modeling of hybrid systems. *IEEE Transactions on Automatic Control*, 43(4) :483–490, 1998.

[49] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data : experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2) :1542–1552, 2008.

[50] R. Duda, P. Hart, and D. Stork. Pattern classification and scene analysis 2nd ed. *ed : Wiley Interscience*, 1995.

[51] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *IEEE Conference on computer vision and pattern recognition, Miami beach, USA*, 2009.

[52] E. Elhamifar and R. Vidal. Sparse subspace clustering : Algorithms, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 :2765–2781, 2013.

[53] J. Ernst, G. Nau, and Z. Bar-Joseph. Clustering short time series gene expression data. *Bioinformatics*, 21(suppl 1) :i159–i168, 2005.

[54] P. Esling and C. Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1) :12, 2012.

[55] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. *Fast subsequence matching in time-series databases*, volume 23. ACM, 1994.

[56] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3) :37, 1996.

[57] G. Ferrari-Trecate and M. Muselli. Single-linkage clustering for optimal classification in piecewise affine regression. In *IFAC Conference on Analysis and Design of Hybrid Systems, Saint-Malo, France*, 2003.

[58] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2) :250–217, 2003.

[59] E. Fogel and Y. Huang. On the value of information in system identification - bounded noise case. *Automatica*, 18(2) :229–238, 1982.

[60] T. Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1) :164–181, 2011.

[61] H. Gamier and M. Mensler. Contsid— A continuous-time system identification toolbox for matlab®. In *European Control Conference, Karlsruhe, Germany*, 1999.

[62] H. Garnier. "Direct continuous-time approaches to system identification. overview and benefits for practical applications". *European Journal of Control*, 24(1) :50–62, 2015.

[63] H. Garnier, M. Gilson, and T. Bastogne. Identification de modèles paramétriques à temps continu. méthodes, outil logiciel et avantages. In *Journées Identification et Modélisation Expérimentale, Poitiers, France*, 2006.

[64] H. Garnier, M. Gilson, and V. Laurain. The contsid toolbox for matlab : extensions and latest developments. In *IFAC Symposium System Identification, Saint-Malo, France, 2009*, 2009.

[65] H. Garnier, M. Mensler, and A. Richard. Continuous-time model identification from sampled data : implementation issues and performance evaluation. *International Journal of Control*, 76(13) :1337–1357, 2003.

[66] H. Garnier and L. Wang. *Identification of continuous-time models from sampled data*. Springer-Verlag, London, 2008.

[67] A. Garulli, B. Kacewicz, A. Vicino, and G. Zappa. Error bounds for conditional algorithms in restricted complexity set membership identification. *IEEE Transactions on Automatic Control*, 45 :160–164, 2000.

[68] A. Garulli, S. Paoletti, and A. Vicino. A survey on switched and piecewise affine system identification. In *IFAC Symposium on System Identification, Brussels, Belgium*, 2012.

[69] A. Garullia, A. Vicino, and G. Zappa. Conditional central algorithms for worst-case set membership indentification and filtering. *IEEE Transactions on Automatic Control*, 45(1) :14–23, 2000.

[70] M. Gegundez, J. Aroba, and J. Bravo. Identification of piecewise affine systems by means of fuzzy clustering and competitive learning. *Engineering Applications of Artificial Intelligence*, 21(8) :1321–1329, 2008.

[71] A. Germani, C. Manes, and P. Palumbo. Simultaneous system identification and channel estimation : a hybrid system approach. In *Conference on Decision and Control, New Orleans, USA*, 2007.

[72] W. Glover and J. Lygeros. A stochastic hybrid model for air traffic control simulation. *Lecture notes in computer science*, 2993 :372–386, 2004.

[73] A. Goudjil, M. Pouliquen, E. Pigeon, and O. Gehan. Convergence analysis of a real-time identification algorithm for switched linear systems with bounded noise. In *IEEE Conference on Decision and Control, Las Vegas, USA*, 2016.

[74] A. Goudjil, M. Pouliquen, E. Pigeon, and O. Gehan. Identification algorithm for piecewise affine systems with bounded disturbances. In *The 24th Mediterranean Conference on Control and Automation, Athens, Greece.*, 2016.

[75] A. Goudjil, M. Pouliquen, E. Pigeon, and O. Gehan. A real-time identification algorithm for switched linear systems with bounded noise. In *European Control Conference, Aalborg, Denmark*, 2016.

[76] A. Goudjil, M. Pouliquen, E. Pigeon, and O. Gehan. Identification algorithm for MIMO switched output error model in presence of bounded noise. In *IEEE Conference on Decision and Control, Melbourne, Australia*, 2017.

[77] A. Goudjil, M. Pouliquen, E. Pigeon, and O. Gehan. Recursive output error identification algorithm for switched linear systems with bounded noise. In *The 20th IFAC World Congress, Toulouse, France*, 2017.

[78] A. Goudjil, M. Pouliquen, E. Pigeon, O. Gehan, and M. M'Saad. Identification of systems using binary sensors via support vector machines. In *IEEE Conference on Decision and Control, Osaka, Japon.*, 2015.

[79] Y. Guermeur. *SVM multiclasses, théorie et applications*. PhD thesis, Université Henri Poincaré-Nancy I, France, 2007.

[80] J. Guo and Y. Zhao. Recursive projection algorithm on FIR system identification with binary-valued observations. *Automatica*, 49(11) :3396–3401, 2013.

[81] J. Han, J. Pei, and M. Kamber. *Data mining : concepts and techniques*. Elsevier, 2011.

[82] J. Harris. *Algebraic geometry : a first course*, volume 133. Springer Science & Business Media, 2013.

[83] J. Hartigan and M. Wong. Algorithm as 136 : A K-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1) :100–108, 1979.

[84] A. Hartmann, J. Lemos, R. Costa, J. Xavier, and S. Vinga. Identification of switched ARX models via convex optimization and expectation maximization. *Journal of Process Control*, 28 :9–16, 2015.

[85] Y. Hashambhoy and R. Vidal. Recursive identification of switched ARX models with unknown number of models and unknown orders. In *IEEE Conference on Decision and Control, Seville, Spain*, 2005.

[86] V. Hautamaki, P. Nykanen, and P. Franti. Time-series clustering by approximate prototypes. In *19th International Conference on Pattern Recognition, Florida, USA*, 2008.

[87] S. Haykin and N. Network. A comprehensive foundation. *Neural Networks*, 2(2004) :41, 2004.

[88] A. Ingimundarson, J. Bravo, V. Puig, T. Alamo, and P. Guerra. Robust fault detection using zonotope-based set-membership consistency test. *International journal of adaptive control and signal processing*, 23(4) :311–330, 2009.

[89] H. Izakian, W. Pedrycz, and I. Jamal. Clustering spatiotemporal data : An augmented fuzzy c-means. *IEEE Transactions on Fuzzy Systems*, 21(5) :855–868, 2013.

[90] H. Izakian, W. Pedrycz, and I. Jamal. Fuzzy clustering of time series data using dynamic time warping distance. *Engineering Applications of Artificial Intelligence*, 39 :235–244, 2015.

[91] K. Jafari, J. Juillard, and E. Colinet. A recursive system identification method based on binary measurements. In *Control and Decision Conference, Atlanta, USA*, 2010.

[92] K. Jafari, J. Juillard, and M. Roger. "Convergence analysis of an online approach to parameter estimation problems based on binary observations". *Automatica*, 48(11) :2837–2842, 2012.

[93] A. Jain. Data clustering : 50 years beyond K-means. *Pattern recognition letters*, 31(8) :651–666, 2010.

[94] A. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[95] A. Jain, M. Murty, and P. Flynn. Data clustering : a review. *ACM computing surveys (CSUR)*, 31(3) :264–323, 1999.

[96] W. Jiang and H. Fang. Identification of switched linear systems via sparse optimization. *IFAC-PapersOnLine*, 48(28) :680–685, 2015.

[97] D. Joachim and J. D. Jr. Multiweight optimization in optimal bounding ellipsoid algorithms. *Signal Processing, IEEE Transactions on*, 54(2) :679–690, 2006.

[98] K. Johansson, J. Lygeros, and S. Sastry. Modeling of hybrid systems. *in Encyclopedia of Life Support Systems*, 2004.

[99] A. Juloski. *Observer design and identification methods for hybrid systems.* Phd thesis, Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands, 2004.

[100] A. Juloski, W. Heemels, G. Ferrari-Trecate, R. Vidal, S. Paoletti, and J. Niessen. Comparison of four procedures for the identification of hybrid systems. *In Hybrid systems : Computation and Control, Zurich, Switzerland*, 2005.

[101] A. Juloski and S. Weiland. A bayesian approach to the identification of piecewise linear output error models. In *IFAC Symposium on System Identification, Newcastle, Australia*, 2006.

[102] A. Juloski, S. Weiland, and W. Heemels. A bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, 50(10) :1520–1533, 2005.

[103] L. Kaufman and P. J. Rousseeuw. *Finding groups in data : an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[104] D. Keim, G. Andrienko, J. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. Visual analytics : Definition, process, and challenges. *Lecture notes in computer science*, 4950 :154–176, 2008.

[105] E. Keogh and J. Lin. Clustering of time-series subsequences is meaningless : implications for previous and future research. *Knowledge and Information Systems*, 8(2) :154, 2005.

[106] E. Keogh, J. Lin, and A. Fu. Hot sax : Efficiently finding the most unusual time series subsequence. In *International Conference on Data Mining, Houston, USA*, 2005.

[107] E. Keogh and M. Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In *KDD*, 1998.

[108] E. Keogh and C. Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7(3) :358–386, 2005.

[109] A. Khakimova, A. Shamshimova, D. Sharipova, A. Kusatayeva, V. Ten, A. Bemporad, Y. Familiant, A. Shintemirov, and M. Rubagotti. Hybrid model predictive control for optimal energy management of a smart house. In *15th International Conference on Environment and Electrical Engineering, Rome, Italy*, 2015.

[110] K. Kim, K. Jung, and J. Kim. Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift al-

gorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12) :1631–1639, 2003.

[111] N. Krishnan and D. Cook. Activity recognition on streaming sensor data. *Pervasive and mobile computing*, 10 :138–154, 2014.

[112] N. Krishnan and D. Cook. Activity recognition on streaming sensor data. *Pervasive and mobile computing*, 10 :138–154, 2014.

[113] R. Kuang, E. Ie, K. Wang, K. Wang, M. Siddiqi, Y. Freund, and C. Leslie. Profile-based string kernels for remote homology detection and motif extraction. *Journal of bioinformatics and computational biology*, 3(03) :527–550, 2005.

[114] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *Proceedings IEEE International Conference on Data Mining, San Jose, USA*, 2001.

[115] O. Lara and M. Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys & Tutorials*, 15(3) :1192–1209, 2013.

[116] Z. Lassoued and K. Abderrahim. A new clustering technique for the identification of PWARX hybrid models. *IEEE Asian Control Conference, Istanbul, Turkey*, 2013.

[117] Z. Lassoued and K. Abderrahim. New results on PWARX model identification based on clustering approach. *International Journal of Automation and Computing*, 11(2) :180–188, 2014.

[118] F. Lauer, G. Bloch, and R. Vidal. A continuous optimization framework for hybrid system identification. *Automatica*, 47(3) :608–613, 2011.

[119] J. Lee, S. Bohacek, J. Hespanha, and K. Obraczka. Modeling communication networks with hybrid systems. *IEEE/ACM Transactions on Networking*, 15(3) :630–643, 2007.

[120] Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99 :67–81, 2004.

[121] B. Lennartson, M. Tittus, B. Egardt, and S. Pettersson. Hybrid systems in process control. *IEEE Control Systems*, 16(5) :45–56, 1996.

[122] T. Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11) :1857–1874, 2005.

[123] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th workshop on Research issues in data mining and knowledge discovery, California, USA*, 2003.

[124] J. Lin, E. Keogh, L. Wei, and S. Lonardi. Experiencing SAX : a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2) :107–144, 2007.

[125] J. Lin, S. Williamson, K. Borne, and D. DeBarr. Pattern recognition in time series. *Advances in Machine Learning and Data Mining for Astronomy*, 1 :617–645, 2012.

[126] L. Ljung. *System Identification - Theory for the User*. Prentice-Hall, 1999.

[127] B. Logan. Mel frequency cepstral coefficients for music modeling. In *ISMIR*, 2000.

[128] J. Lonardi and P. Patel. Finding motifs in time series. In *Processing of the 2nd Workshop on Temporal Data Mining, Alberta, Canada*, 2002.

[129] Y. Ma and R. Vidal. Identification of deterministic switched ARX systems via identification of algebraic varieties. *International Conference on Hybrid Systems : Computation and Controll, Zurich, Switzerland*, 2005.

[130] E. Maharaj and P. D'Urso. Fuzzy clustering of time series in the frequency domain. *Information Sciences*, 181(7) :1187–1211, 2011.

[131] E. Maharaj and P. D'Urso. Fuzzy clustering of time series in the frequency domain. *Information Sciences*, 181(7) :1187–1211, 2011.

[132] O. Maimon and L. Rokach. *Data mining and knowledge discovery handbook*. Springer, 2005.

[133] I. Maruta and T. Sugie. Identification of PWA models via data compression based on l_1 optimization. In *Conference on Decision and Control and European Control Conference, Orlando, USA*, 2011.

[134] H. Mei, L. Wang, and G. Yin. Almost sure convergence rates for system identification using binary, quantized, and regular sensors. *Automatica*, 50(11) :2120–2127, 2014.

[135] M. Milanese and G. Belforte. Estimation theory and uncertainty intervals evaluation in presence of unknown but bounded errors : Linear families of models and estimators. *IEEE Transactions on Automatic Control*, 27(2) :408–414, 1982.

[136] M. Milanese and A. Vicino. Estimation theory for nonlinear models and set membership uncertainty. *Automatica*, 27(2) :403–408, 1991.

[137] D. Minnen, T. Starner, I. Essa, and C. Isbell. Discovering characteristic actions from on-body sensor data. In *IEEE international symposium on wearable computers, Montreux, Switzerland*, 2006.

[138] C. Möller-Levet, F. Klawonn, K. Cho, and O. Wolkenhauer. Fuzzy clustering of short time-series and unevenly distributed sampling points. In *International Symposium on Intelligent Data Analysis, Berlin, Germany*, 2003.

[139] A. Mueen and N. Chavoshi. Enumeration of time series motifs of all lengths. *Knowledge and Information Systems*, 45(1) :105–132, 2015.

[140] A. Mueen and E. Keogh. Online discovery and maintenance of time series motifs. In *Proceedings of the 16th International Conference on Knowledge discovery and data mining, Washington, USA*, 2010.

[141] J. Nakache and J. Confais. *Approche pragmatique de la classification : arbres hiérarchiques, partitionnements*. Editions Technip, 2004.

[142] H. Nakada, K. Takaba, and T. Katayama. Identification of piecewise affine systems based on statistical clustering technique. *Automatica*, 41(5) :905–913, 2005.

[143] S. Nanda, B. Mahanty, and M. Tiwari. Clustering indian stock market data for portfolio management. *Expert Systems with Applications*, 37(12) :8793–8798, 2010.

[144] M. Naphade and T. Huang. Discovering recurrent events in video using unsupervised methods. In *International Conference on Image Processing, New York, USA*, 2002.

[145] G. Nychis, V. Sekar, D. Andersen, H. Kim, and H. Zhang. An empirical evaluation of entropy-based traffic anomaly detection. In *Proceedings of the 8th Conference on Internet measurement, Vouliagmeni, Greece*, 2008.

[146] H. Ohlsson and L. Ljung. Identification of piecewise affine systems using sum-of-norms regularization. In *18th IFAC World Congress, Milan, Italy*, 2011.

[147] H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX models using sum-of-norms regularization. *Automatica*, 46(6) :1107–1111, 2010.

[148] N. Ozay, C. Lagoa, and M. Sznaier. Set membership identification of switched linear systems with known number of subsystems. *Automatica*, 51(1) :180–191, 2015.

[149] N. Ozay, M. Sznaier, C. Lagoa, and O. Camps. A sparsification approach to set membership identification of switched affine systems. *IEEE Transactions on Automatic Control*, 57(3) :634–648, 2012.

[150] S. Paoletti. *Identification of piecewise affine models*. Phd thesis, University of Siena, Siena, Italy, 2004.

[151] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal. Identification of hybrid systems : a tutorial. *European Journal of Control*, 13(1) :242–260, 2007.

[152] K. Pekpe, G. Mourot, K. Gasso, and J. Ragot. Identification of switching systems using change detection technique in the subspace framework. In *IEEE Conference on Decision and Control, Paradise Island, Bahamas*, 2004.

[153] F. Petitjean, A. Ketterlin, and P. Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, 44(3) :678–693, 2011.

[154] R. Porreca, S. Drulhe, H. de Jong, and G. Ferrari-Trecate. Identification of parameters and structure of piecewise affine models of genetic networks. In *IFAC Symposium on System Identification, Saint-Malo, France*, 2009.

[155] R. Porreca and G. Ferrari-Trecate. Identification of piecewise affine models of genetic regulatory networks : the data classification problem. In *IFAC World Congress, Seoul, Korea*, 2008.

[156] M. Pouliquen, A. Goudjil, O. Gehan, and E. Pigeon. Continuous-time system identification using binary measurements. In *IEEE Conference on Decision and Control, Las Vegas, USA*, 2016.

[157] M. Pouliquen, T. Menard, E. Pigeon, O. Gehan, and A. Goudjil. Recursive system identification algorithm using binary measurements. In *European Control Conference, Aalborg, Denmark*, 2016.

[158] M. Pouliquen, E. Pigeon, and O. Gehan. Output Error identification for multi-input multi-output systems with bounded disturbances. In *IEEE Conference on Decision and Control - European Control Conference, Orlando, USA*, 2011.

[159] M. Pouliquen, E. Pigeon, and O. Gehan. Bounded-error identification for closed-loop systems. *Automatica*, 50(2) :1884 – 1890, 2014.

[160] M. Pouliquen, E. Pigeon, and O. Gehan. Identification scheme for Hammerstein Output Error models with bounded noise. *IEEE Transactions on Automatic Control*, 61(2) :550 – 555, 2016.

[161] M. Prandini and J. Hu. Application of reachability analysis for stochastic hybrid systems to aircraft conflict prediction. *IEEE Transactions on Automatic Control*, 54(4) :913–917, 2009.

[162] J. Quinlan. Induction of decision trees. *Machine learning*, 1(1) :81–106, 1986.

[163] A. Rao and Y. Huang. Recent developments in optimal bounding ellipsoidal parameter estimation. *Mathematics and Computers in Simulation*, 32 :515–526, December 1990.

[164] G. Rao and H. Unbehauen. "Identification of continuous-time systems". *IEE Proceedings - Control Theory and Applications*, 153(2) :185–220, 2006.

[165] P. Rodrigues, J. Gama, and J. Pedroso. Hierarchical clustering of time-series data streams. *IEEE transactions on knowledge and data engineering*, 20(5) :615–627, 2008.

[166] J. Roll, A. Bemporad, and L. Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1) :37–50, 2004.

[167] J. Sander, M. Ester, H. Kriegel, and X. Xu. Density-based clustering in spatial databases : The algorithm gdbscan and its applications. *Data mining and knowledge discovery*, 2(2) :169–194, 1998.

[168] T. Schlegl, M. Buss, and G. Schmidt. A hybrid systems approach toward modeling and dynamical simulation of dextrous manipulation. *IEEE/ASME Transactions on Mechatronics*, 8(3) :352–361, 2003.

[169] B. Scholkopf and A. Smola. *Learning with kernels-Support vector machines, Regularization, Optimisation, and Beyond*. Massachusetts Institue of Technology , Cambridge, 2002.

[170] A. Shah and D. Adhyaru. Parameter identification of PWARX models using fuzzy distance weighted least squares method. *Applied Soft Computing*, 25 :174–183, 2014.

[171] A. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3) :199–222, 2004.

[172] R. Sokal and F. J. Rohlf. The comparison of dendrograms by objective methods. *Taxon*, 11(2) :33–40, 1962.

[173] M. Soltanolkotabi, E. Elhamifar, and E. Candes. Robust subspace clustering. *The Annals of Statistics*, 42(2) :669–699, 2014.

[174] R. Stanković and B. J. Falkowski. The haar wavelet transform : its status and achievements. *Computers & Electrical Engineering*, 29(1) :25–44, 2003.

[175] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Workshop on Artificial Intelligence for Web Search, Texas, USA*, 2000.

[176] Z. Struzik and A. Siebes. The haar wavelet transform in the time series similarity paradigm. In *European Conference on Principles of Data Mining and Knowledge Discovery, Prague, Czech Republic*, 1999.

[177] Z. Sun and S. Ge. *Switched linear systems : Control and design*. Springer Science & Business Media, 2006.

[178] M. TabatabaeiPour, K. Salahshoor1, and B. Moshiri. A K-plane clustering algorithm for identification of hybrid systems. In *International Conference on Automatic Control, Modeling and Simulation, Prague, Czech Republic*, 2006.

[179] G. Tan, C. Wen, and Y. Soh. Identification for systems with bounded noise. *IEEE Transactions on Automatic Control*, 42(7) :998–1001, 1997.

[180] F. Torrisi and A. Bemporad. Hysdel-a tool for generating computational hybrid models for analysis and synthesis problems. *IEEE transactions on control systems technology*, 12(2) :235–249, 2004.

[181] H. Unbehauen and G. Rao. Continuous-time approaches to system identification—a survey. *Automatica*, 26(1) :23–35, 1990.

[182] V. Vapnik. Pattern recognition using generalized portrait method. *Automation and remote control*, 24 :774–780, 1963.

[183] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Series in Statistics, 1995.

[184] V. Vapnik and C. Cortes. Support-vector networks. *Machine Learning*, 20(3) :273–297, 1995.

[185] V. Verdult and M. Verhaegen. Subspace identification of piecewise linear systems. In *IEEE Conference on Decision and Control, Paradise Island, Bahamas*, 2004.

[186] A. Vicino and G. Zappa. Sequential approximation of feasible parameter sets for identification with set membership uncertainty. *IEEE Transactions on Automatic Control*, 41(6) :774–785, 1996.

[187] R. Vidal. Identification of PWARX hybrid models with unknown and possibly different orders. In *IEEE American Control Conference, Boston, USA*, 2004.

[188] R. Vidal. Recursive identification of switched ARX systems. *Automatica*, 44(9) :2274–2287, 2008.

[189] R. Vidal, A. Chuiso, and S. Soatto. Observability and identifiability of jump linear systems. In *IEEE Conference on Decision and Control, Las Vegas, USA*, 2002.

[190] R. Vidal and Y. Ma. A unified algebraic approach to 2-D and 3-D motion segmentation. *Journal of Mathematical Imaging and Vision*, 25(3) :403–421, 2006.

[191] R. Vidal, Y. Ma, and S. Sastry. *Generalized Principal Component Analysis*, volume 40. Springer New York, 2016.

[192] R. Vidal, S. Soatto, and A. Chiuso. Applications of hybrid system identification in computer vision. In *European Control Conference, Kos, Greece*, 2007.

[193] R. Vidal, S. Soatto, Y. Ma, and S. Sastry. An algebraic geometric approach to the identification of a class of linear hybrid systems. In *IEEE Conference on Decision and Control, Maui, Hawaii, USA*, 2003.

[194] M. Vilela, C. Borges, S. Vinga, A. Vasconcelos, H. Santos, E. Voit, and J. Almeida. Automated smoother for the numerical decoupling of dynamics models. *BMC Bioinform*, 8, 2007.

[195] M. Vlachos, J. Lin, E. Keogh, and D. Gunopulos. A wavelet-based anytime algorithm for k-means clustering of time series. In *proceedings Workshop on Clustering High Dimensionality Data and Its Applications, San Francisco, USA*, 2003.

[196] E. Voit and J. Almeida. Decoupling dynamical systems for pathway identification from metabolic profiles. *Bioinformatics (Oxford, England)*, 20(11) :1670–1681, 2004.

[197] V. Vuori and J. Laaksonen. A comparison of techniques for automatic clustering of handwritten characters. In *16th International Conference on Pattern Recognition, Quebec, Canada*, 2002.

[198] J. Wang and T. Chen. Online identification of switched linear output error models. In *IEEE International Symposium on Computer-Aided Control System Design, Denver, USA*, 2011.

[199] L. Wang, T. Gu, X. .Tao, and J. Lu. A hierarchical approach to real-time activity recognition in body sensor networks. *Pervasive and Mobile Computing*, 8(1) :115–130, 2012.

[200] L. Wang, G. Yin, and J. Zhang. Joint identification of plant rational models and noise distribution functions using binary-valued observations. *Automatica*, 42(4) :535–547, 2006.

[201] L. Wang, G. Yin, J. Zhang, and Y. Zhao. *System identification with quantized observations*. Springer, 2010.

[202] L. Wang, G. Yin, Y. Zhao, and J. Zhang. Identification input design for consistent parameter estimation of linear systems with binary-valued output observations. *IEEE Transactions on Automatic Control*, 53(4) :867–880, 2008.

[203] L. Wang, J. Zhang, and G. Yin. System identification using binary sensors. *IEEE Transactions on Automatic Control*, 48(11) :1892–1907, 2003.

[204] X. Wang, K. Smith, and R. Hyndman. Characteristic-based clustering for time series data. *Data mining and knowledge Discovery*, 13(3) :335–364, 2006.

[205] D. Yankov, E. Keogh, J. Medina, B. Chiu, and V. Zordan. Detecting time series motifs under uniform scaling. In *Proceedings of the 13th International Conference on Knowledge discovery and data mining, San Jose, USA*, 2007.

[206] P. Young. "Parameter estimation for continuous-time models - a survey". *Automatica*, 17(1) :23–39, 1981.

[207] P. Young, C. J. Taylor, W. Tych, D. Pedregal, and P. G. McKenna. The captain toolbox. *Centre for Research on Environmental Systems and Statistics*, 2010.

[208] P. Young, W. Tych, and C. J. Taylor. The captain toolbox for matlab. In *IFAC Symposium System Identification, Saint-Malo, France, 2009*, 2009.

[209] X. Zhang, J. Liu, Y. Du, and T. Lv. A novel clustering method on time series data. *Expert Systems with Applications*, 38(9) :11891–11900, 2011.

[210] N. Zhong, Y. Li, and S. Wu. Effective pattern discovery for text mining. *IEEE transactions on knowledge and data engineering*, 24(1) :30–44, 2012.

[211] Y. Zhu. *Multivariable system identification for process control*. Elsevier, 2001.

[212] S. Zolhavarieh, S. Aghabozorgi, and Y. Teh. A review of subsequence time series clustering. *The Scientific World Journal*, 2014, 2014.

Résumé

Les travaux de cette thèse portent sur l'identification des systèmes et l'extraction de motifs à partir de données. Dans le cadre de l'identification des systèmes, nous nous intéressons plus précisément à l'identification des systèmes dynamiques hybrides et l'identification des systèmes dynamiques linéaires ayant une sortie binaire. Deux classes très populaires des systèmes hybrides sont les systèmes linéaires à commutations et les systèmes affines par morceaux. Nous faisons tout d'abord un état de l'art sur les méthodes d'identification de ces deux classes. Nous proposons ensuite un algorithme basé sur une méthode d'identification de type OBE "Outer Bounding Ellipsoid" pour l'identification en temps réel des systèmes à commutations soumis à un bruit borné. Nous présentons ensuite plusieurs extensions de l'algorithme soit pour l'identification des systèmes affines par morceaux, l'identification des systèmes à commutations décrits par un modèle du type erreur de sortie et l'identification des systèmes MIMO à commutations. Nous abordons ensuite le problème d'identification des systèmes linéaires ayant une sortie binaire en introduisant un point de vue original consiste à formuler le problème d'identification comme un problème de classification. Ceci permet de proposer deux algorithmes d'identification basés sur l'utilisation des SVMs. Le premier algorithme est dédié à l'identification des systèmes à temps discret et le deuxième algorithme est dédié à l'identification des systèmes à temps continu. Dans le cadre de l'extraction de motifs, nous présentons dans un premier temps un état de l'art sur les algorithmes d'extraction de motifs et sur les techniques de la classification non supervisée. Ensuite, nous proposons un algorithme d'extraction de motifs à partir des données basé sur des techniques de classification non supervisée.

Mots clés : Identification, systèmes à commutations, systèmes affines par morceaux, bruit borné, sortie binaire, motifs.

Summary

In this thesis, we deal with the identification of systems and the extraction of patterns from data. In the context of system identification, we focus precisely on the identification of hybrid systems and the identification of linear systems using binary sensors. Two very popular classes of hybrid systems are switched linear systems and piecewise affine systems. First, we give an overview of the different approaches available in the literature for the identification of these two classes. Then, we propose a new real-time identification algorithm for switched linear systems, it's based on an Outer Bounding Ellipsoid (OBE) type algorithm suitable for system identification with bounded noise. We then present several extensions of the algorithm either for the identification of piecewise affine systems, the identification of switched linear systems described by an output error model and the identification of MIMO switched linear systems. After this, we address the problem of the identification of linear systems using binary sensors by introducing an original point of view. We formulate the identification problem as a classification problem. This formulation allows the use of supervised learning algorithms such as Support Vector Machines (SVMs) for the identification of discrete time systems and the identification of continuous-time systems using binary sensors. In the context of pattern extraction, we first present an overview of the different pattern extraction algorithms and clustering techniques available in the literature. Next, we propose an algorithm for extracting patterns from data based on clustering techniques.

Key words : Identification, switched systems, piecewise affine systems, bounded noise, binary sensors, pattern