UNIVERSITE LILLE 1 - SCIENCES ET TECHNOLOGIES **Numéro d'ordre : XXXX**

Modélisation et classification de comportements dynamiques des systèmes hybrides

THÈSE

présentée et soutenue publiquement le 12 juillet 2011

pour l'obtention du

Doctorat de l'Université Lille 1 - Sciences et Technologies

(spécialité Automatique, Génie Informatique, Traitement du Signal et des Images)

par

Khaled BOUKHAROUBA

Composition du jury

Président :	L. Belkoura	MCF HdR, Université Lille 1
Rapporteurs :	G. Bloch	Professeur des Universités, Université Nancy 1
	A. Rakotomamonjy	Professeur des Universités, Université de Rouen
Examinateurs :	D. Lefebvre	Professeur des Universités, Université du Havre
	L. Belkoura	MCF HdR, Université Lille 1
	Y. Delignon	Professeur des Universités, TELECOM Lille 1
Directeur de thèse :	S. Lecœuche	Professeur des Écoles des Mines, Mines de Douai
Co-encadrant :	L. Bako	Maître Assistant, Mines de Douai

École des Mines de Douai, Unité de Recherche en Informatique et Automatique (URIA).

Mis en page avec la classe thloria.

Remerciements

Je souhaite remercier en premier lieu mon directeur de thèse, **Stéphane Lecœuche**, Adjoint Chargé de la Recherche - Directeur de l'URIA, pour m'avoir accueilli au sein du département « IA ». Je lui suis également reconnaissant pour sa disponibilité, son aide précieuse et ses qualités pédagogiques et scientifiques. J'ai beaucoup appris à ces côtés et je lui adresse toute ma gratitude.

J'adresse mes vifs remerciements à mon co-encadrant de thèse, **Laurent Bako**, pour m'avoir conseillé, encouragé et soutenu avec patience et disponibilité. Je salue ses qualités scientifiques et ses conseils judicieux apportés tout au long de cette thèse.

J'exprime ma sincère reconnaissance à Monsieur **Gérard Bloch**, Professeur à l'Université Nancy 1, ainsi qu'à Monsieur **Alain Rakotomamonjy**, Professeur à l'Université de Rouen, pour avoir accepté de juger ce travail en qualité de rapporteurs au sein du jury. Je remercie également Messieurs **Dimitri Lefebvre**, **Lotfi Belkoura** et **Yves Delignon** de m'avoir fait l'honneur de participer au jury en tant qu'examinateurs.

Je remercie mes amis et collègues du département Informatique et Automatique, pour l'ambiance conviviale qu'ils ont contribuée à entretenir, les bons moments passés en leur compagnie et leur sympathie. En particulier, je tiens à saluer Lyes, Moussa, Dulin, Ines, Aznul, Eric, Christiane et Frédéric.

Enfin, mes remerciements les plus chaleureux vont à mes parents, mon frère et mes sœurs, ainsi qu'à toute ma famille, pour leur soutien moral et leurs encouragements.

A mes parents.
A mon frère et mes sœurs.

Table des matières

Table des figures

Introduction générale

1 Contexte et motivation

De façon générale, un système dynamique peut être défini comme une entité qui interagit avec son environnement ou avec d'autres systèmes suivant des entrées et sorties bien définies. Pour être en mesure d'analyser et de contrôler ce système, on a besoin d'un modèle décrivant la relation entre ses entrées et ses sorties. Un modèle est alors une description (plus ou moins approximative) du système. Il reflète le plus fidèlement possible le comportement du système, c'est-à-dire qu'il est capable de prédire la sortie du système s'il subit l'influence d'une entrée donnée. Un modèle doit en général vérifier trois propriétés [75], qui sont : (i) la simplicité dans la compréhension, l'interprétation et l'utilisation du modèle, (ii) la précision dans la description et la prédiction de la dynamique du système, (iii) la capacité à rendre compte complètement du comportement du système. Malheureusement, il existe un conflit inhérent entre ces trois propriétés idéales, c'est-à-dire qu'un modèle très précis sera en général complexe et peu généralisable, ce qui nous oblige à trouver un compromis entre ces trois propriétés.

Dans le cas de la modélisation de systèmes complexes, construire un seul modèle mathématique est susceptible de déboucher sur une structure non-linéaire très compliquée qui peut être très difficile à exploiter en pratique. Une alternative serait alors de recourir à des approches d'identification favorisant les trois propriétés citées précédemment. Les systèmes dynamiques hybrides (SDH) [71] sont un concept rigoureux pour modéliser des systèmes complexes. Les systèmes dynamiques hybrides sont des systèmes dynamiques faisant intervenir simultanément des phénomènes de type dynamique continue et événementielle. L'exemple classique est le cas des processus continus supervisés. La commutation d'un mode de fonctionnement à un autre est régie par l'évolution de variables internes du système ou par une loi externe non maîtrisée.

Dans cette thèse, nous nous intéressons à l'identification d'une classe particulière de systèmes dynamiques hybrides, celle des systèmes dynamiques affines par morceaux (PieceWise Affine - PWA en anglais). Cette classe est probablement l'une des plus impor-

tantes des SDH. C'est une collection de modèles linéaires/affines partageant un même état continu, reliés par des commutateurs déterminés par une partition polyédrique de l'espace décrit à partir de l'état et de l'entrée [71]. Les PWA sont utilisés pour modéliser un grand nombre de processus physiques. Ils ont été introduits en vue d'approximer des dynamiques non linéaires via la linéarisation du comportement non linéaire en différents points de fonctionnement autour desquels des modèles linéaires peuvent être définis.

Le travail de thèse présenté dans ce manuscrit s'inscrit dans la continuité des travaux développés par notre équipe, à savoir le développement de techniques d'identification récursive et de techniques de classification dynamique. Ces techniques ont pour objectif d'assurer le suivi de systèmes complexes modélisés sous la forme de systèmes à plusieurs modes de fonctionnements.

2 Contribution

La contribution de cette thèse comporte essentiellement les points suivants :

- La proposition d'une nouvelle approche d'identification des modèles dynamiques affines par morceaux. Cette approche est basée sur une technique innovante de classification non supervisée combinée avec une technique de régression linéaire pour grouper les données de régression relatives au même sous-modèle affine. Elle consiste à alterner l'assignation des données de régression à un sous-modèle et l'estimation des paramètres de ce sous-modèle [19]. L'assignation des données est ensuite améliorée par l'utilisation d'une classification de données basée sur une modélisation de connaissances incertaines exploitant la théorie de Dempster-Shafer [15]. Cette théorie permet de représenter explicitement, à partir d'outils mathématiques, l'incertitude liée aux connaissances.
- La proposition d'un algorithme récursif d'identification de modèles affines par morceaux et de modèles non linéaires par morceaux dont les paramètres des sous-modèles et des régions de validité peuvent varier dans le temps. Une technique de régression par les LS-SVM (Least Squares-Support Vector Machines) récursifs permettant l'adaptation des fonctions de régression après l'incorporation ou le retrait d'une donnée sera ainsi proposée. L'adaptation des paramètres des régions de validité variant dans le temps sera assurée par un nouvel algorithme de classification incrémentale et décrémentale multi-classe à vecteurs de support [16]. Grâce à deux procédures incrémentale et décrémentale, cet algorithme est capable d'estimer de façon récursive les paramètres des frontières séparatrices délimitant les régions et donc de permettre le suivi de l'évolution des régions dans le temps.
- La validation de techniques développées dans ce manuscrit sur plusieurs types d'ap-

plications. Nous procédons d'abord à la modélisation d'un système hydraulique à surface libre [20] puis à l'identification d'une machine de montage de composants électroniques sur circuits imprimés [15]. Ensuite, nous montrons comment la segmentation temporelle de vidéos en différentes scènes peut être effectuée en se basant sur une estimation de sous-modèles linéaires locaux, chaque sous-modèle dynamique correspondant à une scène [18] [17]. Enfin, une approche de classification automatique et de reconnaissance de visages sous différents seuils d'éclairage par l'estimation de sous-modèles dynamiques locaux sera proposée.

3 Organisation

Ce mémoire, décomposé en cinq chapitres, est organisé de la façon suivante :

Chapitre 1

Le premier chapitre est dédié à un état de l'art sur l'identification des systèmes dynamiques hybrides. Nous dressons d'abord quelques formulations du problème d'identification pour la classe des modèles dynamiques affines par morceaux (PWA). Nous décrivons, ensuite, les méthodes d'identification des modèles PWA, qui sont disponibles dans la littérature. Puis, nous détaillons plus particulièrement les méthodes qui sont basées sur les techniques de classification [35][67] et sur la régression par les machines à vecteurs de support [54]. Nous montrons, ainsi, les avantages et les inconvénients de chaque méthode suivant les hypothèses faites sur le nombre de sous-modèles, les ordres du système, le coût de calcul, les connaissances *a priori* sur le système, etc.

Chapitre 2

Ce chapitre traite le problème de classification des données dans le but de les séparer selon leurs sous-modèles affines respectifs et d'estimer les paramètres associés à ces sous-modèles. Des algorithmes sont alors développés afin d'effectuer cette tâche. Les algorithmes proposés rentrent dans la catégorie des méthodes basées sur les techniques de classification. A part les ordres du système, nos algorithmes ne nécessitent aucune connaissance *a priori* sur la nature des données ni sur le nombre de sous-modèles. De plus, le nombre de paramètres nécessaires à leur utilisation est réduit à un seul paramètre.

Chapitre 3

Ce chapitre est consacré à l'estimation de la loi de commutation dans le cas des PWA. Cette loi est définie par un partitionnement polyédrique de l'ensemble des données de régression. Chaque sous-modèle est alors valide dans une région bien définie. L'estimation de ces régions constitue l'estimation de l'état discret de la classe PWA des systèmes dynamiques hybrides. Nous utilisons, dans ce chapitre, deux catégories de classification pour effectuer cette tâche. La première catégorie combine des séries de classifieurs SVM binaires en faisant appel à des schémas de décomposition du type "un-contre-reste" ou "un-contre-un". La deuxième catégorie consiste à résoudre le problème multi-classe en une seule étape sans le décomposer en un ensemble de sous-problèmes binaires.

Chapitre 4

Ce chapitre traite de l'identification récursive de modèles affines par morceaux ou de modèles non linéaires par morceaux dont les paramètres des sous-modèles et des régions peuvent varier dans le temps. Un algorithme générique d'identification récursive est alors proposé. Des procédures permettant d'adapter les paramètres définissant chaque sous-modèle aussi bien dans le cas de sous-modèles linéaires que dans le cas de sous-modèles non linéaires sont présentés. Un classifieur incrémental et décrémental multi-classe à vecteurs de support est également proposé pour l'estimation en ligne des frontières séparatrices des régions de validité. Les procédures d'adaptation permettent de prendre en compte les modifications locales et le suivi de l'évolution des modes au travers de la mise à jour de leurs paramètres.

Chapitre 5

Ce chapitre est réservé à la validation expérimentale. Plusieurs processus sont étudiés et modélisés. Tout d'abord, les approches proposées sont appliquées à la modélisation de systèmes hydrauliques à surface libre. Ces systèmes, caractérisés par des dynamiques non linéaires, sont soumis à de larges plages de fonctionnement. Ensuite, une machine de montage de composants électroniques sur circuits imprimés est modélisée par des modèles dynamiques affines par morceaux puis par des modèles dynamiques non linéaires par morceaux. Puis, une nouvelle méthode de segmentation temporelle de vidéos en différentes scènes basée sur une estimation de sous-modèles locaux linéaires est proposée. Chaque sous-modèle correspond à une scène. La procédure de segmentation basée sur l'estimation de sous-modèles locaux est appliquée à trois types de séquences vidéo. Enfin, ce chapitre s'achève par la proposition d'une dernière application. Cette application concerne la clas-

sification automatique et la reconnaissance de visages sous différents seuils d'éclairage par l'estimation de sous-modèles dynamiques locaux.

1

Un état de l'art sur l'identification des systèmes dynamiques hybrides

Sommaire

1.1 Introduction

On appelle système dynamique hybride (SDH), tout système faisant intervenir simultanément des phénomènes de type dynamique continue et événementielle. Les SDH sont régis par des équations différentielles continues et des variables discrètes. Ils résultent généralement de l'interaction entre des algorithmes discrets de planification et des algorithmes continus de commande (voir Fig. 1.1). Par exemple, dans plusieurs systèmes industriels, les stratégies de décision et de coordination de haut niveau sont modifiées par les résultats des processus physiques de bas niveau. De nombreux exemples de systèmes dynamiques hybrides peuvent être trouvés dans la vie réelle : les systèmes embarqués, où l'on notera une interaction entre la partie discrète (le programme) et la partie physique (le système), les chaînes de production industrielle, les réseaux de télécommunication, les systèmes de gestion de trafic aérien, etc. Les variables discrètes peuvent être régies par l'évolution de variables internes du système ou par une loi externe non maitrisée. En réalité, les commutations entre les modèles ne sont pas rigoureusement instantanées. Elles sont considérées comme telles parce que leur durée est très inférieure aux durées d'évolution des phénomènes physiques continus mis en jeu dans le SDH.

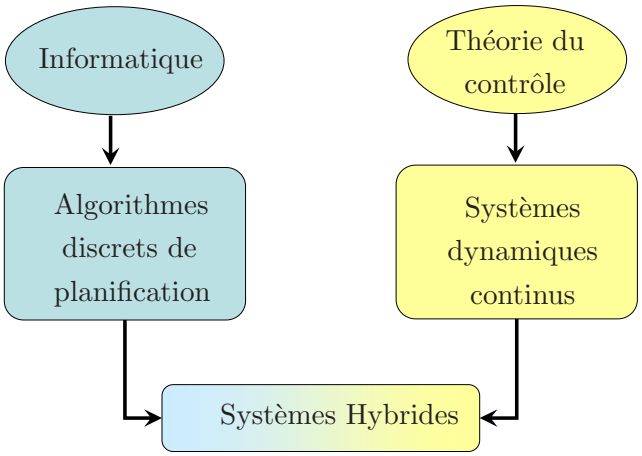


FIGURE 1.1 – Principe des systèmes dynamiques hybrides.

Selon la description de la loi gérant la variation de la variable discrète qu'on appelle *loi de commutation*, diverses représentations des systèmes hybrides sont obtenues. Quelques exemples de modèles sont : les modèles linéaires à sauts (Jump Linear) [83], les modèles linéaires à sauts markoviens (Jump Markov Linear) [31], les modèles MLD (Mixed Logic Dynamical) [9], les modèles LC (Linear Complementarity) [40][29], les modèles ELC (Extended Linear Complementarity) [27], les modèles affines par morceaux ou modèles PWA

(PieceWise Affine) [79], etc. L'équivalence entre les différentes classes de modèles a été montrée dans [7][39]. L'objectif de ce chapitre est de définir les systèmes hybrides et plus particulièrement les systèmes dynamiques affines par morceaux puis de présenter le principe des méthodes d'identification des SDH existantes dans la littérature et de discuter leurs performances actuelles.

Le plan du chapitre est le suivant. Dans la Section 1.2, nous définissons les modèles dynamiques affines par morceaux, d'abord sous la forme d'une représentation d'état puis sous la forme entrée-sortie. Ensuite, dans la Section 1.3, nous dressons quelques formulations du problème d'identification pour cette classe de systèmes hybrides. Enfin, dans la Section 1.4, nous présentons quelques méthodes développées dans la littérature pour l'identification de tels systèmes en donnant une brève description de leurs principes. Nous détaillons plus particulièrement les méthodes qui rentrent dans la catégorie des méthodes basées sur les techniques de classification [35][67] et de la méthode basée sur la régression par les machines à vecteurs supports [54].

1.2 Les systèmes dynamiques affines par morceaux

Un système dynamique affine par morceaux (PWA) est une collection de systèmes linéaires/affines partageant un même état continu, reliés par des commutateurs déterminés par une partition polyédrique de l'ensemble état + entrée [71]. Les PWA sont utilisés pour modéliser un grand nombre de processus physiques, ou pour approximer une dynamique non linéaire via la linéarisation du comportement non linéaire en différents points de fonctionnement. Dans cette section, les systèmes dynamiques affines par morceaux sont définis, d'abord sous forme de représentation d'état, ensuite sous forme entrée-sortie.

1.2.1 Représentation sous forme d'état

Dans le cas général, les systèmes commutants sont décrits sous forme d'état par des équations de la forme :

$$\begin{aligned} x(k+1) &= A_{\sigma(k)}x(k) + B_{\sigma(k)}u(k) + f_{\sigma(k)} + w(k) \\ y(k) &= C_{\sigma(k)}x(k) + D_{\sigma(k)}u(k) + g_{\sigma(k)} + v(k) \end{aligned} \tag{1.1}$$

où $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^p$ et $y(k) \in \mathbb{R}^q$ sont respectivement, l'état continu, l'entrée et la sortie du système à l'instant $k \in \mathbb{Z}$. $w(k) \in \mathbb{R}^n$ et $v(k) \in \mathbb{R}^q$ sont respectivement les bruits d'état et de sortie du modèle.
Les matrices A_i, B_i, C_i et D_i, $i = 1, \ldots, s$, sont réelles et de dimensions appropriées ; elles décrivent la dynamique du système dans chaque sous-modèle. Par conséquent, le

modèle (1.1) peut être considéré comme une collection des modèles linéaires (affines) à état continu $x(k)$ indexés par l'état discret $\sigma(k)$.

L'état discret $\sigma(k)$ indexe le modèle ou le régime actif à chaque instant k et prend ses valeurs dans un ensemble fini $\{1, \ldots, s\}$, où s représente le nombre de sous-modèles. En général cet état peut être une fonction de k, $x(k)$ et $u(k)$ ou d'autres entrées externes.
L'évolution de l'état discret peut être décrite de diverses façons. Dans les modèles linéaires à sauts ou modèles JL, $\sigma(k)$ est une entrée inconnue, déterministe. Dans les modèles linéaires à sauts markoviens ou modèles JML, la dynamique de $\sigma(k)$ est modélisée comme une chaîne de Markov irréductible régie par la probabilité de transition : $\pi(i,j) \triangleq P(\sigma(k+1) = j \,|\sigma(k) = i)$.
Dans le cas des systèmes dynamiques affines par morceaux (PieceWise Affine), $\sigma(k)$ est donnée par la loi

$$\sigma(k) = i \qquad \text{ssi} \quad \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \Re_i, \quad i = 1, \ldots, s \tag{1.2}$$

où $\{\Re_i\}_{=1}^{s}$ est une partition complète du domaine état-entrée $\Re \subset \mathbb{R}^{n+p}$; les régions $\Re_i$ sont supposées être des polyèdres convexes. Chaque polyèdre est décrit par

$$\Re_i = \left\{ \begin{bmatrix} x \\ u \end{bmatrix} \in \mathbb{R}^{n+p} : \; \bar{H}_i \begin{bmatrix} x \\ u \\ 1 \end{bmatrix} \preceq 0 \right\} \tag{1.3}$$

où $\bar{H}_i \in \mathbb{R}^{\mu_i \times (n+p+1)}, i = 1, \ldots, s$ et μ_i est le nombre d'inégalités linéaires définissant la *ième* région $\Re_i$.

1.2.2 Représentation sous forme entrée-sortie

Le modèle sous forme entrée-sortie d'un système dynamique affine par morceaux (PWARX) peut être représenté sous la forme

$$y(k) = \theta_{\sigma(k)}^{\top} \begin{bmatrix} \varphi(k) \\ 1 \end{bmatrix} + e(k) \tag{1.4}$$

où $\varphi(k) = \left[y(k-1)^{\top} \ldots y(k-n_a)^{\top} \; u(k)^{\top} \; u(k-1)^{\top} \ldots u(k-n_b)^{\top}\right]^{\top}$ est le vecteur de régression avec n_a et n_b les ordres du système, $\sigma(k) \in \{1, \ldots, s\}$, l'état discret indexant le régime actif à l'instant k. $\theta_i, \quad i = 1, \ldots, s$, sont les matrices de paramètres associés aux différents sous-modèles du système, et $e(k) \in \mathbb{R}^q$ est le bruit.

Dans ce qui suit le vecteur $\bar{\varphi}(k) = \begin{bmatrix}\varphi(k)^\top & 1\end{bmatrix}^\top$ sera appelé vecteur de régression étendu. Comme pour les modèles sous forme d'état, l'évolution de l'état discret $\sigma(k)$ peut être décrite de diverses façons. Dans le cas des modèles dynamiques affines par morceaux, le mécanisme de commutation est déterminé par une partition polyédrale du domaine de régression $\Re \subset \mathbb{R}^d$, où $d = q.n_a + p.(n_b + 1)$ (voir figure 1.2). Cela signifie que pour ces modèles, l'état discret $\sigma(k)$ est donné par :

$$\sigma\,(k) = i \quad \text{ssi} \quad \varphi(k) \in \Re_i, \quad i = 1, \ldots, s \tag{1.5}$$

où $\{\Re_i\}_{i=1}^{s}$ est une partition complète de $\Re$, chaque région $\Re_i$ est un polyèdre convexe décrit par :

$$\Re_i = \left\{\varphi \in \mathbb{R}^d : \; H_i\bar{\varphi} \preceq 0\right\} \tag{1.6}$$

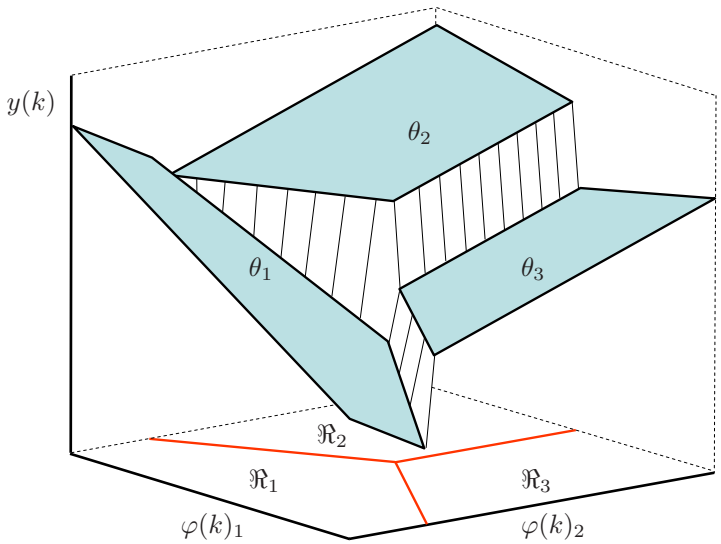


FIGURE 1.2 – Représentation d'un système PWARX possédant 3 sous-modèles ($s = 3$).

Dans le cas des systèmes dynamiques hybrides, le passage modèle d'état/modèle entrée-sortie (et vice versa) peut être établi pour certaines formes particulières de modèles. Nous nous intéressons, dans ce qui suit, à la littérature existante sur l'identification des systèmes dynamiques affines par morceaux sous forme entrée-sortie. Dans la section suivante, nous dressons quelques formulations du problème d'identification des modèles dynamiques affines par morceaux.

1.3 Problèmes d'identification des modèles dynamiques affines par morceaux

Dans cette section, nous posons le problème d'identification des modèles dynamiques affines par morceaux MISO sous forme entrée-sortie. Avant cela, nous formulons le problème d'identification des systèmes à commutation MISO sous forme entrée-sortie. Nous rappelons que le modèle MISO sous forme entrée-sortie est donnée par

$$y(k) = \theta_{\sigma(k)}^{\top} \begin{bmatrix} \varphi(k) \\ 1 \end{bmatrix} + e(k) \tag{1.7}$$

Dans ce cas, les paramètres $\theta_i, \quad i = 1, \ldots, s$, ne sont plus des matrices (comme dans le cas du modèle (1.4)) mais des vecteurs de paramètres associés aux différents sous-modèles du système. Le vecteur de régression $\varphi(k)$ est défini par $\varphi(k) = [y(k-1) \ldots y(k-n_a)\ u(k)^{\top} u(k-1)^{\top} \ldots u(k-n_b)^{\top}]^{\top}$.
Dans le cas général, le problème d'identification des systèmes linéaires commutants (modèle (1.7)), où la loi de commutation est totalement arbitraire, peut être formulé comme :

Problème 1.1 *Étant donné les mesures entrée-sortie* $(u(k), y(k))$, $k = 1, \ldots, N$, *estimer les ordres* n_a *et* n_b *du modèle, le nombre* s *de sous-modèles, et les vecteurs de paramètres* θ_i, $i = 1, \ldots, s$. *De plus, estimer l'état discret* $\sigma(k)$ *pour tout* $k > \bar{n}$ *où* $\bar{n} = \max\{n_a, n_b\}$.

Si les ordres du modèle n_a et n_b sont connus, le problème d'identification consiste à ajuster les données à un certain nombre d'hyperplans. Ce problème a été abordé dans le domaine de l'analyse des données, et plusieurs approches sont proposées, où s est soit estimé à partir des données, soit fixé *a priori*. Une façon d'estimer s est de résoudre le problème suivant

Problème 1.2 *Étant donné* $\delta > 0$, *trouver le plus petit nombre* s *de vecteurs de paramètres* θ_i, $; i = 1, \ldots, s$ *et l'application* $k \to \sigma(k)$, *avec* $\sigma(k) \in \{1, \ldots, s\}$, *tel que*

$$\left| y(k) - \theta_{\sigma(k)}^{\top} \varphi(k) \right| \leq \delta, \tag{1.8}$$

pour tout $k = \bar{n} + 1, \ldots, N$.

Le paramètre δ dans (1.8) ne doit pas être nécessairement fixé *a priori*, il peut être plutôt ajusté (après plusieurs essais) afin de trouver un meilleur compromis entre simplicité et précision du modèle. En fait, plus δ est petit, plus le nombre de sous-modèles

nécessaires est grand. Tandis que plus δ est grand, plus mauvaise est la précision du modèle.

Dans le cas des modèles dynamiques affines par morceaux où l'état discret est défini par l'équation (1.5), le problème d'identification peut être reformulé comme suit

Problème 1.3 *Étant donné les mesures entrée-sortie* $(u(k), y(k)),\ k = 1, \ldots, N$, *estimer les ordres* n_a *et* n_b *du modèle, le nombre* s *de sous-modèles, les vecteurs de paramètres* $\theta_i,\ i = 1, \ldots, s$ *et les régions* $\Re_i,\ i = 1, \ldots, s$.

Notons que dans ce cas, la partition du domaine de régression en différentes régions $\Re_i$ est équivalente à l'estimation de l'état discret. La plupart des techniques développées pour l'identification des modèles PWARX, supposent les ordres n_a et n_b connus. L'estimation des ordres du modèle peut être réalisée par des techniques algébriques [82], ou des techniques classiques de sélection d'ordres de modèles [60].
Sous l'hypothèse que les ordres du système sont connus *a priori*, le problème d'identification considéré consiste à trouver le modèle PWARX qui correspond le mieux aux données entrée-sortie fournies selon un critère d'ajustement spécifié. Il s'agit d'estimer

1. Le nombre s de sous-modèles du système.
2. Les vecteurs de paramètres $\theta_i,\ i = 1, \ldots, s$, associés aux sous-modèles.
3. Les coefficients $H_i, i = 1, \ldots, s$, définissant les frontières délimitant le partitionnement de l'ensemble de régression.

Ce problème traite à la fois des problèmes de classification et de régression. L'estimation simultanée de toutes les quantités mentionnées ci-dessus est un problème très difficile.
L'une des principales préoccupations est de savoir comment choisir correctement s. Des pénalités sur l'accroissement de s peuvent être mises en place pour maintenir le nombre de sous-modèles raisonnablement faible, afin d'éviter que le modèle soit difficile à exploiter.
Dans la situation où le nombre s de sous-modèles du système est connu, le problème d'identification sera équivalent à la recherche des paramètres d'une fonction affine à partir du jeu de données entrée-sortie. En introduisant la quantité

$$\chi_{k,i} = \begin{cases} 1 & \text{si} \quad \varphi(k) \in \Re_i \\ 0 & \text{sinon} \end{cases} \tag{1.9}$$

le problème d'identification (1.3) peut se ramener au problème de minimisation suivant

$$\min_{\theta_i} \frac{1}{N} \sum_{k=1}^{N} \sum_{i=1}^{s} l(y(k) - \theta_i^\top \bar{\varphi}(k)) \chi_{k,i}, \tag{1.10}$$

où l est une fonction non négative. Les deux choix les plus courants pour la fonction l sont la norme euclidienne au carré, c'est-à-dire $l(\epsilon) = \epsilon^2$ et la norme 1, c'est-à-dire $l(\epsilon) = |\epsilon|$.

Le problème d'optimisation (1.10) est généralement non convexe avec des minima locaux. Pour estimer efficacement le vecteur de paramètres correspondant à chaque sous-modèle, il faut déjà connaitre les données qui caractérisent ce sous-modèle. Ceci est donc un problème de classification non supervisée. Une difficulté supplémentaire est la manière d'exprimer efficacement la contrainte selon laquelle les régions doivent former une partition complète de l'ensemble de régression $\Re$.

1.4 Techniques d'identification des systèmes hybrides

Dans cette section, nous allons présenter quelques méthodes développées pour l'identification de systèmes à commutations. A la suite de cela, nous allons décrire les principales techniques d'identification existantes dans la littérature des modèles dynamiques affines par morceaux.

1.4.1 Introduction à l'identification de systèmes à commutations

La plupart des méthodes d'identification des modèles sous forme entrée-sortie concernent l'identification des modèles dynamiques affines par morceaux. Néanmoins, on trouve dans la littérature quelques méthodes développées pour l'identification des systèmes commutants SARX. Sans doute, la méthode la plus connue est celle proposée par Vidal et al. [85][82]. Cette approche traite l'identification des modèles SARX comme un problème algébro-géométrique. Elle fournit une solution analytique au problème d'identification qui est théoriquement fondée dans un cadre déterministe. La méthode algébrique consiste à considérer l'identification de plusieurs modèles ARX comme l'identification d'un seul modèle qui inclut simultanément tous les sous-modèles ARX et ne dépend pas de la séquence de commutation. L'idée de base est de trouver une équation vérifiée par toutes les données collectées quelque soit le modèle générateur. Cette équation est donnée par

$$\prod_{i=1}^{s} \left(b_i^\top z(k)\right) = 0,\ \forall k > \bar{n} = \max(n_a, n_b) \tag{1.11}$$

où $b_i = [1\ \theta_i^\top]^\top$ et $z(k) = [-y(k)\ \varphi(k)^\top]^\top$. L'équation (1.11) est appelée « contrainte de découplage hybride » car elle est indépendante de la séquence de commutation et du mécanisme générant la transition.
Les paramètres de chaque sous-modèle ARX peuvent être estimés à partir de la dérivée

d'un certain polynôme multivariable de degré s issu du développement de la contrainte de découplage hybride (1.11).
En présence de bruit, la contrainte de découplage hybride (1.11) n'est plus strictement vérifiée. Le produit considéré dans cette équation ne sera plus nul. Cela peut dégrader considérablement les performances de cette méthode. En plus, la dimension du vecteur de coefficients définissant le polynôme multivariable augmente d'une façon exponentielle en fonction de la dimension du système à estimer.
Pour pallier le problème des données bruitées les auteurs proposent de résoudre l'équation (1.11) dans le sens des moindres carrés en minimisant :

$$J = \sum_{k=1}^{N} \prod_{i=1}^{s} \left(b_i^T z(k)\right)^2 \tag{1.12}$$

Bako et al. [5] ont proposé une extension de cette méthode algébrique à l'estimation de systèmes MIMO commutants, dans le contexte général où les ordres sont inconnus et peuvent différer d'un sous-modèle à un autre et le nombre de sous-modèles est également inconnu. Contrairement au cas de l'identification de modèles SISO SARX [85] où un seul polynôme multivariable est utilisé pour réaliser une transformation polynomiale de données appartenant à un mélange d'hyperplans, l'identification de modèles MIMO SARX implique un nombre inconnu de polynômes homogènes indépendants qui s'annulent sur des sous-espaces qui ne sont plus des hyperplans. Les auteurs commencent d'abord par estimer les ordres des sous-modèles ainsi que le nombre d'états discrets à partir de conditions de rang portant sur les données entrée-sortie. Ensuite, sous certaines hypothèses, ils calculent le nombre de polynômes multivariable et ils identifient les paramètres des modèles ARX à partir des dérivées des polynômes.
Inspirés de la méthode algébro-géométrique, Lauer et al [57] ont introduit un estimateur appelé Produit-des-Erreurs (PE) pour l'identification des modèles SARX. Cet estimateur est défini par

$$\min_{f_i} \frac{1}{s} \sum_{i=1}^{s} R(f_i) + \frac{C}{N} \sum_{k=1}^{N} \prod_{i=1}^{s} l(y(k) - f_i(\varphi(k))) \tag{1.13}$$

où $R(f_i)$ est un régularisateur pour le sous-modèle $f_i(\varphi(k)) = \theta_i^\top \varphi(k)$ et l est une fonction de perte robuste. Le deuxième terme de l'équation (1.13) correspond à l'erreur algébrique définie dans (1.12) quand $l(e) = e^2$. Cependant l'estimateur Produit-des-Erreurs (PE) implique la résolution d'un problème d'optimisation non linéaire et non convexe.
Une autre méthode développée pour l'identification des systèmes commutants SARX a été proposée par Bako et al. [3] [4]. C'est une méthode d'identification récursive. L'approche commence d'abord par donner des valeurs initiales aux vecteurs de paramètres représentant les différents sous-modèles. Ensuite, la procédure alterne entre la classification des

données de régression et l'estimation des paramètres des sous-modèles. A chaque instant, l'état discret est déterminé comme étant l'indice du sous-modèle qui, en termes de prédiction d'erreur (ou d'erreur *a posteriori*), apparaît comme celui qui a le plus de chance d'avoir généré la donnée de régression à cet instant. Étant donné l'état discret, le vecteur de paramètres associé est alors adapté par un algorithme de moindres carrés récursifs.

1.4.2 Principales techniques d'identification des modèles dynamiques affines par morceaux PWARX

La méthode HFA (Hinge Finding Algorithm) : Utilisée pour l'approximation des fonctions non-linéaires, cette méthode a été proposée par Breiman [23]. Elle est basée sur la détermination des hyperplans représentant les modèles locaux de type HHARX. Cette classe de modèle est une sous-classe des modèles PWARX. Les fonctions affines par morceaux des modèles HHARX sont toujours continues sur les frontières des polyèdres définissant la partition polyédrique alors qu'elles peuvent être discontinues dans le cas des modèles PWARX. Un modèle HHARX constitué de demi-hyperplans est décrit par l'équation $y(k) = h(\varphi(k))$ où h est défini par

$$\begin{aligned} h(\varphi) &= \max\left(\theta_+^\top \varphi, \theta_-^\top \varphi\right) \\ \text{ou } h(\varphi) &= \min\left(\theta_+^\top \varphi, \theta_-^\top \varphi\right) \end{aligned} \tag{1.14}$$

avec θ_+ et θ_- les vecteurs de paramètres des deux hyperplans. Les deux hyperplans se rejoignent à $\left\{\varphi : (\theta_+ - \theta_-)^\top \varphi = 0\right\}$. L'intersection des deux hyperplans est appelée *gond*. L'équation (1.14) signifie que toutes données générées par un modèle local sont réparties sur un demi-hyperplan limité par l'intersection avec l'hyperplan de l'autre modèle. Cependant, deux hyperplans peuvent ne pas être suffisants pour représenter au mieux le système. Breiman a démontré que toute fonction f (non linéaire) définie et continue sur un support compact peut être approchée sur une base d'hyperplans de Breiman par

$$\hat{f}(\varphi) = \sum_{i=1}^{s} h_i(\varphi), \tag{1.15}$$

où s est le nombre d'hyperplans nécessaires pour l'obtention des performances souhaitées. La détermination des différents hyperplans h_i est effectuée par l'algorithme HFA. La méthode HFA se base sur une détermination des hyperplans deux à deux. La procédure commence d'abord par estimer deux hyperplans expliquant la sortie du modèle puis si la performance du modèle obtenu n'est pas satisfaisante, une nouvelle sortie qui est la différence entre la sortie du système et les prédictions du modèle est alors calculée. Malheureusement, l'algorithme HFA souffre de problèmes de convergence. En effet, la convergence de l'algorithme proposé dépend du découpage initial des données entre les

deux hyperplans. Pucar et al. [73] ont pu apporter quelques améliorations pour assurer la convergence de l'algorithme vers un minimum local.

La méthode de la programmation mixte en nombres entiers : Roll et al. [10] ont reformulé le problème d'identification comme un problème de programmation quadratique mixte en nombres entiers ("MIQP" de l'anglais "Mixed Integer Quadratic Programing"). L'algorithme est développé pour des modèles ARX du type hyperplans de Breiman (HHARX) [23]. Contrairement aux autres méthodes qui vont être présentées ci-après, cette approche garantit la convergence vers l'optimum global. Cependant le coût important de calcul restreint l'utilisation de cette méthode à un nombre très faible de données. En effet, la complexité des algorithmes MIQP croît exponentiellement en fonction du nombre de données et de l'ordre du système. Pour remédier à ce problème, les auteurs recommandent de subdiviser l'ensemble des données en de petits groupes de taille très faible puis de partitionner chaque groupe. La méthode élaborée par ces auteurs peut être utilisée dans le cas où il n'y a pas de données sur l'intersection des hyperplans (ce qui n'est pas le cas de la méthode précédente).

La méthode de l'erreur bornée : Inspiré par l'idée des méthodes ensemblistes d'estimation (voir par exemple [66]), Bemporad et al. [8] traitent l'identification des modèles PWARX comme un problème de partitionnement en un nombre minimum de sous-systèmes réalisables (MIN PFS). Le principe de cette méthode, appelée méthode de l'erreur bornée, est d'imposer que l'erreur définie comme étant la différence entre la sortie $y(k)$ et $\theta_i^\top \bar{\varphi}(k)$ soit bornée par un nombre réel positif $\delta > 0$ pour tous les échantillons de données $(y(k), \bar{\varphi}(k))$, c'est-à-dire

$$\left| y(k) - \theta_i^\top \bar{\varphi}(k) \right| \leq \delta, \ \forall k = \bar{n} + 1, \ldots, N. \tag{1.16}$$

La solution proposée à ce problème souffre de deux inconvénients. Le premier est lié à la sous-optimalité de la méthode utilisée, c'est-à-dire que l'obtention du nombre minimum de sous-modèles n'est pas garantie. Le second est lié au problème des points de données dits indécidables, c'est-à-dire les points pour lesquels l'inégalité (1.16) peut être vérifiée pour plusieurs sous-modèles. Pour faire face à ces inconvénients, une procédure de raffinement itératif de réaffectation des données est appliquée. La procédure de raffinement alterne entre la réaffectation des données et la mise à jour des vecteurs de paramètres définissant les sous-modèles.
Le paramètre de réglage δ permet de fixer la précision du modèle recherché ainsi que le nombre de sous-modèles réalisables. Plus δ est grand, plus le nombre de sous-modèles est petit et plus aussi le modèle estimé est moins précis. A l'inverse un δ petit conduit à une surestimation du nombre de sous-modèles.

Méthode basée sur le choix des fonctions de pondération adaptées : Ragot et al. [74] proposent une méthode pour identifier les paramètres des modèles locaux PWARX basée sur le choix d'une fonction de pondération adaptée. Cette fonction permet de sélectionner les données pour lesquelles chaque modèle local est actif. En effet, la méthode proposée est capable de résoudre simultanément la répartition des données et l'estimation des paramètres. La méthode consiste à réduire au minimum la fonction d'erreur suivante :

$$\Phi = \sum_{j=1}^{s} \sum_{k=1}^{N} (y(k) - \theta_j^{\top} \varphi(k))^2 p_{k,j} \tag{1.17}$$

où les poids $p_{k,j}$, doivent être conçus de telle sorte que le modèle local j est adapté uniquement avec les données qu'il a générées.
En général, l'indice de performance (1.17) doit être minimisé par rapport aux vecteurs de paramètres θ_j, pour toutes les partitions disjointes possibles de l'ensemble des données de régression et pour tous les nombres possibles de sous-modèles. Ragot et al. ont restreint le problème d'identification au cas où le nombre de sous-modèles est choisi *a priori*.
Un algorithme itératif a été alors introduit pour adapter les fonctions de pondération. Chaque itération se compose de deux étapes. Le première étape consiste à déterminer une estimation des fonctions de pondération $p_{k,j}$ compte tenu des modèles locaux. La deuxième étape permet d'identifier les modèles locaux en se basant sur les fonctions de pondération. Il convient de noter que l'algorithme proposé estime séquentiellement ces modèles locaux par l'utilisation d'une allocation de données en série. Plus précisément, l'algorithme estime le premier modèle local, le deuxième modèle local explique les résidus du premier modèle local et ainsi de suite. La méthode basée sur le choix des fonctions de pondération suppose le nombre de sous-modèles et les ordres du système connus *a priori*. De plus, elle est sensible à l'initialisation des paramètres.

La méthode bayésienne : La procédure bayésienne proposée par Juloski et al. [47][49] est basée sur l'idée d'exploiter les connaissances *a priori* disponibles sur les modes et les paramètres des systèmes PWARX. Les vecteurs de paramètres θ_i sont considérés comme des variables aléatoires, et sont décrits par leurs fonctions de densité de probabilité (f.d.p.) $p_{\theta_i}(.)$. Un algorithme itératif sous-optimal a été alors proposé pour mettre à jour la f.d.p. de chaque mode. Une méthode de filtrage particulaire est utilisée pour une mise en oeuvre numérique de la procédure proposée. Le partitionnement de données est basé sur la réaffectation de chaque donnée au mode ayant la plus forte probabilité. Enfin, l'estimation des régions est basée sur une version modifiée de la méthode de programmation linéaire multi-classes [11]. L'approche bayésienne exige la connaissance *a priori* des ordres du modèle n_a et n_b, et du nombre de sous-modèles s. De plus, les fonctions de densité de

probabilité (f.d.p.) sont estimées au prix d'un coût de calcul prohibitif.

Nous avons exposé dans la section précédente quelques méthodes dédiées à l'identification des modèles dynamiques affines par morceaux. On notera la méthode de détermination d'hyperplans de Breiman "Hinging Hyperplanes" introduite par Breiman [23]. Cette méthode fut améliorée par Bemporad et al. [10] pour assurer la convergence vers le minimum global au prix d'un coût de calcul prohibitif. Nous avons ensuite présenté la méthode de l'erreur bornée [8] qui partitionne les données de régression en un nombre minimum de sous-systèmes réalisables (MIN PFS). Enfin nous avons présenté la méthode basée sur le choix des fonctions de pondération adaptées [74] et la méthode bayésienne [49].
Comme nous l'avons évoqué dans la section 1.3, la principale difficulté dans la reconstruction des modèles dynamiques affines par morceaux est que l'estimation de sous-modèles linéaires est très rarement dissociée du problème de classification des données, à savoir, l'attribution de chaque donnée de régression au sous-modèle qui l'a générée. Nous exposons dans la section suivante des méthodes exploitant des techniques de classification pour l'identification des modèles dynamiques affines par morceaux.

1.5 Méthodes basées sur les techniques de classification

Récemment, quelques auteurs ont introduit des méthodes d'identification exploitant l'utilisation combinée de la classification non supervisée (ou clustering) et des techniques d'identification linéaire. Chaque mode est alors caractérisé par une classe pour laquelle l'affectation de données doit être effectuée suivant une mesure de similarité spécifiée. En effet, les techniques de classification offrent des possibilités de catégorisation des données de régression qui s'avèrent être très intéressantes. Contrairement aux méthodes basées sur la programmation quadratique par exemple, les méthodes de classification peuvent être appliquées à un nombre important de données. Elles peuvent également prendre en considération le bruit dans les données par le rejet de l'information aberrante (outliers). L'idée consisterait par exemple à utiliser dans un premier niveau de décision une approche par modélisation pour rejeter les outliers, puis classer les données ne présentant aucune ambigüité. Les méthodes de classification pour l'identification peuvent effectuer une double tâche à savoir le partitionnement des données de régression mais aussi l'estimation des zones de fonctionnement (régions). Par exemple, si on suppose que les données appartenant à un sous-modèle peuvent être modélisé par un modèle gaussien, alors l'ellipsoïde

définissant ce modèle pourra constituer la frontière délimitant sa région de validité dans le cas où les régions ne sont pas définies par des hyperplans linéaires.

Dans cette section nous présentons deux méthodes basées sur des techniques de classification. La première méthode est développée par [35], elle utilise une variante de l'algorithme des K-means. Cette méthode a été ensuite modifiée en utilisant une approche de classification hiérarchique dite du lien-unique (single-linkage) [34]. La deuxième méthode est proposée par [67], elle est basée sur une technique de classification statistique des données au moyen d'un modèle de mélange gaussien.

1.5.1 Méthode basée sur la technique de classification K-means

Ferrari-Trecate et al. [35] ont proposé une méthode d'identification basée sur une technique de classification exploitant un algorithme des K-means [33] modifié. Cette méthode suppose connus le nombre s de sous-modèles et les ordres n_a et n_b. Elle est basée sur la subdivision des données en plusieurs classes ou ensembles locaux. Les vecteurs de paramètres identifiés à partir de ces petits ensembles locaux devront "ressembler" aux vecteurs de paramètres des vrais sous-modèles. Par conséquent, les paramètres des sous-modèles peuvent être obtenus par regroupement des vecteurs de paramètres locaux.

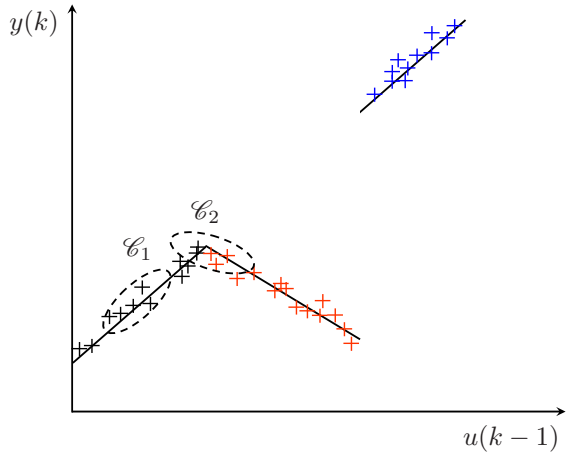


FIGURE 1.3 – Ensembles locaux de données dans l'espace de régression augmenté, $\mathscr{C}_1$: ensemble local pur, $\mathscr{C}_2$: ensemble local mixte.

La procédure proposée par [35] comporte quatre étapes qui sont :

– (i) la construction d'ensembles locaux de données à partir des données fournies ;

- (ii) l'identification d'un vecteur de paramètres pour chaque ensemble local et la construction d'un vecteur de caractéristiques basé sur ce vecteur de paramètres ;
- (iii) le partitionnement des vecteurs de caractéristiques en s groupes disjoints par l'utilisation d'une variante de l'algorithme des K-means ;
- (iv) la classification des données de régression et l'estimation des vecteurs de paramètres des s sous-modèles.

(i) Pour chaque donnée $(y(j), \varphi(j)), j = \bar{n}+1, \ldots, N$, un ensemble local de données $\mathscr{C}_j$ de taille c contenant la donnée $(y(j), \varphi(j))$ et ses $c-1$ plus proches voisins est construit. Parmi les ensembles locaux constitués, on distingue des ensembles locaux ne contenant que des points de données générés par le même sous-modèle, ces ensembles sont dits "ensembles locaux purs" et des ensembles contenant des données générées par plusieurs sous-modèles, ils sont dits "ensembles locaux mixtes" (cf. Fig. 1.3). Le nombre des ensembles locaux mixtes dépend à la fois de la façon dont les données sont disposées dans l'espace de régression et du paramètre c, supposé strictement supérieur à $dim(\theta)$.

(ii) Un vecteur de paramètres θ_k^{LS} est ensuite calculé à partir de chaque ensemble local $\mathscr{C}_j$ par la méthode des moindres carrés ordinaires :

$$\theta_j^{LS} = \left(\Phi_j \Phi_j^\top\right)^{-1} \Phi_j \mathscr{Y}_j, \tag{1.18}$$

où $\Phi_j = \left[\bar{\varphi}(k_j^1), \ldots, \bar{\varphi}(k_j^c)\right]$ et $\mathscr{Y}_j = \left[y(k_j^1), \ldots, y(k_j^c)\right]^\top$, les k_j^i étant les indices des plus proches voisins de la *jème* donnée.

Un résultat classique de la théorie des moindres carrés [60] assure que si les ensembles $\mathscr{C}_j$ sont des ensembles locaux purs, alors les vecteurs de paramètres estimés sont des vecteurs aléatoires gaussiens de moyenne θ_r, r étant l'indice du sous-modèle qui a généré les points de données dans $\mathscr{C}_j$. En outre, la matrice de covariance empirique V_j du vecteur θ_j^{LS} est donnée par [60]

$$V_j = \frac{SRR_j}{c-(n+1)} \left(\Phi_j \Phi_j^\top\right)^{-1} \tag{1.19}$$

$$\text{ou} \quad SRR_j = \mathscr{Y}_j^\top \left(I - \Phi_j^\top \left(\Phi_j \Phi_j^\top\right)^{-1} \Phi_j\right) \mathscr{Y}_j. \tag{1.20}$$

où $c > n+1$.
La matrice de covariance V_j est une mesure pouvant distinguer les ensembles locaux purs des ensembles locaux mixtes. En effet, la covariance des ensembles locaux purs ne dépend que du niveau de bruit et devrait être plus petite que la covariance des ensembles locaux mixtes.

Afin de mesurer la dispersion des données dans les ensembles locaux $\mathscr{C}_j$, les auteurs font appel à la matrice de dispersion des régresseurs [33] définie par :

$$Q_j = \sum_{k,\varphi(k)\in\mathscr{C}_j} (\varphi(k) - m_j)(\varphi(k) - m_j)^\top \tag{1.21}$$

$$m_j = \frac{1}{c} \sum_{k,\varphi(k)\in\mathscr{C}_j} \varphi(k). \tag{1.22}$$

La prochaine étape consiste à construire ce que les auteurs appellent "le vecteur de caractéristiques" ξ_j. Chaque donnée $(y(j), \varphi(j))$ est transformée en un vecteur de caractéristiques $\xi_j = \left[\left(\theta_j^{LS}\right)^\top \quad m_j^\top\right]^\top$.
Ce vecteur représente la moyenne d'un nuage de caractéristiques distribuées selon une distribution gaussienne de variance :

$$R_j = \begin{bmatrix} V_j & 0 \\ 0 & Q_j \end{bmatrix}. \tag{1.23}$$

(iii) Classification des vecteurs de caractéristiques : les vecteurs de caractéristiques sont ensuite partitionnés en s groupes disjoints $\mathscr{D}_i$. Ferrari-Trecate et al. utilisent un algorithme des "K-means" en exploitant convenablement les mesures de confiance (les matrices R_j) définies sur les vecteurs de caractéristiques. Les mesures de confiance permettent de réduire l'influence des valeurs aberrantes. Le regroupement des vecteurs de caractéristiques est basé sur la minimisation d'un coût de regroupement fonctionnel exprimé par

$$J(\{\mathscr{D}_i\}_{i=1}^s, \{\mu_i\}_{i=1}^s) = \sum_{i=1}^{s} \sum_{\xi_j\in\mathscr{D}_i} \|\xi_j - \mu_i\|^2_{R_j^{-1}}, \tag{1.24}$$

où μ_i est le centre de la classe (de l'ensemble) $\mathscr{D}_i$ et $\|\xi_j - \mu_i\|^2_{R_j^{-1}} = (\xi_j - \mu_i)^\top R_j^{-1}(\xi_j - \mu_i)$. L'objectif du regroupement optimal consiste à trouver les ensembles $\mathscr{D}_i$ et les centres μ_i qui minimisent le critère (1.24). La classification non supervisée des vecteurs de caractéristiques est accomplie grâce à l'algorithme de classification 1.1, donné ci-après.

(iv) En utilisant l'application bijective existant entre les vecteurs de caractéristiques ξ_j et les points de données, les données d'origine peuvent maintenant être partitionnées. En fait, chaque vecteur de caractéristiques ξ_j correspond à un point de donnée $(y(j), \varphi(j))$. Par conséquent, on peut former des sous-ensembles disjoints $\mathscr{F}_j$ selon la règle suivante : si $\xi_j \in \mathscr{D}_j$ alors $(y(j), \varphi(j)) \in \mathscr{F}_j$. Une fois les données partitionnées, un algorithme des moindres carrés ordinaires est utilisé pour estimer les vecteurs de paramètres $\theta_j,\ j = 1, \ldots, s$ des différents sous-modèles.

La technique proposée par Ferrari-Trecate et al. remet à jour de manière itérative les centres et les matrices de covariance. Il est bien connu que la convergence de ce genre

Algorithme 1.1

1. Initialiser les centres $\mu_i^{(0)}, i = 1, \ldots, s$ des classes, poser $r = 0$ et trouver les domaines $\mathscr{D}_i$ des vecteurs ξ_j qui minimisent le critère

$$J(\left\{ \mathscr{D}_i^{(0)} \right\}_{i=1}^{s}, \{ \mu_i^{(0)} \}_{i=1}^{s}) = \sum_{i=1}^{s} \sum_{\xi_j \in \mathscr{D}_i} \left\| \xi_j - \mu_i^{(0)} \right\|_{R_j^{-1}}^2, \tag{1.25}$$

2. Recalculer les centres des classes $\mu_i^{(r+1)}$ pour tout $i = 1, \ldots, s$ en résolvant le système linéaire suivant :

$$\left(\sum_{j, \xi_j \in \mathscr{D}_i^{(r)}} R_j^{-1} \right) \mu_i^{(r+1)} = \sum_{j, \xi_j \in \mathscr{D}_i^{(r)}} R_j^{-1} \xi_j, \tag{1.26}$$

Si $J(\left\{ \mathscr{D}_i^{(r)} \right\}_{i=1}^{s}, \{ \mu_i^{(r+1)} \}_{i=1}^{s}) = J(\left\{ \mathscr{D}_i^{(r)} \right\}_{i=1}^{s}, \{ \mu_i^{(r)} \}_{i=1}^{s})$ alors poser $\mathscr{D}_i^* = \mathscr{D}_i^{(r)}$, $\mu_i^* = \mu_i^{(r)}$, terminer l'algorithme, sinon aller à l'étape (3).

3. Trouver le domaine $\mathscr{D}_i^{(r+1)}$ des vecteurs ξ_j qui minimise le critère :

$$J(\left\{ \mathscr{D}_i^{(r+1)} \right\}_{i=1}^{s}, \{ \mu_i^{(r+1)} \}_{i=1}^{s}), \tag{1.27}$$

Si $J(\{ \mathscr{D}_i^{(r+1)} \}_{i=1}^{s}, \{ \mu_i^{(r+1)} \}_{i=1}^{s}) = J(\{ \mathscr{D}_i^{(r)} \}_{i=1}^{s}, \{ \mu_i^{(r+1)} \}_{i=1}^{s})$ alors poser $\mathscr{D}_i^* = \mathscr{D}_i^{(r)}$, $\mu_i^* = \mu_i^{(r+1)}$, terminer l'algorithme, sinon faire $r = r + 1$ et aller à l'étape (2).

d'algorithme dépend des conditions initiales utilisées. Cette approche exige que les ordres n_a et n_b du modèle, et le nombre s de sous-modèles soient connus. En plus de tous ces paramètres, le paramètre c doit être convenablement réglé pour avoir une convergence de l'algorithme proposé.

Une modification de l'approche [35] a été proposée dans [34] pour permettre l'estimation simultanée du nombre de sous-modèles au fur et à mesure que les données sont classées. Les auteurs de [34] utilisent une méthode de classification dite du lien-unique (single-linkage) [33] qui est une méthode de classification hiérarchique. L'idée est de définir une distance entre deux ensembles locaux disjoints

$$d(\mathscr{C}_q, \mathscr{C}_p) = \min_{\theta \in \mathscr{C}_q, \theta' \in \mathscr{C}_p} \|\theta_q - \theta_p\|. \tag{1.28}$$

Ces ensembles sont ensuite fusionnés au fur et à mesure si un critère de fusion basé sur cette distance est vérifié. L'algorithme hiérarchique du "lien unique" est décrit dans l'Algorithme 1.2.

Algorithme 1.2

1. Partitionner les données en N ensembles locaux $\mathscr{C}^{(0)} = \{\mathscr{C}_1, \ldots, \mathscr{C}_N\}$ comme à l'étape (i) (cf. page 24), fixer $l > 0$,
Poser $i = 0$,

2. Tant que $card(\mathscr{C}^{(i)}) > 1$ et $\delta^{(i)} = \min_{\mathscr{C}_q, \mathscr{C}_{q'} \in \mathscr{C}^{(i)}} d(\mathscr{C}_q, \mathscr{C}_{q'}) < l$
Faire
   1. $\mathscr{C}_{\bar{q}} = \mathscr{C}_{\bar{q}} \cup \mathscr{C}_{\tilde{q}}$ où $\mathscr{C}_{\bar{q}}$ et $\mathscr{C}_{\tilde{q}}$, avec $\bar{q} < \tilde{q}$, correspondent aux ensembles locaux qui minimisent la distance (1.28),
   2. $\mathscr{C}^{(i+1)} = \mathscr{C}^{(i)} - \{\mathscr{C}_{\tilde{q}}\}$,
   3. Faire $i = i + 1$.

Le paramètre l doit être fourni par l'utilisateur. Il est à noter que le nombre de sous-modèles s dépend fortement de l, une petite valeur de l produirait un nombre important de sous-modèles. Par contre, si l est trop grand, cela pourrait déboucher sur un seul sous-modèle linéaire. Certes, la modification apportée à la procédure de Ferrari-Trecate et al. a permis de s'affranchir de la nécessité de connaître *a priori* le nombre s de sous-modèles. Cependant, cela s'accompagne de l'introduction d'un nouveau paramètre l à régler en plus de la variable c. Notons que l'un des intérêts de l'algorithme d'identification que nous proposons dans le chapitre suivant est qu'il ne nécessite que la connaissance d'un seul paramètre.
Nous exposons dans la section suivante une méthode basée sur une classification statistique au moyen d'un modèle de mélange gaussien. Contrairement à la méthode de Ferrari-Trecate et al., cette méthode pose une hypothèse un peu forte sur la nature des données. Les auteurs supposent que la densité de probabilité des données suit un modèle de mélange Gaussien.

1.5.2 Méthode basée sur la classification EM

Nakada et al. [67] ont introduit une méthode pour l'identification des modèles PWARX basée sur une classification statistique des données mesurées au moyen d'un modèle de

mélange gaussien. Ils supposent que le nombre s de sous-modèles ainsi que les ordres n_a et n_b sont connus *a priori*. Cette approche comporte trois étapes qui sont : (i) classification des données. Dans cette phase, les mesures entrée-sortie sont partitionnées en plusieurs groupes en utilisant une technique de classification statistique de données par des modèles de mélange Gaussien. (ii) estimation des hyperplans définissant les régions dans l'espace de régression. (iii) estimation des vecteurs de paramètres de chaque sous-modèle par application de la méthode des moindres carrés ordinaires.

Phase 1 : Classification des données

Pour pouvoir effectuer la tache de classification de données, Nakada et al. supposent que la densité de probabilité des données $(y(k), \varphi(k)), k = \bar{n}+1, \ldots, N$ suit un modèle de mélange Gaussien. En posant $z_k = \left[\varphi(k)^\top y(k)\right]^\top$, $z_k \in \mathbb{R}^{n+p}$ on définit

$$p(z_k; \Phi) = \sum_{i=1}^{s} \alpha_i p_i(z_k; \mu_i, \Sigma_i) \tag{1.29}$$

où Φ est défini par $\Phi = (\alpha, \mu, \Sigma)$, avec $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_s)$ des coefficients vérifiant $\sum_{i=1}^{s} \alpha_i = 1$, μ et Σ représentent respectivement les vecteurs moyennes $\mu = (\mu_1, \mu_2, \ldots, \mu_s)$ de dimension $(n+p)$ et les matrices de covariance $\Sigma = (\Sigma_1, \Sigma_2, \ldots, \Sigma_s)$, chaque matrice Σ_i étant de dimension $(n+p)(n+p)$.

La fonction $p_i(z; \mu_i, \Sigma_i)$ est une densité de probabilité Gaussienne définie par :

$$\begin{array}{c} p_i(z; \mu_i, \Sigma_i) = \frac{1}{(2\pi)^{(n+p)/2}[\det(\Sigma_i)]^{1/2}} \times \exp\left\{-\frac{1}{2}(z-\mu_i)^\top \Sigma_i^{-1}(z-\mu_i)\right\} \\ i = 1, \ldots, s \end{array} \tag{1.30}$$

A partir des données z_k, $; k = 1, \ldots, N$, l'objectif consiste à trouver le paramètre Φ qui atteint le maximum de la fonction log-vraisemblance définie par :

$$\begin{aligned} L(\Phi) &= \sum_{k=1}^{N} \ln p(z_k; \Phi) \\ &= \sum_{k=1}^{N} \ln \left(\sum_{i=1}^{s} \alpha_i p_i(z_k; \mu_i, \Sigma_i) \right) \end{aligned} \tag{1.31}$$

de façon à ce que le modèle de mélange corresponde le mieux aux données.
Une fois le paramètre Φ qui décrit la densité de probabilité des données est obtenu, la probabilité que z_k soit dans la classe C_i est donnée par

$$P(z_k \in C_i) = \frac{\alpha_i p_i(z_k; \mu_i, \Sigma_i)}{p(z_k; \Phi)} \tag{1.32}$$

Cette probabilité sert de critère pour la classification des données. Pour que la classification soit faite de façon univoque, chaque donnée z_k doit être affectée à la classe C_i

pour laquelle $P(z_k \in C_i)$ est maximale pour tout $i = 1, \ldots, s$. L'obtention du meilleur paramètre Φ est basée sur l'estimation du maximum de vraisemblance.

Le maximum (éventuellement local) est obtenu par l'utilisation de l'algorithme EM qui consiste à mettre à jour itérativement le paramètre Φ. Cet algorithme est essentiellement composé de deux étapes : l'étape "Expectation" (étape E) et l'étape "Maximisation" (étape M). L'étape M consiste à maximiser la fonction log-vraisemblance qui est redéfinie après chaque itération dans l'étape E (cf. Algorithme 1.3).

Algorithme 1.3

1. Fixer les valeurs initiales $\Phi^{(0)} = (\alpha^{(0)}, \mu^{(0)}, \Sigma^{(0)})$ et mettre le compteur d'itération l à 0 ($l = 0$).

2. Effectuer pour tout $\Phi^{(l)} = (\alpha^{(l)}, \mu^{(l)}, \Sigma^{(l)})$ les procédures suivantes :
Etape E **:** Calculer

$$\psi_{ik}^{(l)} = \frac{\alpha_i^{(l)} p_i(z_k; \mu_i^{(l)}, \Sigma_i^{(l)})}{p(z_k; \Phi^{(l)})},\ k = 1, \ldots,\ N, i = 1, \ldots, s,$$
$$\Psi_i^{(l)} = \sum_{k=1}^{N} \psi_{ik}^{(l)},\ i = 1, \ldots, s.$$

Etape M **:** Mettre à jour $\Phi^{(l)} = (\alpha^{(l)}, \mu^{(l)}, \Sigma^{(l)})$ par

$$\alpha_i^{(l+1)} = \frac{\Psi_i^{(l)}}{N},\ i = 1, \ldots, s,$$
$$\mu_i^{(l+1)} = \frac{1}{\Psi_i^{(l)}} \sum_{k=1}^{N} \psi_{ik}^{(l)} z_k,\ i = 1, \ldots, s,$$
$$\Sigma_i^{(l+1)} = \frac{1}{\Psi_i^{(l)}} \sum_{k=1}^{N} \psi_{ik}^{(l)} (z_k - \mu_i^{(l+1)})(z_k - \mu_i^{(l+1)})^\top,\ i = 1, \ldots, s,$$

3. Si la condition de convergence définie par

$$\max \left\{ \frac{\left\| \alpha_i^{(l+1)} - \alpha_i^{(l)} \right\|_2}{\left\| \alpha_i^{(l)} \right\|_2}, \frac{\left\| \mu_i^{(l+1)} - \mu_i^{(l)} \right\|_2}{\left\| \mu_i^{(l)} \right\|_2}, \frac{\left\| \Sigma_i^{(l+1)} - \Sigma_i^{(l)} \right\|_2}{\left\| \Sigma_i^{(l)} \right\|_2},\ i = 1, \ldots, s \right\} \leq \varepsilon,$$
$$\varepsilon << 1$$

est satisfaite, alors mettre $l^* = l + 1$ et terminer. Le paramètre Φ est alors donné par $\Phi^* = \Phi^{(l^*)}$. Sinon, faire $l = l + 1$ et aller à l'étape 2.

Phase 2 : Estimation des hyperplans définissant les régions

Nakada et al. utilisent un classifieur binaire SVM [81] pour l'identification des paramètres des hyperplans définissant les régions dans l'espace de régression. L'objectif consiste à trouver l'hyperplan $\mathscr{H}_{ij} = \{ \varphi : a_{ij}^\top \varphi + b_{ij} = 0 \}$, séparant deux classes C_i et C_j où $a_{ij} \in \mathbb{R}^n$ et $b_{ij} \in \mathbb{R}$. À cette fin, ils résolvent le problème de programmation quadratique (QP) [81] suivant :

$$\begin{aligned} &\text{minimiser } \|a_{ij}\|^2 + \gamma \sum_{h \in \{i,j\}, z_k \in C_h} \nu_{hk} \\ &\text{sous contrainte } a_{ij}^\top \varphi(k) + b_{ij} \geq 1 - \nu_{ik},\ \nu_{ik} \geq 0, z_k \in C_i \\ &\qquad\qquad a_{ij}^\top \varphi(k) + b_{ij} \leq -(1 - \nu_{jk}),\ \nu_{jk} \geq 0, z_k \in C_j \end{aligned} \tag{1.33}$$

Ici, $\sum_{h \in \{i,j\}, z_k \in C_h} \nu_{hk}$ est une mesure du degré de classification erronée que l'on souhaite minimiser. La quantité $\|a_{ij}\|$ dans la fonction objectif représente l'inverse de la marge entre les deux classes. Le scalaire γ est une constante réelle positive.

Phase 3 : Estimation des vecteurs de paramètres des sous-modèles
A partir des données regroupées en différents sous-modèles disjoints, les vecteurs de paramètres $\hat{\theta}_i,\ i = 1, 2, \ldots, s$, sont estimés par l'algorithme des moindres carrés. Pour tout $i = 1, 2, \ldots, s$, le paramètre de chaque sous-modèle peut être estimé par la formule

$$\begin{aligned} &\hat{\theta}_i = (X_i^\top X_i)^{-1} X_i^\top Y_i, \\ &X_i = [\bar{\varphi}(k_{i1}), \bar{\varphi}(k_{i2}), \ldots, \bar{\varphi}(k_{iN_i})]^\top \end{aligned}$$

où $\bar{\varphi}(k_{il}) = \left[\varphi^\top(k_{il})\ 1 \right]^\top$, $Y_i = [y(k_{i1}), y(k_{i2}), \ldots, y(k_{iN_i})]^\top$ et $C_i = \left\{ z_{k_{i1}}, z_{k_{i2}}, \ldots, z_{k_{iN_i}} \right\}$. Ici, la quantité N_i désigne la cardinalité de C_i.

L'approche proposée par [67] pour partitionner les données est très discutable. En effet, les auteurs ne se basent que sur le modèle de mélange gaussien pour séparer complètement les données sans tenir compte des vecteurs de paramètres. Cette approche risque de ne pas être efficace si les données des différents sous-modèles ne forment pas des classes assez distantes dans l'espace de régression. De plus, du fait de l'équipartition des différents modèles, certains modèles se retrouvent favorisés par rapport aux autres lors de la prise de décision.

Nakada et al. suggèrent une technique d'extraction du nombre de sous-modèles lorsque les ordres sont connus sur la base du critère associé à l'estimation du maximum de vraisemblance. Tout d'abord, ils fixent deux entiers positifs s_{min} et s_{max} tels que $s_{min} \leq s \leq s_{max}$. Ensuite, pour tout $s \in [s_{min}, \ldots, s_{max}]$, ils estiment le paramètre Φ_s, où Φ_s désigne l'estimée de Φ pour un s donné. Puis, l'estimation de s est donnée par

$$\hat{s} = \underset{s = s_{\min}, \ldots, s_{\max}}{\arg\min} J(\Phi_s, s) \tag{1.34}$$

où J est un critère spécifié choisi parmi les critères d'information existants pour la sélection de modèle [43]. Il est défini par

$$J\left(\Phi_s, s\right) = -2L(\Phi_s) + A(N)D(s) \tag{1.35}$$

où L est la fonction de log-vraisemblance définie par (1.31). Le second terme pénalise le nombre de données et de sous-modèles. Dans (1.35), $D(s)$ représente le nombre de paramètres indépendants, tandis que $A(N)$ est une fonction dépendant du nombre de données N.

Dans la section suivante, nous présentons une méthode basée sur la régression par les machines à vecteurs de support [54].

1.6 Méthode basée sur la régression par les machines à vecteurs de support (SVR)

Lauer et al. [54] [56] [55] proposent une approche basée sur la régression par les machines à vecteurs de support (SVR). Le problème d'identification est reformulé comme un programme d'optimisation non-linéaire sous contraintes linéaires. Les non-linéarités impliquées dans le critère de ce programme d'optimisation sont des produits de variables continues et positives. Le critère est donc donné sous la forme d'une fonction régulière, contrairement à l'approche basée sur l'optimisation mixte impliquant des variables entières. La méthode ainsi proposée par [54] est ensuite étendue pour traiter les systèmes hybrides non-linéaires.

Le modèle considéré est défini par $\hat{y}(i) = f_{\lambda_i}(\bar{\varphi}(i))$ où $\lambda_i \in \{1, \ldots, s\}$ est l'état discret à l'instant i. Les s sous-modèles affines sont de la forme :

$$f_j(\varphi) = w_j^\top \bar{\varphi},\ j = 1, \ldots, s \tag{1.36}$$

Suivant l'approche de régression par les machines à vecteurs de support (SVR), les sous-modèles (1.36) sont estimés en minimisant les normes des vecteurs de paramètres $\|w_j\|_2$. En exploitant l'idée de la contrainte de l'erreur bornée (1.16), Lauer et Bloch reformulent le problème d'identification comme le problème d'optimisation suivant

$$\begin{aligned} &\min_{w_j} \sum_{j=1}^{s} \|w_j\|_2^2 \\ &-\delta \leq y(i) - w_{\lambda_i}^\top \bar{\varphi}(i) \leq \delta,\ i = 1, \ldots, N \end{aligned} \tag{1.37}$$

où l'erreur absolue $|e_{ij}| = \left|y(i) - w_j^\top \bar{\varphi}(i)\right|$ doit être inférieure au paramètre δ uniquement pour le sous-modèle f_j. Cependant, sans connaissance *a priori* de l'état discret (ou de la région $\Re_j$ dans le cas des systèmes dynamiques affines par morceaux) le problème ne pourra pas être résolu. Pour contourner cette difficulté, les auteurs considèrent une fonction de perte non négative qui permet de pénaliser les erreurs supérieures à ε. Une solution approximative du problème (1.37) peut alors être trouvée en résolvant le problème suivant

$$\begin{aligned} &\min_{w_j, \xi_{ij} \geq 0} \sum_{j=1}^{s} \|w_j\|_2^2 \\ &-\xi_{ij} - \delta \leq y(i) - w_j^\top \bar{\varphi}(i) \leq \delta + \xi_{ij},\ i = 1, \ldots, N, j = 1, \ldots, s \\ &\prod_{j=1}^{s} \xi_{ij} = 0,\ i = 1, \ldots, N, \end{aligned} \tag{1.38}$$

qui met en oeuvre cette fonction de perte pour $\varepsilon = \delta$ avec des variables ressort [1] : $\xi_{ij} \geq l(e_{ij}),\ i = 1, \ldots, N,\ j = 1, \ldots, s$. La variable ressort ξ_{ij} peut être vue comme l'erreur du sous-modèle j pour le point de donnée i sous le seuil δ.

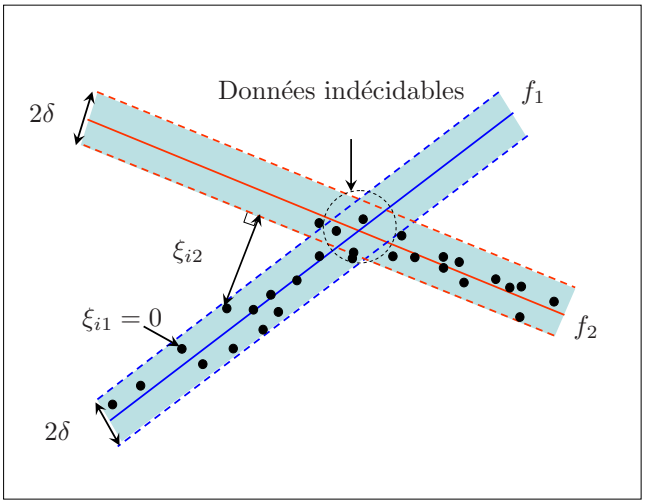


FIGURE 1.4 – Representation des variables ressort ξ_{ij} dans le cas de l'identification d'un système dynamique affine par morceaux.

Les égalités non linéaires ne sont pas faciles à traiter du point de vue de l'optimisation. En outre, le fait d'imposer des contraintes dures, telles que dans (1.38), n'est pas robuste

1. En anglais : slack variables

aux valeurs aberrantes, car le problème peut devenir impossible. Les contraintes sont donc relâchées et le problème est enfin formulé sous la forme

$$\begin{aligned} &\min_{w_j,\xi_{ij}\geq 0} \sum_{j=1}^{s} w_j^\top w_j + C\sum_{i=1}^{N}\prod_{j=1}^{s}\xi_{ij} \\ &-\xi_j - \delta\mathbf{1}_N \leq y - \Phi w_j \leq \delta\mathbf{1}_N + \xi_j,\ j=1,\ldots,s \end{aligned} \tag{1.39}$$

où $\Phi = \left[\bar{\varphi}(1)\ \bar{\varphi}(2)\ \ldots\ \bar{\varphi}(N)\right]^\top$, $y = \left[y(1)\ y(2)\ \ldots\ y(N)\right]^\top$, $\xi_j = \left[\xi_{1j}\ \xi_{2j}\ \ldots\ \xi_{Nj}\right]^\top$, $\mathbf{1}_N = [1\ 1\ldots 1]^\top \in \mathbb{R}^N$ et C est une constante de régularisation.

La résolution du problème (1.39) permet d'obtenir les vecteurs de paramètres $w_j,\ j = 1,\ldots,s$. En outre, l'état discret λ_i peut être déduit à partir des variables ξ_{ij}. En effet $\hat{\lambda}_i = j$, quand $\xi_{ij} = 0$.
Pour tenir compte des niveaux de bruit qui peuvent différer d'un sous-modèle à un autre, [54] propose de remplacer le paramètre δ par δ_j. Cependant, cette solution augmente le nombre de paramètres et la quantité d'informations à connaitre *a priori*.
La régularisation peut également être obtenue en minimisant la norme $l1$ des vecteurs de paramètres w_j au lieu de $\|w_j\|_2$ dans la fonction objectif (1.39). Cela est équivalent au problème d'optimisation suivant :

$$\begin{aligned} &\min_{w_j,a_j\geq 0,\xi_{ij}\geq 0} \sum_{j=1}^{s} 1^\top a_j + C\sum_{i=1}^{N}\prod_{j=1}^{s}\xi_{ij} \\ &-\xi_j - \delta_j 1 \leq y - \Phi w_j \leq \delta_j 1 + \xi_j,\ j=1,\ldots,s \\ &-a_j \leq w_j \leq a_j,\ j=1,\ldots,s \end{aligned} \tag{1.40}$$

où les vecteurs a_j sont des variables positives introduites pour borner $\|w_j\|_1$ par $\mathbf{1}_N^\top a_j$.

Lauer et Bloch ont également proposé des procédures d'estimation des ordres du système et du nombre de sous-modèles s à partir d'une surestimation des ordres et du nombre de sous-modèles. Néanmoins, ces procédures nécessitent quelques connaissances *a priori* sur le système à modéliser.

Pour estimer les régions correspondant à chaque sous-modèle, Lauer et Bloch ont fait appel à un classifieur non linaire binaire SVM (voir Annexe 1). La frontière séparatrice non linéaire S est donnée, dans ce cas, par

$$h(\varphi) = \sum_{i=1}^{N} \beta_i k(\varphi(i),\varphi) + b = 0. \tag{1.41}$$

Compte tenu de ce qui a été présenté précédemment, la procédure du regroupement de données est présentée dans Algorithme 1.4. Le problème d'identification (1.39) est un pro-

Algorithme 1.4

1. Estimer les vecteurs de paramètres en résolvant le problème d'optimisation (1.39).
2. Estimer l'état discret par $\hat{\lambda}_i = \arg\min_{j=1,\ldots,s} |e_{ij}|\,,\ i = 1,\ldots,N$.
3. Identifier l'ensemble U des données dites indécidables pour lesquelles le critère $|e_{ij}| < \delta_j$ est satisfait pour plusieurs sous-modèles f_j.
4. Estimer les frontières séparatrices S_i en utilisant le classifieur h (Eq. (1.41)) sur les données décidables $\varphi(i),\ i \in \{1,\ldots,N\} \setminus U$.
5. Pour tout $i \in U$ si $\left|y(i) - f_{\hat{\lambda}_i}(\varphi(i))\right| < \delta_{\hat{\lambda}_i}$, assigner la donnée $\varphi(i)$ en se basant sur $h(\varphi(i))$.

blème d'optimisation non linéaire sous contraintes linéaires. Ces méthodes sont connues pour leur coût calculatoire important. De plus, la méthode proposée nécessite la connaissance *a priori* de quelques paramètres.

1.7 Conclusion

Ce premier chapitre a été consacré à un état de l'art sur l'identification des systèmes dynamiques hybrides. Après avoir défini et reformulé le problème d'identification des systèmes hybrides (plus particulièrement les systèmes dynamiques affines par morceaux), nous avons décrit les principales techniques d'identification existantes dans la littérature. Nous avons notamment détaillé les méthodes basées sur les techniques de classification et la méthode basée sur la régression par les machines à vecteurs de support (SVR). Chacune des contributions mentionnées ci-dessus possède des avantages et des inconvénients suivant les hypothèses faites sur le nombre de sous-modèles, les ordres du système, le coût de calcul ou encore suivant les performances réalisables en présence de bruit.
Les approches d'identification que nous proposons dans les chapitres suivants s'appuient sur une classification non supervisée pour grouper les données de régression relatives au même sous-modèle affine. Contrairement aux autres méthodes citées, elles ne nécessitent la connaissance *a priori* que d'un seul paramètre de réglage.

Nous considérons également dans cette thèse des modèles dynamiques affines/non linéaires par morceaux dont les paramètres des sous-modèles et des régions peuvent varier dans le temps. L'objectif sera alors de modéliser les différents sous-modèles et leurs domaines de validité mais aussi de suivre l'évolution de leurs paramètres et de leurs régions dans le temps.

2

Classification de données et estimation de paramètres

Sommaire

2.1 Introduction

Après avoir présenté dans le chapitre précédent un état de l'art sur les principales techniques d'identification de systèmes hybrides existant dans la littérature, nous proposons, dans ce chapitre, une nouvelle approche d'identification de systèmes dynamiques affines par morceaux (PieceWise Affine – PWA en anglais). Les modèles dynamiques affines par morceaux ont été introduits pour décrire les phénomènes non linéaires dans les systèmes réels. Des systèmes dynamiques décrits par les PWA ont été considérés dans de nombreux travaux (réseaux de neurones [6],[37], approximation de fonctions [45],[46] , analyse de séries temporelles [63],[64]).

Comme nous l'avons indiqué dans le chapitre précédent, l'identification des modèles affines par morceaux (PWA) est un problème complexe qui implique à la fois la classification de données, l'estimation des vecteurs de paramètres définissant les différents sous-modèles affines, et l'estimation des coefficients des hyperplans définissant la partition du domaine de régression [71]. Nous traitons dans ce chapitre, le problème de la classification des données en des classes disjointes et de l'estimation des vecteurs de paramètres caractérisant chaque sous-modèle (voir figure 2.1). L'estimation des paramètres des hyperplans définissant la partition du domaine de régression sera traitée dans le chapitre suivant. L'approche proposée, dans ce chapitre, est basée sur une classification non supervisée combinée avec une technique de régression linéaire pour grouper les données de régression relatives au même sous-modèle affine. Cette approche consiste à alterner l'assignation des données de régression à un sous-modèle et l'estimation des paramètres de ce sous-modèle. La mesure de similarité employée est originale ; elle est composée de la distance euclidienne entre les données appartenant au même sous-modèle et de l'erreur entre la sortie du chaque sous-modèle et la sortie réelle.

L'approche que nous proposons dans ce chapitre rentre dans la catégorie des méthodes basées sur les techniques de classification [35][67]. Elle se distingue de ces méthodes par le fait qu'elle ne nécessite pas de connaissance *a priori* sur le nombre de sous-modèles. Seulement les ordres du systèmes doivent être fournis. De plus, nous avons réduit le nombre de paramètres que les autres méthodes exigent à l'utilisateur à un seul paramètre c qui est le nombre de plus proches voisins.

Ce chapitre est organisé de la manière suivante. Dans la section 2.2, nous présentons la structure des systèmes affines par morceaux, et nous formulons le problème de l'identification pour ces systèmes. Dans la section 2.3, nous décrivons une procédure générale de classification des données et d'estimation des vecteurs de paramètres. La procédure de réaffectation de données sera ensuite améliorée via une modélisation de connaissance incertaine par la théorie de Dempster-Shafer [77], [78]. Cette théorie permet de repré-

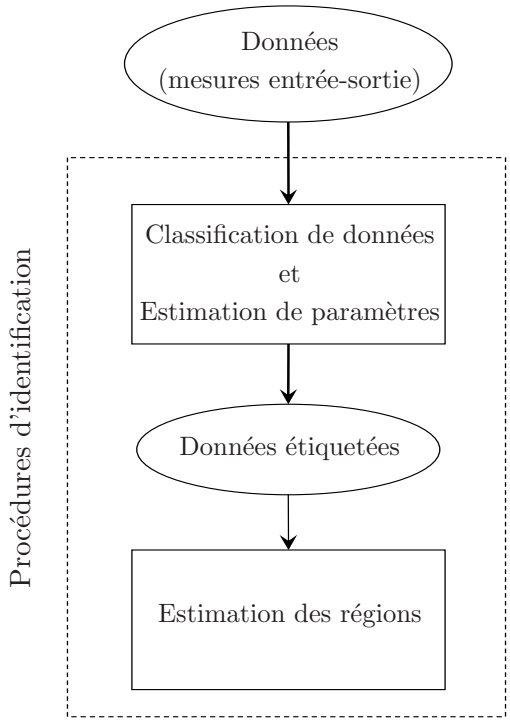


FIGURE 2.1 – Procédures d'identification des PWA.

senter explicitement, à partir d'outils mathématiques, l'incertitude sur les connaissances. Enfin, dans la section 2.4, nous montrons comment cette approche de classification des données peut être adaptée afin de traiter le cas des modèles dynamiques non linéaires par morceaux.

2.2 Motivation et formulation du problème

Les modèles PWA forment une classe particulière et probablement l'une des plus importantes des systèmes hybrides. Ils possèdent la propriété d'approximateurs universels. Ils sont construits en partitionnant l'espace des régression en un nombre fini de régions polyédriques et en associant un modèle local affine (que nous appellerons aussi sous-modèle) à chaque région.

La principale difficulté lors de l'identification des modèles affines par morceaux réside dans le fait que le problème d'identification des vecteurs de paramètres n'est pas proprement séparable de la phase de classification automatique (clustering en anglais) des données caractérisées par des mesures entrée-sortie en des classes disjointes. Chaque donnée est associée au sous-modèle affine le plus approprié. En effet, la connaissance des paramètres décrivant chaque sous-modèle du système permettrait de partitionner immédiatement l'en-

semble des données.
Deux cas peuvent alors être distingués :

1. La partition des données est connue *a priori*.
2. La partition des données est estimée au fur et à mesure que les paramètres des sous-modèles sont estimés.

Dans le premier cas, la classification des données n'est pas nécessaire et l'estimation des sous-modèles peut être effectuée en recourant à des techniques standards d'identification linéaire telles que les moindres carrés.

Dans le second cas, la classification des données ne peut pas s'effectuer sans connaissance des vecteurs de paramètres définissant les sous-modèles. Il est donc nécessaire de traiter simultanément le problème de la séparation des données par sous-modèle et celui de l'estimation d'un vecteur de paramètres pour chaque sous-modèle.

Cette tâche d'identification est encore plus délicate à effectuer quand le nombre de sous-modèles s doit lui aussi être estimé. En effet, si le nombre de sous-modèles n'est pas connu *a priori*, s'ajoute aux problèmes de la classification des données et de l'estimation des vecteurs de paramètres, le problème de trouver le nombre minimal de sous-modèles qui correspondent le mieux aux mesures entrée-sortie. Sans contrainte de minimalité sur le nombre de sous-modèles, on pourrait obtenir un modèle qui expliquerait parfaitement le comportement du système en choisissant autant de sous-modèles que de données. Il est évident que cette solution est à proscrire car elle introduit une surparamétrisation du modèle obtenu. Il est donc nécessaire d'imposer des contraintes sur le nombre de sous-modèles s de façon à minimiser le nombre de sous-modèles identifiés tout en garantissant une erreur d'identification faible.

Considérons un système MISO, dynamique affine par morceaux de s sous-modèles, défini par

$$y\left(k\right) = f\left(\varphi\left(k\right)\right) + \varepsilon\left(k\right), \tag{2.1}$$

où $\varepsilon(k)$ est l'erreur de prédiction et $\varphi(k) \in \mathbb{R}^n$ est le vecteur régression, défini par

$$\varphi(k) = \left[y(k-1), \cdots, y(k-n_a), u(k)^\top, \cdots, u(k-n_b)^\top\right]^\top, \tag{2.2}$$

où $u(k) \in \mathbb{R}^m$ et $y(k) \in \mathbb{R}$ sont respectivement l'entrée et la sortie mesurée du système à l'instant $k \in \mathbb{Z}$, n_a et n_b sont ses ordres.

Dans l'équation (2.1), f est une fonction affine par morceaux de la forme

$$f\left(\varphi(k)\right) = \begin{cases} \theta_1^\top \bar{\varphi}(k) & \text{si } \sigma(k) = 1 \\ \vdots & \\ \theta_s^\top \bar{\varphi}(k) & \text{si } \sigma(k) = s, \end{cases} \tag{2.3}$$

où $\bar{\varphi} = \begin{bmatrix} \varphi^\top & 1 \end{bmatrix}^\top$ est le vecteur de régression étendu et $\{\theta_i\}_{i=1}^s$ sont les vecteurs de paramètres qui définissent les sous-modèles. $\sigma(k)$ est la loi de commutation qui est définie par

$$\sigma\left(k\right) = i \quad \text{ssi} \quad \varphi(k) \in \Re_i, \quad i = 1, \ldots, s. \tag{2.4}$$

Les régions $\{\Re_i\}_{i=1}^s$ forment une partition complète du domaine de régression $\Re \subset \mathbb{R}^n$, avec $n = n_a + m(n_b + 1)$, chaque région $\Re_i$ est un polyèdre convexe décrit par

$$\Re_i = \left\{\varphi \in \mathbb{R}^n : H_i \bar{\varphi} \leq \mathbf{0}\right\}, \tag{2.5}$$

où H_i est une matrice de dimensions appropriées et $\mathbf{0}$ est le vecteur nul.

Étant donné les mesures entrée-sortie $(u(k), y(k))$, générées par un système caractérisé par (2.1), la procédure présentée dans ce chapitre consiste à (i) déterminer le nombre de sous-modèles nécessaires à l'obtention d'une bonne approximation du système, (ii) résoudre un problème de classification des données dont le but est de séparer les données disponibles selon leurs sous-modèles affines respectifs, (iii) estimer les paramètres associés à ces sous-modèles.
Il est à noter que les procédures que nous proposons, dans ce chapitre, permettent l'estimation de ces trois quantités de façon simultanée.

2.3 Classification de données et estimation de paramètres des modèles PWA

L'approche proposée s'appuie sur une classification non supervisée combinée avec des techniques de régression linéaire pour grouper les données de régression relatives au même sous-modèle affine. Cette approche consiste à alterner l'assignation des données de régression à une classe et l'estimation des paramètres du sous-modèle correspondant à cette classe [15],[19].

Dans le cas de la classification non supervisée, les étiquettes (ou indices des classes d'appartenance) des données de régression ne sont pas connues a priori. La méthode de classification a donc pour objectif de construire la partition des données, c'est-à-dire de déterminer l'appartenance des données aux classes. L'affectation des données aux classes se fait à partir de l'évaluation d'un critère de ressemblance.

Pour surmonter la difficulté liée au fait que le nombre de sous-modèles est inconnu *a priori*, la procédure proposée procède, dans sa phase initiale, à la création de N classes singletons $\mathscr{C}_i,\ i = 1, \ldots, N$ avec les N données de régression disponibles. Pour chacune de ces classes singletons, nous commençons par estimer un vecteur de paramètres avec les c vecteurs de régression les plus proches (au sens de la distance euclidienne) du régresseur concerné (avec c un entier donné). A chaque étape de l'algorithme, les données peuvent migrer selon des règles convenues, d'une classe à une autre, entrainant ainsi une réduction progressive du nombre de classes (et donc du nombre de sous-modèles) par élimination des classes vides ou inconsistantes. Chaque vecteur de paramètres est alors adapté suivant l'évolution de la classe correspondante. La réaffectation de chaque donnée est basée sur des mesures d'appartenance originales calculées à partir des c plus proches voisins de la donnée concernée. La procédure de réaffectation de données est ensuite modifiée par une modélisation de données incertaines par la théorie de Dempster-Shafer. Enfin, l'algorithme de classification de données est doté d'un critère d'arrêt permettant d'arrêter la procédure de réaffectation de données quand les vecteurs de paramètres se stabilisent.

Grâce à cette procédure, nous traitons simultanément le problème de la séparation des données par sous-modèle, l'estimation d'un vecteur de paramètres pour chaque sous-modèle (à partir des données contenues dans chaque classe) et l'estimation du nombre de sous-modèles.

Comme il est montré sur le synopsis de la figure 2.2, l'approche que nous proposons est composée de trois étapes qui sont :

1. Initialisation.
2. Réaffectation de données.
3. Test de convergence.

L'algorithme proposé a comme entrées : les données entrée-sortie $\{y(i), \varphi(i)\}_{i=1}^{N}$ et le nombre c fixant le nombre des plus proches voisins, et comme sorties : le nombre de sous-modèles s, les classes $\{\mathscr{C}_i\}_{i=1}^{s}$ et les vecteurs de paramètres $\left\{\hat{\theta}_i\right\}_{i=1}^{s}$. A l'initialisation, les données sont utilisées pour créer N classes, ensuite elles sont réaffectées par itérations successives jusqu'à ce que le critère d'arrêt soit vérifié afin de stopper la procédure de

réaffectation de données. Dans ce qui suit, nous détaillerons les trois étapes constituant notre algorithme.

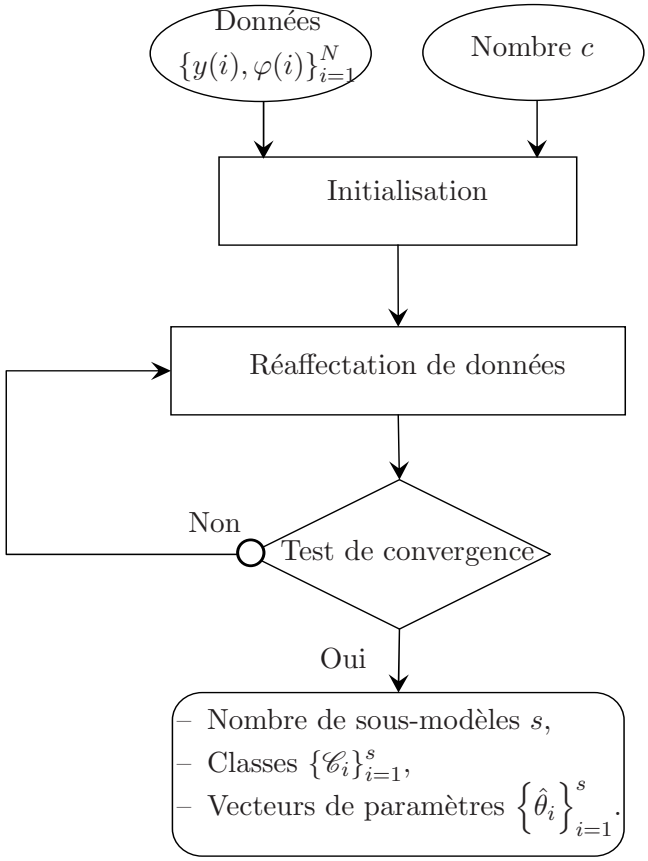


FIGURE 2.2 – Algorithme de classification de données et d'estimation de paramètres des modèles PWA.

2.3.1 Initialisation

Le nombre de sous-modèles étant inconnu, l'idée est alors de procéder à la création de N classes singletons avec les N données de régression disponibles $\mathscr{C} = \{\mathscr{C}_1, \ldots, \mathscr{C}_N\}$. Chaque classe $\mathscr{C}_i$ correspond à un sous-modèle. Cela revient à poser (à l'étape initiale) $\bar{s} = N$, où $\bar{s}$ est l'estimée du nombre de sous-modèles. Les données à classer sont $x(i) = \left[\varphi(i)^\top, y(i)\right]^\top$; $i = 1, \ldots, N$, c'est-à-dire que chaque donnée est obtenue par la concaténation du vecteur de régression et de la sortie mesurée.
A ces N classes, on associe les vecteurs de paramètres $\hat{\theta}_1^{(0)}, \ldots, \hat{\theta}_N^{(0)}$. Cependant, étant donné que chaque classe ne contient qu'une seule donnée, elle peut admettre une infinité de vecteurs de paramètres. Nous proposons pour le calcul de $\hat{\theta}_i^{(0)}$ de considérer la

donnée $x(i)$ de la classe $\mathscr{C}_i$ et ses c plus proches voisins (c-ppv) et d'utiliser la technique des moindres carrés sur ces $c+1$ données de régression ; cette technique est décrite ci-après.

La méthode des moindres carrés

Soient N_i données $(\varphi(k), y(k))$, $k = 1, \ldots, N_i$ générées par un modèle affine, où $\varphi(k)$ et $y(k)$ sont respectivement le vecteur de régression et la sortie du modèle à l'instant k. Un tel modèle s'écrit :

$$y(k) = \theta_i^\top \bar{\varphi}(k) \tag{2.6}$$

où $\bar{\varphi} = \begin{bmatrix} \varphi^\top & 1 \end{bmatrix}^\top$.

La méthode des moindres carrés consiste à calculer le vecteur de paramètres $\hat{\theta}_i$, définissant ce modèle affine, en minimisant

$$V(\theta_i) = \frac{1}{N_i} \sum_{k=1}^{N_i} \left(y(k) - \theta_i^\top \bar{\varphi}(k) \right)^2. \tag{2.7}$$

Ce critère est minimisé pour :

$$\hat{\theta}_i = \left(\sum_{k=1}^{N_i} \bar{\varphi}(k) \bar{\varphi}^\top(k) \right)^{-1} \sum_{k=1}^{N_i} \bar{\varphi}(k) y(k). \tag{2.8}$$

Cette dernière équation peut s'écrire sous la forme suivante :

$$\hat{\theta}_i = \left(\Phi^\top \Phi \right)^{-1} \Phi^\top Y. \tag{2.9}$$

où $\Phi = \begin{bmatrix} \varphi(1) & \varphi(2) & \cdots & \varphi(N_i) \\ 1 & 1 & \cdots & 1 \end{bmatrix}^\top$ et $Y = [y(1), y(2), \ldots, y(N_i)]^\top$.

Le paramètre c devrait être convenablement choisi afin d'obtenir une bonne initialisation des vecteurs de paramètres $\hat{\theta}_i^{(0)}$. Si le rapport signal sur bruit est élevé, une faible valeur du nombre de plus proches voisins c pourrait induire une estimation correcte des vecteurs de paramètres alors que si le paramètre c augmente, cela augmenterait le nombre de vecteurs de paramètres aberrants calculés à partir des plus proches voisins issus de différents sous-modèles. Cependant lorsque le niveau de bruit n'est pas négligeable, une faible valeur du paramètre c produirait une mauvaise estimation des vecteurs de paramètres (une estimation avec une forte variance). Dans ce cas, la solution serait d'augmenter la valeur de c pour lisser l'influence du bruit. Néanmoins, si la valeur de c dépasse le nombre de données de régression dans chaque sous-modèle, cela donnerait des vecteurs de paramètres initiaux erronés.

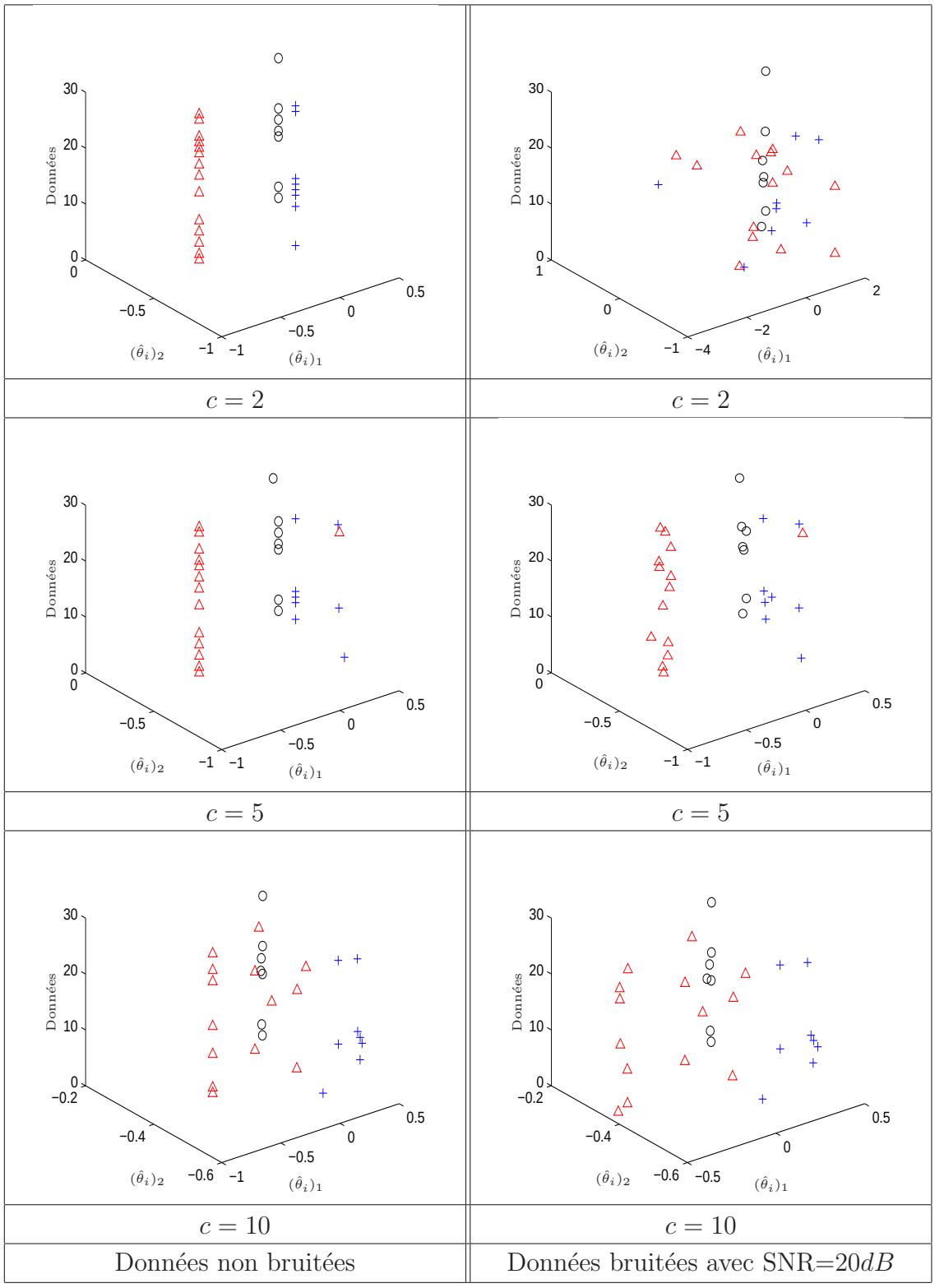


FIGURE 2.3 – Représentation des vecteurs de paramètres initiaux pour différents niveaux de bruit et pour différentes valeurs de c pour $s = N = 30$.

Pour illustrer l'influence du paramètre c sur l'étape d'initialisation, nous considérons un système dynamique affine par morceaux d'ordres $n_a = n_b = 1$ à trois sous-modèles ($s = 3$). Un ensemble de $N = 30$ données est généré suivant ce modèle. Sur la figure 2.3, nous représentons les deux composantes des vecteurs de paramètres initiaux correspondants aux 30 classes singletons, dans le cas de données non bruitées (colonne 1) et dans le cas de données bruitées avec un rapport signal sur bruit SNR=20db (colonne 2). Dans la première colonne, nous constatons que l'initialisation est meilleure pour $c = 2$ où les trois classes des vecteurs de paramètres se distinguent bien. Plus le nombre c augmente, plus les trois classes de vecteurs de paramètres se chevauchent. Ceci est dû au fait que certains vecteurs de paramètres sont calculés à partir de plus proches voisins issus de différents sous-modèles. Dans le cas des données bruitées (colonne 2), nous constatons que l'initialisation est meilleure pour $c = 5$. En effet pour $c = 2$, suite aux effets du bruit, les vecteurs de paramètres sont estimés avec une forte variance. Pour $c = 10$, les vecteurs de paramètres sont calculés à partir de plus proches voisins issus de différents sous-modèles. Notons qu'une mauvaise initialisation n'influe pas beaucoup sur la convergence de la procédure de classification de données et d'estimation de paramètres vers les bonnes valeurs. En effet, comme on peut le constater plus tard, dans l'exemple 2.2, quelle que soit la valeur de c, la procédure proposée permet l'obtention du modèle PWA qui correspond le mieux aux données entrée-sortie fournies. Le paramètre c est donc un paramètre de réglage qui pourra être ajusté en fonction de la précision souhaitée.

Pour diminuer le nombre de classes, il est nécessaire d'utiliser une règle de décision efficace pour réaffecter les données initialement partitionnées en N classes. Les données pourront alors migrer vers les classes les plus représentatives. Les classes les moins représentatives deviennent vides et sont alors éliminées permettant ainsi de diminuer le nombre de classes (de sous-modèles).

2.3.2 Réaffectation de données

Cette étape de la procédure consiste à regrouper les données de régression (initialement réparties en N classes) en un nombre minimal s de classes disjointes. Dans le cas des PWA, les données appartiennent à des modèles localement affines. Cela signifie que deux données $x(i)$ et $x(j)$ suffisamment proches (au sens de la distance euclidienne) sont susceptibles d'appartenir au même sous-modèle affine. C'est pourquoi notre stratégie vise à minimiser simultanément la distance euclidienne entre chaque paire de données appartenant au même sous-modèle ainsi que l'erreur entre la sortie mesurée et la sortie de chaque sous-modèle affine.

L'affectation des données aux sous-modèles se fait à l'aide d'un certain critère de similarité. Ce critère permettra de déterminer l'apport d'information (d'un point de vue de la ressemblance) des données de régression $x(i)$ par rapport aux N classes $\mathscr{C}_i$ initialement créées afin de réaffecter ces données et donc de diminuer le nombre de classes. Les informations prises en compte dans la construction des mesures de similarité peuvent être de différentes natures, et ceci en exploitant toutes les formes de ressemblance qui peuvent exister entre les données.
Étant donné que les données de régression $x(i), i = 1, \ldots, N$ appartiennent à des modèles *localement* affines, leur réaffectation peut donc être basée sur les apports d'information (sur la proximité géométrique et la linéarité locale) fournis par leurs c plus proches voisins (c-ppv). Le choix des plus proches voisins est motivé par le fait qu'ils représentent des sources d'information pouvant balayer un environnement local. En effet, ces informations peuvent nous renseigner sur le modèle qui englobe la donnée concernée. Ainsi, la donnée $x(i)$ peut migrer vers l'une des classes contenant ses plus proches voisins suivant une règle de décision.
Soit $\Gamma_c(x(i))$ l'ensemble des c-plus proches voisins de $x(i)$ et soit $x(j) \in \Gamma_c(x(i))$, $j \neq i$ un des plus proches voisins de $x(i)$ appartenant à la classe $\mathscr{C}_{\hat{\sigma}(j)}, \hat{\sigma}(j) \in \{1, \ldots, \bar{s}\}$. Nous introduisons la mesure de similarité

$$\phi_j^i = \exp\left(-\alpha_{\hat{\sigma}(j)}\|x(i) - x(j)\|^2 - \beta_{\hat{\sigma}(j)}\left(y(i) - \hat{\theta}_{\hat{\sigma}(j)}^\top \bar{\varphi}(i)\right)^2\right), \tag{2.10}$$

pour caractériser l'information fournie par $x(j)$ sur l'éventuelle appartenance de $x(i)$ à la classe $\mathscr{C}_{\hat{\sigma}(j)}$.
La plus grande valeur que peut prendre cette mesure de similarité est 1. Cela ne se produit que si $x(i) = x(j)$ et l'erreur entre la sortie réelle $y(i)$ et la sortie reconstruite $\hat{\theta}_{\hat{\sigma}(j)}^\top \bar{\varphi}(i)$ est nulle. A l'inverse, la valeur minimum de cette mesure de similarité est 0. Cela se produit si la donnée $x(i)$ se situe à l'infini par rapport à $x(j)$ ou si l'erreur entre la sortie réelle et la sortie reconstruite est infinie. Dans ce cas, nous disons que la source d'information fournie par $x(j)$ apporte une information selon laquelle $x(i)$ ne peut pas être attribué à la classe $\mathscr{C}_{\hat{\sigma}(j)}$.

Dans l'équation (2.10), $\hat{\theta}_{\hat{\sigma}(j)}$ est le vecteur de paramètres associé à la classe $\mathscr{C}_{\hat{\sigma}(j)}$. Ce vecteur de paramètres est calculé en utilisant la technique des moindres carrés sur l'ensemble des données de la classe $\mathscr{C}_{\hat{\sigma}(j)}$ via la formule (2.9). La mesure (2.10) vise à la fois à minimiser la distance euclidienne entre les vecteurs $x(i)$ et $x(j)$ et l'erreur entre la sortie du sous-modèle $\hat{\sigma}(j)$ et la sortie $y(i)$ (voir Figure 2.4).
Les paramètres $\alpha_{\hat{\sigma}(j)}$ et $\beta_{\hat{\sigma}(j)}, \hat{\sigma}(j) \in \{1, \ldots, \bar{s}\}$ sont strictement positifs, et sont calculés

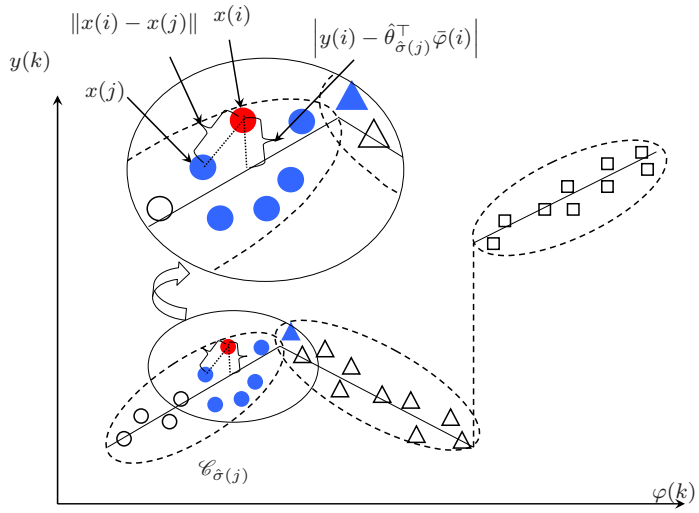


FIGURE 2.4 – Illustration de trois sous-modèles affines dans l'espace de régression augmenté.

selon les formules

$$\alpha_{\hat{\sigma}(j)} = \frac{1}{\frac{1}{\left|\mathscr{C}_{\hat{\sigma}(j)}\right|^2}\sum_{k,\hat{\sigma}(k)=\hat{\sigma}(j)}\sum_{l,\hat{\sigma}(l)=\hat{\sigma}(j)}\|x(k)-x(l)\|^2} \tag{2.11}$$

$$\beta_{\hat{\sigma}(j)} = \frac{1}{\frac{1}{\left|\mathscr{C}_{\hat{\sigma}(j)}\right|}\sum_{k,\hat{\sigma}(k)=\hat{\sigma}(j)}\left(y(k)-\hat{\theta}^{\top}_{\hat{\sigma}(j)}\bar{\varphi}(k)\right)^2} \tag{2.12}$$

où $|\mathscr{C}_q|$ représente la cardinalité de $\mathscr{C}_q$.

Le paramètre $\alpha_{\hat{\sigma}(j)}$ représente l'inverse de la moyenne du carré la distance entre chaque paire de données appartenant à la classe $\mathscr{C}_{\hat{\sigma}(j)}$. En d'autres termes, il représente l'inverse de la distance intra-classe de la classe $\mathscr{C}_{\hat{\sigma}(j)}$. Le paramètre $\beta_{\hat{\sigma}(j)}$ est l'inverse de la valeur moyenne du carré de l'erreur entre la sortie mesurée et la sortie du sous-modèle correspondant à la classe $\mathscr{C}_{\hat{\sigma}(j)}$.

A présent, nous disposons de c mesures de similarité pour décider sur la réaffectation de la donnée $x(i)$ aux $\bar{s}$ classes existantes. Il est alors nécessaire de combiner ces c mesures afin d'obtenir $\bar{s}$ critères d'appartenance. Ainsi, nous introduisons le critère d'appartenance de $x(i)$ à la classe $\mathscr{C}_q$ comme :

$$P\left(x(i)\in\mathscr{C}_q\right) = \frac{\sum_{k/x(k)\in\{\Gamma_c(x(i))\cap\mathscr{C}_q\}}\phi_k^i}{\sum_{k=1}^{c}\phi_k^i},\ q\in\{1,\ldots,\bar{s}\}\,. \tag{2.13}$$

Cette mesure sert de critère pour la réaffectation de données. Elle représente la somme de toutes les informations fournies par les plus proches voisins appartenant à la classe

$\mathscr{C}_q$ sur la somme de toutes les informations fournies par les c-ppv. On peut noter que $\sum_{q=1}^{\bar{s}} P(x(i) \in \mathscr{C}_q) = 1$ et $0 \leq P\left(x(i) \in \mathscr{C}_q\right) \leq 1$. Le critère d'appartenance $P\left(x(i) \in \mathscr{C}_q\right)$ vaut 1 si tous les éléments de $\Gamma_c\left(x(i)\right)$ appartiennent à la classe $\mathscr{C}_q$ et il vaut 0 si aucun élément de $\Gamma_c\left(x(i)\right)$ n'appartient à $\mathscr{C}_q$.

La décision se fait par l'affectation de $x(i)$ à la classe $\mathscr{C}_{ret}, ret \in \{1, \ldots, \bar{s}\}$ qui maximise le critère (2.13). Dans ce cas, l'affectation de $x(i)$ s'effectue par :

$$\mathscr{C}_{ret} = \mathscr{C}_{ret} \cup \{x\left(i\right)\} \quad \text{tel que} \quad ret = \underset{q=1,\ldots,\bar{s}}{\arg\max} \left(P\left(x(i) \in \mathscr{C}_q\right)\right). \tag{2.14}$$

Les données $x(i),\ i = 1, \ldots, N$, sont réaffectées après chaque itération. Les paramètres α_q et $\beta_q, q \in \{1, \ldots, \bar{s}\}$ ainsi que le nombre de classes $\bar{s}$ sont adaptés à chaque itération (après l'affectation de la donnée) afin de tenir compte de l'évolution des classes. La procédure de réaffectation de données est arrêtée dès que le critère d'arrêt est satisfait.

2.3.3 Critère d'arrêt

La mise en place d'un critère d'arrêt est nécessaire pour détecter la stabilisation de la partition de données. La procédure peut être arrêtée quand les performances n'augmentent plus, c'est-à-dire quand l'erreur d'identification (l'erreur calculée entre le système réel et le modèle reconstruit) ne diminue plus. Ainsi, le critère d'arrêt de la procédure de classification des données et d'identification des paramètres est basée sur la comparaison entre les vecteurs de paramètres antérieurs $\Theta^{(r)}$ et les vecteurs de paramètres postérieurs $\Theta^{(r+1)}$ où r est l'indice de l'itération et $\Theta^{(r)} = \left[\hat{\theta}_1^{(r)}, \ldots, \hat{\theta}_{\bar{s}}^{(r)}\right]$. Le critère d'arrêt est le suivant

$$\left\|\Theta^{(r+1)} - \Theta^{(r)}\right\| \leq \upsilon. \tag{2.15}$$

où υ est un seuil au choix de l'utilisateur. Dans nos simulations, un seuil υ de l'ordre de 10^{-5} a été utilisé.
Si le critère (2.15) est satisfait, alors la procédure de réaffectation des données est arrêtée, sinon on fait $r = r + 1$ et on continue la réaffectation des données.

En tenant compte de ce qui précède, la procédure de classification de données et d'estimation de paramètres est exposée dans la table **Algorithme 2.1**. Le paramètre c, c'est-à-dire le nombre de plus proches voisins, est le paramètre de réglage de notre l'algorithme. Les simulations montrent que plus c est petit, plus le nombre de sous-modèles obtenus par l'algorithme est grand. Ainsi, le choix de c dépend de la précision souhaitée sur le modèle estimé.

Algorithme 2.1

Étape 1 **Initialisation :**

- Fixer c, $\bar{s} = N$, $r = 0$, poser $\mathscr{C}_i = \{x(i)\}$, $i = 1, \ldots, \bar{s}$.
- Calculer les paramètres $\Theta^{(0)} = \left[\hat{\theta}_1^{(0)}, \ldots, \hat{\theta}_{\bar{s}}^{(0)}\right]$.

Étape 2 **Réaffectation des données :**

Pour $i = 1, \ldots, N$
- Pour tous les c-ppv $x(j) \in \Gamma_c(x(i))$, $j = 1, \ldots, c$, calculer ϕ_i^j, Eq. (2.10).
- Calculer les mesures d'appartenance $P(x(i) \in \mathscr{C}_q), q \in \{1, \ldots, \bar{s}\}$, Eq. (2.13).
- Décider sur l'affectation de $x(i)$, Eq. (2.14).

Fin pour.

- $\bar{s}$ = nombre de classes non vides.
- Adaptation des paramètres :
  - Adapter $\Theta^{(r)}$ en utilisant la méthode des moindres carrés sur les données de chaque classe non vide.
  - Adapter les paramètres α_q et $\beta_q, q \in \{1, \ldots, \bar{s}\}$, Eq. (2.11) et (2.12).

Étape 3 **Test de convergence :**

Si $\left\|\Theta^{(r+1)} - \Theta^{(r)}\right\| \leq \upsilon$, $s = \bar{s}$, terminer l'algorithme. Sinon faire $r = r + 1$ et retourner à l'étape 2.

Pour atténuer les problèmes des erreurs de classification de données de régression situées dans les zones de chevauchement de sous-modèles dans l'espace de régression, nous présentons dans la section suivante une classification de données basée sur une modélisation de connaissances incertaines par la théorie de Dempster-Shafer. Cette théorie permet de représenter explicitement, à partir d'outils mathématiques, l'incertitude liée aux connaissances.

2.3.4 Réaffectation de données basée sur la théorie de Dempster-Shafer

La théorie de Dempster-Shafer [77], [78] (ou théorie des fonctions de croyance) est un formalisme utilisé pour la modélisation de connaissances incertaines. Contrairement à la théorie bayésienne, elle ne se base pas sur une quantification probabiliste, mais sur un modèle plus général basé sur les fonctions de croyance. En effet, elle permet de raisonner dans l'incertain avec un degré de croyance suffisamment général pour englober les ap-

proches probabiliste classique et possibiliste. Le but de cette théorie est de permettre de combiner des informations distinctes pour décider sur une hypothèse définie.
La théorie de Dempster-Shafer repose sur deux niveaux de perception des informations. Le premier niveau est appelé "niveau crédal". Il permet la modélisation et la fusion de l'information par les masses de croyance. Le deuxième niveau est appelé "niveau pignistique". Ce niveau est entièrement dédié à la prise de décision et est clairement séparé de la modélisation des données.

2.3.4.1 Quelques rappels sur la théorie de Dempster-Shafer

Dans cette section, nous présentons le principe de la théorie de Dempster-Shafer. Nous définissons tout d'abord ce qu'on appelle une fonction de masse de croyance. Ensuite, nous montrons comment des masses de croyance issues de plusieurs sources d'information distinctes peuvent être combinées pour donner une seule masse de croyance grâce à la règle de combinaison de Dempster. Enfin, nous présentons le niveau pignistique de cette théorie où les mesures de croyance sont transformées en une probabilité pignistique permettant la prise de décision.

Définition de la fonction de masse : Considérons une variable aléatoire x prenant sa valeur dans un ensemble $\mathscr{C}$. Une connaissance partielle quant à l'hypothèse de l'appartenance de la valeur de x à un sous-ensemble de $\mathscr{C}$ ($x \in A$) peut être représentée par une masse de croyance [77], [78], définie comme une fonction m de $2^{\mathscr{C}}$ (ensemble des parties de $\mathscr{C}$) vers $[0,1]$, vérifiant :

$$\begin{aligned} m(\emptyset) &= 0 \\ \sum_{A \subseteq \mathscr{C}} m(A) &= 1. \end{aligned} \tag{2.16}$$

Les sous-ensembles de $\mathscr{C}$ qui reçoivent une masse non nulle sont appelés des éléments focaux de m. L'absence totale d'information concernant l'appartenance éventuelle de la valeur de x à un sous-ensemble de $\mathscr{C}$ peut être modélisée par une fonction telle que $m(\mathscr{C}) = 1$ et $m(A) = 0$ pour tout sous-ensemble strict de $\mathscr{C}$. Une connaissance parfaite conduit à l'allocation de la totalité de la masse à un singleton de $\mathscr{C}$. On parle alors de masse certaine.

Étant donnée une fonction de masse m, il est possible de définir une fonction de plausibilité pl et une fonction de croyance bel de $2^{\mathscr{C}}$ dans [0,1]

$$\begin{aligned} pl(A) &= \sum_{B|B \cap A \neq \emptyset} m(B), \\ bel(A) &= \sum_{B|B \subseteq A} m(B). \end{aligned} \tag{2.17}$$

La plausibilité *pl* désigne le degré avec lequel les informations disponibles ne discréditent pas l'hypothèse $x \in A$. La croyance *bel* représente le degré avec lequel les informations disponibles accréditent l'hypothèse $x \in A$.

Règle de combinaison des masses des croyances : La règle de combinaison originale, connue en tant que Règle de combinaison de Dempster, est une généralisation du théorème de Bayes. Cette règle met clairement en valeur l'accord entre des sources multiples et ignore les conflits grâce à un facteur de normalisation.
Plusieurs masses de croyances $m_1,\ m_2, \dots m_J$ issues de différentes sources d'information peuvent être combinées par la règle de combinaison de Dempster pour donner une nouvelle masse $m = m_1 \oplus m_2 \oplus \cdots \oplus m_J$ appelée la somme orthogonale de $m_1,\ m_2, \dots m_J$ et définie par :

$$\begin{aligned}&(m_1 \oplus m_2 \oplus \cdots \oplus m_J)(A) = \tfrac{1}{1-K} \sum_{A_1, A_2, \dots A_J \mid \bigcap_j A_j = A} \prod_{j=1}^{J} m_j(A_j),\\ &\forall A \subseteq \mathscr{C}, A \neq \emptyset.\end{aligned} \tag{2.18}$$

où $1 - K$ est un coefficient de normalisation tel que $K = \sum_{A_1, A_2, \dots A_J \mid \bigcap_j A_j = \emptyset} \prod_{j=1}^{J} m_j(A_j)$ représente la masse affectée à l'ensemble vide. En d'autres termes, la masse m représente la somme sur les ensembles d'éléments incluent dans $2^{\mathscr{C}}$ tels que l'intersection des éléments d'un ensemble donne A. Si on ne prend pas en compte la masse affectée à l'ensemble vide, on obtient la règle de combinaison non normalisée de Dempster appelée règle de combinaison conjonctive. Cette règle est définie par :

$$\begin{aligned}&(m_1 \textcircled{\cap} m_2 \textcircled{\cap} \cdots \textcircled{\cap} m_J)(A) = \sum_{A_1, A_2, \dots A_J \mid \bigcap_j A_j = A} \prod_{j=1}^{J} m_j(A_j),\\ &\forall A \subseteq \mathscr{C}, A \neq \emptyset.\end{aligned} \tag{2.19}$$

Prise de décision : Une caractéristique de la théorie de Dempster-Shafer est l'existence de deux niveaux d'interprétation des croyances : le niveau crédal et le niveau pignistique. Au niveau crédal, la connaissance est modélisée par l'intermédiaire des différentes mesures de croyance. Le niveau pignistique est, quant à lui, dédié à la prise de décision. Les mesures de croyance sont alors transformées en une distribution de probabilité, appelée probabilité pignistique $BetP$, au moyen d'une opération appelée transformation pignistique. Cette probabilité est définie par

$$BetP(A) = \sum_{A \subset B} \frac{m(B)}{|B|}, \quad \forall A \subset \mathscr{C}. \tag{2.20}$$

où $|B|$ représente la cardinalité de B.
A partir de cette distribution de probabilité, il est alors possible d'utiliser les outils classiques de la théorie de la décision statistique pour la prise de décision. Notons que la fonction $BetP$ satisfait à l'inégalité $bel\,(A) \leq BetP\,(A) \leq pl\,(A)$.

Exemple explicatif :
Le Docteur Dupont a été retrouvé mort la veille de son 30ème anniversaire. Trois personnes $\{\text{Pierre}, \text{Paul}, \text{Marie}\}$ sont suspectées de son assassinat. Les enquêteurs disposent de deux éléments d'enquête : un témoignage et un indice. En effet, un témoin croit avoir vu un homme sortir de la maison du docteur la nuit de sa mort avec une confiance de 0.7. D'autre part, un mégot a été retrouvé sur la scène du crime ce qui amène les enquêteurs à penser que le tueur est un fumeur avec une confiance de 0.6. Deux des trois suspects sont fumeurs : Pierre et Marie. Se basant sur ces deux éléments les enquêteurs doivent retrouver le meurtrier. Les enquêteurs décident alors de résoudre ce problème en utilisant la théorie de Dempster-Shafer.
L'ensemble de tous les suspects est $\mathscr{C} = \{\text{Pierre}, \text{Paul}, \text{Marie}\}$. L'ensemble des parties de $\mathscr{C}$ est donné par $2^{\mathscr{C}} = \{\emptyset, \{\text{Pierre}\}, \{\text{Paul}\}, \{\text{Marie}\}, \{\text{Pierre}, \text{Paul}\}, \{\text{Pierre}, \text{Marie}\}, \{\text{Paul}, \text{Marie}\}, \mathscr{C}\}$. Les enquêteurs disposent de deux sources d'informations qui sont :

Source 1 (Témoignage) : La masse allouée à l'hypothèse que $\{\text{Pierre}, \text{Paul}\}$ soient les meurtriers est égale à 0.7. Cette masse reste attachée à l'ensemble $\{\text{Pierre}, \text{Paul}\}$, elle ne peut être distribuée à ses éléments en l'absence d'informations complémentaires. La masse allouée à l'ignorance, c'est-à-dire à l'ensemble de tous les suspects $\mathscr{C}$ est donc égale à 0.3. Ceci se traduit par :

$$\begin{aligned} &m_1(\{\text{Pierre}, \text{Paul}\}) = 0.7, \\ &m_1(\mathscr{C}) = 0.3, \\ &m_1(\text{A}) = 0, \forall A \in 2^{\mathscr{C}} \setminus \{\{\text{Pierre}, \text{Paul}\}, \mathscr{C}\}. \end{aligned} \tag{2.21}$$

Source 2 (Indice) : La masse allouée à l'hypothèse que $\{\text{Pierre}, \text{Marie}\}$ soient les meurtriers est égale à 0.6. La masse allouée à l'ensemble de tous les suspects $\mathscr{C}$ est égale à 0.4. On a alors :

$$\begin{aligned} &m_2(\{\text{Pierre}, \text{Marie}\}) = 0.6, \\ &m_2(\mathscr{C}) = 0.4, \\ &m_2(\text{A}) = 0, \forall A \in 2^{\mathscr{C}} \setminus \{\{\text{Pierre}, \text{Marie}\}, \mathscr{C}\}. \end{aligned} \tag{2.22}$$

La règle de combinaison de Dempster (2.18) est ensuite utilisée pour combiner les masses issues des deux sources

$$m(A) = (m_1 \oplus m_2)(A), \ \forall A \in 2^{\mathscr{C}}.$$

Calculons, par exemple, la masse $m(\{\text{Pierre}\})$ issue de la combinaison des masses des deux sources d'information :

$$m(\{\text{Pierre}\}) = \frac{1}{1-K} \sum_{A_1,A_2 |\, A_1 \cap A_2 = \{\text{Pierre}\}} m_1(A_1) m_2(A_2),$$

On a

$$\begin{aligned}
&\sum_{A_1,A_2 |\, A_1 \cap A_2 = \{\text{Pierre}\}} m_1(A_1) m_2(A_2) = \\
&\quad m_1(\{\text{Pierre}\}) m_2(\{\text{Pierre}\}) + m_1(\{\text{Pierre}\}) m_2(\{\text{Pierre, Paul}\}) \\
&\quad + m_1(\{\text{Pierre}\}) m_2(\{\text{Pierre, Marie}\}) + m_1(\{\text{Pierre}\}) m_2(\mathscr{C}) \\
&\quad + m_1(\{\text{Pierre, Paul}\}) m_2(\{\text{Pierre}\}) + m_1(\{\text{Pierre, Paul}\}) m_2(\{\text{Pierre, Marie}\}) \\
&\quad + m_1(\{\text{Pierre, Marie}\}) m_2(\{\text{Pierre}\}) + m_1(\{\text{Pierre, Marie}\}) m_2(\{\text{Pierre, Paul}\}) \\
&\quad + m_1(\mathscr{C}) m_2(\{\text{Pierre}\}) \\
&\quad = 0.7 \times 0.6 = 0.42.
\end{aligned}$$

Et

$$\begin{aligned}
K &= \sum_{A_1,A_2 |\, A_1 \cap A_2 = \emptyset} m_1(A_1) m_2(A_2) \\
&= m_1(\{\text{Pierre}\}) m_2(\{\text{Paul}\}) + m_1(\{\text{Pierre}\}) m_2(\{\text{Marie}\}) \\
&+ m_1(\{\text{Paul}\}) m_2(\{\text{Pierre}\}) + m_1(\{\text{Paul}\}) m_2(\{\text{Marie}\}) \\
&+ m_1(\{\text{Marie}\}) m_2(\{\text{Pierre}\}) + m_1(\{\text{Marie}\}) m_2(\{\text{Paul}\}) \\
&+ m_1(\{\text{Pierre}\}) m_2(\{\text{Paul, Marie}\}) + m_1(\{\text{Paul}\}) m_2(\{\text{Pierre, Marie}\}) \\
&+ m_1(\{\text{Marie}\}) m_2(\{\text{Pierre, Paul}\}) + m_1(\{\text{Pierre, Paul}\}) m_2(\{\text{Marie}\}) \\
&+ m_1(\{\text{Pierre, Marie}\}) m_2(\{\text{Paul}\}) + m_1(\{\text{Paul, Marie}\}) m_2(\{\text{Pierre}\}) \\
&= 0.
\end{aligned}$$

Alors $m(\{\text{Pierre}\}) = 0.42$.
La masse issue de la règle de combinaison, la plausibilité *pl*, la croyance *bel* ainsi que la probabilité pignistique $BetP$ sont calculées et présentées sur le tableau 2.1. A partir de ce tableau, on constate que la probabilité pignitique la plus élevée est celle de Pierre. Cela signifie que le coupable serait Pierre.

2.3.4.2 Réaffectation de données par les plus proches voisins basés sur la théorie de Dempster-Shafer

Comme nous l'avons évoqué dans la section 2.3.2, puisque les données appartiennent à des modèles localement affines, leur réaffectation peut donc être basée sur les informations fournies par leurs c plus proches voisins (c-ppv). Les plus proches voisins basés utilisant

A	$m_1(A)$	$m_2(A)$	$m(A)$	$bel(A)$	$pl(A)$	$BetP(A)$	Décision
{Pierre}	0	0	0.42	0.42	1	0.69	Coupable
{Paul}	0	0	0	0	0.4	0.18	Innocent
{Marie}	0	0	0	0	0.3	0.13	Innocente
{Pierre, Paul}	0.7	0	0.28	0.7	1	x	x
{Pierre, Marie}	0	0.6	0.18	0.6	1	x	x
{Paul, Marie}	0	0	0	0	0.58	x	x
$\mathscr{C}$	0.3	0.4	0.12	1	1	x	x
	Niveau crédal					Niveau pignistique	

TABLE 2.1 – Masses de croyance, plausibilités, croyances et probabilités pignistiques des différents sous-ensembles de $\mathscr{C}$.

la théorie de Dempster-Shafer [28] sont une alternative à la procédure de réaffectation de données présentée précédemment pour modéliser les connaissances incertaines. La réaffectation d'une nouvelle donnée de régression $x(i)$ est basée sur les apports d'information des masses de croyance fournies par ses c plus proches voisins. De plus, la théorie de Dempster-Shafer est dotée de règles efficaces de fusion de masses de croyance permettant de fusionner les informations apportées par les c plus proches voisins.
Le degré de croyance qu'offre $x(j)$, un des plus proches voisins de $x(i)$, peut être assimilé à une fonction de croyance qui ne possède que deux éléments focaux qui sont $\left\{\mathscr{C}_{\hat{\sigma}(j)}\right\}$ et $\mathscr{C} = \{\mathscr{C}_1, \ldots, \mathscr{C}_{\bar{s}}\}$ (ensemble de toutes les classes). Une partie de la croyance sera affectée à la classe $\mathscr{C}_{\hat{\sigma}(j)}$ et le reste à l'ensemble $\mathscr{C}$. Dans ce cas, l'information fournie par $x(j)$ sur l'éventuelle appartenance de $x(i)$ à la classe $\mathscr{C}_{\hat{\sigma}(j)}$ peut être donnée par la masse m_j^i définie par

$$m_j^i(A) = \begin{cases} \gamma\phi_j^i & \text{si } A = \left\{\mathscr{C}_{\hat{\sigma}(j)}\right\} \\ 1 - \gamma\phi_j^i & \text{si } A = \mathscr{C} \\ 0 & \text{si } A \in 2^{\mathscr{C}} \setminus \left\{\left\{\mathscr{C}_{\hat{\sigma}(j)}\right\}, \mathscr{C}\right\}, \end{cases} \tag{2.23}$$

où ϕ_j^i est défini par l'équation (2.10).
$m_j^i(\mathscr{C})$ est la fraction de masse allouée à l'ignorance sur l'affectation de la donnée $x(i)$ à la classe $\mathscr{C}_{\hat{\sigma}(j)}$. γ est un paramètre tel que $0 << \gamma < 1$. Il empêche l'allocation de toute la masse de croyance à la classe $\mathscr{C}_{\hat{\sigma}(j)}$. En effet, on a vu que $0 < \phi_j^i \leq 1$. Alors si $\phi_j^i = 1$ et si $\gamma = 1$ toute la masse sera affectée à $\left\{\mathscr{C}_{\hat{\sigma}(j)}\right\}$ puisque $\sum_{A \subseteq \mathscr{C}} m_j^i(A) = 1$.
Chacun des plus proches voisins pourra alors être considéré comme une part de croyance quant à l'appartenance de $x(i)$ à l'une des classes existantes. Si nous considérons les c-plus proches voisins de $x(i)$, nous obtenons alors c masses de croyances $m_1^i, m_2^i, \ldots, m_c^i$.

Dans notre cas, on ne prend pas en compte la masse affectée à l'ensemble vide. Pour combiner les c masses de croyance, on utilise alors la règle de combinaison conjonctive (règle de combinaison non normalisée de Dempster). Cette règle donne

$$m^i(\{\mathscr{C}_q\}) = (m_1^i \textcircled{\cap} \ldots \textcircled{\cap} m_c^i)(\{\mathscr{C}_q\}),\ \forall q = 1, \ldots, \bar{s} \tag{2.24}$$

$$m^i(\mathscr{C}) = (m_1^i \textcircled{\cap} \ldots \textcircled{\cap} m_c^i)(\mathscr{C}) \tag{2.25}$$

$$m^i(A) = (m_1^i \textcircled{\cap} \ldots \textcircled{\cap} m_c^i)(A) = 0,\ \forall A \in 2^{\mathscr{C}} \setminus \left\{ \{\mathscr{C}_q\}_{q=1}^{\bar{s}}, \mathscr{C} \right\}. \tag{2.26}$$

Calcul de $m^i(\mathscr{C})$: Pour calculer $m^i(\mathscr{C})$, on applique directement la formule (2.19) permettant la fusion de plusieurs masses de croyance comme suit :

$$m^i(\mathscr{C}) = \sum_{A_1, A_2, \ldots A_c | \bigcap_j A_j = \mathscr{C}} \prod_{j=1}^{c} m_j^i(A_j). \tag{2.27}$$

Il est évident que $\forall A_i \in 2^{\mathscr{C}}$: $\left(\bigcap_j A_j = \mathscr{C} \right) \Rightarrow (A_j = \mathscr{C}),\ j = 1, \ldots, c$. L'expression (2.27) devient alors

$$m^i(\mathscr{C}) = \prod_{j=1}^{c} m_j^i(\mathscr{C}) \tag{2.28}$$

$$= \prod_{j=1}^{c} (1 - \gamma \phi_j^i). \tag{2.29}$$

Calcul de $m^i(\{\mathscr{C}_q\})\ \forall q = 1, \ldots, \bar{s}$: L'utilisation directe de la formule (2.19) dans ce cas n'est pas évidente. Pour calculer $m^i(\{\mathscr{C}_q\})$, on fusionne, dans un premier temps, les masses de croyance $m_j^i,\ x(j) \in \mathscr{C}_q$ issues des plus proches voisins appartenant à la classe $\mathscr{C}_q$. On appelle la masse résultante $\mu_q^i(\{\mathscr{C}_q\})$. Dans un deuxième temps, on fusionne les masses de croyance $m_j^i,\ x(j) \notin \mathscr{C}_q$ issues des plus proches voisins appartenant aux autres classes $\mathscr{C}_p,\ p = 1, \ldots, \bar{s}, p \neq q$. On appelle la masse résultante $\mu_{\bar{q}}^i(\{\mathscr{C}_q\})$. La fusion des deux masses $\mu_q^i(\{\mathscr{C}_q\})$ et $\mu_{\bar{q}}^i(\{\mathscr{C}_q\})$ donne la masse $m^i(\{\mathscr{C}_q\})$. Cette démarche est possible car la règle de combinaison conjonctive (2.19) est commutative et associative [76]. On a alors

$$m^i(\{\mathscr{C}_q\}) = (m_1^i \textcircled{\cap} \ldots \textcircled{\cap} m_c^i)(\{\mathscr{C}_q\}),\ \forall q = 1, \ldots, \bar{s} \tag{2.30}$$

$$= (\mu_q^i \textcircled{\cap} \mu_{\bar{q}}^i)(\{\mathscr{C}_q\}). \tag{2.31}$$

L'application de la formule (2.19) sur l'équation (2.31) donne :

$$(\mu_q^i \textcircled{\cap} \mu_{\bar{q}}^i)(\{\mathscr{C}_q\}) = \mu_q^i(\{\mathscr{C}_q\}) \cdot \mu_{\bar{q}}^i(\{\mathscr{C}_q\}) + \mu_q^i(\{\mathscr{C}_q\}) \cdot \mu_{\bar{q}}^i(\mathscr{C}) + \mu_q^i(\mathscr{C}) \cdot \mu_{\bar{q}}^i(\{\mathscr{C}_q\}). \tag{2.32}$$

En réalité, la masse $\mu^i_{\bar{q}}(\{\mathscr{C}_q\})$ est nulle car, comme il est indiqué dans l'équation (2.23), les masses $m^i_j(\{\mathscr{C}_q\}) = 0$ si $x(j) \notin \mathscr{C}_q$. Par conséquent, l'équation (2.32) s'écrit :

$$(\mu^i_q \textcircled{\cap} \mu^i_{\bar{q}})(\{\mathscr{C}_q\}) = \mu^i_q(\{\mathscr{C}_q\}) \cdot \mu^i_{\bar{q}}(\mathscr{C}). \tag{2.33}$$

Pour calculer la masse $m^i(\{\mathscr{C}_q\})$, il suffit alors de calculer les deux masses $\mu^i_q(\{\mathscr{C}_q\})$ et $\mu^i_{\bar{q}}(\{\mathscr{C}\})$ et de les multiplier.

Calculons d'abord la masse $\mu^i_q(\{\mathscr{C}_q\})$. On sait que $\sum_{A \subseteq \mathscr{C}} \mu^i_q(A) = 1$, alors $\mu^i_q(\{\mathscr{C}_q\}) + \mu^i_q(\mathscr{C}) = 1$. On a donc :

$$\mu^i_q(\{\mathscr{C}_q\}) = 1 - \mu^i_q(\mathscr{C}) \tag{2.34}$$

$$= 1 - \prod_{j/x(j)\in\mathscr{C}_q} \left(1 - \gamma\phi^j_i\right). \tag{2.35}$$

Par application de la règle (2.19), la masse $\mu^i_{\bar{q}}(\mathscr{C})$ est donnée par

$$\mu^i_{\bar{q}}(\mathscr{C}) = \prod_{\substack{p=1 \\ p\neq q}}^{\bar{s}} \prod_{j/x(j)\in\mathscr{C}_p} \left(1 - \gamma\phi^j_i\right). \tag{2.36}$$

Enfin, l'expression de la masse de croyance $m^i(\{\mathscr{C}_q\})$, $\forall q = 1, \ldots, \bar{s}$ peut être donnée par

$$m^i(\{\mathscr{C}_q\}) = (1 - \prod_{j/x(j)\in\mathscr{C}_q} \left(1 - \gamma\phi^j_i\right)) \prod_{\substack{p=1 \\ p\neq q}}^{\bar{s}} \prod_{j/x(j)\in\mathscr{C}_p} \left(1 - \gamma\phi^j_i\right). \tag{2.37}$$

La croyance et la plausibilité sont déduites de l'équation (2.17) et elles sont données par

$$\begin{aligned} pl^i\left(\{\mathscr{C}_q\}\right) &= m^i\left(\{\mathscr{C}_q\}\right) + m^i\left(\mathscr{C}\right) \\ bel^i\left(\{\mathscr{C}_q\}\right) &= m^i\left(\{\mathscr{C}_q\}\right). \end{aligned} \tag{2.38}$$

D'après l'équation (2.20), la probabilité pignistique est alors exprimée par

$$BetP^i(\{\mathscr{C}_q\}) = m^i\left(\{\mathscr{C}_q\}\right) + \frac{m^i(\mathscr{C})}{\bar{s}}. \tag{2.39}$$

Une fois que les masses de croyance de chaque classe sont transformées en probabilité pignistique, la décision peut être déduite immédiatement. En effet, la décision se fait par l'affectation de $x(i)$ à la classe $\mathscr{C}_{ret}, ret \in \{1, \ldots, \bar{s}\}$ qui réalise le maximum de probabilité pignistique $BetP$. Dans ce cas, l'affectation de $x(i)$ s'effectue par la règle :

$$\mathscr{C}_{ret} = \mathscr{C}_{ret} \cup \{x(i)\} \quad \text{tel que} \quad ret = \max_q \left(BetP^i\left(\{\mathscr{C}_q\}\right)\right). \tag{2.40}$$

La majeure différence entre cette approche est celle présentée dans la section 2.3.2, peut se résumer en deux points. Le premier point est que dans l'approche basée sur la théorie de Dempster-Shafer, contrairement à la première approche, une partie de la croyance concernant l'hypothèse de l'appartenance d'une donnée quelconque à un ensemble $\mathscr{C}_q$ est attribuée à l'ensemble $\mathscr{C}$ de toutes les classes. Cette quantité représente la fraction de masse allouée à l'ignorance. En d'autres termes, nous ne pouvons jamais affirmer qu'une source d'information (ici un plus proche voisin) est entièrement fiable. Comme nous pouvons le constater dans l'expression de la probabilité pignistique (équation (2.39)), la masse allouée à l'ignorance est redistribuée sur les différentes classes existantes. Le deuxième point réside dans la façon dont les mesures de similarité fournies par les c-ppv sont combinées. En effet, la règle de combinaison de Dempster fournit une relation efficace pour combiner les mesures de confiance provenant de différentes sources d'information.

La procédure modifiée de classification de données et d'estimation de paramètres est présentée dans l'**Algorithme 2.2**. Il est à noter que l'étape d'initialisation et le critère d'arrêt resteront les mêmes que ceux de l'algorithme 2.1.

2.3.5 Exemples numériques

Exemple 2.1. Approximation de la fonction sinus

Les modèles dynamiques affines par morceaux sont une alternative très attractive pour approximer les systèmes non linéaires. Pour montrer les potentialités de notre approche en termes d'approximation de non-linéarité, nous considérons dans cet exemple la fonction sinusoïdale $y(k) = \sin(\varphi(k))$ où $\varphi(k) = 2\pi k/125$. Le domaine de définition considéré est $[0, 2\pi]$. Un ensemble de $N = 125$ données est alors généré. La modélisation par les modèles PWA décompose le domaine de définition en un nombre fini d'intervalles. Sur chacun de ces intervalles, un modèle affine est utilisé pour approximer la non-linéarité.

La procédure de classification des données et d'estimation de paramètres est appliquée pour différentes valeurs de c ($c = 10$ et $c = 15$). Les deux algorithmes (Algorithme 2.1 et 2.2) présentés précédemment sont employés. Nous représentons sur la figure 2.5 la fonction sinus ainsi que les modèles affines estimés et leurs domaines de validité.

Pour $c = 15$, cinq modèles affines ont été identifiés par les deux approches de réaffectation de données alors que pour $c = 10$, six modèles affines ont été identifiés par l'Algorithme 2.1 et sept par l'Algorithme 2.2. Les vecteurs de paramètres estimés par les deux approches sont données dans les tables 2.2 et 2.3.

Pour mesurer la similitude entre la fonction sinus et son approximée par des modèles locaux affines par morceaux, nous calculons la moyenne des erreurs quadratiques définie

Algorithme 2.2

Étape 1 **Initialisation :**

– Fixer c, $\bar{s} = N$, $r = 0$, $\gamma = 0.9$, poser $\mathscr{C}_i = \{x(i)\}$, $i = 1, \ldots, \bar{s}$.
– Calculer les paramètres $\Theta^{(0)} = \left[\hat{\theta}_1^{(0)}, \ldots, \hat{\theta}_{\bar{s}}^{(0)}\right]$.

Étape 2 **Réaffectation des données :**

Pour $i = 1, \ldots, N$
– Pour tous les c-ppv $x(j) \in \Gamma_c(x(i))$, $j = 1, \ldots, c$, calculer ϕ_i^j, Eq. (2.10).
– Combiner toutes les masses de croyance en utilisant la règle de combinaison de Dempster, Eq. (2.29) et (2.37).
– Calculer les probabilités pignistiques $BetP^i(\{\mathscr{C}_q\}), q \in \{1, \ldots, \bar{s}\}$, Eq. (2.39).
– Décider sur l'affectation de $x(i)$, Eq. (2.40).
Fin pour.

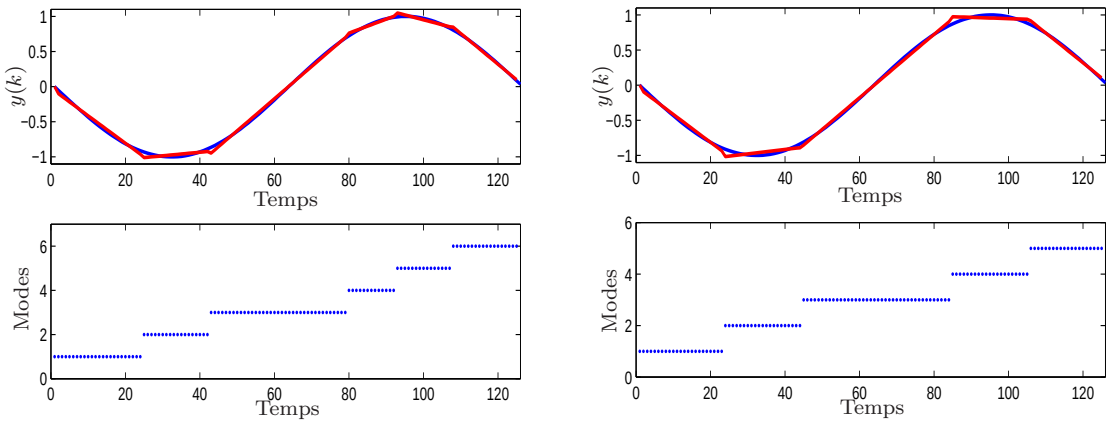
– $\bar{s}$ = nombre de classes non vides.
– Adaptation des paramètres :
  - Adapter $\Theta^{(r)}$ en utilisant la méthode des moindres carrés sur les données de chaque classe non vide.
  - Adapter les paramètres α_q et $\beta_q, q \in \{1, \ldots, \bar{s}\}$, Eq. (2.11) et (2.12).

Étape 3 **Test de convergence :**

Si $\left\|\Theta^{(r+1)} - \Theta^{(r)}\right\| \leq \upsilon$, $s = \bar{s}$, terminer l'algorithme. Sinon faire $r = r + 1$ et retourner à l'étape 2.

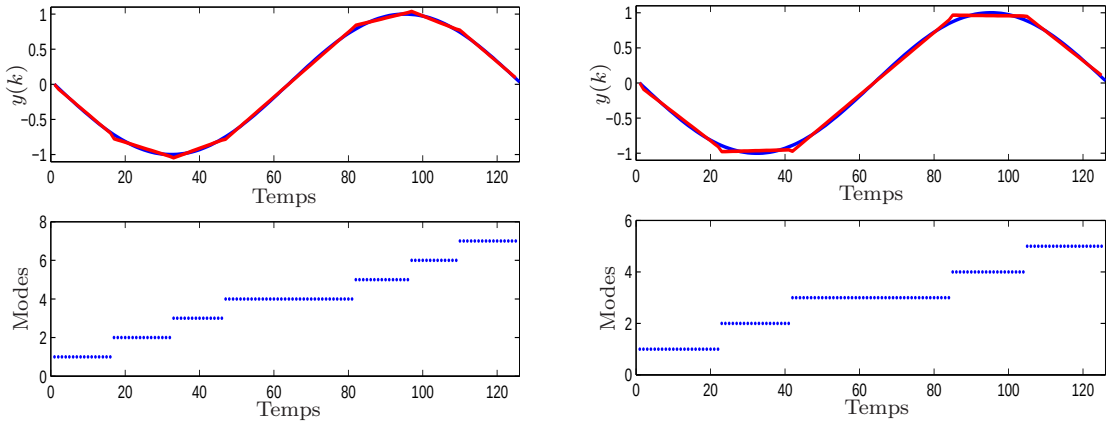
c	Les vecteurs de paramètres estimés
10	$\hat{\theta}_1 = \left[-0.798\ -2.563\right]$, $\hat{\theta}_2 = \left[0.004\ -0.967\right]$, $\hat{\theta}_3 = \left[0.779\ -0.071\right]$, $\hat{\theta}_4 = \left[0.854\ 0.024\right]$, $\hat{\theta}_5 = \left[0.036\ 0.909\right]$, $\hat{\theta}_6 = \left[-0.807\ 2.587\right]$,
15	$\hat{\theta}_1 = \left[-0.798\ -2.563\right]$, $\hat{\theta}_2 = \left[0.125\ -0.766\right]$, $\hat{\theta}_3 = \left[0.900\ -0.005\right]$, $\hat{\theta}_4 = \left[-0.036\ 1.013\right]$, $\hat{\theta}_5 = \left[-0.852\ 2.710\right]$

TABLE 2.2 – Les paramètres estimés de la fonction sinus par l'Algorithme 2.1.



$c = 10$ $c = 15$

Approximation par l'Algorithme 2.1



$c = 10$ $c = 15$

Approximation par l'Algorithme 2.2

FIGURE 2.5 – Approximation de la fonction sinus par des modèles affines par morceaux.

par :

$$\mathrm{SSE} = \frac{1}{N-n} \sum_{k=n+1}^{N} \left(y\left(k\right) - \hat{y}\left(k\right) \right)^2 \tag{2.41}$$

Les moyennes des erreurs quadratiques SSE des différents cas sont présentées dans le tableau 2.4. On peut constater que les moyennes des erreurs quadratiques obtenues dans le cas de la deuxième méthode de réaffectation de données (Algorithme 2.2) sont inférieures à celles trouvées par la première méthode (Algorithme 2.1). Cela montre que, sur cet exemple, l'approche basée sur la théorie de Dempster-Shafer est plus performante puisque elle a donné de meilleurs résultats. On constate également que plus le nombre de sous-modèles augmente, plus la moyenne des erreurs quadratiques diminue. Ceci est justifié puisque la précision augmente avec le nombre de sous-modèles.

c	Les vecteurs de paramètres estimés
10	$\hat{\theta}_1 = \begin{bmatrix} -0.858 & -2.728 \end{bmatrix}$, $\hat{\theta}_2 = \begin{bmatrix} -0.216 & -1.340 \end{bmatrix}$, $\hat{\theta}_3 = \begin{bmatrix} 0.456 & -0.373 \end{bmatrix}$, $\hat{\theta}_4 = \begin{bmatrix} 0.926 & -0.003 \end{bmatrix}$, $\hat{\theta}_5 = \begin{bmatrix} 0.255 & 0.608 \end{bmatrix}$, $\hat{\theta}_6 = \begin{bmatrix} -0.419 & 1.732 \end{bmatrix}$, $\hat{\theta}_7 = \begin{bmatrix} -0.903 & 2.855 \end{bmatrix}$,
15	$\hat{\theta}_1 = \begin{bmatrix} -0.814 & -2.606 \end{bmatrix}$, $\hat{\theta}_2 = \begin{bmatrix} 0.028 & -0.918 \end{bmatrix}$, $\hat{\theta}_3 = \begin{bmatrix} 0.889 & -0.000 \end{bmatrix}$, $\hat{\theta}_4 = \begin{bmatrix} -0.012 & 0.978 \end{bmatrix}$, $\hat{\theta}_5 = \begin{bmatrix} -0.837 & 2.670 \end{bmatrix}$

TABLE 2.3 – Les paramètres estimés de la fonction sinus par l'Algorithme 2.2.

c	SSE par Algorithme 2.1	SSE par Algorithme 2.2
10	$7.40 \cdot 10^{-4}$	$2.85 \cdot 10^{-4}$
15	$15.2 \cdot 10^{-4}$	$9.21 \cdot 10^{-4}$

TABLE 2.4 – Les moyennes des erreurs quadratiques SSE de la fonction sinus des deux approches pour différentes valeurs de c.

Exemple 2.2. Identification d'un modèle PWARX

A titre illustratif, nous considérons maintenant un modèle affine par morceaux SISO composé de trois sous-modèles ($s = 3$) d'ordre un ($n_a = n_b = 1$) défini par les équations suivantes :

$$y(k) = \begin{cases} \begin{bmatrix} 0.4 & 0.5 & 0.3 \end{bmatrix} \bar{\varphi}(k) + \varepsilon(k) & \text{si } \varphi(k) \in \Re_1, \\ \begin{bmatrix} -0.7 & 0.6 & -0.5 \end{bmatrix} \bar{\varphi}(k) + \varepsilon(k) & \text{si } \varphi(k) \in \Re_2, \\ \begin{bmatrix} 0.4 & -0.2 & -0.2 \end{bmatrix} \bar{\varphi}(k) + \varepsilon(k) & \text{si } \varphi(k) \in \Re_3, \end{cases} \tag{2.42}$$

où $\bar{\varphi}(k) = \begin{bmatrix} \varphi^\top(k) & 1 \end{bmatrix}^\top$ et $\varphi(k) = [y(k-1)\ u(k-1)]^\top$

avec

$$\Re_1 = \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} 1 & 0.3 & 0 \end{bmatrix} \bar{\varphi} \geq 0 \text{ et } \begin{bmatrix} 0 & 0.5 & 0 \end{bmatrix} \bar{\varphi} > 0 \right\}$$
$$\Re_2 = \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} 1 & 0.3 & 0 \end{bmatrix} \bar{\varphi} \leq 0 \text{ et } \begin{bmatrix} 1 & -0.3 & 0 \end{bmatrix} \bar{\varphi} < 0 \right\}$$
$$\Re_3 = \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} 1 & -0.3 & 0 \end{bmatrix} \bar{\varphi} \geq 0 \text{ et } \begin{bmatrix} 0 & 0.5 & 0 \end{bmatrix} \bar{\varphi} \leq 0 \right\}$$

L'entrée d'excitation $u(k)$ et le bruit $\varepsilon(k)$ sont tous les deux générés selon une loi normale de moyenne nulle et de variance 0.5 et 0.05 respectivement. Le rapport signal sur bruit

SNR est de 20 dB.

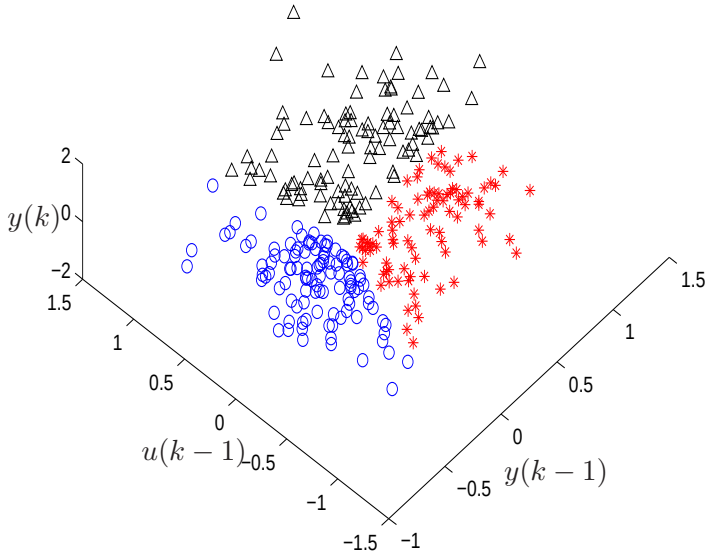


FIGURE 2.6 – Classification de données dans l'espace $\mathbb{R}^{n+1}$ avec $c = 50$

.

Nous effectuons sur le modèle (2.42) une simulation de Monte-Carlo de taille 100 avec différentes réalisations du bruit et de l'entrée d'excitation. Un ensemble de 300 données est généré suivant ce modèle pour chaque réalisation.
La procédure de classification des données et d'estimation de paramètres de l'algorithme 2.2 a été appliquée à ce jeu de données en choisissant le paramètre de réglage (le nombre des plus proches voisins) $c = 50$. Les résultats de la classification des données pour une de ces réalisations sont représentés sur la figure 2.6. Le nombre de classes obtenues est de 3. On représente sur la figure 2.7, la sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié. La séquence des modes estimés ainsi que l'erreur d'identification sont également tracées. On constate que la sortie reconstruite est presque identique à la sortie réelle et que l'erreur d'identification ne dépasse pas le bruit introduit dans le modèle.
Nous avons également calculé, pour chaque réalisation, la moyenne des erreurs quadratiques (SSE) définie dans (2.41) et le nombre d'itérations. Les résultats de ces calculs sont représentés sur les histogrammes de la figure 2.8. Nous notons que l'algorithme converge après un nombre d'itérations très réduit qui ne dépasse pas 12 itérations. La valeur maxi-

male du SSE avoisine 3×10^{-3} et la valeur minimale est de 1.7×10^{-3}. Le temps total pour identifier les modèles PWA de toutes les réalisations est de 162.90 secondes, soit un temps moyen de 16,29 secondes par réalisation. Ce temps semble être correct pour un jeu de 300 données. Les vecteurs de paramètres estimés $\hat{\theta}_i$ sont donnés par

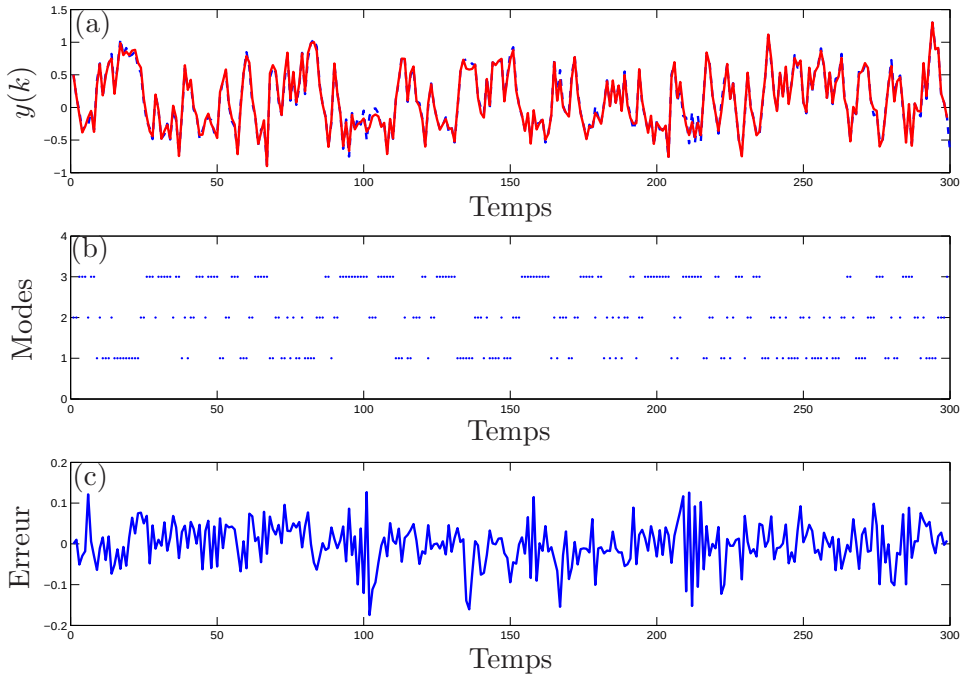


FIGURE 2.7 – (a) La sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié. (b) La séquence des modes estimés. (c) L'erreur d'identification.

$$\hat{\theta}_1 = \begin{bmatrix} 0.3962 & 0.5102 & 0.3010 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} -0.7160 & 0.6119 & -0.4974 \end{bmatrix},$$
$$\hat{\theta}_3 = \begin{bmatrix} 0.4054 & -0.2386 & -0.2140 \end{bmatrix}.$$

On peut remarquer que les vecteurs de paramètres estimés sont très proches des vecteurs de paramètres réels, et ce, malgré le niveau élevé du bruit.

Pour voir l'influence du paramètre c (le nombre des plus proches voisins) sur l'estimation du nombre de sous-modèles, nous avons appliqué la procédure de classification de données et d'estimation de paramètres pour différentes valeurs de c sur le même jeu de données (d'une seule réalisation). La figure 2.9 montre la variation du nombre de sous-modèles s en fonction du paramètre c. On peut noter que plus le nombre de plus proches voisins c est petit, plus le nombre de sous-modèles estimés est grand sur tout l'intervalle [5,40].

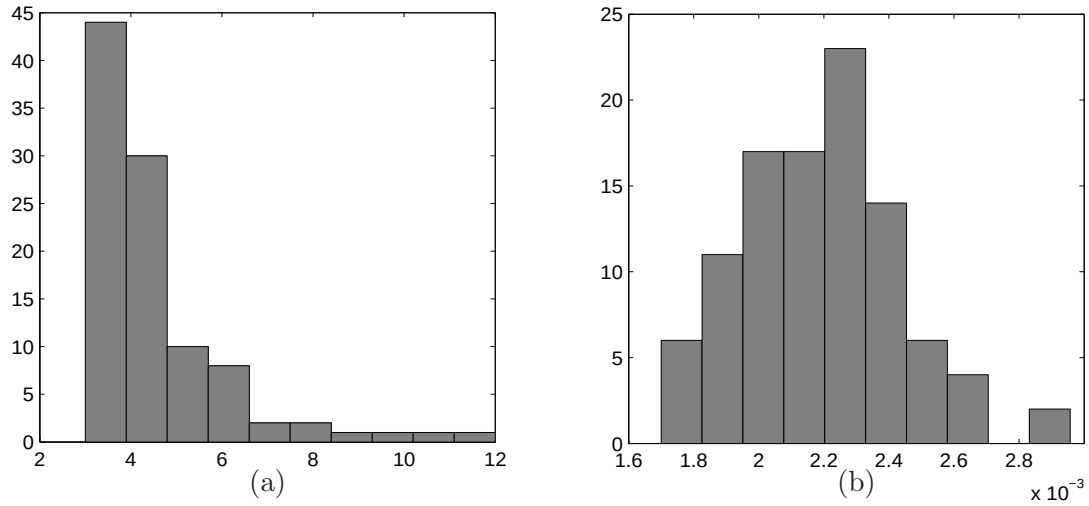


FIGURE 2.8 – Histogramme d'une simulation de Monte-Carlo (100 réalisations de 300 données), (a) Nombre d'itérations, (b) Moyenne des erreurs quadratiques (SSE).

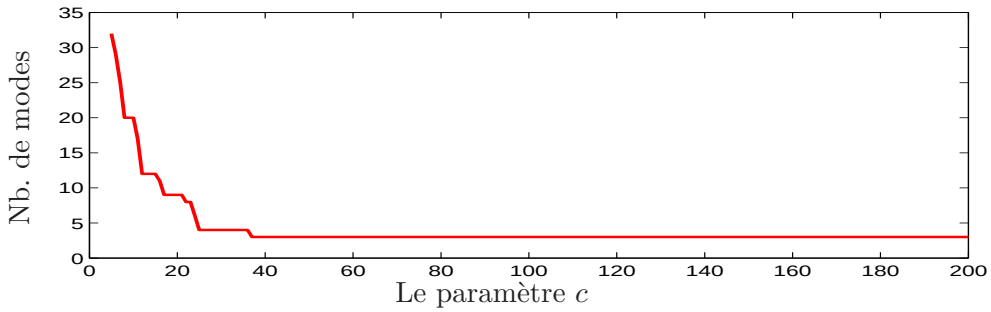


FIGURE 2.9 – Variation du nombre de sous-modèles s en fonction du paramètre c.

Néanmoins, à partir de la valeur $c = 40$, on constate que le nombre de sous-modèles estimés se stabilise et qu'il est égal à 3. Si le paramètre c augmente plus (par exemple $c = N$), le nombre de sous-modèles serait égal à 1.

2.4 Extension aux modèles dynamiques non linéaires par morceaux

Les modèles dynamiques affines par morceaux ont été introduits en vue d'approximer des systèmes non linéaires. Quand les systèmes présentent des non linéarités fortes et très différentes, leur approximation par des modèles PWA avec une faible erreur d'identification peut nécessiter un nombre important de sous-modèles affines. Afin de diminuer le nombre de sous-modèles composant le modèle PWA, une alternative serait alors de

modéliser ce système par des modèles dynamiques non linéaires par morceaux (en anglais : PWNARX- PieceWise Nonlinear AutoRegressive), notés PWN par la suite. En effet, chaque sous-modèle non linéaire peut remplacer un certain nombre de sous-modèles affines avec approximativement la même précision. La procédure d'identification de paramètres et de classification de données des modèles PWARX présentée dans la section 2.3 peut être modifiée pour traiter la modélisation par des modèles dynamiques non linéaires par morceaux.

2.4.1 Définition de modèles dynamiques non linéaires par morceaux

Les modèles dynamiques non linéaires par morceaux sont définis par :

$$y\left(k\right) = f_{\sigma(k)}(\varphi(k)) + e(k). \tag{2.43}$$

où $\varepsilon(k)$ est l'erreur de prédiction et $\varphi(k)$ est le vecteur de régression, il est défini par l'équation (2.2).

$f_{\sigma(k)}$ est une fonction non linéaire. On considère qu'il existe un espace de dimension plus élevé, possiblement infini, dans lequel $f_{\sigma(k)}$ peut être considérée comme linéaire par rapport à un vecteur $\eta(\varphi)$ où η est une fonction non linéaire choisie *a priori*. La fonction $f_{\sigma(k)}$ peut alors s'écrire comme suit :

$$f_{\sigma(k)}(\varphi) = w_{\sigma(k)}^{\top}\eta(\varphi) + \rho_{\sigma(k)}, \tag{2.44}$$

avec $w_{\sigma(j)}$ un vecteur de poids et $\rho_{\sigma(j)}$ un biais.

$\sigma(k)$ est la loi de commutation, elle est définie par l'équation (2.4). Les régions $\Re_i$ peuvent être définies par une loi plus générale que celle donnée en (2.5). Les frontières séparatrices considérées dans (2.5) sont des hyperplans et donc des frontières linéaires. Cependant, cette condition peut être relâchée en considérant des frontières non linéaires délimitant les différentes régions. Dans ce cas, les régions $\Re_i$ sont définies par :

$$\Re_i = \{\varphi \in \mathbb{R}^n : h_i(\varphi) \leq 0\} \tag{2.45}$$

où h_i est la fonction non linéaire délimitant la région $\Re_i$.

2.4.2 Identification des modèles dynamiques non linéaires par morceaux

L'objectif est de séparer les données de régression selon leurs sous-modèles non linéaires respectifs. Puisque les données appartiennent à des modèles localement non linéaires, la stratégie employée dans le cas des PWA (c'est-à-dire que deux données $x(i)$ et $x(j)$ suffisamment proches sont susceptibles d'appartenir à un même sous-modèle linéaire) peut être utilisée ici en modifiant la mesure de similarité (2.10). Cette mesure devient dans ce cas :

$$\phi_i^j = \exp\left(-\alpha_{\hat{\sigma}(j)} \left\|x(i) - x(j)\right\|^2 - \beta_{\hat{\sigma}(j)} \left(y(i) - f_{\hat{\sigma}(j)}(\varphi(i))\right)^2\right). \tag{2.46}$$

Dans ce cas, les paramètres $\alpha_{\hat{\sigma}(j)}$ et $\beta_{\hat{\sigma}(j)}, \hat{\sigma}(j) \in \{1, \ldots, \bar{s}\}$ sont calculés par :

$$\alpha_{\hat{\sigma}(j)} = \frac{1}{\frac{1}{\left|\mathscr{C}_{\hat{\sigma}(j)}\right|^2} \sum\limits_{k,\hat{\sigma}(k)=\hat{\sigma}(j)} \sum\limits_{l,\hat{\sigma}(l)=\hat{\sigma}(j)} \left\|x(k) - x(l)\right\|^2} \tag{2.47}$$

$$\beta_{\hat{\sigma}(j)} = \frac{1}{\frac{1}{\left|\mathscr{C}_{\hat{\sigma}(j)}\right|} \sum\limits_{k,\hat{\sigma}(k)=\hat{\sigma}(j)} \left(y(k) - f_{\hat{\sigma}(j)}(\bar{\varphi}(k))\right)^2} \tag{2.48}$$

Les algorithmes 2.1 ou 2.2 proposés précédemment peuvent être modifiés pour traiter le cas des modèles dynamiques non linéaires par morceaux. Les principales modifications concernent les points suivants. La procédure d'initialisation, dans le cas des PWN, estime pour chaque donnée $x(i)$ une fonction non linéaire f_i en se basant sur l'ensemble formé par $x(i)$ et ses c plus proches voisins dans l'espace de régression. Cette estimation sera assurée par l'algorithme de régression par les LS-SVM [80]. Les procédures de réaffectation de données proposées dans la section précédente peuvent s'appliquer directement au partitionnement de données de régression en des classes disjointes où chaque classe caractérise un sous-modèle non linéaire. Seule la mesure de similarité sera remplacée par celle donnée dans (2.46). Le critère d'arrêt se base, dans ce cas, sur la comparaison entre les paramètres antérieurs $w_i^{(r)}$, $\rho_i^{(r)}$ et les paramètres postérieurs $w_i^{(r+1)}$, $\rho_i^{(r+1)}$ de toutes les fonctions f_i, $i = 1, \ldots, \bar{s}$.

Nous présentons dans le paragraphe suivant la méthode de régression par les LS-SVM [80] employée pour l'identification des paramètres des fonctions f_i.

2.4.2.1 Régression par les LS-SVM

La méthode de régression par les LS-SVM a été proposée par Suykens et Vanderwalle [80] pour la classification et la régression non linéaire. La solution est obtenue en résolvant un système d'équations linéaires dont le nombre de lignes est égal à $N_i + 1$, N_i étant le

nombre de données.
Soit un ensemble de N_i données $\{\varphi(k), y(k)\}_{k=1}^{N_i}$. L'objectif est de trouver une fonction f_i régulière telle que :

$$y(k) = f_i(\varphi(k)) + e_i(k).$$

où f_i est définie par : $f_i(\varphi) = w_i^\top \eta(\varphi) + \rho_i$.

L'estimation des paramètres w_i et ρ_i de la fonction f_i s'effectue en résolvant le problème de minimisation quadratique sous contraintes suivant [80] :

$$\begin{array}{l} \min\limits_{w_i,\rho_i,e_i} \frac{1}{2} w_i^\top w_i + \frac{\gamma}{2} \sum\limits_{k=1}^{N_i} e_i(k)^2 \\ s.c.\ \ y(k) = w_i^\top \eta(\varphi(k)) + \rho_i + e_i(k),\ k = 1, \ldots, N_i \end{array} \tag{2.49}$$

La solution de ce problème d'optimisation correspond au point selle du Lagrangien suivant :

$$L(w_i, \rho_i, e_i; \alpha_i) = \frac{1}{2} w_i^\top w_i + \frac{\gamma}{2} \sum_{k=1}^{N_i} e_i^2(k) - \sum_{k=1}^{N_i} \alpha_i(k)(w_i^\top \eta(\varphi(k)) + \rho_i + e_i(k) - y(k)) \tag{2.50}$$

où $\alpha_i(k)$ sont les multiplicateurs de Lagrange associés aux contraintes du problème original. Le Lagrangien L doit être minimisé par rapport à w_i, ρ_i et $e_i(k)$ et maximisé par rapport à α_i.
Les conditions d'optimalité du Lagrangien (ou conditions nécessaires à l'obtention d'un minimum) sont données par :

$$\frac{\partial L}{\partial w_i} = 0 \quad \rightarrow \quad w = \sum_{k=1}^{N_i} \alpha_i(k) \eta(\varphi(k)) \tag{2.51}$$

$$\frac{\partial L}{\partial \rho_i} = 0 \quad \rightarrow \quad \sum_{k=1}^{N_i} \alpha_i(k) = 0 \tag{2.52}$$

$$\frac{\partial L}{\partial e_i(k)} = 0 \quad \rightarrow \quad \alpha_k = \gamma e_i(k),\ k = 1, \ldots, N_i \tag{2.53}$$

$$\frac{\partial L}{\partial \alpha_i(k)} = 0 \quad \rightarrow \quad y(k) = w_i^\top \eta(\varphi(k)) + \rho_i + e_i(k),\ k = 1, \ldots, N_i \tag{2.54}$$

En remplaçant les expressions (2.51) et (2.52) dans (2.54) et en tenant compte de (2.53), on obtient alors le système d'équations linéaires suivant :

$$\begin{bmatrix} 0 & 1_{N_i}^\top \\ 1_{N_i} & \Omega + \gamma^{-1} I_{N_i} \end{bmatrix} \begin{bmatrix} \rho_i \\ \alpha_i \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix} \tag{2.55}$$

où $1_{N_i} = [1, \ldots, 1]^\top \in \mathbb{R}^{N_i}$ et $y = [y(1), \ldots, y(N_i)]^\top$. I_{N_i} est la matrice identité de dimension $N_i \times N_i$ et $\Omega \in \mathbb{R}^{N_i \times N_i}$ est une matrice définie par $\Omega_{ij} = \eta(\varphi(i))^\top \eta(\varphi(j))$. Ω_{ij} peut être calculé par $\Omega_{ij} = \kappa(\varphi(i), \varphi(j))$ [81] où $\kappa : \mathbb{R}^{N_i} \times \mathbb{R}^{N_i} \to \mathbb{R}^+$ est une fonction définie positive appelée fonction noyau [80].

En remplaçant l'expression de w_i dans f_i, cette fonction peut s'écrire comme suit :

$$f_i(\varphi) = \sum_{k=1}^{N_i} \alpha_i(k) \kappa\left(\varphi(k), \varphi\right) + \rho_i \tag{2.56}$$

où $\alpha_i(k)$, $k = 1, \ldots, N_i$ et ρ_i sont obtenus à partir de l'équation (2.55).

2.4.3 Exemple numérique

Nous considérons maintenant un modèle SISO composé de deux sous-modèles ($s = 2$) d'ordre un ($n_a = n_b = 1$) défini par les équations suivantes :

$$y(k) = \begin{cases} 0.6y(k-1)^2 + 0.2u(k-1) & si \quad \varphi(k) \in \Re_1 \\ y(k-1) - 0.5u(k-1) & si \quad \varphi(k) \in \Re_2 \end{cases} \tag{2.57}$$

où $\varphi(k) = [y(k-1)\ u(k-1)]^\top$
et

$$\Re_1 = \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} -0.5 & -1 & 0 \end{bmatrix} \bar{\varphi} \geq 0 \right\}$$
$$\Re_2 = \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} -0.5 & -1 & 0 \end{bmatrix} \bar{\varphi} < 0 \right\}$$

L'entrée d'excitation $u(k)$ est générée selon une loi normale de moyenne 0 et de variance 0.4 (voir Figure 2.10). Un ensemble de $N = 100$ données a été généré suivant le modèle (2.57).

Pour montrer l'intérêt de la modélisation par les modèles dynamiques non linéaires par morceaux (PWNARX), nous comparons la sortie reconstruite par ces modèles à celles reconstruites par des modèles dynamiques affines par morceaux (PWARX) et par un seul modèle non linéaire (NARX) pour le même jeu de données.

Nous supposons que le type de non linéarité des sous-modèles est inconnu. Sans cette connaissance *a priori*, nous décidons d'utiliser un noyau gaussien (très connu pour sa capacité de modélisation) dans le cas de la modélisation par les modèles dynamiques non linéaires par morceaux (PWNARX) et dans la modélisation par un seul modèle non linéaire (NARX) :

$$\kappa(\varphi(i), \varphi(j)) = \exp\left(-\left\|\varphi(i) - \varphi(j)\right\|^2 / \sigma^2\right)$$

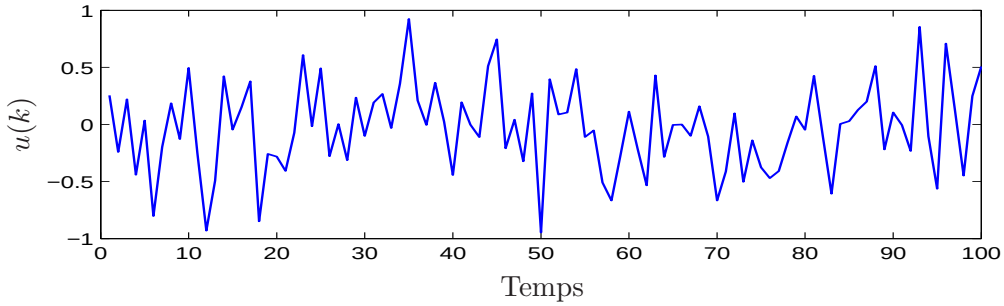


FIGURE 2.10 – L'entrée d'excitation $u(k)$ du modèle PWNARX.

avec $\sigma = 1$.

	PWNARX ($s = 2$)	PWARX ($s = 6$)	PWARX ($s = 2$)	NARX ($s = 1$)
SSE	$3.68 \cdot 10^{-5}$	$2.45 \cdot 10^{-5}$	$3 \cdot 10^{-3}$	$31 \cdot 10^{-3}$

TABLE 2.5 – Comparaison des moyennes des erreurs quadratiques SSE des différentes méthodes (PWNARX, PWARX, NARX).

La sortie réelle ainsi que les sorties reconstruites dans les différents cas sont présentées sur la figure 2.11. Nous présentons dans le tableau 2.5 les moyennes des erreurs quadratiques SSE des différents cas. A partir du tableau 2.5 et de la figure 2.11, nous constatons que la modélisation par un seul modèle non linéaire donne un SSE considérable comparé à celui obtenu par les PWN pour la même valeur de σ. Nous constatons également que pour obtenir un SSE équivalent à celui obtenu par la modélisation par les modèles dynamiques non linéaires par morceaux avec seulement deux sous-modèles non linéaires il a fallu identifier six sous-modèles affines par la modélisation par les modèles dynamiques affines par morceaux.

2.5 Conclusion

Ce chapitre a été consacré au problème de la classification de données de régression en des classes disjointes et à celui de l'estimation des vecteurs de paramètres caractérisant chaque sous-modèle. Ce problème a été traité d'abord dans le cas des modèles dynamiques affines par morceaux puis il a été étendu au cas des modèles dynamiques non linéaires par morceaux. L'approche proposée repose sur une technique de classification non supervisée combinée avec une technique de régression linéaire pour grouper les données de

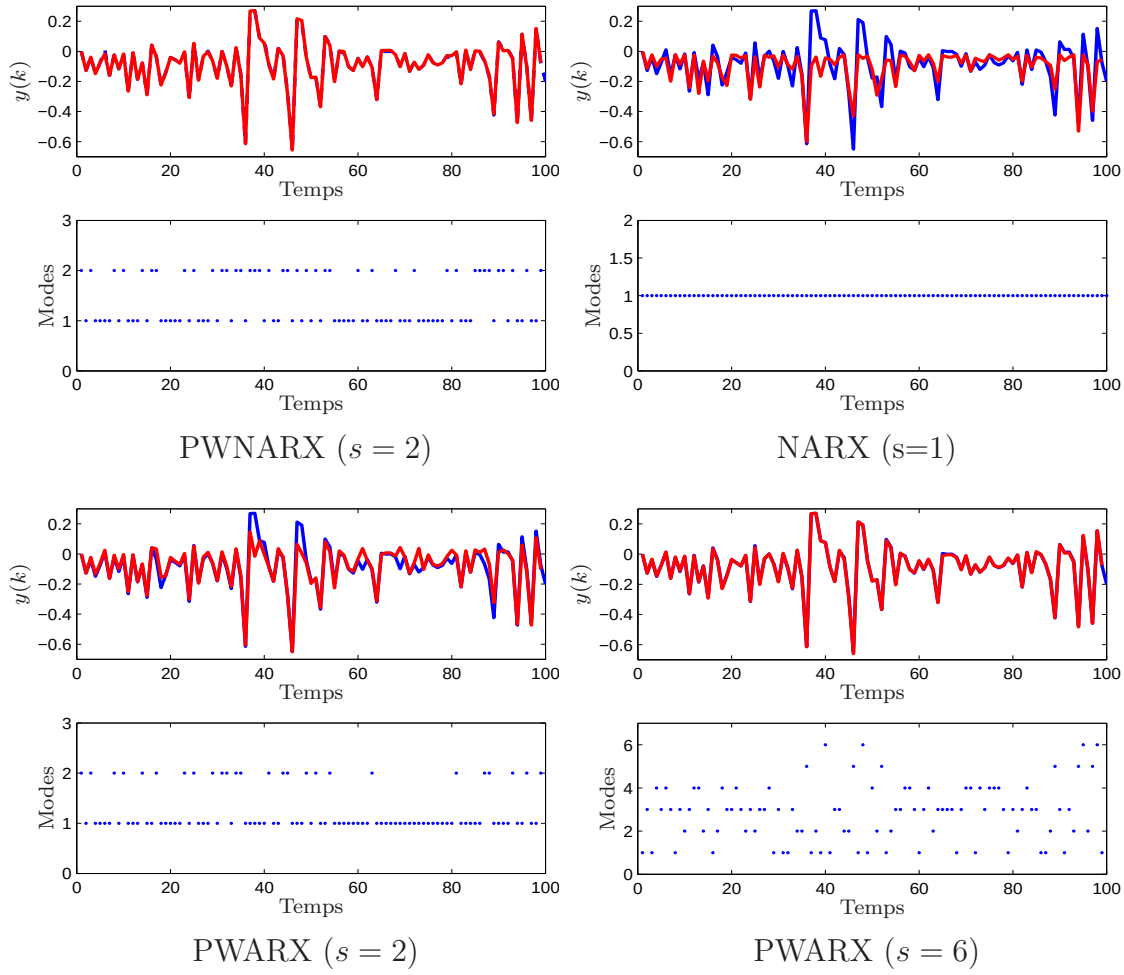


FIGURE 2.11 – Sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié ; Séquence des modes estimés par différentes méthodes de modélisation.

régression relatives au même sous-modèle affine. Elle consiste à alterner l'assignation des données de régression aux différents sous-modèles et l'estimation des vecteurs de paramètres correspondants. La réaffectation de données a été modifiée par l'utilisation d'une classification de données basée sur une modélisation de connaissances incertaines par la théorie de Dempster-Shafer. L'avantage majeur de la procédure de classification de données et d'estimation de paramètres présentée dans ce chapitre est qu'elle ne nécessite qu'un seul paramètre de réglage qui est le nombre des plus proches voisins c. Ensuite, nous avons vu que l'utilisation des modèles dynamiques non linéaires par morceaux diminue considérablement le nombre de sous-modèles tout en gardant une bonne erreur d'identification.

Nous traitons, dans le chapitre suivant, l'estimation des régions caractérisant le mécanisme qui gère les commutations d'un sous-modèle à un autre. Le problème consiste à

trouver une partition complète $\{\Re_i\}_{i=1}^{s}$ de l'espace de régression $\Re$. Ce problème revient à trouver les frontières qui délimitent chaque région dans la partition complète et donc à séparer les s régions par des séparateurs linéaires (hyperplans).

3

Détermination de la loi de commutation - Estimation des régions

Sommaire

3.1 Introduction

Ce chapitre traite l'estimation de la règle gérant les commutations d'un sous-modèle à un autre. Dans le cas des modèles dynamiques affines par morceaux, cette règle est définie par un partitionnement polyédrique du domaine de régression. Chaque sous-modèle est alors valide dans une région bien définie. L'estimation de ces régions constitue l'étape finale de la procédure d'identification des modèles PWARX (voir figure 2.1) après l'étape de classification des données et d'identification des vecteurs de paramètres (cf. Chapitre 2). Elle consiste à trouver une partition complète $\{\Re_i\}_{i=1}^{s}$ de l'espace de régression $\Re$. Ce problème revient à trouver les frontières qui délimitent chaque région dans la partition complète et donc à séparer les s régions par des séparateurs linéaires (hyperplans).

Les séparateurs les plus utilisés en pratique sont les Machines à Vecteurs de Support ou encore Séparateurs à Vaste Marge (SVM) [81] qui sont connus pour leur capacité de généralisation. Le principe est de trouver l'hyperplan qui minimise la somme des erreurs associées aux mauvaises classifications tout en maximisant la marge de séparation entre les régions.

Le problème de la séparation linéaire de s régions $\Re_1, \ldots, \Re_s$ peut être abordé en utilisant deux catégories différentes de classification :

1. La première catégorie combine des séries de classifieurs linéaires binaires en faisant appel à des schémas de décomposition du type "un-contre-reste" ou "un-contre-un".
2. La deuxième catégorie consiste à résoudre le problème multi-classe en une seule étape sans le décomposer en une collection de sous-problèmes binaires.

La décomposition "un-contre-reste" consiste à construire s classifieurs binaires à vecteurs de support où chaque classifieur sépare une région $\Re_i$ du reste des régions tandis que la décomposition "un-contre-un" consiste à construire $s(s-1)/2$ classifieurs binaires, chacun séparant uniquement deux régions tout en ignorant les autres régions. L'approche multi-classe permet de séparer les régions en considérant simultanément l'ensemble de toutes les données et en recherchant directement un séparateur linéaire par morceaux. Cette approche nécessite la résolution d'un seul problème d'optimisation quadratique sous contraintes linéaires.

Après avoir formulé le problème de l'identification de la loi de commutation dans la Section 3.2, nous traitons dans la Section 3.3, l'estimation de séparateurs linéaires dans le cas de $s > 2$ régions. Nous présentons les différents schémas de décomposition (approche un-contre-reste et approche un-contre-un). Ensuite, nous détaillons le séparateur linéaire par morceaux. Enfin, nous montrons comment les techniques de classification linéaire

peuvent être étendues pour traiter le cas où la loi de commutation est définie non pas par un partitionnement polyédrique mais par un partitionnement où les frontières séparatrices sont non linéaires.

3.2 Formulation du problème

Dans les modèles dynamiques affines par morceaux (PWARX), la loi de commutation est déterminée par une partition polyédrique de l'espace de régression $\Re$. Cela signifie que, pour ces modèles, l'état discret $\sigma(k)$ est donné par :

$$\sigma\left(k\right) = i \quad \text{ssi} \quad \varphi(k) \in \Re_i, \quad i = 1, \ldots, s \tag{3.1}$$

avec $\varphi(k) = \left[y(k-1), \cdots, y(k-n_a), u(k)^\top, \cdots, u(k-n_b)^\top\right]^\top$.
$\{\Re_i\}_{i=1}^s$ est une partition complète de l'ensemble de régression $\Re$, c'est-à-dire

$$\begin{cases} \cup_{i=1}^s \Re_i = \Re \\ \Re_i \cap \Re_j = \emptyset, \ \forall i \neq j. \end{cases} \tag{3.2}$$

Chaque région $\Re_i$ est un polyèdre convexe décrit par l'inégalité suivante :

$$\Re_i = \left\{\bar{\varphi} \in \mathbb{R}^{n+1} : H_i \bar{\varphi} \leq \mathbf{0}\right\}, \tag{3.3}$$

avec $\bar{\varphi} = \left[\varphi^\top \ 1\right]^\top$, $H_i \in \mathbb{R}^{m_i \times (n+1)}$ où m_i désigne le nombre d'hyperplans contribuant à la définition de la région $\Re_i$ et $\mathbf{0}$ est le vecteur nul.

L'estimation des régions du modèle affine par morceaux (PWARX) consiste à trouver une partition complète $\{\Re_i\}_{i=1}^s$ de l'espace de régression $\Re$ de telle sorte que, si $x(k) \in \mathscr{C}_i$, alors $\varphi(k) \in \Re_i$. On rappelle que le regroupement des données $x(k) = \left[\varphi(k)^\top, y(k)\right]^\top$ en des classes $\mathscr{C}_i$, a été effectué dans le chapitre précédent par la procédure de classification de données et d'estimation de paramètres. On cherche, ici, à retrouver les paramètres $\{H_i\}_{i=1}^s$ des polyèdres convexes définissant les régions de validité de chaque sous-modèle.

Dans le cas où la partition $\{\Re_i\}_{i=1}^s$ est délimitée par des fonctions non linéaires, chaque région $\Re_i$ est définie par :

$$\Re_i = \{\varphi \in \mathbb{R}^n : h_i(\varphi) \leq 0\} \tag{3.4}$$

où h_i est la fonction non linéaire délimitant la région $\Re_i$.
L'objectif revient, dans ce cas, à retrouver les paramètres des fonctions $\{h_i\}_{i=1}^s$.

Dans la section suivante nous traitons l'estimation de séparateurs linéaires pour plus de deux régions ($s > 2$). Nous présentons d'abord les différents schémas de décomposition binaire, ensuite nous présentons l'approche multi-classe par les séparateurs linéaires par morceaux.

3.3 Classification multi-classe pour l'estimation des régions

La technique SVM binaire, présentée dans l'Annexe 1, a été étendue au problème de la classification multi-classe [42]. Cette généralisation peut être divisée en deux catégories de méthodes différentes. La première catégorie fait appel à des schémas de décomposition du type "un-contre-reste" ou "un-contre-un". La règle de décision multi-classe est ensuite dérivée en combinant toutes les règles de décision binaire issues du schéma de décomposition adopté. La deuxième catégorie de méthodes consiste à résoudre le problème multi-classe en une seule étape. Cette méthode revient à résoudre un unique problème d'optimisation quadratique conformément à ce qui est fait lorsqu'il s'agit de deux régions. Ces deux catégories sont détaillées dans les parties qui suivent.

3.3.1 Approche "un-contre-reste"

L'approche un-contre-reste (en anglais one-against-all) est une généralisation du cas binaire pour traiter le cas de la classification multi-classe. Cette approche est la plus simple et probablement la plus ancienne, utilisée pour les machines à vecteurs de support multi-classes. Elle consiste à construire s classifieurs binaires à vecteurs de support où s est le nombre total de régions. La construction du *ième* classifieur s'effectue en considérant toutes les données de la *ième* région comme ayant une étiquette positive $z = +1$ et toutes les autres données comme ayant une étiquette négative $z = -1$. Le *ième* classifieur s'obtient en résolvant le problème d'optimisation suivant (voir Annexe 1)

$$\begin{array}{ll} \min\limits_{w_i,b_i,\xi_j^i} & \frac{1}{2}\left\|w_i\right\|^2 + C\sum\limits_{j=1}^{N}\xi_j^i \\ sc. & w_i^\top\varphi(j) + b_i \geq 1 - \xi_j^i, \quad \forall\varphi(j) \in \Re_i \\ & w_i^\top\varphi(j) + b_i < -1 + \xi_j^i, \quad \forall\varphi(j) \in \Re \setminus \{\Re_i\} \\ & \xi_j^i \geq 0,\ j = 1,\ldots,N \end{array} \tag{3.5}$$

En se basant sur l'Annexe 1, le problème d'optimisation (3.5) est équivalent au problème

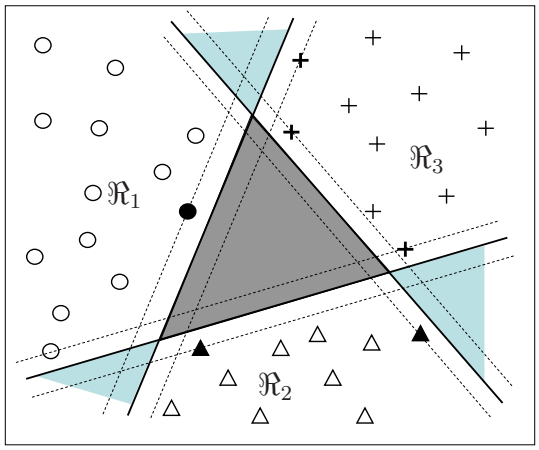


FIGURE 3.1 – Séparateurs "un-contre-reste" de trois régions.

dual suivant

$$\begin{aligned} \max_{\alpha_l^i} & \sum_{\varphi(l)\in\Re} \alpha_l^i - \tfrac{1}{2} \sum_{\varphi(l)\in\Re} \sum_{\varphi(k)\in\Re} \alpha_l^i \alpha_k^i z_l^i z_k^i \varphi(l)\varphi(k) \\ sc. \quad & \sum_{\varphi(l)\in\Re} \alpha_l^i z_l^i = 0 \\ & 0 \le \alpha_l^i \le C, \quad \forall \varphi(l) \in \Re \end{aligned} \tag{3.6}$$

où $z_l^i = 1$ si $\varphi(l) \in \Re_i$ et $z_l^i = -1$ si $\varphi(l) \in \Re \setminus \{\Re_i\}$.

La résolution du problème (3.6) pour chaque valeur de $i \in \{1, \ldots, s\}$ donne lieu à s fonctions de décision :

$$h_i(\varphi) = w_i^\top \varphi + b_i, \quad i = 1, \ldots, s. \tag{3.7}$$

où $w_i = \sum_{\varphi(k)\in\Re} \alpha_l^i z_l^i \varphi(l)$.

On obtient ainsi s hyperplans $H_i,\ i = 1, \ldots, s$, chaque hyperplan étant obtenu en séparant la région $\Re_i$ de l'union des $s-1$ régions restantes, et il est défini par :

$$H_i = \left[-w_i^\top \ -b_i \right]. \tag{3.8}$$

La donnée $\varphi(l)$ appartient à la région $\Re_i$ si $h_i(\varphi(l)) \ge 0$. Cette égalité peut être satisfaite pour plus d'une région. Dans ce cas, l'observation $\varphi(l)$ est dite non-classable. Ainsi, les régions estimées par cette classification peuvent ne pas constituer une partition complète de l'espace de régression $\Re$ (équation (3.2) non vérifiée) car l'intersection des régions peut ne pas être vide (régions de couleur turquoise schématisées dans la figure 3.1) et l'union de toutes les régions ne forme pas tout l'espace de régression. La région grisée au centre de la figure 3.1 n'est incluse dans aucune des régions $\Re_i$.

Pour éviter l'existence de la région grisée, on applique le principe "le gagnant emporte le tout"[2]. L'étiquette retenue est celle associée au classifieur ayant renvoyé la valeur la plus élevée. Ainsi, une donnée φ sera affectée à la région $\Re_i$ selon la règle de décision suivante

$$i = \arg\max_{1\leq j\leq s} h_j(\varphi). \tag{3.9}$$

L'espace de régression $\Re$ sera subdivisé en s régions convexes. Cependant, l'inconvénient majeur de cette heuristique est qu'elle ne conserve pas les s frontières de séparation.

3.3.2 Approche "un-contre-un"

Cette approche procède en recherchant un hyperplan séparateur pour chaque paire de régions $(\Re_i, \Re_j)$. Cette méthode de décomposition attribuée à Knerr et ses co-auteurs [50] a été utilisée pour la première fois dans le contexte des machines à vecteurs de support par Kreβel [52]. Elle consiste à construire $s(s-1)/2$ classifieurs binaires, chacun séparant uniquement deux régions tout en ignorant les autres régions.

L'hyperplan séparateur des deux régions ($\Re_i$ et $\Re_j$) a pour équation

$$w_{ij}^\top \varphi + b_{ij} = 0, \tag{3.10}$$

où w_{ij} et b_{ij} sont les solutions du problème d'optimisation suivant :

$$\begin{aligned} &\min_{w_{ij},b_{ij},\xi^{ij}} \tfrac{1}{2}\|w_{ij}\|^2 + C\sum_{k=1}^{N_{ij}} \xi_k^{ij} \\ &s.c. \quad w_{ij}^\top \varphi(k) + b_{ij} \geq 1 - \xi_k^{ij}, \quad \forall \varphi(k) \in \Re_i \\ &\qquad w_{ij}^\top \varphi(k) + b_{ij} < -1 + \xi_k^{ij}, \quad \forall \varphi(k) \in \Re_j \\ &\qquad \xi_k^{ij} \geq 0,\ k = 1, \ldots, N_{ij} \end{aligned} \tag{3.11}$$

où N_{ij} est le nombre des observations issues des régions $\Re_i$ et $\Re_j$.
Le problème dual équivalent au problème d'optimisation (3.11) est donné par

$$\begin{aligned} &\max_{\alpha_l^{ij}} \sum_{\varphi(l)\in\Re_{ij}} \alpha_l^{ij} - \tfrac{1}{2}\sum_{\varphi(l)\in\Re_{ij}}\sum_{\varphi(k)\in\Re_{ij}} \alpha_l^{ij}\alpha_k^{ij} z_l^{ij} z_k^{ij} \varphi(l)\varphi(k) \\ &sc. \quad \sum_{\varphi(l)\in\Re_{ij}} \alpha_l^{ij} z_l^{ij} = 0 \\ &\qquad 0 \leq \alpha_l^{ij} \leq C, \quad \forall \varphi(l) \in \Re_{ij} \end{aligned} \tag{3.12}$$

où $z_l^{ij} = 1$ si $\varphi(l) \in \Re_i$ et $z_l^{ij} = -1$ si $\varphi(l) \in \Re_j$ et $\Re_{ij} = \Re_i \cup \Re_j$.
La résolution du problème (3.12) pour chaque valeur de $i \in \{1, \ldots, s\}$ et $j \in \{i+1, \ldots, s\}$ donne lieu à $s(s-1)/2$ fonctions de séparation linéaire :

$$h_{ij}(\varphi) = w_{ij}^\top \varphi + b_{ij}, \quad i = 1, \ldots, s,\ j = i+1, \ldots, s. \tag{3.13}$$

2. En anglais : winner-takes-all

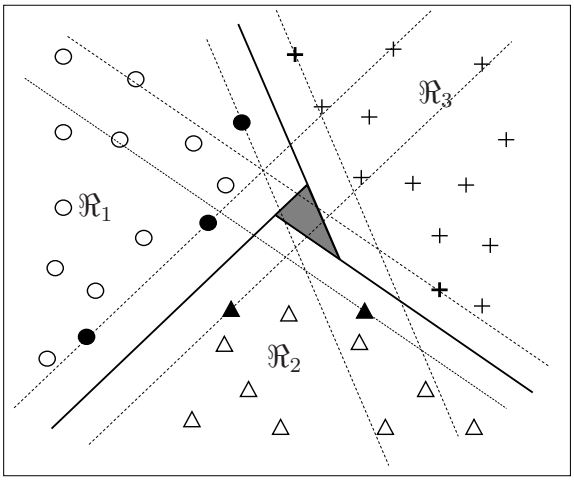


FIGURE 3.2 – Séparateurs "un-contre-un" de trois régions (l'espace hachuré représente la région d'ambiguïté).

L'ensemble des hyperplans qui séparent la région $\Re_i$ du reste des régions est défini par :

$$H_i = \begin{bmatrix} -w_{i1} & \cdots & -w_{ij,j\neq i} & \cdots & -w_{is} \\ -b_{i1} & \cdots & -b_{ij,j\neq i} & \cdots & -b_{is} \end{bmatrix}^\top . \tag{3.14}$$

Pour décider sur l'affectation d'une donnée, Friedman [36] définit une nouvelle fonction de décision de la façon suivante :

$$h_i(\varphi) = \sum_{j=1}^{s} \text{signe}\,(h_{ij}(\varphi)), \quad i = 1, \ldots, s \tag{3.15}$$

et introduit la règle de classification d'une donnée φ à la classe k comme

$$k = \arg\max_{1\leq i\leq s} h_i(\varphi). \tag{3.16}$$

Cette règle est connue sous le nom de "vote majoritaire"[3]. Comme dans le cas un-contre-reste, la règle (3.16) peut être satisfaite pour plusieurs classes. Également, Les régions estimées par cette méthode peuvent ne pas constituer une partition complète de l'espace de régression $\Re$ (voir section grisée schématisée dans la figure 3.2). Néanmoins, cette approche peut être la plus adéquate dans le cas où les régions sont adjacentes deux à deux (Figure 3.3). Un test d'adjacence de région permettra de détecter ce cas. Dans ce cas, il n'est pas nécessaire de résoudre $s(s-1)/2$ problèmes de séparation linéaire mais seulement $s-1$ problèmes de séparation linéaire.

3. En anglais : max-wins voting

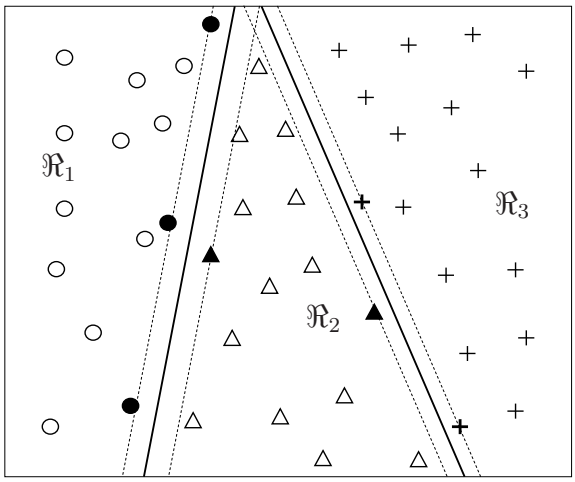


FIGURE 3.3 – Séparateurs "un-contre-un" de trois régions adjacentes deux à deux.

Détection de l'adjacence des régions : Il est nécessaire de connaître la façon dont les régions sont disposées dans l'ensemble de régression avant toute opération de décomposition en série de classifieurs binaires. Un classifieur binaire est nécessaire pour séparer deux régions si ces deux régions sont adjacentes. Par conséquent, afin de connaître le nombre de classifieurs linéaires nécessaires pour effectuer une classification multi-classe, les régions $\Re_i,\ i = 1, \ldots, s$ sont soumises à un test permettant de détecter une éventuelle adjacence, deux à deux, entre elles. Deux régions $\Re_i$ et $\Re_j$ sont dites adjacentes si le critère suivant est satisfait :

$$\text{card}\left\{\varphi(k) \in \Re_i / \ \Gamma_c(\varphi(k)) \cap \Re_j \neq \emptyset\right\} \geq N_{adj}. \tag{3.17}$$

On rappelle que $\Gamma_c(\varphi(k))$ est l'ensemble des c plus proches voisins de $\varphi(k)$.
Cela veut dire que les deux régions $\Re_i$ et $\Re_j$ sont adjacentes si le nombre de données appartenant à la région $\Re_i$ dont l'ensemble des c plus proches voisins peut contenir des données de la région $\Re_j$ est supérieur à un seuil N_{adj}. Le seuil de cardinalité N_{adj} sera choisi par l'utilisateur en fonction de la taille des régions.

3.3.3 Approche tous-contre-tous : séparateur linéaire par morceaux

L'approche tous-contre-tous [4] consiste à construire un séparateur linéaire par morceaux pour séparer les $s > 2$ régions, en considérant simultanément l'ensemble de toutes les

4. En anglais : all-together

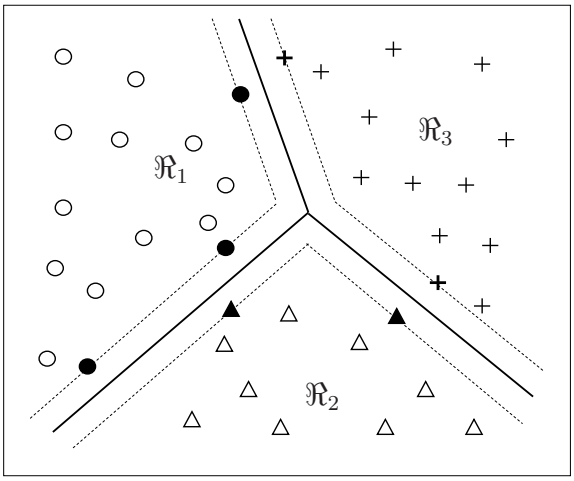


FIGURE 3.4 – Séparateur linéaire par morceaux de trois régions.

données [22][16][86][58]. Dans le cas de données linéairement séparables, le problème de la construction d'un classifieur linéaire par morceaux consiste à trouver $(w_1, b_1), \ldots, (w_s, b_s)$, tels que

$$w_i^\top \varphi + b_i > w_j^\top \varphi + b_j,\ \forall \varphi \in \Re_i, \quad \forall j \in \{1, \ldots, s\}, \quad j \neq i. \tag{3.18}$$

Par la suite, la fonction de décision de ce classifieur est définie par

$$h(\varphi) = \max_{i=1,\ldots,s} (h_i(\varphi)) \tag{3.19}$$

où chaque fonction h_i est définie par

$$h_i(\varphi) = w_i^\top \varphi + b_i. \tag{3.20}$$

Pour ce problème de séparation linéaire par morceaux, il existe un nombre infini d'hyperplans (w_i, b_i) satisfaisant l'équation (3.18). Intuitivement, les hyperplans optimaux (w_i, b_i), sont ceux qui maximisent la marge de séparation. Ainsi, par une approche analogue à celle de la classification binaire par les machines à vecteurs de support (SVM), nous allons ajouter des termes de régularisation.

Les distances entre les lignes en pointillés dans la figure 3.4 représentent les marges de chaque séparateur $(w_i - w_j, b_i - b_j)$ de la fonction linéaire de séparation par morceaux. La marge de séparation entre deux régions $\Re_i$ et $\Re_j$, c'est-à-dire la distance entre les hyperplans

$$\begin{aligned} (w_i - w_j)^\top \varphi + (b_i - b_j) &= 1 \quad \text{et} \\ (w_i - w_j)^\top \varphi + (b_i - b_j) &= -1 \end{aligned}$$

est égale à $2/\left\|w_i - w_j\right\|_2$. La maximisation de cette distance revient donc à minimiser la quantité $\left\|w_i - w_j\right\|_2^2$, pour tous $i = 1, \ldots, s; j = i + 1, \ldots, s$. Bredensteiner et Bennett

[22] ont ajouté un terme de régularisation supplémentaire $\sum_{i=1}^{s} \|w_i\|_2^2$ et ont proposé de résoudre le problème d'optimisation sous contraintes suivant :

$$\begin{array}{l} \min\limits_{w_i,b_i} \frac{1}{2} \sum\limits_{i=1}^{s} \sum\limits_{j=i+1}^{s} \|w_i - w_j\|_2^2 + \frac{1}{2} \sum\limits_{i=1}^{s} \|w_i\|_2^2 \\ s.c. \quad (w_i - w_j)^\top \varphi(l) + (b_i - b_j) - 1 \geq 0, \forall \varphi(l) \in \Re_i \\ \qquad (w_i - w_j)^\top \varphi(l) + (b_i - b_j) + 1 < 0, \forall \varphi(l) \in \Re_j \\ i = 1, \ldots, s; j = i+1, \ldots, s. \end{array} \tag{3.21}$$

Le terme de régularisation $\sum_{i=1}^{s} \|w_i\|_2^2$ permet d'avoir un vecteur w_i de norme minimale. Dans le cas général où les données peuvent ne pas être linéairement séparables par morceaux (suite à la présence du bruit par exemple), on doit trouver la règle de décision qui minimise les erreurs de mauvaise classification. A cet effet, on peut minimiser aussi la fonction

$$\sum_{i=1}^{s} \sum_{j=i+1}^{s} \sum_{x_l \in \Re_{ij}} \xi_l^{ij} \tag{3.22}$$

où ξ_l^{ij} sont des variables ressorts et $\Re_{ij} = \Re_i \cup \Re_j$.
En relâchant les contraintes du problème d'optimisation (3.21), nous proposons de résoudre le problème d'optimisation suivant [16] :

$$\begin{array}{l} \min\limits_{w_i,b_i} \frac{1}{2} \sum\limits_{i=1}^{s} \sum\limits_{j=i+1}^{s} \|w_i - w_j\|_2^2 + \frac{1}{2} \sum\limits_{i=1}^{s} \|w_i\|_2^2 + C \sum\limits_{i=1}^{s} \sum\limits_{j=i+1}^{s} \sum\limits_{\varphi(l) \in \Re_{ij}} \xi_l^{ij} \\ s.c. \quad (w_i - w_j)^\top \varphi(l) + (b_i - b_j) - 1 + \xi_l^{ij} \geq 0, \forall \varphi(l) \in \Re_i \\ \qquad (w_i - w_j)^\top \varphi(l) + (b_i - b_j) + 1 - \xi_l^{ij} < 0, \forall \varphi(l) \in \Re_j \\ \qquad \xi_l^{ij} \geq 0 \\ i = 1, \ldots, s; j = i+1, \ldots, s; \end{array} \tag{3.23}$$

où C est un paramètre de régularisation.
Cette écriture du problème est appelée formulation primale. Nous sommes ainsi face à un problème d'optimisation quadratique convexe sous des contraintes de type inégalités linéaires. Les méthodes classiques de programmation mathématique peuvent donc être utilisées [21].

D'un point de vue théorique, une formulation primale possède une forme duale. Dans le cas où la fonction objectif et les contraintes sont strictement convexes, ce qui est bien le cas pour le problème (3.23), la résolution de l'expression duale du problème est équivalente à la solution de la formulation primale.

Pour résoudre ce type de problème, on utilise une fonction que l'on appelle Lagrangien qui incorpore des informations sur la fonction objectif et sur les contraintes. Plus précisément, le lagrangien est défini comme étant la somme de la fonction objectif et d'une

combinaison linéaire des contraintes dont les coefficients sont appelés *multiplicateurs de Lagrange* α^{ij}.
Dans le cas traité ici, le Lagrangien $\mathbb{L}$ est exprimé par

$$\begin{aligned}\mathbb{L} = & \tfrac{1}{2}\sum_{i=1}^{s}\sum_{j=i+1}^{s}\|w_i - w_j\|_2^2 + \tfrac{1}{2}\sum_{i=1}^{s}\|w_i\|_2^2 + C\sum_{i=1}^{s}\sum_{j=i+1}^{s}\sum_{\varphi(l)\in\Re_{ij}}\xi_l^{ij} \\ & - \sum_{i=1}^{s}\sum_{j=i+1}^{s}\sum_{\varphi(l)\in\Re_{ij}}\alpha_l^{ij} z_l^{ij}\left((w_i - w_j)\,\varphi(l) + (b_i - b_j) - 1 + \xi_l^{ij}\right) \\ & - \sum_{i=1}^{s}\sum_{j=i+1}^{s}\sum_{\varphi(l)\in\Re_{ij}}\mu_l^{ij}\xi_l^{ij}\end{aligned} \tag{3.24}$$

où α_l^{ij} et μ_l^{ij} sont des coefficients de Lagrange, $z_l^{ij} = 1$ si $\varphi(l) \in \Re_i$ et $z_l^{ij} = -1$ si $\varphi(l) \in \Re_j$.

Le Lagrangien $\mathbb{L}$ doit être minimisé par rapport à w_i, b_i et ξ_l^{ij} et maximisé par rapport à α_l^{ij} et μ_l^{ij}. Le théorème de Kuhn-Tucker [53] démontre que le problème primal et sa formulation duale ont la même solution qui correspond à un point-selle du Lagrangien. Au point-selle, la dérivée du Lagrangien $\mathbb{L}$ par rapport aux variables primaires s'annule :

$$\frac{\partial\mathbb{L}}{\partial w_i} = sw_i - \sum_{\substack{j=1\\ j\neq i}}^{s} w_j - \sum_{\substack{j=1\\ j\neq i}}^{s}\Big(\sum_{\varphi(l)\in\Re_i}\alpha_l^{ij}\varphi(l) - \sum_{\varphi(l)\in\Re_j}\alpha_l^{ij}\varphi(l)\Big) = 0; \tag{3.25}$$

$$\frac{\partial\mathbb{L}}{b_i} = \sum_{\substack{j=1\\ j\neq i}}^{s}\Big(\sum_{\varphi(l)\in\Re_i}\alpha_l^{ij} - \sum_{\varphi(l)\in\Re_j}\alpha_l^{ij}\Big) = 0; \tag{3.26}$$

$$\frac{\partial\mathbb{L}}{\xi_l^{ij}} = C - \alpha_l^{ij} - \mu_l^{ij} = 0. \tag{3.27}$$

De l'équation (3.27) on a $\alpha_l^{ik} + \mu_l^{ik} = C$ et donc $0 \leq \alpha_l^{ik} \leq C$, $i = 1, \ldots, s$,$k = i+1, \ldots, s$, cela signifie que tous les poids α_l^{ik} sont majorés par la constante de régularisation C.

Le paramètre C est une constante positive fixée à l'avance qui permet de contrôler l'importance de l'erreur que l'on s'autorise par rapport à la taille de la marge. Plus C est important, moins les erreurs sont autorisées. Si la valeur de C tend vers l'infini, la fonction de séparation risque d'être plus complexe. La majoration des coefficients α_l^{ij} par C permettra ainsi d'ajuster à la fois la précision et la complexité des fonctions séparatrices.

En vue de déterminer l'expression de w_i, à partir de l'équation (3.25), nous sommons les

lignes suivantes

$$
\begin{aligned}
sw_1 - \sum_{\substack{j=1\\ j\neq 1}}^{s} w_j &= \sum_{\substack{j=1\\ j\neq 1}}^{s} \Big(\sum_{\varphi(l)\in\Re_1} \alpha_l^{1j}\varphi(l) - \sum_{\varphi(l)\in\Re_j} \alpha_l^{1j}\varphi(l)\Big)\\
sw_2 - \sum_{\substack{j=1\\ j\neq 2}}^{s} w_j &= \sum_{\substack{j=1\\ j\neq 2}}^{s} \Big(\sum_{\varphi(l)\in\Re_2} \alpha_l^{2j}\varphi(l) - \sum_{\varphi(l)\in\Re_j} \alpha_l^{2j}\varphi(l)\Big)\\
&\vdots\\
2(sw_i - \sum_{\substack{j=1\\ j\neq i}}^{s} w_j) &= 2\sum_{\substack{j=1\\ j\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ij}\varphi(l) - \sum_{\varphi(l)\in\Re_j} \alpha_l^{ij}\varphi(l)\Big)\\
&\vdots\\
sw_s - \sum_{\substack{j=1\\ j\neq s}}^{s} w_j &= \sum_{\substack{j=1\\ j\neq s}}^{s} \Big(\sum_{\varphi(l)\in\Re_s} \alpha_l^{sj}\varphi(l) - \sum_{\varphi(l)\in\Re_j} \alpha_l^{sj}\varphi(l)\Big)\\
\hline
(s+1)w_i &= \sum_{\substack{j=1\\ j\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ij}\varphi(l) - \sum_{\varphi(l)\in\Re_j} \alpha_l^{ij}\varphi(l)\Big)
\end{aligned}
$$

Ce résultat est facile à obtenir, sachant que $\alpha_l^{ij} = \alpha_l^{ji}$. Cela a permis d'annuler un grand nombre de termes de la sommation de la partie droite des égalités.
Il résulte de cette sommation, l'expression suivante de w_i

$$w_i = \frac{1}{s+1}\sum_{\substack{j=1\\ j\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ij}\varphi(l) - \sum_{\varphi(l)\in\Re_j} \alpha_l^{ij}\varphi(l)\Big). \tag{3.28}$$

Remarque 3.1. *On peut noter qu'à partir de l'expression de w_i (équation (3.28)), on peut facilement déduire que $\sum_{i=1}^{s} w_i = 0$.*

En remplaçant l'expression (3.28) de w_i dans l'équation (3.20), la fonction définissant chaque séparateur linéaire s'écrira comme suit :

$$h_i(\varphi) = \frac{1}{s+1}\sum_{\substack{j=1\\ j\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ij}(\varphi(l)^{\top}\varphi) - \sum_{\varphi(l)\in\Re_j} \alpha_l^{ij}(\varphi(l)^{\top}\varphi)\Big) + b_i \tag{3.29}$$

Selon le théorème classique de Kuhn-Tucker, aux points selle : $\alpha_l^{ij}((w_i - w_j)^{\top}\varphi(l) + (b_i - b_j) - 1 + \xi_l^{ij}) = 0$ et $\mu_l^{ij}\xi_l^{ij} = 0$; $i = 1,\ldots,s; j = i+1,\ldots,s$. Par conséquent, en remplaçant w_i par son expression dans $\mathbb{L}$ (équation (3.24)), le problème dual correspondant

au problème d'optimisation (3.23) s'écrit

$$\begin{aligned}
\min_{\alpha_l^{ij},b_i} \; & \delta \sum_{i=1}^{s}\sum_{j=i+1}^{s} \left\| \sum_{\varphi(l)\in\Re_i} (2\alpha_l^{ij} + \sum_{\substack{k=1\\k\neq i,j}}^{s} \alpha_l^{ik})\varphi(l) - \sum_{\varphi(l)\in\Re_j} (2\alpha_l^{ij} + \sum_{\substack{k=1\\k\neq i,j}}^{s} \alpha_l^{kj})\varphi(l) \right. \\
& \left. - \sum_{\substack{k=1\\k\neq i,j}}^{s} \sum_{\varphi(l)\in\Re_k} (\alpha_l^{ik} - \alpha_l^{kj})\varphi(l) \right\|_2^2 + \delta \sum_{i=1}^{s} \left\| \sum_{\substack{k=1\\k\neq i}}^{s} \sum_{\varphi(l)\in\Re_i} \alpha_l^{ik}\varphi(l) - \sum_{\varphi(l)\in\Re_k} \alpha_l^{ik}\varphi(l) \right\|_2^2 \\
& - \sum_{i=1}^{s}\sum_{j=i+1}^{s} \sum_{\varphi(l)\in\Re_i} \alpha_l^{ij} \\
s.c. \quad & \sum_{\substack{j=1\\j\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ij} - \sum_{\varphi(l)\in\Re_j} \alpha_l^{ij} \Big) = 0 \\
& 0 \leq \alpha_l^{ij} \leq C, \quad i = 1,\ldots,s; j = i+1,\ldots,s.
\end{aligned} \tag{3.30}$$

où $\delta = 1/(2(s+1)^2)$.
Le problème revient donc à résoudre la forme duale (3.30) pour laquelle les méthodes classiques de programmation mathématique peuvent être utilisées. La résolution de ce problème permettra de calculer les coefficients α_l^{ij} $i = 1,\ldots,s,\ j = i+1,\ldots,s$ et donc de calculer les paramètres w_{ij} et b_{ij} $i = 1,\ldots,s,\ j = i+1,\ldots,s$
Dans ce cas, l'hyperplan qui sépare la région $\Re_i$ du reste des régions est défini par :

$$H_i = \begin{bmatrix} w_i - w_1 & \cdots & w_i - w_{j,j\neq i} & \cdots & w_i - w_s \\ b_i - b_1 & \cdots & b_i - b_{j,j\neq i} & \cdots & b_i - b_s \end{bmatrix}^\top . \tag{3.31}$$

Contrairement aux deux autres approches (un-contre-reste et un-contre-un), les régions estimées par cette méthode constituent bien une partition complète de l'espace de régression $\Re$.

3.3.4 Exemples numériques

Exemple 3.1. Nous considérons un modèle affine par morceaux SISO composé de trois sous-modèles ($s = 3$) d'ordre un ($n_a = n_b = 1$). Les régions correspondantes aux sous-modèles sont définies par :

$$\begin{aligned}
\Re_1 &= \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} 1 & -0.3 & -0.4 \end{bmatrix} \bar{\varphi} \leq 0 \right\} \\
\Re_2 &= \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} 1 & -0.3 & -0.4 \end{bmatrix} \bar{\varphi} > 0 \text{ et } \begin{bmatrix} 3 & 0.3 & 0.8 \end{bmatrix} \bar{\varphi} \leq 0 \right\} \\
\Re_3 &= \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} 3 & 0.3 & 0.8 \end{bmatrix} \bar{\varphi} > 0 \right\}
\end{aligned}$$

où $\bar{\varphi} = \begin{bmatrix} \varphi^\top & 1 \end{bmatrix}^\top$.
Un ensemble de 300 données a été généré suivant trois vecteurs de paramètres θ_1, θ_2 et

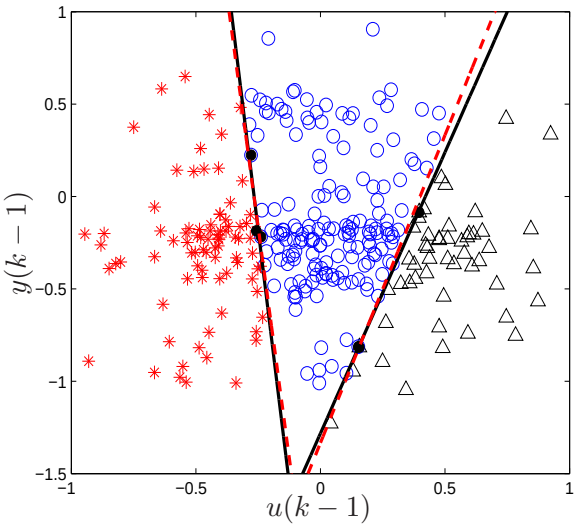


FIGURE 3.5 – Séparateurs linéaires réels (en pointillé) et estimés (en continu) par l'approche "un-contre-un".

θ_3. Nous supposons ici que nous connaissons les étiquettes de toutes les données.
La procédure de détection de l'adjacence entre les régions a été appliquée en choisissant $N_{adj} = N/10 = 30$. Les résultats montrent que les régions sont adjacentes deux par deux. Ainsi un classifieur binaire a été utilisé pour chaque paire $(\Re_i,\Re_j)$ partageant une frontière commune. Le nombre de classifieurs nécessaires dans ce cas est de 2.

Les hyperplans $\hat{H}_i$ identifiés définissant les trois régions sont donnés par : $\hat{H}_1 = \hat{h}_1$, $\hat{H}_2 = \left[-\hat{h}_1^\top \ \hat{h}_2^\top \right]^\top$ et $\hat{H}_3 = -\hat{h}_2^\top$.
où

$$\hat{h}_1 = \left[1 \ -0.328 \ -0.421\right],$$
$$\hat{h}_2 = \left[3 \ 0.273 \ 0.799\right].$$

On peut constater que les hyperplans estimés sont très proches des hyperplans réels. Les séparateurs linéaires réels et ceux estimés sont représentés sur la figure 3.5.

Exemple 3.2. Dans cet exemple, nous identifions la partition polyédrique de l'exemple 2.2 (section 2.3.5). La procédure de classification des données et d'estimation des paramètres (cf. Chapitre 2) a permis de regrouper les données $x(i)$ en trois classes différentes. A ces données (déjà étiquetées) nous avons appliqué la procédure de détection de l'adjacence avec $N_{adj} = N/10 = 30$. Ce test d'adjacence a montré que chaque région est frontalière avec les deux autres régions. Par conséquent, l'approche la plus adéquate à appliquer dans

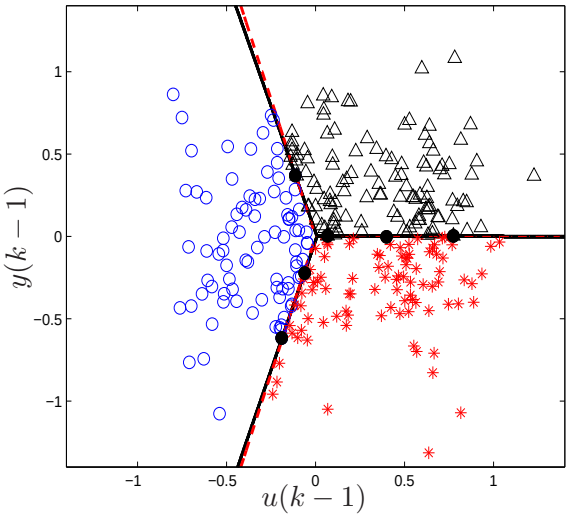


FIGURE 3.6 – Séparateurs linéaires réels (en pointillé) et estimés (en continu) par l'approche "tous-contre-tous".

ce cas est l'approche "tous-contre-tous" en utilisant un séparateur linéaire par morceaux. Les hyperplans $\hat{H}_i$ identifiés définissant les trois régions sont donnés par :

$$\hat{H}_1 = \left[\hat{h}_1^\top - \hat{h}_2^\top \;\; \hat{h}_1^\top - \hat{h}_3^\top\right]^\top,$$
$$\hat{H}_2 = \left[\hat{h}_2^\top - \hat{h}_1^\top \;\; \hat{h}_2^\top - \hat{h}_3^\top\right]^\top,$$
$$\hat{H}_3 = \left[\hat{h}_3^\top - \hat{h}_1^\top \;\; \hat{h}_3^\top - \hat{h}_2^\top\right]^\top.$$

où

$$\hat{h}_1 = \left[1 \; 0.3097 \; 0.0021\right],$$
$$\hat{h}_2 = \left[-0.0184 \; 0.5 \; 0.0029\right],$$
$$\hat{h}_3 = \left[1 \; -0.3335 \; 0.0005\right].$$

Les paramètres des hyperplans identifiés sont très proches des paramètres réels (voir exemple 2.2). Les données de régression des différentes régions ainsi que les séparateurs linéaires réels et estimés sont représentés sur la figure 3.6.

3.3.5 Estimation de frontières séparatrices non-linéaires

Les techniques de classification multi-classe présentées précédemment peuvent être étendues pour traiter le cas où la loi de commutation est définie non pas par un partitionnement polyédrique mais par un partitionnement où les frontières séparatrices sont

non linéaires. L'astuce qui fait vraiment la force des SVM repose sur ce qu'on appelle "les noyaux reproduisants". L'idée de Boser et al. [14] fut alors de projeter les données φ dans un espace $\mathbb{H}$ de dimension plus élevée, voire infinie, appelé espace de Hilbert à noyau reproduisant [5] à l'aide d'une fonction non-linéaire : $\eta : \mathbb{R}^n \to \mathbb{H}$ choisie *a priori*.

Cette projection permet de surmonter les inconvénients du cas des données non linéairement séparables puisque l'hyperplan estimé sera linéaire dans l'espace de projection RKHS. Intuitivement, plus la dimension de l'espace de projection est grande, plus la probabilité de pouvoir trouver un hyperplan séparateur optimal est élevée.
Pour estimer des séparateurs non-linéaires, il suffit de remplacer dans les problèmes d'optimisation duales ((3.6), (3.12) où (3.30)) précédents le produit scalaire $\varphi(i)^\top\varphi(j)$ par le produit $\eta(\varphi(i))^\top\eta(\varphi(j))$.
Grâce au théorème de Mercer [81], le produit scalaire $\eta(\varphi(i))^\top\eta(\varphi(j))$ peut se calculer facilement à l'aide d'une fonction symétrique κ, dite fonction noyau, définie par :

$$\kappa(\varphi(i),\varphi(j)) = \eta\left(\varphi(i)\right)^\top \eta\left(\varphi(j)\right). \tag{3.32}$$

En pratique, quelques familles de fonctions noyau paramétrables sont connues et il revient à l'utilisateur d'effectuer des tests pour déterminer celle qui convient le mieux. On peut citer les exemples de noyaux suivants :

- Noyau linéaire : $\kappa(\varphi(i),\varphi(j)) = \varphi(i)^\top\varphi(j)$.
- Noyau polynomial : $\kappa(\varphi(i),\varphi(j)) = \left(c + \varphi(i)^\top\varphi(j)\right)^d$.
- Noyau Gaussien : $\kappa(\varphi(i),\varphi(j)) = \exp\left(-\left\|\varphi(i)-\varphi(j)\right\|^2/\sigma^2\right)$.

où c, d, et σ sont des paramètres réels.
Dans la littérature, on retrouve aussi les termes de noyau reproduisant, noyau de Mercer, noyau admissible.

Exemple 3.3. Dans cet exemple, nous illustrons le cas où la loi de commutation est définie par un partitionnement de régions assuré par des frontières séparatrices non linéaires. Pour cela, nous considérons un système SISO composé de trois sous-modèles ($s = 3$) d'ordre un ($n_a = n_b = 1$) dont les régions de validité des différents sous-modèles sont définies par :

$$\Re_1 = \left\{\varphi \in \mathbb{R}^2 : h_1(\varphi) \geq 0 \ \text{ et } \ h_2(\varphi) \geq 0\right\} \tag{3.33}$$
$$\Re_2 = \left\{\varphi \in \mathbb{R}^2 : h_1(\varphi) < 0 \ \text{ et } \ h_3(\varphi) < 0\right\} \tag{3.34}$$
$$\Re_3 = \left\{\varphi \in \mathbb{R}^2 : h_2(\varphi) < 0 \ \text{ et } \ h_3(\varphi) \geq 0\right\} \tag{3.35}$$

5. En anglais RKHS : Reproducing Kernel Hilbert Space

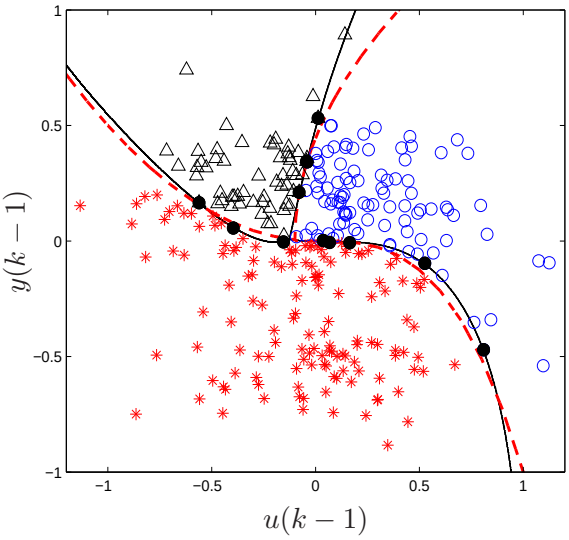


FIGURE 3.7 – Séparateurs non linéaires réels (en pointillé) et estimés (en continu) par l'approche "tous-contre-tous".

où

$$h_1(\varphi(k)) = y(k-1)^2 - 2u(k-1) - 0.2,$$
$$h_2(\varphi(k)) = 2y(k-1) - u(k-1)^2,$$
$$h_3(\varphi(k)) = -y(k-1) - u(k-1)^3.$$

Un ensemble de 300 données a été généré par trois sous-modèles affines respectant les régions de validité définies dans (3.33)-(3.35). L'approche tous-contre-tous a été appliquée en utilisant un noyau Gaussien : $\kappa(\varphi(i), \varphi(j)) = \exp\left(-\|\varphi(i) - \varphi(j)\|^2 / \sigma^2\right)$, avec $\sigma = 1$. Les frontières séparatrices non linéaires réelles et estimées ainsi que les données de régression sont représentées sur la figure 3.7. Les frontières séparatrices estimées discriminent bien les trois régions malgré la légère différence qui existe entre ces estimés et les frontières réelles.

3.4 Conclusion

Nous avons étudié dans ce chapitre le problème de l'estimation de la loi de commutation d'un sous-modèle à un autre. Cela consiste à estimer la région de validité de chaque sous-modèle dans le cas des modèles dynamiques affines par morceaux. L'objectif est de trouver les frontières qui délimitent chaque région dans la partition complète et donc à séparer les s régions par des séparateurs linéaires. L'estimation de la loi de commutation constitue l'étape finale de l'identification des modèles dynamiques affines par morceaux après l'étape de classification de données et d'estimation de paramètres.

Deux catégories de classification ont été abordées. La première catégorie combine des séries de classifieurs linéaires binaires en faisant appel à des schémas de décomposition du type un-contre-reste ou un-contre-un. La décomposition un-contre-reste construit pour chaque région, un classifieur binaire à vecteurs de support la séparant du reste des régions. La décomposition un-contre-un considère deux à deux les régions et construit un classifieur binaire à vecteurs de support pour chaque paire de régions. L'inconvénient de ces décompositions est que le partitionnement de l'espace de régression obtenu peut ne pas être complet, c'est-à-dire que certaines régions de l'espace de régression peuvent n'être associées à aucun mode. Une solution consiste à affecter les points appartenant à ces régions au mode dont la région de fonctionnement est la plus proche. Pour pallier cette difficulté nous avons présenté une approche qui consiste à résoudre le problème multi-classe en une seule étape sans le décomposer en une collection de sous-problèmes binaires. Elle permet de séparer les régions en considérant simultanément l'ensemble de toutes les données et en recherchant directement un séparateur linéaire par morceaux. Cette approche nécessite la résolution d'un seul problème d'optimisation quadratique sous contraintes linéaires. Nous avons vu que ces techniques de classification binaire ou multi-classe peuvent être étendues pour traiter le cas où la loi de commutation est définie par un partitionnement où les frontières séparatrices de chaque région sont non linéaires. Les données sont alors projetées dans un espace de dimension plus élevée, voire infinie, appelé espace de Hilbert à noyau reproduisant.

Nous proposons dans le chapitre suivant un algorithme d'identification récursive de modèles dynamiques affines par morceaux et des modèles dynamiques non linéaires par morceaux dont les paramètres des sous-modèles et des régions peuvent varier dans le temps.

4

Identification récursive de modèles affines par morceaux et de modèles non linéaires par morceaux

Sommaire

4.1 Introduction

Nous avons présenté dans les deux chapitres précédents des procédures pour la classification de données, l'estimation des paramètres et l'estimation des régions caractérisant les différents sous-modèles des modèles PWA et PWN. Les algorithmes que nous avons développés, dans le chapitre 2, supposent toutes les données disponibles lors de l'identification. Cependant, dans beaucoup de problèmes réels, les données sont acquises l'une après l'autre de façon continue. Dans ce cas, chaque donnée doit être traitée dès qu'elle se présente. De ce fait, les modèles doivent être récursivement estimés dans le temps. De plus, à cause des changements que peut subir un système donné, les paramètres des différents modes de fonctionnement qui le constituent peuvent évoluer dans le temps. Par exemple, une défaillance ou une dégradation lente d'un ou plusieurs des composants d'un système se manifeste par un changement graduel dans les paramètres définissant le mode concerné. Une adaptation des paramètres des différents modes est alors nécessaire pour prendre en compte ces évolutions. Cela est généralement réalisé grâce à une identification récursive sur une fenêtre glissante de données. Nous rappelons qu'un mode est décrit par un sous-modèle et sa région de validité. La tâche d'adaptation consiste alors à actualiser les paramètres du sous-modèle concerné et de sa région après l'incorporation de chaque nouvelle donnée. Cette tâche doit se faire en utilisant les connaissances extraites des données déjà acquises et sans attendre que toutes les informations contenues dans les données futures soient disponibles. Cette tâche doit également mettre à jour les paramètres du mode après le retrait de la donnée la plus ancienne de la fenêtre glissante considérée.

Dans ce chapitre, une approche d'identification récursive de modèles dynamiques affines par morceaux et de modèles non linéaires par morceaux variant dans le temps est proposée. Cette approche permet d'estimer récursivement les différents modes en attribuant les données, une après l'autre, au sous-modèle concerné en se basant sur une mesure de similarité spécifique. Après l'attribution de chaque donnée, les paramètres du mode concerné sont mis à jour grâce à des procédures d'adaptation. Des procédures permettant d'adapter les paramètres définissant chaque sous-modèle qu'il soit linéaire ou non linéaire, sont présentées. Un classifieur incrémental et décrémental multi-classe à vecteurs de support est également proposé pour l'estimation récursive des frontières séparatrices des régions de validité.

Ce chapitre est organisé de la manière suivante. Dans la Section 4.2, nous présentons le modèle théorique des systèmes considérés et nous formulons le problème. Dans la Section 4.3, nous décrivons l'algorithme générique pour l'identification récursive des modèles considérés. Nous commençons d'abord par détailler le principe de cet algorithme. Nous décrivons ensuite des procédures permettant d'adapter les paramètres définissant les dif-

férents sous-modèles. L'adaptation des sous-modèles affines est effectuée par l'algorithme des moindres carrés récursifs à fenêtre glissante [30]. Nous proposons pour l'adaptation des sous-modèles non linéaires une procédure basée sur des LS-SVM récursifs (LS-SVMR). Puis, nous décrivons une procédure d'adaptation des paramètres définissant les frontières séparatrices des régions. Un classifieur multi-classe incrémental et décrémental à vecteurs de support (MID-SVM) a été développé pour effectuer cette tâche. Enfin, nous présentons, dans la section 4.4, des exemples de simulation afin de montrer les performances de notre algorithme et d'étudier l'influence de ses paramètres de réglage sur l'estimation récursive.

4.2 Modèle théorique et formulation du problème

Dans ce chapitre, nous considérons des modèles dynamiques affines par morceaux et des modèles dynamiques non linéaires par morceaux dont les paramètres des sous-modèles et des régions peuvent varier dans le temps. Nous nous intéressons plus particulièrement à des systèmes MISO, définis par :

$$y\left(t\right) = f_{\sigma(t)}^{(t)}(\varphi(t)) + \varepsilon\left(t\right), \tag{4.1}$$

où $\varepsilon(t)$ est l'erreur de prédiction, $\sigma(t)$ est l'état discret et $\varphi(t) \in \mathbb{R}^n$ est le vecteur de régression défini par

$$\varphi(t) = \left[y(t-1), \cdots, y(t-n_a), u(t)^\top, \cdots, u(t-n_b)^\top\right]^\top \tag{4.2}$$

avec $u(t) \in \mathbb{R}^m$ et $y(t) \in \mathbb{R}$ respectivement l'entrée et la sortie mesurée du système à l'instant $t \in \mathbb{N}$, n_a et n_b sont ses ordres.
La fonction de régression $f_{\sigma(t)}^{(t)}$ représente le modèle du mode actif $\mathscr{M}_{\sigma(t)}$ à l'instant t. Cette fonction peut varier dans le temps suivant les variations que peut subir ce mode. Si le sous-modèle représentant le mode $\mathscr{M}_{\sigma(t)}$ est linéaire alors cette fonction s'écrit :

$$f_{\sigma(t)}^{(t)}\left(\varphi\right) = {\theta_{\sigma(t)}^{(t)}}^\top \left[\varphi^\top \; 1\right]^\top. \tag{4.3}$$

avec $\theta_{\sigma(t)}^{(t)}$ est le vecteur de paramètres définissant ce sous-modèle.
Si le sous-modèle représentant le mode $\mathscr{M}_{\sigma(t)}$ est non linéaire alors la fonction $f_{\sigma(t)}^{(t)}$ s'écrit :

$$f_{\sigma(t)}^{(t)}(\varphi) = {w_{\sigma(t)}^{(t)}}^\top \eta(\varphi) + b_{\sigma(t)}^{(t)}. \tag{4.4}$$

où η est une fonction non linéaire choisie *a priori*, $w_{\sigma(t)}^{(t)}$ est un vecteur de poids et $b_{\sigma(t)}^{(t)}$ est un biais.
Les sous-modèles constituant le modèle théorique considéré dans ce chapitre sont soit tous linéaires soit tous non linéaires.

La loi de commutation $\sigma(t)$ est définie par

$$\sigma\left(t\right) = i \quad \text{ssi} \quad \varphi(t) \in \Re_i, \quad i = 1, \ldots, s. \tag{4.5}$$

Les régions $\{\Re_i\}_{i=1}^{s}$ forment une partition complète du domaine de régression $\Re \subset \mathbb{R}^n$, avec $n = n_a + m(n_b + 1)$, chaque région $\Re_i$ est décrite par :

$$\Re_i = \left\{ \varphi \in \mathbb{R}^n : h_i^{(t)}(\varphi) > h_j^{(t)}(\varphi),\ \forall j \in \{1, \ldots, s\}, j \neq i \right\} \tag{4.6}$$

où $h_i^{(t)}$ est une fonction définissant la frontière séparatrice qui délimite la région $\Re_i$. Cette fonction peut être linéaire ou non linéaire.

Dans ce chapitre, nous faisons une hypothèse sur l'évolution de la loi de commutation $\sigma(t)$. Nous supposons que les instants de commutation sont séparés par un temps minimum de séjour τ_{sej}. Aucune transition n'est autorisée pendant ce temps.

4.3 Algorithme générique d'identification récursive

Un algorithme d'identification récursive de modèles dynamiques affines par morceaux et de modèles non linéaires par morceaux est proposé dans cette section. La structure de cet algorithme est présentée dans le synopsis de la figure 4.1. L'objectif de cet algorithme est d'estimer récursivement les modes en leur attribuant les données $x(t) = \left[\varphi(t)^\top, y(t)\right]^\top$ selon un critère d'appartenance spécifique. Ce dernier permet de décider à quel mode doit être affectée la donnée. Ainsi, la donnée est assignée au mode qui, de tous les modes existants, est le générateur le plus vraisemblable de cette donnée. Une fois la donnée affectée, les paramètres de ce sous-modèle et les paramètres des différentes régions sont mis à jour.

4.3.1 Principe de l'algorithme

La conception de notre algorithme passe par plusieurs étapes. A l'instant initial, la première donnée $x(t) = x(1)$ est acquise, le premier mode $\mathscr{M}_1$ est automatiquement créé. Toutes les données suivantes appartiendront au même sous-modèle que la première donnée pendant un certain temps minimum τ. Le paramètre τ est défini par l'utilisateur. Ce paramètre doit être inférieur ou égal au temps minimum théorique de séjour τ_{sej}. Le sous-modèle est mis à jour au fur et à mesure que les données sont acquises. Cette mise à jour s'effectue par l'adaptation de la fonction de régression $f_1^{(t)}$. Après que le temps minimum τ soit écoulé, les données suivantes peuvent appartenir au premier mode comme elles peuvent contribuer à la création d'un nouveau mode. Une mesure de similarité permettant

de mesurer l'adéquation entre les observations et les sous-modèles est donc nécessaire. Cela sera expliqué dans le cas général ci-après.

On suppose que l'on dispose, à l'instant t, de $\bar{s}$ modes. Après l'acquisition de la donnée $x(t+1)$, si le temps minimum τ du mode en cours n'est pas encore atteint, alors la donnée est affectée à ce même mode, c'est-à-dire $\hat{\sigma}(t+1) = \hat{\sigma}(t)$. La fonction de régression $f^{(t+1)}_{\hat{\sigma}(t+1)}$ ainsi que les fonctions $h_i^{(t+1)}$ définissant les frontières séparatrices sont mises à jour par les procédures d'adaptation. Dans le cas contraire (le temps minimum τ du mode en cours est écoulé), un test de similarité avec les $\bar{s}$ sous-modèles déjà identifiés doit être effectué. On dénombre alors deux cas. Cas 1 : la donnée $x(t+1)$ n'est affectée à aucun mode. Une commutation est alors détectée. Cette donnée contribue à la création d'un nouveau mode $\mathscr{M}_{\bar{s}+1}$. Cas 2 : la donnée $x(t+1)$ est affectée à un des modes existants. Dans ce deuxième cas, on distingue deux possibilités : (a) la donnée $x(t+1)$ appartient au même mode actif à l'instant t. Aucune commutation n'est alors détectée. (b) la donnée $x(t+1)$ appartient à un mode différent de celui actif à l'instant t. Dans ce cas, une commutation est détectée. Après chaque commutation, les données qui succèderont, doivent respecter la condition d'appartenance au même sous-modèle pendant tout le temps minimum τ.

Cas 1 : un nouveau mode est détecté si la donnée $x(t+1)$ n'est générée par aucun des sous-modèles déjà identifiés. Selon notre critère, cela signifie que l'observation $x(t+1)$ a des degrés d'appartenance, aux différents modes existants, tous inférieurs à un certain seuil de confiance. En effet, un nouveau mode est détecté si

$$\forall q = 1, \ldots, \bar{s}, P_q^{(t+1)} < \theta_{seuil} \tag{4.7}$$

où $P_q^{(t+1)}$ est un critère d'appartenance de $x(t+1)$ au mode $\mathscr{M}_q$, il est donné par

$$P_q^{(t+1)} = \sum_{k/x(k)\in\{\Gamma_c(x(t+1))\cap\mathscr{M}_q\}} \phi_k^{t+1},\ q \in \{1, \ldots, \bar{s}\}\,. \tag{4.8}$$

Nous rappelons que $\Gamma_c\left(x(t+1)\right)$ est l'ensemble des c plus proches voisins de $x(t+1)$. Nous rappelons également que la mesure ϕ_k^{t+1} caractérise l'information fournie par $x(k)$ (une donnée générée par $\mathscr{M}_q$), située dans le voisinage de $x(t+1)$, sur l'éventuelle appartenance de $x(t+1)$ au mode $\mathscr{M}_q$. Cette mesure est définie par

$$\phi_k^{t+1} = \exp\left(-\alpha_{\hat{\sigma}(k)} \left\|x(t+1) - x(k)\right\|_2^2 - \beta_{\hat{\sigma}(k)} \left(y(t+1) - f_{\hat{\sigma}(k)}(\varphi(t+1))\right)^2\right). \tag{4.9}$$

Le seuil de détection θ_{seuil} peut être choisi suite à différents essais. Le nombre de sous-modèles identifiés peut dépendre de ce seuil. Les données qui suivent l'instant de commutation vers ce nouveau mode serviront à l'estimation récursive de ses paramètres pendant tout le temps minimum τ. La fonction de régression $f^{(t+1)}_{\bar{s}+1}$ est mise à jour ainsi que les

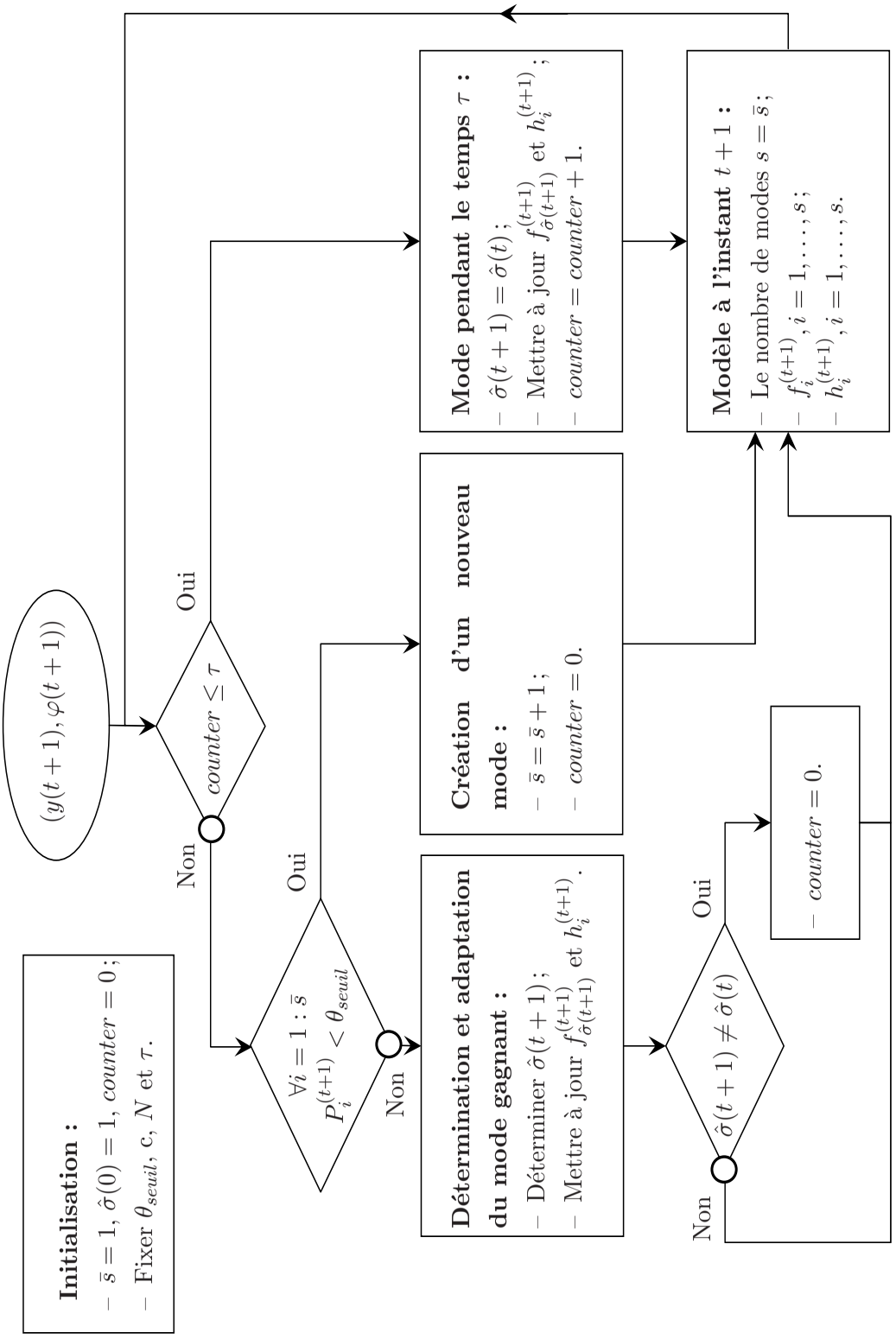


FIGURE 4.1 – Structure de l'algorithme d'identification récursive

fonctions $h_i^{(t+1)}$ définissant les frontières séparatrices des régions.

Cas 2 : dans le cas contraire où il existe $q \in \{1, \ldots, \bar{s}\}$ tel que $P_q^{(t+1)} \geq \theta_{seuil}$, nous devons décider sur l'affectation de la nouvelle donnée $x(t+1)$ à un seul mode parmi les modes existants. Appelons M^{win} l'ensemble de sous-modèles définis par :

$$\mathrm{M}^{win} = \left\{ \mathscr{M}_q \text{ tel que } P_q^{(t+1)} \geq \theta_{seuil} \right\}, q \left\{ \in 1, \ldots, \bar{s} \right\}, \tag{4.10}$$

une compétition se fait entre les sous-modèles de l'ensemble M^{win} pour décider sur l'affectation de la donnée $x(t+1)$. Cette compétition est basée sur les fonctions de décision définissant les frontières séparatrices des régions. En effet, l'affectation de $x(t+1)$ s'effectue par la règle de décision suivante :

$$x(t+1) \rightarrow \mathscr{M}_{ret} \text{ tel que } ret = \underset{i/\mathscr{M}_i \in \mathrm{M}^{win}}{\arg\max} \left(h_i^{(t)}(\varphi(t+1)) \right) \tag{4.11}$$

avec $h_i^{(t)}$ la fonction définissant la région $\Re_i$; cette fonction est définie (voir chapitre 3 p. 83) par :

$$\begin{aligned} h_i^{(t)}(\varphi(t+1)) = \tfrac{1}{s+1} \sum_{\substack{j=1 \\ j \neq i}}^{s} \Bigg[& \sum_{\varphi(l) \in \Re_i} \alpha_l^{ij} (\eta(\varphi(l))^\top \eta(\varphi(t+1))) \\ & - \sum_{\varphi(l) \in \Re_j} \alpha_l^{ij} (\eta(\varphi(l))^\top \eta(\varphi(t+1))) \Bigg] + b_i. \end{aligned} \tag{4.12}$$

Dans ce cas $\hat{\sigma}(t+1) = ret$. Si $\hat{\sigma}(t+1) = \hat{\sigma}(t)$, cela veut dire que le sous-modèle actif à l'instant t reste actif à l'instant $t+1$. Aucune commutation ne s'est donc produite à l'instant $t+1$. Dans le cas contraire (c'est-à-dire $\hat{\sigma}(t+1) \neq \hat{\sigma}(t)$), le système a commuté de $\mathscr{M}_{\hat{\sigma}(t)}$ vers $\mathscr{M}_{\hat{\sigma}(t+1)}$. Les données postérieures à l'instant $t+1$ appartiendront au même sous-modèle pendant un durée minimum τ correspondante à l'hypothèse de temps séjour. Après l'affectation de la donnée $x(t+1)$ au mode $\mathscr{M}_{\hat{\sigma}(t+1)}$, les paramètres de son sous-modèle $f_{\hat{\sigma}(t+1)}^{(t+1)}$ sont mis à jour ainsi que les fonctions $h_i^{(t+1)}$ définissant les frontières séparatrices des régions. L'algorithme d'identification récursive est résumé dans l'**Algorithme 4.1**.

Dans ce qui suit, nous décrivons, dans un premier temps, des techniques d'adaptation permettant de mettre à jour les paramètres des sous-modèles définis par la fonction de régression $f_{\hat{\sigma}(t+1)}^{(t+1)}$. Si les modèles traités sont des modèles dynamiques affines par morceaux, l'estimation récursive des sous-modèles est effectuée par l'algorithme des moindres carrés récursifs à fenêtre glissante [30]. Dans le cas des modèles dynamiques non linéaires par morceaux, l'estimation récursive des sous-modèles est assurée par un nouvel algorithme appelé LS-SVMR. Dans un deuxième temps, nous présentons un classifieur incrémental

Algorithme 4.1

Initialisation :

- Fixer le seuil de confiance θ_{seuil}, le nombre des plus proches voisins c, la taille de la fenêtre glissante N et le temps de séjour τ.
- Faire $\bar{s} = 1$, $\hat{\sigma}(0) = 1$, $counter = 0$.

Identification récursive :

Pour $t = 1, \ldots, \infty$
  Si $counter \leq \tau$
  - L'état discret $\hat{\sigma}(t+1) = \hat{\sigma}(t)$;
  - Mettre à jour le mode $\mathscr{M}_{\hat{\sigma}(t+1)}$:
    - Adapter les paramètres du sous-modèle $f^{(t+1)}_{\hat{\sigma}(t+1)}$;
    - Adapter les paramètres des frontières séparatrices $h_i^{(t+1)}, i = 1, \ldots, \bar{s}$;
  - $counter = counter + 1$.

  Sinon
    Si $\forall i = 1 : \bar{s}, P_i^{(t+1)} < \theta_{seuil}$
    - Créer un nouveau mode et faire $\bar{s} = \bar{s} + 1$;
    - Faire $counter = 0$.

    Sinon
    - Déterminer $\hat{\sigma}(t+1)$;
    - Mettre à jour le mode $\mathscr{M}_{\hat{\sigma}(t+1)}$:
      - Adapter les paramètres du sous-modèle $f^{(t+1)}_{\hat{\sigma}(t+1)}$;
      - Adapter les paramètres des frontières séparatrices $h_i^{(t+1)}, i = 1, \ldots, \bar{s}$;

      Si $\hat{\sigma}(t+1) \neq \hat{\sigma}(t)$
      - Faire $counter = 0$.

      FinSi.

    FinSi.

  FinSi.

FinPour.

et décrémental multi-classe à vecteurs de support qui permet d'adapter les frontières séparatrices des différentes régions dans l'espace de régression. Il s'agit de la sous-routine MID-SVM. Les procédures d'adaptation cherchent à produire des estimations successives des différents modes en utilisant les connaissances déjà acquises et les informations des nouvelles données. Ces procédures permettent le suivi de l'évolution des modes dans le

temps. L'objectif ici, consiste à déterminer de façon récursive les paramètres sur une fenêtre glissante de taille N en ajoutant une nouvelle donnée et en retirant une donnée ancienne.

4.3.2 Procédures d'adaptation des sous-modèles

4.3.2.1 Adaptation des paramètres des sous-modèles affines : les moindres carrés récursifs à fenêtre glissante

L'adaptation des sous-modèles affines est effectuée par l'algorithme des moindres carrés récursifs à fenêtre glissante [6] [30]. Cet algorithme est une variante de l'algorithme classique des moindres carrés récursifs [61]. La majeure différence entre les deux algorithmes est que l'algorithme classique des moindres carrés récursifs possède un facteur d'oubli lui permettant d'oublier les données qui correspondent à un passé distant ; l'algorithme des moindres carrés récursifs à fenêtre glissante, comme son nom l'indique, estime récursivement les vecteurs de paramètres sur une fenêtre de taille fixe en ajoutant une nouvelle donnée et retirant la plus ancienne donnée.

Disposant d'une valeur initiale du vecteur de paramètres $\theta_i^{(t)}$ et d'une valeur initiale d'une matrice $P_i(t) = \mu I_n$, avec μ un scalaire, les moindres carrés récursifs à fenêtre glissante permettent d'adapter le vecteur de paramètres au fur à mesure qu'une nouvelle donnée est acquise sur une fenêtre de taille N.
Cet algorithme récursif permet de mettre à jour le vecteur de paramètres $\hat{\theta}_i^{(t+1)}$ où $i = \sigma(t+1)$ après l'ajout de la nouvelle donnée $(y(t+1), \varphi(t+1))$ et le retrait de l'ancienne donnée $(y(t-N+1), \varphi(t-N+1))$.
L'adaptation du vecteur de paramètres $\hat{\theta}_i^{(t+1)}$ à partir du vecteur de paramètres $\hat{\theta}_i^{(t)}$ s'effectue par les formules suivantes [30] :

$$\tilde{P}_i(t) = P_i(t) + \frac{P_i(t)\bar{\varphi}(t-N+1)\bar{\varphi}(t-N+1)^\top P_i(t)}{1+\bar{\varphi}(t-N+1)^\top P_i(t)\bar{\varphi}(t-N+1)}, \tag{4.13}$$

$$\tilde{\theta}_i^{(t)} = \hat{\theta}_i^{(t)} - \tilde{P}_i(t)\bar{\varphi}(t-N+1)^\top \left[y(t-N+1) - \hat{\theta}_i^{(t)\top}\bar{\varphi}(t-N+1)\right], \tag{4.14}$$

$$P_i(t+1) = \tilde{P}_i(t) - \frac{\tilde{P}_i(t)\bar{\varphi}(t+1)\bar{\varphi}(t+1)^\top \tilde{P}_i(t)}{1+\bar{\varphi}(t+1)^\top \tilde{P}_i(t)\bar{\varphi}(t+1)}, \tag{4.15}$$

$$\hat{\theta}_i^{(t+1)} = \tilde{\theta}_i^{(t)} + P_i(t+1)\bar{\varphi}(t+1)\left[y(t+1) - \tilde{\theta}_i^{(t)\top}\bar{\varphi}(t+1)\right], \tag{4.16}$$

où N est la taille de la fenêtre glissante et $\bar{\varphi}(t) = \begin{bmatrix}\varphi(t)^\top & 1\end{bmatrix}^\top$.
L'équation (4.16) permet de mettre à jour le vecteur de paramètres $\theta_i^{(t+1)}$ après l'ajout de la nouvelle donnée $(y(t+1), \varphi(t+1))$ et l'élimination de la donnée $(y(t-N+1), \varphi(t-N+1))$.

6. En anglais : Finite-data-window recursive least squares FDW-RLS

Dans le paragraphe suivant, nous décrivons ce que nous appelons les LS-SVM récursifs qui permettent de mettre à jour les paramètres des sous-modèles non linéaires.

4.3.2.2 Adaptation des paramètres des sous-modèles non linéaires : régression par les LS-SVM récursifs

Dans cette section, nous proposons une méthode de régression par des LS-SVM récursifs. La procédure d'adaptation LS-SVMR présentée ici, permet l'estimation récursive des paramètres de la fonction de régression non linéaire $f_i^{(t)}$ où $i = \hat{\sigma}(t+1)$. L'estimation récursive est élaborée ici à partir d'un calcul analytique ayant pour but de déterminer la récurrence sur les paramètres de $f_i^{(t)}$. L'estimation se fait sur une fenêtre glissante de taille N par l'ajout d'une nouvelle donnée et l'élimination d'une donnée ancienne. Si la taille de l 'ensemble de données définissant le modèle $f_i^{(t)}$ est inférieure à N, l'élimination de la donnée la plus ancienne n'est pas effectuée.

A l'instant t, nous disposons d'un modèle $f_i^{(t)}$ estimé à partir d'un ensemble de données $\{y(k), \varphi(k)\}_{k=t-N+1}^{t}$ de taille N généré par le modèle $\mathscr{M}_i$. L'objectif consiste à estimer récursivement le modèle $f_i^{(t+1)}$ après l'ajout d'une nouvelle donnée $(y(t+1), \varphi(t+1))$ et le retrait de l'ancienne donnée $(y(t-N+1), \varphi(t-N+1))$. Le modèle $f_i^{(t+1)}$ est donc défini à partir d'un ensemble $\{y(k), \varphi(k)\}_{k=t-N+2}^{t+1}$ de taille N.
Le modèle de la fonction de régression à l'instant t est défini par :

$$f_i^{(t)}(\varphi) = \sum\nolimits_{k=t-N+1}^{t} \alpha_i^{(t)}(k)\kappa\left(\varphi(k), \varphi\right) + \rho_i^{(t)} \tag{4.17}$$

Nous rappelons que les paramètres de la fonction $f_i^{(t)}$ ($\alpha_i^{(t)}$ et $\rho_i^{(t)}$) sont calculés selon l'équation suivante (cf. section 2.4.2.1) :

$$\begin{bmatrix} \rho_i^{(t)} \\ \alpha_i^{(t)} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & \mathbf{1}_N^\top \\ \mathbf{1}_N & \Omega + \gamma^{-1} I_N \end{bmatrix}}_{K_N}^{-1} \begin{bmatrix} 0 \\ \mathbf{y}^{(t)} \end{bmatrix} \tag{4.18}$$

où $\mathbf{1}_N = [1, \ldots, 1]^\top \in \mathbb{R}^N$ et $\mathbf{y}^{(t)} = [y(t-N+1), \ldots, y(t)]^\top$, $\alpha_i^{(t)} = \left[\alpha_i^{(t)}(1), \ldots, \alpha_i^{(t)}(N)\right]^\top$.
I_N est une matrice identité de dimension $N \times N$ et $\Omega \in \mathbb{R}^{N \times N}$ est une matrice définie par $\Omega_{ij} = \kappa(\varphi(i), \varphi(j))$ où κ est la fonction noyau ; elle peut être polynomiale, gaussienne, etc.
L'estimation récursive de $f_i^{(t+1)}$ consiste à estimer les paramètres $\alpha_i^{(t+1)}$ et $\rho_i^{(t+1)}$ à partir des paramètres $\alpha_i^{(t)}$ et $\rho_i^{(t)}$. Dans notre démarche, nous estimons des paramètres intermédiaires $\tilde{\alpha}_i^{(t+1)}$ et $\tilde{\rho}_i^{(t+1)}$ du modèle $\tilde{f}_i^{(t+1)}$ défini à partir de l'ensemble $\{y(k), \varphi(k)\}_{k=t-N+1}^{t+1}$ de taille $N+1$ après l'ajout de $(y(t+1), \varphi(t+1))$. Nous donnons, d'abord une relation

reliant les paramètres $\tilde{\alpha}_i^{(t+1)}$ et $\tilde{\rho}_i^{(t+1)}$ aux paramètres $\alpha_i^{(t)}$ et $\rho_i^{(t)}$ puis une relation reliant les paramètres $\alpha_i^{(t+1)}$ et $\rho_i^{(t+1)}$ aux $\tilde{\alpha}_i^{(t+1)}$ et $\tilde{\rho}_i^{(t+1)}$:

$$\left(\rho_i^{(t)}, \alpha_i^{(t)}\right) \overset{\text{ajout de } x(t+1)}{\longrightarrow} \left(\tilde{\rho}_i^{(t+1)}, \tilde{\alpha}_i^{(t+1)}\right) \overset{\text{retrait de } x(t-N+1)}{\longrightarrow} \left(\rho_i^{(t+1)}, \alpha_i^{(t+1)}\right).$$

Ajout d'une nouvelle donnée $(y(t+1), \varphi(t+1))$ **:** après l'ajout d'une nouvelle donnée $(y(t+1), \varphi(t+1))$, les coefficients de Lagrange $\tilde{\alpha}_i^{(t+1)}$ ainsi que le biais $\tilde{\rho}_i^{(t+1)}$ sont alors donnés par

$$\begin{bmatrix} \tilde{\rho}_i^{(t+1)} \\ \tilde{\alpha}_i^{(t+1)} \end{bmatrix} = \begin{bmatrix} K_N & u \\ u^\top & \delta_1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \tilde{\mathbf{y}}^{(t+1)} \end{bmatrix} \tag{4.19}$$

où $u = [1, \kappa(\varphi(t-N+1), \varphi(t+1)), \ldots, \kappa(\varphi(t), \varphi(t+1))]^\top$, $\tilde{\mathbf{y}}^{(t+1)} = [y(t-N+1), \ldots, y(t+1)]^\top$ et $\delta_1 = \kappa(\varphi(t+1), \varphi(t+1)) + \gamma^{-1}$.
L'inverse d'une matrice peut être calculé en utilisant la formule d'inversion analytique par bloc [12] suivante

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -(D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix} \tag{4.20}$$

où A, B, C et D sont des blocs de taille appropriée, les blocs A et D doivent être des matrices carrées. L'application de cette formule à l'équation (4.19) donne l'expression suivante

$$\begin{bmatrix} \tilde{\rho}_i^{(t+1)} \\ \tilde{\alpha}_i^{(t+1)} \end{bmatrix} = \begin{bmatrix} K_N^{-1} + \frac{1}{\nu_1} K_N^{-1} u u^\top K_N^{-1} & -\frac{1}{\nu_1} K_N^{-1} u \\ -\frac{1}{\nu_1} u^\top K_N^{-1} & \frac{1}{\nu_1} \end{bmatrix} \begin{bmatrix} 0 \\ \tilde{\mathbf{y}}^{(t+1)} \end{bmatrix} \tag{4.21}$$

où $\nu_1 = (\delta_1 - u^\top K_N^{-1} u)$, à condition que $\nu_1 \neq 0$.

Après simplification de l'équation (4.21) et en tenant compte de l'équation (4.18), on obtient la formule de récurrence suivante

$$\begin{bmatrix} \tilde{\rho}_i^{(t+1)} \\ \tilde{\alpha}_i^{(t+1)} \end{bmatrix} = \begin{bmatrix} \rho_i^{(t)} \\ \alpha_i^{(t)} \\ 0 \end{bmatrix} + \frac{1}{\nu_1} v v^\top \begin{bmatrix} 0 \\ \tilde{\mathbf{y}}^{(t+1)} \end{bmatrix} \tag{4.22}$$

avec $v = \begin{bmatrix} K_N^{-1} u & -1 \end{bmatrix}^\top$.
L'équation (4.22) permet la mise à jour de la fonction $f_i^{(t)}$ après l'ajout de la donnée $(y(t+1), \varphi(t+1))$.

Retrait d'une ancienne donnée $(y(t-N+1), \varphi(t-N+1))$ **:** l'équation (4.21) s'écrit

$$\begin{bmatrix} \tilde{\rho}_i^{(t+1)} \\ \tilde{\alpha}_i^{(t+1)} \end{bmatrix} = R_{N+1} \begin{bmatrix} 0 \\ \tilde{\mathbf{y}}^{(t+1)} \end{bmatrix} \tag{4.23}$$

Avant de calculer la récurrence, nous devons d'abord changer l'ordre des éléments dans le vecteur des coefficients $\tilde{\alpha}_i^{(t+1)}$ et le vecteur des sorties $\tilde{\mathbf{y}}^{(t+1)}$ de la façon suivante :

$$\begin{bmatrix} \tilde{\rho}_i^{(t+1)} \\ \tilde{\alpha}_i^{(t+1)}(2:N+1) \\ \tilde{\alpha}_i^{(t+1)}(1) \end{bmatrix} = S_{N+1} \begin{bmatrix} 0 \\ \mathbf{y}^{(t+1)} \\ y(t-N+1) \end{bmatrix} \tag{4.24}$$

où $\mathbf{y}^{(t+1)} = [y(t-N+2), \ldots, y(t+1)]^\top$ et S_{N+1} est obtenue en déplaçant la deuxième ligne et et la deuxième colonne de la matrice R_{N+1} respectivement à la dernière ligne et à la dernière colonne.
L'objectif maintenant consiste à trouver la matrice $K_N'^{-1}$ tel que

$$S_{N+1}^{-1} = \begin{bmatrix} K_N' & o \\ o^T & \delta_2 \end{bmatrix} \tag{4.25}$$

Écrivant S_{N+1} sous la forme suivante $S_{N+1} = \begin{bmatrix} S_N & \omega \\ \omega^\top & \delta_3 \end{bmatrix}$, l'utilisation de la formule d'inversion d'une matrice par bloc (Eq. 4.20), donne

$$S_{N+1}^{-1} = \begin{bmatrix} S_N^{-1} + \frac{1}{\nu_2} S_N^{-1}\omega\omega^\top S_N^{-1} & -\frac{1}{\nu_2} S_N^{-1}\omega \\ -\frac{1}{\nu_2}\omega^\top S_N^{-1} & \frac{1}{\nu_2} \end{bmatrix} \tag{4.26}$$

avec $\nu_2 = (\delta_3 - \omega^\top S_N^{-1}\omega)$. En utilisant la formule de Sherman-Morrison-Woodbury [12] : $A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} = (A - BD^{-1}C)^{-1}$, l'expression (4.26) s'écrit alors :

$$S_{N+1}^{-1} = \begin{bmatrix} \left(S_N - \frac{\omega\omega^\top}{\delta_3}\right)^{-1} & -\frac{1}{\nu_2} S_N^{-1}\omega \\ -\frac{1}{\nu_2}\omega^\top S_N^{-1} & \frac{1}{\nu_2} \end{bmatrix}. \tag{4.27}$$

Par comparaison entre (4.25) et (4.27), on constate que $K_N'^{-1} = S_N - \left(\omega\omega^\top/\delta_3\right)$ et donc $S_N = K_N'^{-1} + \left(\omega\omega^\top/\delta_3\right)$.
D'autre part, l'équation (4.24) s'écrit

$$\begin{bmatrix} \tilde{\rho}_i^{(t+1)} \\ \tilde{\alpha}_i^{(t+1)}(2:N+1) \\ \tilde{\alpha}_i^{(t+1)}(1) \end{bmatrix} = \begin{bmatrix} S_N & \omega \\ \omega^\top & \delta_3 \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{y}^{(t+1)} \\ y(t-N+1) \end{bmatrix}. \tag{4.28}$$

En remplaçant S_N par $K_N'^{-1} + \left(\omega\omega^\top/\delta_3\right)$ et après quelques simplifications, on a la formule de récurrence sur les coefficients définissant la fonction de régression donnée par

$$\begin{bmatrix} \rho_i^{(t+1)} \\ \alpha_i^{(t+1)} \end{bmatrix} = \begin{bmatrix} \tilde{\rho}_i^{(t+1)} \\ \tilde{\alpha}_i^{(t+1)}(2:N+1) \end{bmatrix} - \omega y(t-N+1) - \frac{\omega\omega^\top}{\delta_3} \begin{bmatrix} 0 \\ \mathbf{y}^{(t+1)} \end{bmatrix}. \tag{4.29}$$

La formule récursive (4.29) permet de mettre à jour la fonction $\tilde{f}_i^{(t+1)}$ après l'élimination de $(y(t-N+1), \varphi(t-N+1))$.

Dans la section suivante, nous présentons un nouvel algorithme qui permet la mise à jour des paramètres définissant les frontières séparatrices des différentes régions dans l'espace de régression.

4.3.3 Adaptation des régions : classifieur incrémental et décrémental multi-classe à vecteurs de support

Dans cette section, nous proposons un classifieur incrémental et décrémental multi-classe à vecteurs de support MID-SVM permettant de mettre à jour les paramètres définissant les frontières séparatrices des $(s > 2)$ régions [16]. Cet algorithme est la version en ligne du classifieur multi-classe présenté dans le chapitre précédent (Section 3.3.3). Ce classifieur dynamique comporte deux procédures d'adaptation : (i) la procédure incrémentale, (ii) la procédure décrémentale. La procédure incrémentale permet d'adapter les paramètres des frontières séparatrices après l'ajout d'une nouvelle donnée. La procédure décrémentale quant à elle, permet de mettre à jour les paramètres des frontières séparatrices après le retrait de l'ancienne donnée. De cette façon, la taille de l'ensemble de données formant chaque région reste fixée à N. Cette mise à jour permettra de suivre l'évolution des régions dans le temps. Notre algorithme fournit des résultats équivalents à ceux de l'approche basée sur la programmation quadratique (QP).
En classification multi-classe, l'ajout ou le retrait d'une observation d'une région donnée peut influencer les paramètres des fonctions séparatrices de toutes les régions. Ainsi, l'objectif ici est de mettre à jour les fonctions $h_i^{(t+1)}$ définissant les différentes régions :

$$\begin{aligned} h_i^{(t+1)}(\varphi(t+1)) = \tfrac{1}{s+1} \sum_{\substack{j=1 \\ j\neq i}}^{s} \Bigg[& \sum_{\varphi(l)\in\Re_i} \alpha_l^{ij} (\eta(\varphi(l))^\top \eta(\varphi(t+1))) \\ & - \sum_{\varphi(l)\in\Re_j} \alpha_l^{ij} (\eta(\varphi(l))^\top \eta(\varphi(t+1))) \Bigg] + b_i. \end{aligned} \tag{4.30}$$

En fait, les coefficients α_l^{ij} et b_i sont fonction du temps. Pour des raisons de simplicité, l'indice temps (t) ne figurera pas sur ces coefficients.

Nous avons vu dans le chapitre précédent que pour déterminer les paramètres α_l^{ij} $i = 1, \ldots, s, j = i+1, \ldots, s$, il a fallu résoudre le problème d'optimisation sous forme duale suivant :

$$\begin{aligned}
\min_{w_i, b_i} \; & \delta \sum_{i=1}^{s} \sum_{j=i+1}^{s} \Bigg\| \sum_{\varphi(l)\in\Re_i} (2\alpha_l^{ij} + \sum_{\substack{k=1 \\ k\neq i,j}}^{s} \alpha_l^{ik})\eta(\varphi(l)) - \sum_{\varphi(l)\in\Re_j} (2\alpha_l^{ij} + \sum_{\substack{k=1 \\ k\neq i,j}}^{s} \alpha_l^{kj})\eta(\varphi(l)) \\
& - \sum_{\substack{k=1 \\ k\neq i,j}}^{s} \sum_{\varphi(l)\in\Re_k} (\alpha_l^{ik} - \alpha_l^{kj})\eta(\varphi(l)) \Bigg\|_2^2 + \delta \sum_{i=1}^{s} \Bigg\| \sum_{\substack{k=1 \\ k\neq i}}^{s} \sum_{\varphi(l)\in\Re_i} \alpha_l^{ik}\eta(\varphi(l)) - \sum_{\varphi(l)\in\Re_k} \alpha_l^{ik}\eta(\varphi(l)) \Bigg\|_2^2 \\
& - \sum_{i=1}^{s} \sum_{j=i+1}^{s} \sum_{\varphi(l)\in\Re_i} \alpha_l^{ij} \\
s.c. \quad & \sum_{\substack{j=1 \\ j\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ij} - \sum_{\varphi(l)\in\Re_j} \alpha_l^{ij} \Big) = 0 \\
& 0 \leq \alpha_l^{ij} \leq C, \quad i = 1, \ldots, s; j = i+1, \ldots, s.
\end{aligned} \tag{4.31}$$

où $\delta = 1/(2(s+1)^2)$.

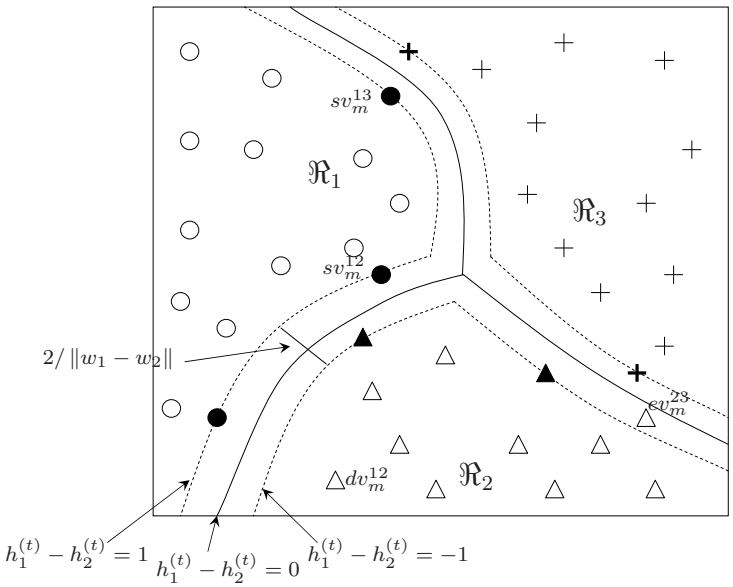


FIGURE 4.2 – Séparateur non linéaire par morceaux de trois régions.

Au lieu de résoudre ce problème d'optimisation à chaque instant t ce qui peut être coûteux en termes de temps de calcul, notre objectif consiste à estimer la variation que peuvent subir les paramètres α_l^{ij} et b_{ij} $i = 1, \ldots, s, j = i+1, \ldots, s$ à chaque instant t après l'incorporation ou l'élimination d'une donnée. De cette façon, la taille de chaque région est N. En partant du problème dual (4.31) et en ajoutant les contraintes dans la fonction objectif avec les coefficients b_i comme étant des multiplicateurs de Lagrange, ce

problème sera équivalent à la minimisation de la fonction objectif suivante :

$$\begin{aligned}\mathbb{W} = \delta \sum_{i=1}^{s}\sum_{j=i+1}^{s} \Bigg\| \sum_{\varphi(l)\in\Re_i} (2\alpha_l^{ij} + \sum_{\substack{k=1\\k\neq i,j}}^{s} \alpha_l^{ik})\eta(\varphi(l)) - \sum_{\varphi(l)\in\Re_j} (2\alpha_l^{ij} + \sum_{\substack{k=1\\k\neq i,j}}^{s} \alpha_l^{kj})\eta(\varphi(l)) \\ - \sum_{\substack{k=1\\k\neq i,j}}^{s} \sum_{\varphi(l)\in\Re_k} (\alpha_l^{ik} - \alpha_l^{kj})\eta(\varphi(l)) \Bigg\|_2^2 + \delta \sum_{i=1}^{s} \Bigg\| \sum_{\substack{k=1\\k\neq i}}^{s} \sum_{\varphi(l)\in\Re_i} \alpha_l^{ik}\eta(\varphi(l)) - \sum_{\varphi(l)\in\Re_k} \alpha_l^{ik}\eta(\varphi(l)) \Bigg\|_2^2 \\ - \sum_{i=1}^{s}\sum_{j=i+1}^{s} \sum_{\varphi(l)\in\Re_i} \alpha_l^{ij} - \sum_{i=1}^{s} b_i \sum_{\substack{k=1\\k\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ik} - \sum_{\varphi(l)\in\Re_k} \alpha_l^{ik} \Big)\end{aligned} \tag{4.32}$$

Pour exprimer les conditions KKT en un point de donnée $\varphi(m) \in \Re_{ij} = \Re_i \cup \Re_j$ pour tout $i = 1, \ldots, s$, $j = i+1, \ldots, s$, nous calculons le gradient suivant

$$\begin{aligned} g_m^{ij} = \frac{\partial \mathbb{W}}{\partial \alpha_m^{ij}} = \frac{y_m^{ij}}{(s+1)} \Bigg[& \sum_{\varphi(l)\in\Re_i} (2\alpha_l^{ij} + \sum_{\substack{k=1\\k\neq i,j}}^{s} \alpha_l^{ik})\kappa(\varphi(l),\varphi(m)) \\ & - \sum_{\varphi(l)\in\Re_j} (2\alpha_l^{ij} + \sum_{\substack{k=1\\k\neq i,j}}^{s} \alpha_l^{kj})\kappa(\varphi(l),\varphi(m)) \\ & - \sum_{\substack{k=1\\k\neq i,j}}^{s} \sum_{\varphi(l)\in\Re_k} (\alpha_l^{ik} - \alpha_l^{kj})\kappa(\varphi(l),\varphi(m)) + (b_i - b_j) \Bigg] - 1 \end{aligned} \tag{4.33}$$

$$\frac{\partial \mathbb{W}}{\partial b_i} = \sum_{\substack{k=1\\k\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \alpha_l^{ik} - \sum_{\varphi(l)\in\Re_k} \alpha_l^{ik} \Big) \tag{4.34}$$

où $\kappa(\varphi(l),\varphi(m)) = \eta(\varphi(l))^\top \eta(\varphi(m))$, $y_m^{ij} = 1$ si $\varphi(m) \in \Re_i$ et $y_m^{ij} = -1$ si $\varphi(m) \in \Re_j$.

Les conditions KKT divisent les données $\varphi(m) \in \Re_{ij}$ en trois catégories [72] (voir figure 4.2 et 4.3) en fonction de la valeur de g_m^{ij} :

$$g_m^{ij} \begin{cases} = 0, & \text{si} \quad 0 < \alpha_m^{ij} < C \\ > 0, & \text{si} \quad \alpha_m^{ij} = 0 \\ < 0, & \text{si} \quad \alpha_m^{ij} = C \end{cases} \tag{4.35}$$

qui sont :

- Les vecteurs de support sv_m^{ij} situés sur les frontières séparatrices $h_i^{(t)} - h_j^{(t)} = \pm 1$, pour lesquels $g_m^{ij} = 0$.
- Les vecteurs erreurs ev_m^{ij} situés à l'intérieur de la marge (c'est-à-dire entre les frontières $h_i^{(t)} - h_j^{(t)} = 1$ et $h_i^{(t)} - h_j^{(t)} = -1$), pour lesquels $g_m^{ij} < 0$. Ces vecteurs ne sont pas nécessairement mal classés.

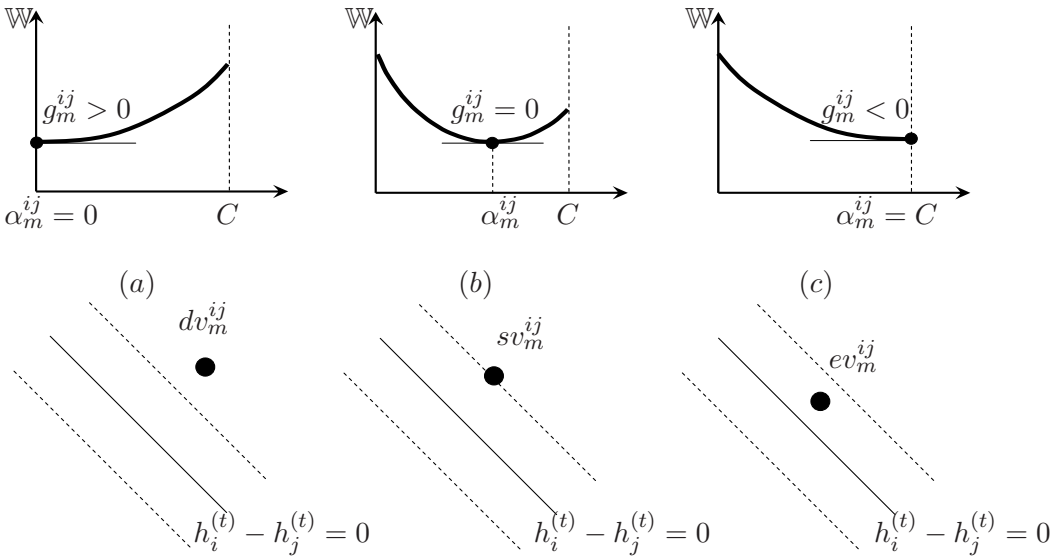


FIGURE 4.3 – Illustration des différents cas de figure de la fonction g_m^{ij}.

– Les vecteurs dv_m^{ij} situés à l'intérieur des frontières $h_i^{(t)} - h_j^{(t)} = \pm 1$, pour lesquels $g_m^{ij} > 0$.

Incrémentation adiabatique :

L'incrémentation adiabatique [24] est basée sur l'idée suivante : quand une donnée candidate $\varphi(c)$ est ajoutée à une région $\Re_p$, son coefficient $\alpha_c^{pq}, p \in \{1, \ldots, s\}, q \in \{p+1, \ldots, s\}$, initialement égale à 0, sera incrémenté jusqu'à une certaine valeur telle que les conditions KKT ne soient pas violées. Ces conditions peuvent être exprimées de façon différentielle par

pour tout $i = 1, \ldots, s\,;\, j = i+1, \ldots, s$

$$\Delta g_m^{ij} = y_{ij}^m \Big(\beta_c^{ij,pq} \Delta\alpha_c^{pq} \kappa\left(\varphi(c), \varphi(m)\right) + \sum_{\varphi(l)\in\Re_i} (2\Delta\alpha_l^{ij} + \sum_{\substack{k=1 \\ k\neq i,j}}^{s} \Delta\alpha_l^{ik}) \kappa\left(\varphi(l), \varphi(m)\right)$$
$$- \sum_{\varphi(l)\in\Re_j} (2\Delta\alpha_l^{ij} + \sum_{\substack{k=1 \\ k\neq i,j}}^{s} \Delta\alpha_l^{kj}) \kappa\left(\varphi(l), \varphi(m)\right) - \sum_{\substack{k=1 \\ k\neq i,j}}^{s} \sum_{\varphi(l)\in\Re_k} (\Delta\alpha_l^{ik} - \Delta\alpha_l^{kj}) \kappa\left(\varphi(l), \varphi(m)\right)$$
$$+ \Delta b_i - \Delta b_j \Big) \tag{4.36}$$

et pour tout $i = 1, \ldots, s$

$$\gamma_c^{i,pq} \Delta\alpha_c^{pq} + \sum_{\substack{k=1 \\ k\neq i}}^{s} \Big(\sum_{\varphi(l)\in\Re_i} \Delta\alpha_l^{ik} - \sum_{\varphi(l)\in\Re_j} \Delta\alpha_l^{ik} \Big) = 0 \tag{4.37}$$

où $\beta_c^{ij,pq}$ et $\gamma_c^{i,pq}$ sont définis par

$$\beta_c^{ij,pq} = \begin{cases} 2 & \texttt{si } (p,q) = (i,j) \text{ et } \varphi(c) \in \Re_i \\ -2 & \texttt{si } (p,q) = (i,j) \text{ et } \varphi(c) \in \Re_j \\ 1 & \texttt{si } p = i, q \neq j \text{ et } \varphi(c) \in \Re_i \quad \text{ou} \quad p \neq i, q = j \text{ et } \varphi(c) \notin \Re_{ij} \\ -1 & \texttt{si } p \neq i, q = j \text{ et } \varphi(c) \in \Re_j \quad \text{ou} \quad p = i, q \neq j \text{ et } \varphi(c) \notin \Re_{ij} \\ 0 & \text{sinon} \end{cases}$$

$$\gamma_c^{i,pq} = \begin{cases} 1 & \texttt{si } \varphi(c) \in \Re_i \\ -1 & \texttt{si } \varphi(c) \notin \Re_i \end{cases}.$$

Le paramètre $\beta_c^{ij,pq}$ est le coefficient qui doit être multiplié à $\Delta\alpha_c^{pq}$. Ce paramètre a été déduit de l'équation (4.33). Il suffit de voir par exemple que le coefficient qui doit être multiplié à $\Delta\alpha_l^{ij}$ (ie. $(p,q) = (i,j)$), dans cette équation, est égal à 2 si $\varphi(l) \in \Re_i$ et il est égal à -2 si $\varphi(l) \in \Re_j$. Le paramètre $\gamma_c^{i,pq}$ est déduit de la même façon de l'équation (4.34).

Afin d'assurer l'unicité de la solution optimale, on ajoute aux deux équations (4.36) et (4.37) une contrainte supplémentaire qui est la contrainte appelée "la somme à zéro" : $\sum_{k=1}^{s} h_k(\varphi) = \sum_{k=1}^{s} (w_k^\top \eta(\varphi) + b_k) = 0$. Cependant on sait déjà que $\sum_{k=1}^{s} w_k = 0$ (cf. Remarque 3.1). Cela signifie que "la somme à zéro" peut être réduite à la contrainte suivante : $\sum_{k=1}^{s} b_k = 0$ qui peut être exprimée de façon différentielle par

$$\sum\nolimits_{k=1}^{s} \Delta b_k = 0 \tag{4.38}$$

Soit $b = [b_1, \ldots, b_s]^\top$ et $a^{ij} = \left[a_i^{ij}, a_j^{ij}\right], i < j$, tel que a_x^{ij} est un vecteur contenant les poids des vecteurs de support sv_m^{ij} appartenant à la région $\Re_x$. Définissons $a = [a_1^{12}, a_2^{12}, \ldots, a_1^{1s},\ a_s^{1s}, a_2^{23}, \ldots a_i^{ij}, a_j^{ij}, \ldots, a_{s-1}^{(s-1)s}, a_s^{(s-1)s}\Big]$.

A partir de l'équation (4.35), on a pour tous les vecteurs de support $sv_m^{ij}; m = 1, \ldots, s_{ij};, i = 1, \ldots s; j = i+1, \ldots, s$: $g_m^{ij}(sv_m^{ij}) = 0$ et donc $\Delta g_m^{ij}(sv_m^{ij}) = 0$ où s_{ij} est le nombre de vecteurs de support dans $h_i^{(t)} - h_j^{(t)}$. Par conséquent, les équations (4.36), (4.37) et (4.38) peuvent s'écrire sous la forme matricielle suivante :

$$-\begin{bmatrix} 0 \\ \Gamma \\ \mathrm{B} \end{bmatrix} \Delta\alpha_c^{pq} = \left[\begin{array}{c|c} \mathbf{1}_s^\top & 0 \\ 0 & \mathscr{L} \\ \mathscr{J} & \mathscr{K} \end{array}\right] \begin{bmatrix} \Delta b \\ \Delta a \end{bmatrix} \tag{4.39}$$

avec $\mathbf{1}_s = [1, \ldots, 1]^\top \in \mathbb{R}^s$.
Les expressions des vecteurs Γ, B et des matrices $\mathscr{L}$, $\mathscr{J}$ et $\mathscr{K}$ sont données dans l'Annexe A2.

La solution de l'équation (4.39) est alors donnée par :

$$\begin{bmatrix} \Delta b \\ \Delta a \end{bmatrix} = - \underbrace{\begin{bmatrix} \mathbf{1}_s^\top & | & 0 \\ 0 & | & \mathscr{L} \\ \mathscr{J} & | & \mathscr{K} \end{bmatrix}^\dagger}_{R} \underbrace{\begin{bmatrix} 0 \\ \Gamma \\ \mathrm{B} \end{bmatrix}}_{H^{pq}} \Delta\alpha_c^{pq} \tag{4.40}$$

où le symbole † fait référence à la pseudo-inverse.

La procédure incrémentale

Affecter $\varphi(c)$ à $\Re_p$, $p = \hat{\sigma}(t+1)$
Fixer $step > 0$
Pour $q = 1, \ldots, s,\ q \neq p$
- $\alpha_c^{pq} \leftarrow 0$;
- $z = [b, a]$;
- Calculer H^{pq} (Eq. (4.40)) ;
- **Tant que** ($g_c^{pq} < 0$ et $\alpha_c^{pq} < C$)
  Faire :
  - $\alpha_c^{pq} = \alpha_c^{pq} + step$;
  - $z = z - R \cdot H^{pq} \cdot step$;
  - $g_l^{ij} = g_l^{ij} + \Delta g_l^{ij},\ i = 1, \ldots, s,\ j = i+1, \ldots, s$,
  - Trouver parmi les α_m^{ij} ceux qui sont nuls. Les vecteurs de support correspondant migrent vers l'ensemble $\{dv_m^{ij}\}$. Adapter R, z et H^{pq}.

  Fin de Tant que
- Si $\alpha_c^{pq} = C$, alors $\varphi(c)$ est ajouté à $\Re_p$ comme étant un vecteur erreur, sinon c'est un vecteur de support sv_c^{pq}.

Fin de pour

TABLE 4.1 – La procédure incrémentale.

La procédure incrémentale :
L'incorporation d'une nouvelle donnée $\varphi(c)$ passe par la procédure incrémentale (Table 4.1). Si pour tout $p = 1, \ldots, s$, $q = p+1, \ldots, s$: $g_c^{pq} > 0$, alors on ajoute $\varphi(c)$ à la région $\Re_p$ sans adaptation des frontières des régions. Sinon si $g_c^{pq} < 0$, on incrémente le coefficient α_c^{pq} correspondant à $\varphi(c)$: $\alpha_c^{pq} = \alpha_c^{pq} + step$ ($step$ est un paramètre très inférieur à C), jusqu'à ce que l'un des deux cas soit satisfait : $g_c^{pq} = 0$ ou $\alpha_c^{pq} = C$. A chaque itération les coefficients (b, a) sont mis à jour par l'équation (4.40). Cela peut entraîner une migration des données entre les ensembles $\{sv_m^{ij}\}$, $\{ev_m^{ij}\}$ et $\{dv_m^{ij}\}$. Dans de telles situations, la matrice R (voir équation (4.40)) est mise à jour. Si un des vecteurs de support s'éloigne de la

La procédure décrémentale

Retirer $\varphi(o)$ de $\Re_p$, $p \in \{1, \ldots, s\}$
Fixer $step > 0$
Pour $q = 1, \ldots, s,\ q \neq p$
– $\alpha_o^{pq} \leftarrow$ poids de φ(o) ;
– $z = [b, a]$;
– Calculer H^{pq} (Eq. (4.40)) ;
– **Tant que** ($\alpha_o^{pq} > 0$ et $g_o^{pq} > -1$)
Faire :
– $\alpha_c^{pq} = \alpha_c^{pq} - step$;
– $z = z + R \cdot H^{pq} \cdot step$;
– $g_l^{ij} = g_l^{ij} + \Delta g_l^{ij},\ i = 1, \ldots, s,\ j = i + 1, \ldots, s,$
– Trouver parmi les vecteurs dv_l^{ij} ceux pour lesquels ($g_l^{ij} < 0$). Ces données sont candidates pour devenir vecteurs de support :
- Interrompre la procédure décrémentale et appliquer la procédure incrémentale sur ces données ;
- Retourner à la procédure décrémentale.
Fin de Tant que
Fin de pour
Si $\forall q,\ \alpha_o^{pq} = 0$, alors retirer $\varphi(o)$ de la base ; sinon $\varphi(o)$ est un vecteur erreur par défaut.

TABLE 4.2 – La procédure décrémentale.

frontière séparatrice, la ligne et la colonne correspondantes de la matrice R sont éliminées. Enfin, à l'arrêt de la procédure itérative, si $\alpha_c^{pq} = C$ alors $\varphi(c)$ est ajouté à l'ensemble de données comme étant un vecteur erreur, sinon, il est ajouté en tant que vecteur de support.

La procédure décrémentale :
La procédure incrémentale est réversible. Ainsi la procédure décrémentale (Table 4.2) est utilisée si une donnée $\varphi(o)$ est éliminée d'une région $\Re_p$ de façon à ce que $\Re_p$ reste de taille N. Si la donnée $\varphi(o)$ n'est ni vecteur de support, ni vecteur erreur, alors elle peut être retirée sans aucune nécessité de mise à jour des frontières des régions. Sinon, son poids correspondant α_o^{pq} sera décrémenté : $\alpha_o^{pq} = \alpha_o^{pq} - step$ jusqu'à ce que $\alpha_o^{pq} = 0$ ou $g_o^{pq} < -1$. A chaque itération les coefficients (b, a) sont mis à jour par l'équation (4.40). L'application de cette procédure peut causer des migrations de données de l'ensemble $\{dv_m^{ij}\}$ à l'ensemble $\{sv_m^{ij}\}$. S'il y a des données dv_l^{ij} pour lesquelles ($g_l^{ij} < 0$), la procédure

décrémentale doit être interrompue et la procédure incrémentale doit être appliquée à ces données jusqu'à ce qu'elles deviennent des vecteurs de support. Puis, la procédure décrémentale est reprise. Enfin, si $\forall q$, $\alpha_o^{pq} = 0$ alors $\varphi(o)$ est retirée de la base d'apprentissage, sinon si $\exists q$, $g_o^{pq} < -1$, alors $\varphi(o)$ est un vecteur erreur par défaut. Ce vecteur ne peut donc pas être retiré de la région $\Re_p$ et il sera ignoré.

Pour résumer, les procédures incrémentale et décrémentale sont présentées respectivement dans les tables 4.1 et4.2.

Exemple numérique :
Pour illustrer l'algorithme de mise à jour des frontières séparatrices, nous considérons un exemple de simulation de trois régions dans $\mathbb{R}^2$ avec des données non linéairement séparables. Les régions sont définies comme suit : la région 1 est composée d'un mélange de trois distributions gaussiennes de moyenne $(-1,-1)$, $(1,0)$ et $(3,-0.5)$ et de covariance $0.2 \times I_2$. Les données de la région 2 sont générées à partir d'une distribution gaussienne avec une moyenne de $(1,-2)$ et une covariance de $0,2 \times I_2$. Pour la région 3, un mélange de quatre distributions gaussiennes de moyenne $(1.5,-4.5)$, $(3,-3)$, $(-1,-3.5)$ et $(0-4)$ et de covariance $0.3 \times I_2$ a été créé. Un ensemble de données de taille $N = 300$ a été généré à partir des distributions spécifiées avec 100 échantillons pour chaque région. La procédure d'adaptation de frontières séparatrices a été appliquée en choisissant un noyau gaussien $\kappa(\varphi(i), \varphi(j)) = \exp((-1/\sigma^2) \cdot \|\varphi(i) - \varphi(j)\|^2)$. Ici, le paramètre σ du noyau a été fixé à $1,5$, *step* à 0.005 et C à 200. Les données sont acquises en ligne de façon continue, à raison d'une donnée à la fois.

La figure 4.4 montre le nuage des points des trois régions, ainsi que les frontières séparatrices pour différents instants. Le nombre final des vecteurs de support est de 10, 11 et 9 pour les trois fonctions de décision. A des fins de comparaison, nous avons également appliqué un algorithme hors ligne qui utilise l'approche QP pour minimiser la fonction objectif (4.31) sur le même ensemble de données. Le résultat est exactement le même que celui de la méthode récursive proposée (les mêmes vecteurs de support et les mêmes poids correspondants).

4.4 Exemples numériques

Exemple 4.1. Approximation récursive de la fonction sinus
Nous considérons dans cet exemple l'approximation récursive de la fonction sinus $y(t) = sin(\varphi(t))$ utilisée dans l'exemple 2.1 où $\varphi(t) = 2\pi t/125$. L'algorithme 4.1 a été appliqué sur 125 données générées par cette fonction avec $\tau = 10$ et $c = 10$. Le paramètre N est

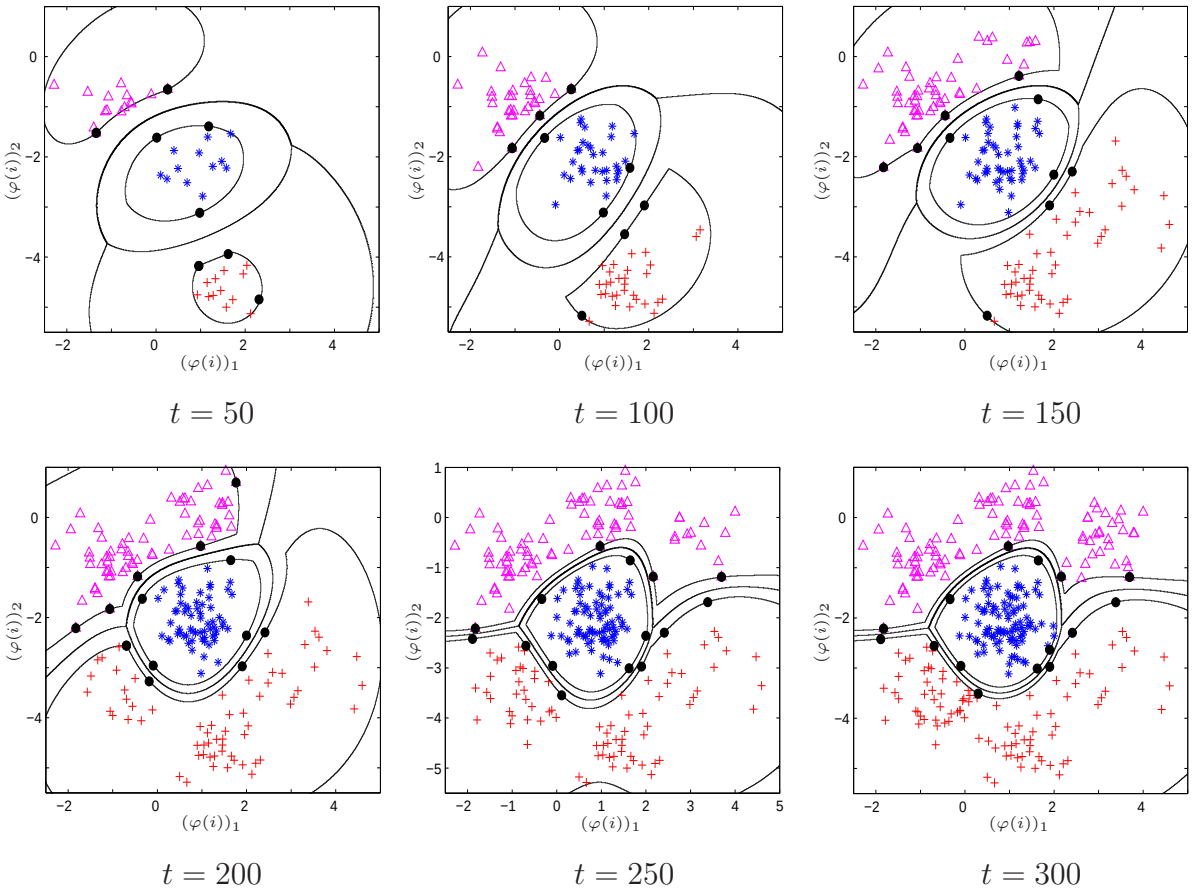


FIGURE 4.4 – Frontières séparatrices de trois régions, estimées à différents instants (Région 1 : triangles, Région 2 : étoiles, et Région 3 : croix).

choisi égal au nombre de données car les modes constituant le modèle de la fonction sinus ne varient pas dans le temps.

θ_{seuil}	Les vecteurs de paramètres estimés finaux
4	$\hat{\theta}_1 = \begin{bmatrix} -0.636 & -2.132 \end{bmatrix}$, $\hat{\theta}_2 = \begin{bmatrix} 0.798 & 0.005 \end{bmatrix}$, $\hat{\theta}_3 = \begin{bmatrix} -0.607 & 2.056 \end{bmatrix}$,
8	$\hat{\theta}_1 = \begin{bmatrix} -0.782 & -2.519 \end{bmatrix}$, $\hat{\theta}_2 = \begin{bmatrix} 0.149 & -0.735 \end{bmatrix}$, $\hat{\theta}_3 = \begin{bmatrix} 0.900 & -0.005 \end{bmatrix}$, $\hat{\theta}_4 = \begin{bmatrix} -0.012 & 0.978 \end{bmatrix}$, $\hat{\theta}_5 = \begin{bmatrix} -0.837 & 2.670 \end{bmatrix}$

TABLE 4.3 – Les vecteurs de paramètres estimés de la fonction sinus par l'Algorithme 4.1.

Pour voir l'influence du paramètre θ_{seuil} sur le nombre de sous-modèles identifiés, nous

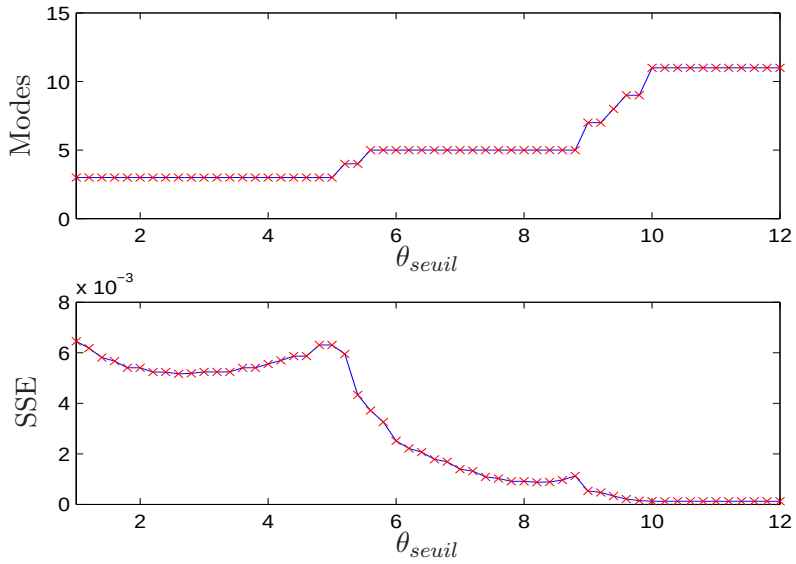
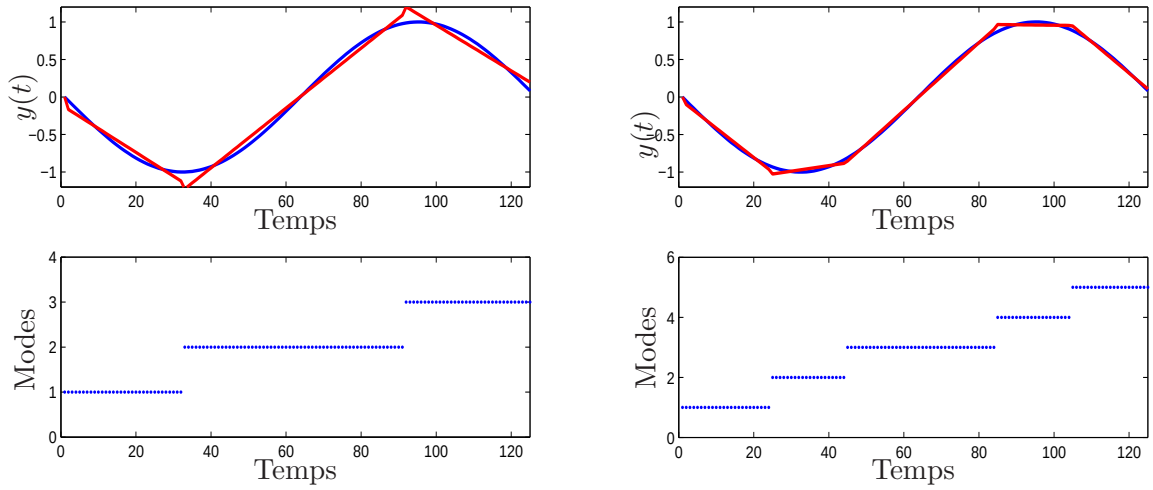


FIGURE 4.5 – Influence du paramètre θ_{seuil} sur l'approximation de la fonction sinus (Haut) Nombre de sous-modèles en fonction de θ_{seuil}. (Bas) La moyenne des erreurs quadratiques SSE en fonction de θ_{seuil}.



$\theta_{seuil} = 4$ $\theta_{seuil} = 8$

FIGURE 4.6 – Approximation récursive de la fonction sinus par des modèles affines par morceaux par l'algorithme 4.1

appliquons l'algorithme 4.1 sur le même jeu de données avec différentes valeurs de θ_{seuil}. Le paramètre θ_{seuil} balaye l'intervalle [1,12] avec un pas égal à 0.2. Pour chaque valeur de θ_{seuil}, nous calculons le nombre de sous-modèles et la moyenne des erreurs quadratiques SSE. Nous rapportons le résultat trouvé sur la figure 4.5. D'après cette figure, nous

constatons que le paramètre θ_{seuil} influe sur le nombre de sous-modèles identifiés et donc la précision du modèle. Plus θ_{seuil} augmente, plus le nombre de sous-modèles est grand et plus le SSE est petit. Nous constatons également que le nombre de sous-modèles identifiés reste égal à 3 sur tout l'intervalle [1,5] de θ_{seuil} et reste égal à 5 sur tout l'intervalle [5.5,9]. Nous représentons sur la figure 4.6 la fonction sinusoïdale ainsi que les modèles affines récursivement estimés et leurs domaines de validité pour $\theta_{seuil} = 4$ et pour $\theta_{seuil} = 8$. Les domaines de validité sont : Mode 1 $[0, \pi/2[$, Mode 2 $[\pi/2, 3\pi/2[$, Mode 3 $[3\pi/2, 2\pi]$ pour $\theta_{seuil} = 4$ et Mode 1 $[0, 2\pi/5[$, Mode 2 $[2\pi/5, 0.72\pi[$, Mode 3 $[0.72\pi, 1.35\pi[$, Mode 4 $[1.35\pi, 1.68\pi[$, Mode 5 $[1.68\pi, 2\pi]$ pour $\theta_{seuil} = 8$.

Exemple 4.2. Identification récursive d'un modèle PWARX

A titre illustratif, nous considérons maintenant un modèle affine par morceaux SISO dont les paramètres des sous-modèles ne varient pas dans le temps. Ce modèle est composé de trois sous-modèles ($s = 3$) d'ordre un ($n_a = n_b = 1$) définis par les équations suivantes :

$$y(t) = \begin{cases} \begin{bmatrix} 0.4 & 0.5 & 0.3 \end{bmatrix} \bar{\varphi}(t) + \varepsilon(t) & \text{si } \varphi(t) \in \Re_1, \\ \begin{bmatrix} -0.7 & 0.6 & -0.5 \end{bmatrix} \bar{\varphi}(t) + \varepsilon(t) & \text{si } \varphi(t) \in \Re_2, \\ \begin{bmatrix} 0.4 & -0.2 & -0.2 \end{bmatrix} \bar{\varphi}(t) + \varepsilon(t) & \text{si } \varphi(t) \in \Re_3, \end{cases} \tag{4.41}$$

où $\bar{\varphi} = \begin{bmatrix} \varphi^\top & 1 \end{bmatrix}^\top$ tel que $\varphi(t) = [y(t-1)\ u(t-1)]^\top$
et

$$\begin{aligned} \Re_1 &= \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} -.3 & 2 & -1 \end{bmatrix} \bar{\varphi} \geq 0 \right\} \\ \Re_2 &= \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} -.3 & 2 & -1 \end{bmatrix} \bar{\varphi} < 0 \text{ et } \begin{bmatrix} 0.3 & 2 & 1 \end{bmatrix} \bar{\varphi} > 0 \right\} \\ \Re_3 &= \left\{ \varphi \in \mathbb{R}^2 : \begin{bmatrix} 0.3 & 2 & 1 \end{bmatrix} \bar{\varphi} \leq 0 \right\} \end{aligned} \tag{4.42}$$

L'entrée d'excitation $u(t)$ est représentée sur la figure 4.7. Cette entrée a été générée de sorte que la loi de commutation du modèle (4.41) respecte (4.42) avec un temps minimum réel de séjour dans chaque mode $\tau_{sej} = 12$. Le bruit $\varepsilon(t)$ a été généré selon une loi normale de moyenne nulle et de variance 0.04. Un ensemble de 1000 données a été généré suivant le modèle (4.41).

L'algorithme 4.1 a été appliqué à ce jeu de données en choisissant $\theta_{seuil} = 1$, $\tau = 10$ et $c = 10$. Comme dans l'exemple précédent, puisque les paramètres de chaque mode ne varient pas dans le temps, le paramètre N a été pris égal au nombre de données.

Nous superposons sur la figure 4.8, la sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié. La séquence des modes réels et estimés est également tracée. La sortie reconstruite est presque identique à la sortie réelle avec une

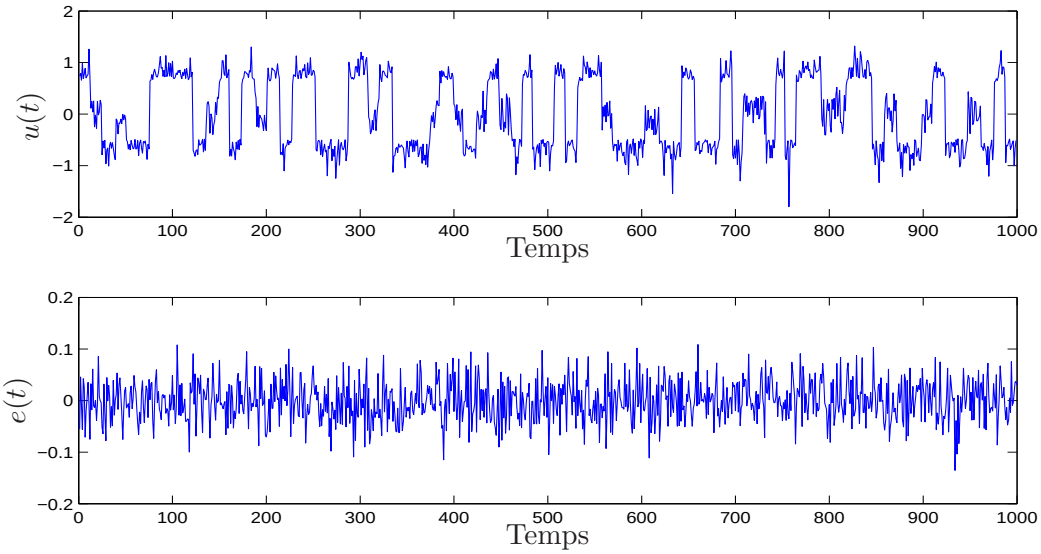


FIGURE 4.7 – (haut) L'entrée d'excitation $u(t)$, (bas) le bruit $\varepsilon(t)$.

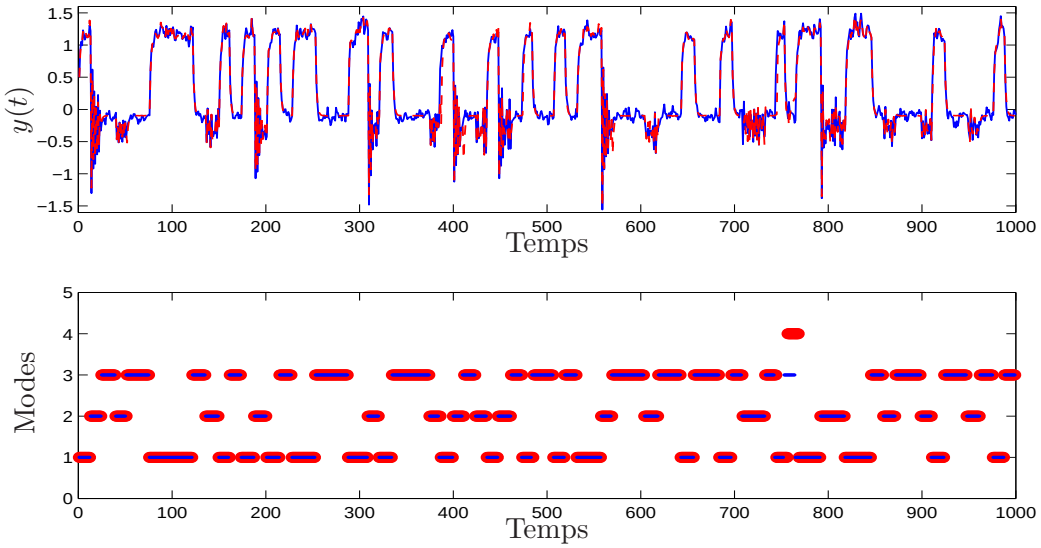


FIGURE 4.8 – (haut) La sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié. (bas) La séquence des modes réels et estimés.

moyenne des erreurs quadratiques SSE= 0.0085. La séquence des modes estimés coïncide presque parfaitement avec la séquence réelle. Nous enregistrons néanmoins une apparition d'un quatrième mode à l'instant $t = 750$ sachant que le nombre réel de sous-modèles est 3. Ceci est dû à l'existence d'un petit groupe de données géométriquement isolées de l'ensemble de données.

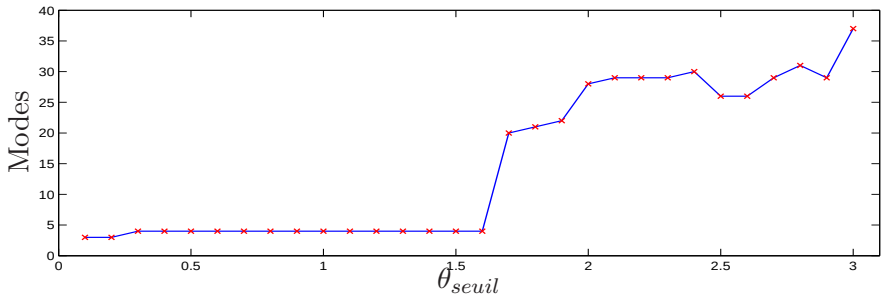


FIGURE 4.9 – Influence du paramètre θ_{seuil} sur le nombre de sous-modèles estimés.

Comme dans l'exemple précédent, pour étudier l'influence du paramètre θ_{seuil} sur le nombre de sous-modèles identifiés, nous faisons varier le paramètre θ_{seuil} dans l'intervalle [0.1,3] avec un pas égal à 0.1. Le résultat est rapporté sur la figure 4.9. Comme prévu, plus le paramètre θ_{seuil} augmente, plus le nombre de sous-modèles est grand. Nous remarquons également que le nombre de sous-modèles est égal à 4 sur tout l'intervalle [0.3, 1.6].

Exemple 4.3. Identification récursive d'un modèle PWARX variant dans le temps

Nous considérons maintenant un modèle affine par morceaux SISO dont les paramètres des sous-modèles varient dans le temps. Il est composé de trois sous-modèles ($s = 3$) d'ordre un ($n_a = n_b = 1$) définis par les équations suivantes :

$$y(t) = \begin{cases} {\theta_1^{(t)}}^\top \bar{\varphi}(t) + \varepsilon(t) \text{ si } \varphi(t) \in \Re_1, \\ {\theta_2^{(t)}}^\top \bar{\varphi}(t) + \varepsilon(t) \text{ si } \varphi(t) \in \Re_2, \\ {\theta_3^{(t)}}^\top \bar{\varphi}(t) + \varepsilon(t) \text{ si } \varphi(t) \in \Re_3, \end{cases} \tag{4.43}$$

où $\varphi(t) = [y(t-1)\ u(t-1)]^\top$.

Les composantes des paramètres $\theta_i^{(t)}$ (voir figure 4.10) varient dans le temps de façon linéaire ou non linéaire comme suit :

$$\begin{aligned} \theta_1^{(t)} &= \theta_1^{(0)} + \left[0 \quad -2 \times 10^{-7} t^2 \quad 10^{-4} t\right], \\ \theta_2^{(t)} &= \theta_2^{(0)} + \left[10^{-6} t \quad -4 \times 10^{-10} t^3 \quad 3 \times 10^{-7} t^2\right], \\ \theta_2^{(t)} &= \theta_2^{(0)} + \left[2 \times 10^{-7} t^2 \quad 0 \quad 3 \times 10^{-4} t\right], \end{aligned}$$

où

$$\theta_1^{(0)} = \begin{bmatrix} 0.4 & 0.5 & 0.3 \end{bmatrix},$$
$$\theta_2^{(0)} = \begin{bmatrix} -0.7 & 0.6 & -0.5 \end{bmatrix},$$
$$\theta_3^{(0)} = \begin{bmatrix} 0.4 & -0.2 & -0.2 \end{bmatrix}.$$

Les régions sont définies par la même loi que dans l'exemple précédent, c'est-à-dire l'équation (4.42).

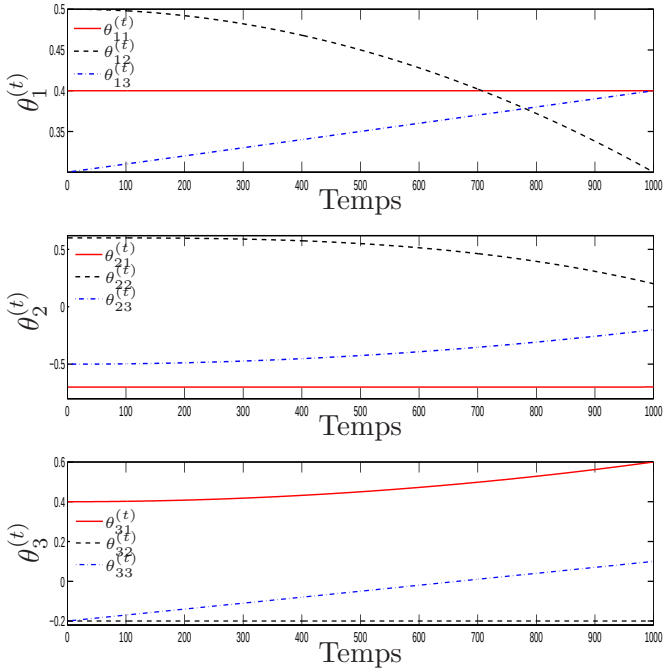


FIGURE 4.10 – Évolution des composantes des paramètres $\theta_i^{(t)},\ i = 1, \ldots, 3$, dans le temps.

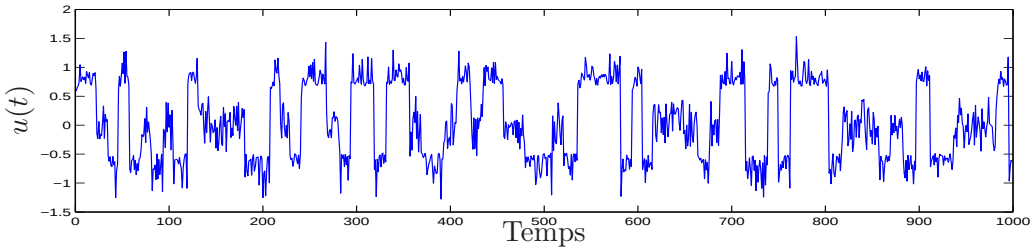


FIGURE 4.11 – L'entrée d'excitation $u(t)$.

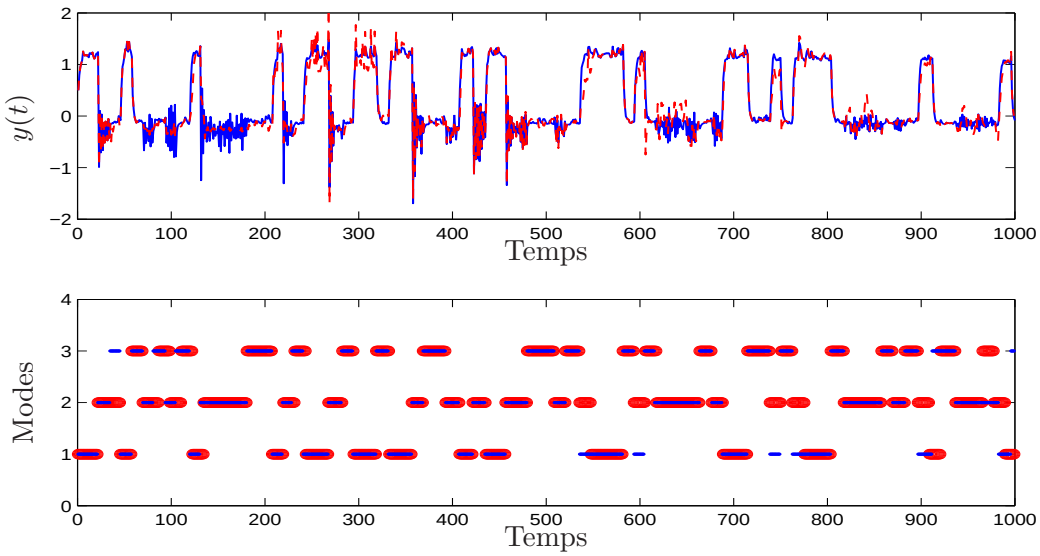


FIGURE 4.12 – (haut) La sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié. (bas) La séquence des modes réels et estimés.

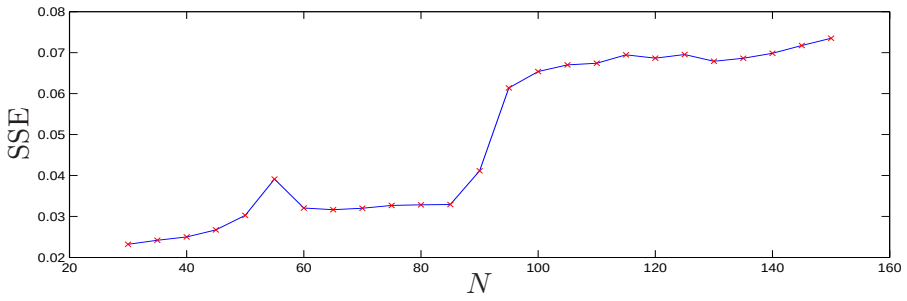


FIGURE 4.13 – Influence du paramètre N sur la précision du modèle estimé.

L'entrée d'excitation $u(t)$ représentée sur la figure 4.11 a été générée de sorte que chaque mode soit actif pendant au minimum un temps de séjour $\tau_{sej} = 12$. Nous générons suivant le modèle (4.43) un ensemble de 1000 données. Nous appliquons à ce jeu de données l'algorithme 4.1 avec $\theta_{seuil} = 1$, $\tau = 10$ et $c = 10$. Étant donné que les paramètres des sous-modèles varient dans le temps, les modes sont récursivement estimés sur une fenêtre de taille N. Pour voir l'influence du paramètre N sur la précision du modèle identifié, nous faisons varier ce paramètre dans l'intervalle [30,150] avec un pas égal à 5. Pour chaque valeur de N, nous calculons la moyenne des erreurs quadratiques SSE.

Sur la figure 4.12, nous avons tracé la sortie reconstruite à partir du modèle identifié superposée sur la sortie mesurée sur le système réel pour le cas $N = 60$. A chaque instant t, la sortie est reconstruite à partir du modèle estimé à partir de l'ensemble des N données

disponibles du mode actif à l'instant t. Malgré cela, la sortie reconstruite est très proche de la sortie réelle. Nous avons également tracé la séquence des modes réels et estimés. La séquence des modes estimés coïncide globalement avec celle des modes réels. Sur la figure 4.13 est représentée la variation de la moyenne des erreurs quadratiques SSE en fonction du paramètre N. Nous constatons que plus N augmente, plus le SSE augmente et donc le modèle devient moins précis. Cela est justifié, puisque la dynamique de chaque mode varie dans le temps. Si N est grand, les données anciennes risquent de ne plus caractériser la dynamique du système à l'instant t. Par conséquent les paramètres $\theta_i^{(t)}$ estimés à chaque instant t à partir de l'ensemble de taille N pour chaque mode peuvent produire une sortie moins précise.

Exemple 4.4. Identification récursive d'un modèle non linéaire par morceaux variant dans le temps

Dans cet exemple, nous considérons un modèle SISO non linéaire par morceaux dont les paramètres des sous-modèles et des régions varient dans le temps. Il composé de deux sous-modèles ($s = 2$) d'ordre un ($n_a = n_b = 1$) définis par les équations suivantes :

$$y(t) = \begin{cases} a_1(t)y(t-1)^2 + a_2(t)u(t-1) & \text{si} \quad \varphi(t) \in \Re_1 \\ b_1(t)y(t-1) - b_2(t)u(t-1)^2 & \text{si} \quad \varphi(t) \in \Re_2 \end{cases} \tag{4.44}$$

où $\varphi(t) = [y(t-1)\ u(t-1)]^\top$.

Les régions $\Re_1$ et $\Re_2$ sont définies par

$$\Re_1 = \left\{\varphi \in \mathbb{R}^2 : c_1(t)y(t-1)^3 + c_2(t)u(t-1) \geq 0\right\}$$
$$\Re_2 = \left\{\varphi \in \mathbb{R}^2 : c_1(t)y(t-1)^3 + c_2(t)u(t-1) < 0\right\},$$

L'évolution des paramètres $a_1(t)$, $a_2(t)$, $b_1(t)$, $b_2(t)$, $c_1(t)$ et $c_2(t)$ dans le temps (voir figure 4.14) est soit linéaire ou non linéaire et elle est donnée par les équations suivantes :

$$\begin{cases} a_1(t) = 0.3 + 5 \times 10^{-4}t \\ a_2(t) = 0.2 \end{cases}, \begin{cases} b_1(t) = 1 - 3 \times 10^{-6}t^2 \\ b_2(t) = -0.1 + 3 \times 10^{-4}t \end{cases} \text{et} \begin{cases} c_1(t) = -1 - 6 \times 10^{-3}t \\ c_2(t) = 2. \end{cases}$$

L'entrée d'excitation $u(t)$ est représentée sur la figure 4.15. Chaque mode doit être actif au minimum sur un temps $\tau_{sej} = 20$. Un ensemble de $N = 800$ données a été généré suivant le modèle (4.44).

Nous appliquons l'algorithme 4.1 en choisissant $\theta_{seuil} = 0.1$, $\tau = 15$, $c = 15$ et $N = 60$. Nous montrons sur la figure 4.16, la sortie reconstruite à partir du modèle identifié

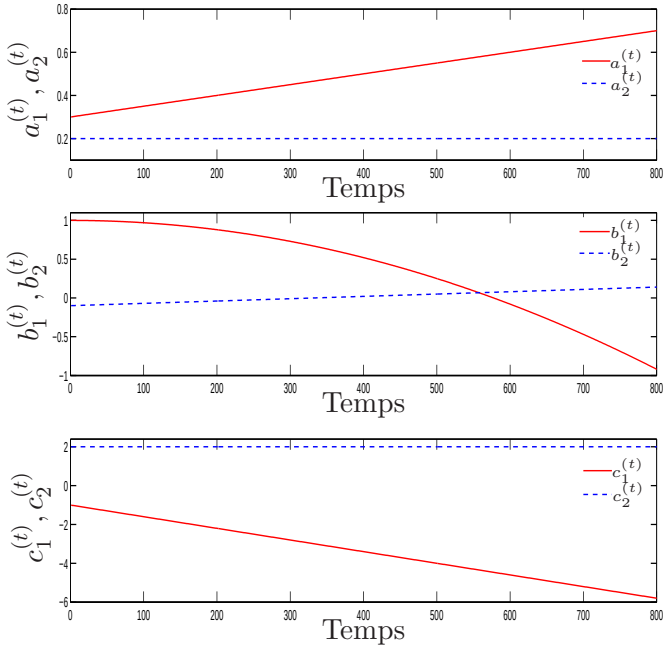


FIGURE 4.14 – Évolution des paramètres $a_1(t)$, $a_2(t)$, $b_1(t)$, $b_2(t)$, $c_1(t)$ et $c_2(t)$ dans le temps.

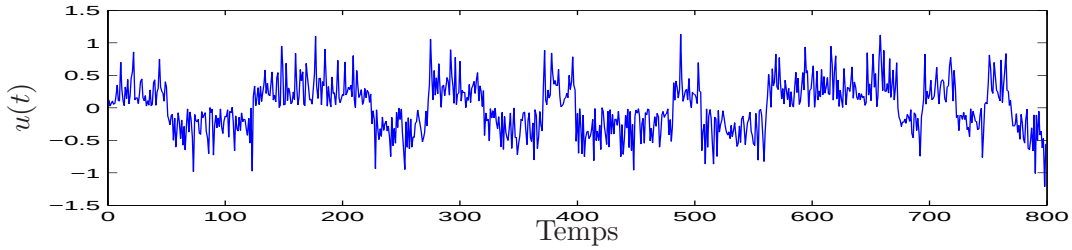


FIGURE 4.15 – L'entrée d'excitation $u(t)$.

superposée sur la sortie mesurée sur le système réel ainsi que la séquence des modes réels et estimés. Comme dans l'exemple précédent, à chaque instant t, la sortie est reconstruite à partir du modèle $f_i^{(t)}$ estimé à partir de l'ensemble des N données disponibles du mode actif à l'instant t. Un noyau polynomial de degré 2 est utilisé dans la procédure d'estimation récursive de $f_i^{(t)}$ par LS-SVMR. La moyenne des erreurs quadratiques calculée entre la sortie réelle et reconstruite est SSE$= 1.52 \times 10^{-4}$. Cette valeur est très faible malgré une différence entre la séquence des modes estimés et celle des modes réels qui peut être remarquée sur l'intervalle [250,290]. A titre illustratif, nous représentons sur la figure 4.17, les frontières séparatrices des deux régions pour différents instants.

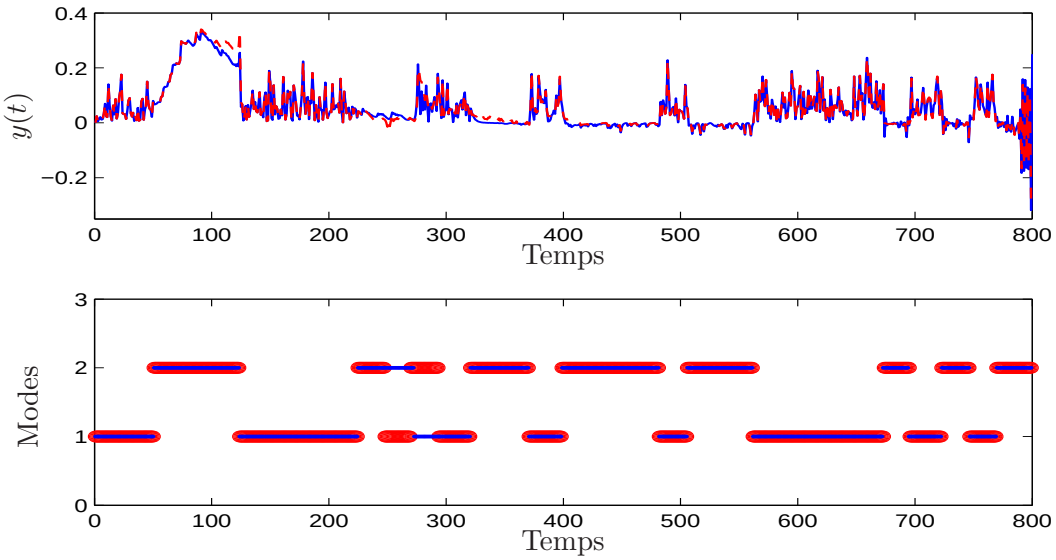


FIGURE 4.16 – (haut) La sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié. (bas) La séquence des modes réels et estimés..

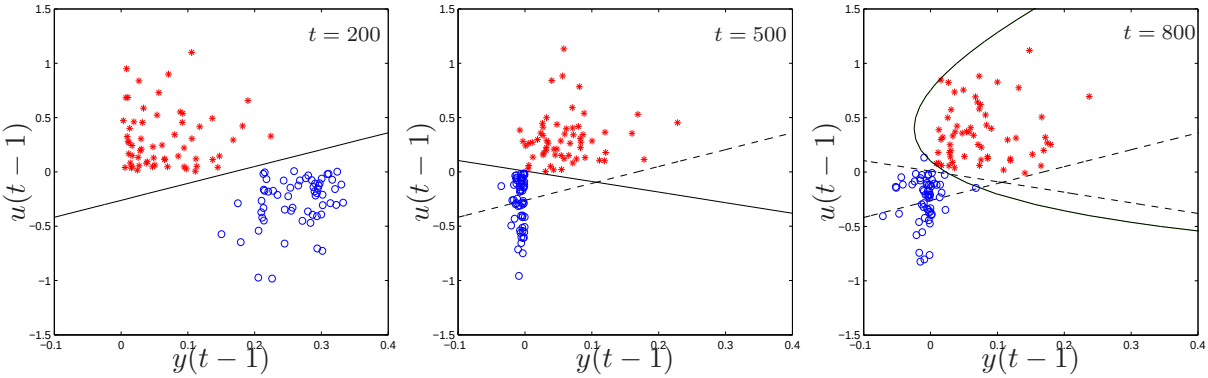


FIGURE 4.17 – Frontières séparatrices des deux régions, estimées à différents instants : $t = 200$, $t = 500$ et $t = 800$ (Région 1 : étoiles, Région 2 : cercles).

4.5 Conclusion

Nous venons de présenter, dans ce chapitre, un algorithme d'identification récursive de modèles dynamique affines et de modèles dynamiques non linéaires par morceaux dont les paramètres varient dans le temps. L'hypothèse principale de travail est que les commutations sont séparées par un temps de séjour minimum. Notre algorithme permet d'estimer récursivement les différents modes après l'acquisition des données, une à la fois. L'affectation des données aux différents modes est effectuée grâce à un critère d'appartenance

spécifique. Cet algorithme est doté de procédures d'adaptation qui permettent de mettre à jour les paramètres des différents modes. Cette adaptation concerne la mise à jour des paramètres des différents sous-modèles $f_i^{(t)}$ et la mise à jour des fonctions $h_i^{(t)}$ définissant les différentes régions. L'adaptation des $f_i^{(t)}$ est assurée par un algorithme des moindres carrés récursifs à fenêtre glissante si les sous-modèles sont affines ou par une procédure de régression par les LS-SVM récursifs si les sous-modèles sont non linéaires. La mise à jour des frontières séparatrices est effectuée par un classifieur incrémental et décrémental multi-classe à vecteurs de support. Enfin, les exemples de simulation présentés ont permis de montrer les performances et la faisabilité de notre algorithme et d'étudier l'influence de ses paramètres sur l'identification récursive.

5

Validation expérimentale

Sommaire

5.1 Introduction

Le dernier chapitre est consacré à la validation expérimentale des méthodes développées. L'objectif est d'évaluer les performances et la faisabilité de nos algorithmes. Nous montrons, à travers ce chapitre, que nos approches peuvent être appliquées à un large domaine d'applications via la proposition de plusieurs types d'exemples réels. Nous étudions et nous montrons l'influence des paramètres utilisés par nos procédures sur la qualité de la modélisation et sur la complexité du calcul.

Tout d'abord, nos approches sont appliquées à la modélisation de systèmes hydrauliques à surface libre. Ces systèmes sont caractérisés par des dynamiques non linéaires et sont soumis à de larges plages de fonctionnement. Ils sont modélisés par des modèles dynamiques affines par morceaux. Chaque modèle dynamique correspond à une plage de fonctionnement. La modélisation par les PWA sera comparée à l'approche de modélisation basée sur les équations physiques. L'influence du paramètre c sur la précision des modèles identifiés, sur le nombre de sous-modèles et sur la complexité du calcul est ensuite étudiée.

Ensuite, une machine de montage de composants électroniques sur circuit imprimé est modélisée par des modèles dynamiques affines par morceaux. Nous identifions les deux modes principaux de la machine, et nous modélisons également le comportement de la machine aux frontières de ces deux modes. Nous étudions, ensuite, l'influence du paramètre c sur la qualité de la modélisation de la machine ainsi que sur le nombre de sous-modèles et sur la complexité du calcul. La machine de montage est également modélisée par des modèles dynamiques non linéaires par morceaux afin d'augmenter la précision du modèle.

Après cela, nous présentons des résultats obtenus par l'algorithme générique d'identification récursive présenté dans le chapitre 4. Nous procédons ainsi à une identification récursive du système hydraulique et de la machine de montage par des modèles dynamiques affines par morceaux. Nous étudions également l'influence du paramètre θ_{seuil} sur le nombre de sous-modèles identifiés.

Puis, nous présentons deux applications dans le domaine de la vision par ordinateur. L'objectif est de montrer la faisabilité des approches développées qui sont basées sur le couplage identification/classification dans un domaine où très généralement seules les méthodes de classification sont appliquées. Une nouvelle méthode de segmentation temporelle de la vidéo en différentes scènes basée sur une estimation de sous-modèles locaux linéaires est proposée. Chaque sous-modèle correspond à une scène. La procédure de segmentation basée sur l'estimation de sous-modèles locaux sera alors appliquée à trois types de séquences vidéo.

Enfin, ce chapitre est achevé par la proposition d'une dernière application. Cette appli-

cation concerne la classification automatique et la reconnaissance de visages sous différents seuils d'éclairage par l'estimation récursive de sous-modèles dynamiques locaux.

5.2 Modélisation de systèmes hydrographiques à surface libre

Les réseaux hydrographiques sont des systèmes de grandes dimensions, composés de systèmes hydrographiques à surface libre tels que les rivières et les canaux. Ces systèmes sont caractérisés par des dynamiques non linéaires à retard variable. Compte tenu de l'impact que peuvent avoir les saisons, les évènements climatiques et l'activité de l'Homme, ils sont soumis à de larges plages de fonctionnement. Ils sont généralement modélisés par les équations aux dérivées partielles de Saint Venant. Dans le but d'obtenir des modèles plus simples à mettre en œuvre, les équations de Saint Venant peuvent être approximées. L'équation simplifiée de diffusion est exprimée par

$$\frac{\partial Q(x,t)}{\partial t} + C(Q,z,x)\frac{\partial Q(x,t)}{\partial x} - D(Q,z,x)\frac{\partial^2 Q(x,t)}{\partial x^2} = 0, \tag{5.1}$$

avec C $[m/s]$ le coefficient de célérité et D $[m^2/s]$ le coefficient de diffusion. $Q(x,t)$ $[m^3/s]$ est le débit, t $[s]$ est la variable de temps, x $[m]$ est la variable spatiale mesurée dans la direction et le sens du courant et z $[m]$ est la variable spatiale correspondant à la hauteur d'eau.

L'équation de diffusion (5.1) peut ensuite être linéarisée autour d'un point de fonctionnement Q_e :

$$\frac{\partial q(x,t)}{\partial t} + C_e\frac{\partial q(x,t)}{\partial x} - D_e\frac{\partial^2 q(x,t)}{\partial x^2} = 0, \tag{5.2}$$

avec $Q = Q_e + q$, où q est la variation du débit autour du débit de linéarisation Q_e. C_e et D_e sont les coefficients de célérité et de diffusion calculés pour le débit Q_e. A partir de l'équation de diffusion linéarisée (5.2), une fonction de transfert reliant le débit amont $Q_{up}(x,t)$ au débit aval $Q_{dns}(x,t)$ pour un système hydrographique de longueur X, est obtenue [59] et exprimée par :

$$G(x,t) = \frac{Q_{dns}(x,t)}{Q_{up}(x,t)} = e^{\frac{XC_e}{2D_e}\left(1-\sqrt{1+4\frac{D_e}{C_e^2}t}\right)}. \tag{5.3}$$

La linéarisation de l'équation de diffusion conduit à l'obtention d'un modèle physique qui représente la dynamique des systèmes hydrographiques autour d'un point de fonctionnement. Cependant, la précision de ce modèle diminue à mesure que le point de fonctionnement du système s'éloigne du point de linéarisation (le modèle obtenu n'est valable qu'autour d'un point de fonctionnement). En plus de la connaissance des points

de linéarisation, cette méthode nécessite aussi une bonne connaissance de la géométrie et de la topographie des systèmes considérés.

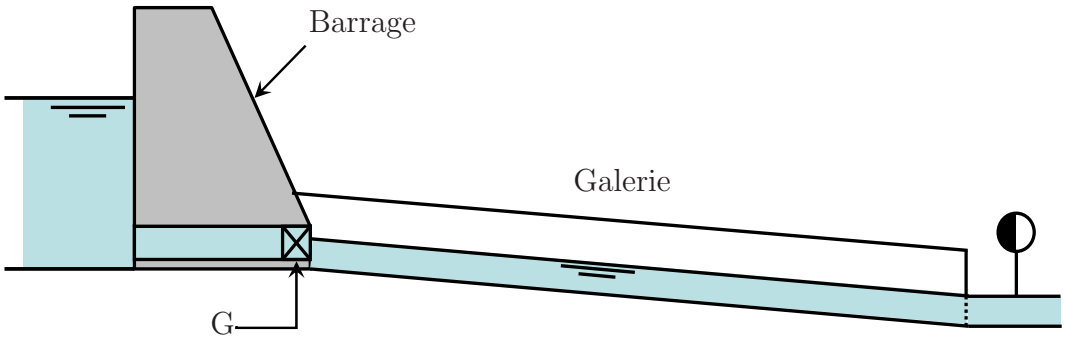


FIGURE 5.1 – Représentation schématique du barrage et de la galerie de la rivière Save.

Contrairement à la modélisation par linéarisation de l'équation de Saint Venant qui est une méthode d'identification boîte blanche, la modélisation proposée ici, estime la dynamique du système hydrographique à surface libre sur une large plage de fonctionnement sans connaissance *a priori* de sa géométrie et de ses caractéristiques. Le système hydrographique considéré est la galerie Lunax-Save située dans le sud-ouest de la France (figure 5.1). Elle est de section circulaire et a une plage de débit allant de $q_{min} = 0.5\ m^3 s^{-1}$ à $q_{max} = 5\ m^3 s^{-1}$. Le problème consiste à modéliser l'écoulement de l'eau entre les deux extrémités du canal. Par conséquent, l'entrée et la sortie du système sont définies respectivement comme le débit volumique d'eau en amont $u(t)$ et le débit volumique d'eau en aval $y(t)$, exprimés tous en $m^3 s^{-1}$ (cf. Figure 5.2).

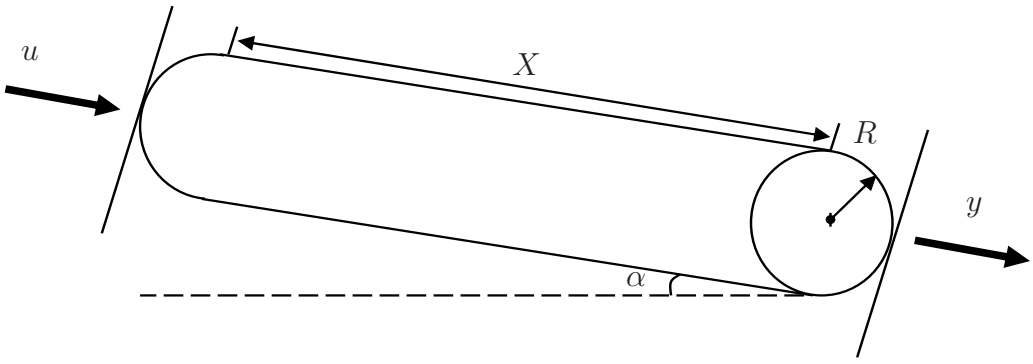


FIGURE 5.2 – Caractéristiques géométriques de la galerie.

En raison de la grande dimension de ces systèmes et de la difficulté à implanter des réseaux de capteurs sur le terrain, les données réelles sont parfois difficiles à collecter. Par conséquent, il est de pratique courante de recourir à des logiciels reproduisant la dynamique des systèmes hydrographiques à surface libre. Le logiciel employé pour générer les données est le logiciel SIC développé par le CEMAGREF de Montpellier. Ainsi, un ensemble de 800 données entrée-sortie a été obtenu. Nous montrons dans les paragraphes suivants les résultats de l'identification de ces systèmes par des modèles dynamiques affines par morceaux.

Identification de deux modes :
En appliquant l'algorithme 2.1 sur les données récoltées avec les paramètres $c = 120$, $n_a = 2$ et $n_b = 2$, un modèle PWA de deux modes a été identifié ($s = 2$). On rappelle que s est le nombre de sous-modèles. Sur la figure 5.3, nous présentons la sortie réelle du système et la sortie reconstruite à partir du modèle. La séquence des modes estimés est également tracée. A partir de la forme de la réponse $y(k)$ et de la séquence des modes estimés, on peut constater que le mode 1 correspond au débit volumique d'eau en aval $0.5\ m^3s^{-1} < y(k) < 2\ m^3s^{-1}$ et le mode 2 correspond à $2\ m^3s^{-1} \leq y(k) < 5\ m^3s^{-1}$. Les

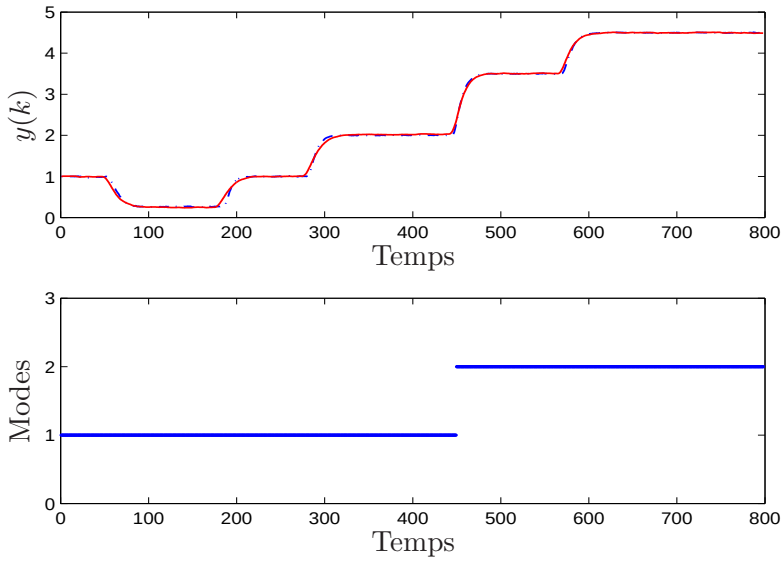


FIGURE 5.3 – Estimation de deux modes. (Haut) La sortie réelle du système (en continu) et la sortie reconstruite à partir du modèle (en pointillé). (Bas) La séquence des modes estimés.

vecteurs de paramètres identifiés et les hyperplans estimés sont donnés par :

$$\hat{\theta}_1 = \begin{bmatrix} 1.5963 & -0.6266 & -0.0151 & 0.0459 & -0.0003 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} 1.3656 & -0.4273 & 0.0208 & 0.0401 & 0.0035 \end{bmatrix},$$

$$\hat{H}_1 = \begin{bmatrix} -\hat{h}_1 \end{bmatrix}, \quad \hat{H}_2 = \begin{bmatrix} \hat{h}_1 \end{bmatrix},$$
$$\hat{h}_1 = \begin{bmatrix} -7.948 & -7.073 & 2.801 & -3.924 & 38.946 \end{bmatrix}.$$

Nous calculons la moyenne des erreurs quadratiques SSE pour mesurer la similarité entre la sortie mesurée sur le système et la sortie reconstruite à partir du modèle. Ainsi, dans ce cas, on trouve SSE= 0.002. Comme on peut le constater aussi sur la figure 5.3, cela prouve que la sortie réelle et celle reconstruite sont presque identiques. Nous présentons dans le paragraphe suivant une identification de trois modes de fonctionnement.

Identification de trois modes :
En choisissant $c = 80$, $n_a = 2$ et $n_b = 2$, et en appliquant notre algorithme sur le même ensemble de données que ci-dessus, un modèle de trois modes est identifié ($s = 3$). La sortie réelle du système superposée à la sortie reconstruite à partir du modèle ainsi que la séquence des modes estimés sont présentées sur la figure 5.4. Dans ce cas, nous avons trouvé SSE= 0.0012. Nous pouvons constater que la précision du modèle estimé semble augmenter avec le nombre de sous-modèles car il y a une réduction du SSE de 40% par rapport au cas ($s = 2$). Le mode 1 correspond au débit volumique d'eau en aval $0.5\ m^3s^{-1} < y(k) < 1.3\ m^3s^{-1}$, le mode 2 correspond à $1.3\ m^3s^{-1} \leq y(k) < 3.8\ m^3s^{-1}$ et mode 3 correspond à $3.8\ m^3s^{-1} \leq y(k) < 5\ m^3s^{-1}$.
Les vecteurs de paramètres identifiés et les hyperplans estimés sont donnés par :

$$\hat{\theta}_1 = \begin{bmatrix} 1.5351 & -0.5632 & -0.0223 & 0.0516 & -0.0007 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} 1.6102 & -0.6473 & 0.0043 & 0.0328 & -0.0001 \end{bmatrix},$$
$$\hat{\theta}_3 = \begin{bmatrix} 0.5234 & 0.3328 & -0.0002 & 0.0162 & 0.5750 \end{bmatrix},$$

$$\hat{H}_1 = \begin{bmatrix} -\hat{h}_1 \end{bmatrix}, \quad \hat{H}_2 = \begin{bmatrix} \hat{h}_1^\top & -\hat{h}_2^\top \end{bmatrix}^\top, \quad \hat{H}_3 = \begin{bmatrix} \hat{h}_2 \end{bmatrix},$$
$$\hat{h}_1 = \begin{bmatrix} -10.225 & -12.544 & -5.921 & -2.844 & 44.475 \end{bmatrix},$$
$$\hat{h}_2 = \begin{bmatrix} -9.290 & -9.062 & -9.253 & -4.724 & 130.084 \end{bmatrix}.$$

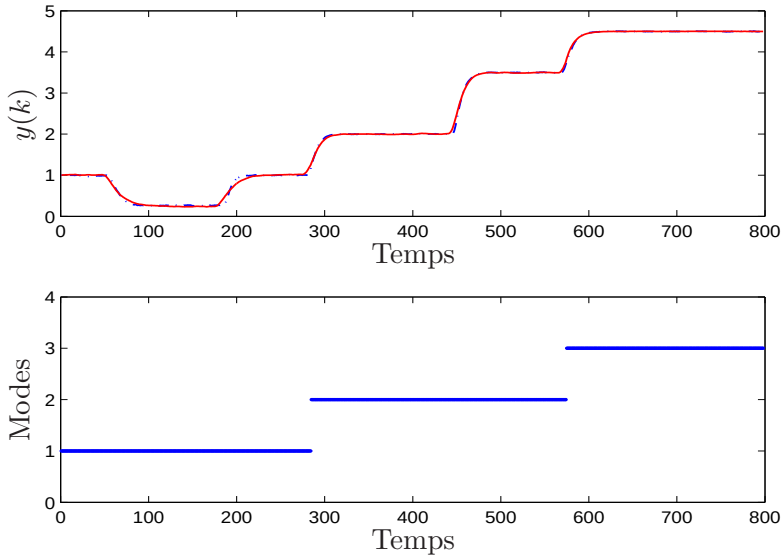


FIGURE 5.4 – Estimation de trois modes. (Haut) La sortie réelle du système (en continu) et la sortie reconstruite à partir du modèle (en pointillé). (Bas) La séquence des modes estimés.

Validation du modèle estimé :

La validation de l'approche proposée est faite sur un ensemble de 750 données entrée-sortie générées par une séquence d'entrée d'excitation différente de celle utilisée pour l'identification. La validation consiste à reconstruire la sortie pour ce nouvel ensemble de données en utilisant les modèles estimés précédemment. Pour cela, on utilise les paramètres identifiés dans le cas où $s = 3$. Le résultat ainsi trouvé est présenté sur la figure 5.5 où la sortie réelle est superposée à la sortie reconstruite. On trouve dans ce cas SSE= 0.0092.
Afin de montrer l'efficacité de l'approche d'identification proposée, la validation du modèle estimé est comparée à l'approche de modélisation basée sur les équations physiques des systèmes hydrauliques à surface libre. Le modèle de la galerie est obtenue par linéarisation de l'équation de diffusion (voir équation (5.2)) autour du débit $Q_e = 1\ m^3/s$. La fonction de transfert obtenue est donnée ci-dessous :

$$F(s) = \frac{e^{-248s}}{1 + 239\ s + 16250\ s^2}. \tag{5.4}$$

La sortie reconstruite par cette méthode de linéarisation est superposée aux deux autres sorties (sortie réelle et sortie reconstruite par les PWA) sur la figure 5.5. Pour la méthode basée sur la linéarisation de l'équation de diffusion, la moyenne des erreurs quadratiques trouvée est SSE= 0.0160. Par comparaison des deux SSE, on peut constater que le modèle identifié par notre méthode (qui est une méthode de type « boîte noire ») est plus précis que celui identifié par la méthode de la linéarisation de l'équation de diffusion. Certes, le

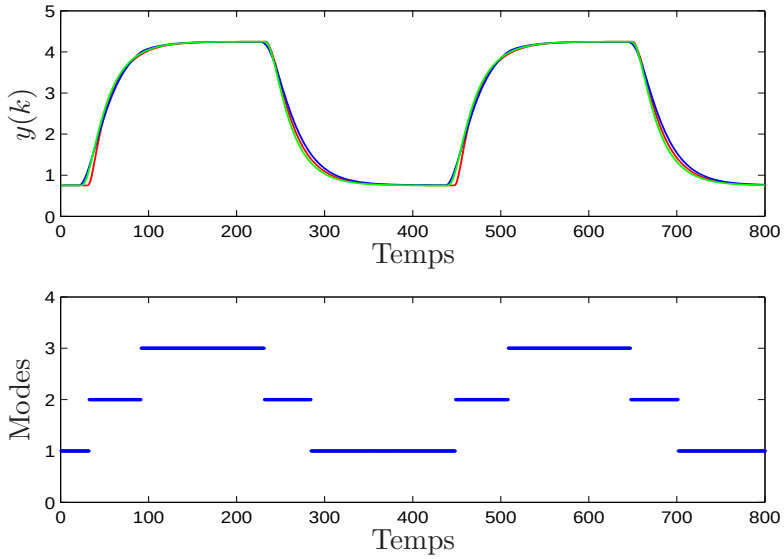


FIGURE 5.5 – (Haut) La sortie réelle du système (en rouge), la sortie reconstruite à partir des PWA modèles (en bleu) et la sortie reconstruite à partir de la linéarisation de l'équation de l'onde de diffusion (en vert). (Bas) La séquence des modes estimés par notre méthode.

modèle (5.4) est plus simple mais il est basé sur des connaissances *a priori* du système. Les sorties reconstruites à partir des modèles estimés sont très proches des sorties réelles et cela, pour différentes séquences d'entrée d'excitation. Les résultats présentés sont satisfaisants et démontrent les potentialités expérimentales de notre méthode.

Influence du paramètre c :
Dans ce paragraphe, nous étudions l'influence du paramètre c sur la précision et la complexité du modèle PWA identifié à partir des données entré-sortie du système hydrographique considéré. Les deux algorithmes 2.1 et 2.2 sont appliqués sur le même jeu de données considéré (les 800 données utilisées dans la partie identification) pour différentes valeur de c. Le paramètre c varie dans l'intervalle [5,150] avec un pas égal à 5. Pour chaque valeur de c, nous calculons le nombre de sous-modèles, le FIT donné par

$$\text{FIT} = \left(1 - \frac{\|\hat{y}\_y\|}{\|y\_\bar{y}\|}\right) \times 100\%,$$

le nombre d'itérations et le temps de calcul. Le résultat de ce calcul est représenté sur la figure 5.6.
Le FIT sert à mesurer la similarité entre la sortie y mesurée sur le système et la sortie $\hat{y}$ reconstruite à partir du modèle. Dans cette formule, $\bar{y}$ est la moyenne des mesures

$y(k), k = 1, \ldots, N$, où N est le nombre de points de mesure disponibles.

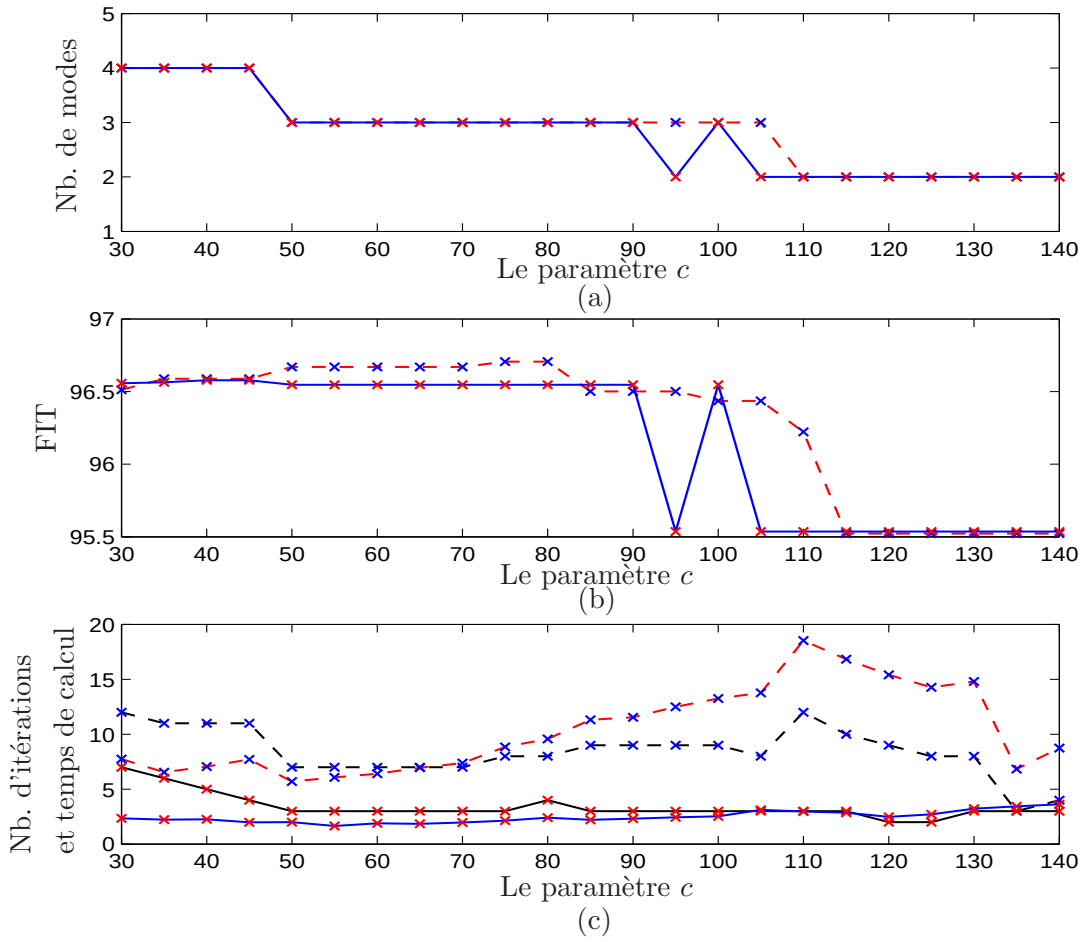


FIGURE 5.6 – Influence du paramètre c sur les résultats fournis par l'algorithme 2.1 (en continu) et par l'algorithme 2.2 (en pointillé) : (a) Nombre de sous-modèles (b) Le FIT (c) Nombre d'itérations (noir) et temps de calcul en secondes (rouge).

Nous constatons que plus c est petit, plus le nombre de sous-modèles est grand avec une évolution monotone sauf pour la valeur $c = 95$. Le FIT diminue avec l'augmentation du paramètre c et ceci pour les deux algorithmes. La précision fournie par les deux algorithmes est équivalente. Le nombre d'itérations et le temps de calcul sont globalement constants pour l'algorithme 2.1. Ils augmentent légèrement en fonction de c pour l'algorithme 2.2. Une concordance entre l'évolution du temps de calcul et du nombre d'itérations peut être remarquée.

5.3 Modélisation d'une machine de montage de composants sur circuit imprimé

La machine "Pick-and-place" est utilisée pour placer automatiquement des composants électroniques sur des circuits imprimés. Le processus se compose d'une tête de montage transportant le composant électronique. A chaque cycle de montage, la tête de montage est entraînée jusqu'à la position exacte du circuit imprimé où le composant doit être fixé. Une fois à la position désirée, le composant est poussé délicatement jusqu'à ce qu'il soit solidement en contact avec la surface du circuit, puis il est relâché. Cette procédure est répétée de manière cyclique pour chaque nouveau composant.
Les données d'entrée-sortie ont été recueillies à partir d'une maquette expérimentale composée d'une tête de montage et d'une surface d'impact représentant le circuit imprimé (figure 5.7-(b)). Ces données nous ont été délivrées par Dr Aleksandar Juloski de l'Université Technologique d'Eindhoven, Pays-Bas. Un modèle physique du dispositif expérimental est représenté sur la figure 5.7-(a). La tête de montage est représentée par la masse M qui est contrainte à se déplacer seulement selon l'axe vertical. Les ressorts c_1 et c_2 caractérisent une élasticité. Les frictions linéaires sont représentées par les amortisseurs d_1 et d_2 et les frottements secs sont représentés par les blocs f_1 et f_2. L'entrée du système est la tension d'alimentation du moteur chargé d'entraîner la tête de montage. Cette tension est représentée par la force $\vec{F}$ sur la figure 5.7-(a). La sortie du système est la position de la tête de montage. Cette maquette a été utilisée pour la validation de divers algorithmes d'identification de systèmes hybrides [48] [49] [62] [8] [2]. Une description détaillée du processus peut être trouvée dans [48].
Quatre modes de fonctionnement du système peuvent être distingués :

- **Le mode libre** : la tête de montage se déplace librement, c'est-à-dire sans être en contact avec la surface d'impact.
- **Le mode contraint** : la tête de montage se déplace en contact avec la surface d'impact.
- **La saturation supérieure** : correspondant à la situation dans laquelle la tête de montage ne peut se déplacer vers le haut, en raison de contraintes physiques.
- **La saturation inférieure** : correspondant à la situation dans laquelle la tête de montage ne peut se déplacer vers le bas, en raison de contraintes physiques.

A partir du modèle physique du système (cf. figure 5.7-(a))), une technique de modélisation boîte blanche peut être utilisée pour déterminer la valeur des paramètres physiques (c_1, c_2, etc.). Cependant, il est montré dans [48] qu'il y a des difficultés majeures qui entravent la modélisation boîte blanche. Pour illustrer ces difficultés, on considère une représentation simplifiée du système. On suppose que les saturations ne sont pas atteintes,

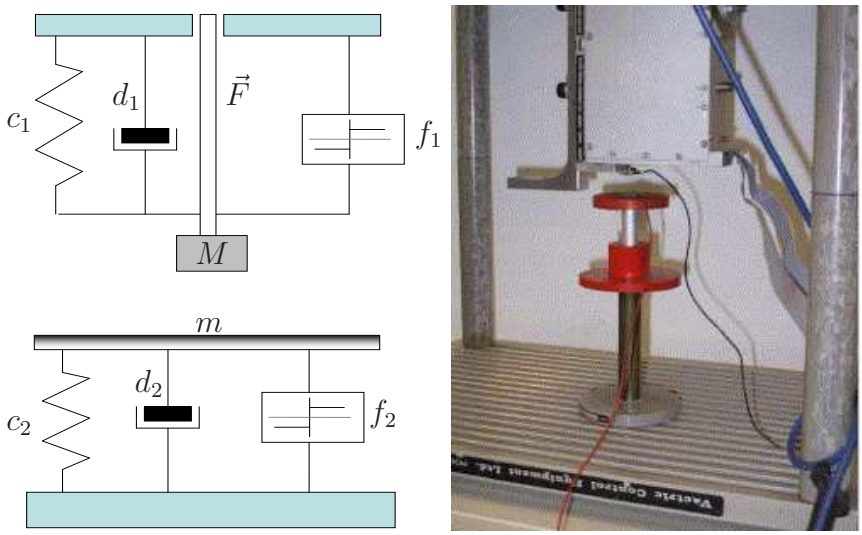


FIGURE 5.7 – (a) Modèle physique du système de montage. (b) Photo de la maquette de montage.

que le frottement sec est absent et que les ressorts et les amortisseurs ont un comportement linéaire. Dans ce cas, les équations reliant la force $\vec{F}$ à la position y de la tête sont données par :

$$\begin{array}{l} \ddot{y} = -\frac{c_1}{M}y - \frac{d_1}{M}\dot{y} + \frac{c_1\bar{y}_1}{M} + \frac{F}{M} \quad \text{(Mode libre)} \\ \ddot{y} = -\frac{c_1+c_2}{M+m}y - \frac{d_1+d_2}{M+m}\dot{y} + \frac{c_1\bar{y}_1+c_2\bar{y}_2}{M+m} + \frac{F}{M+m} \quad \text{(Mode contraint)} \end{array} \tag{5.5}$$

$y = 0$ correspond à la position supérieure de la tête, $\bar{y}_1$ et $\bar{y}_2$ sont définis, respectivement, tels que $y = \bar{y}_1$ quand le ressort c_1 est au repos et $y = \bar{y}_2$ quand le ressort c_2 est au repos. m correspond à la masse de la surface d'impact.

Comme les données sont disponibles en temps discret, on doit alors échantillonner le modèle (5.5) et identifier ses paramètres. Malheureusement, à notre connaissance, il n'existe pas une méthode générale pour discrétiser des systèmes hybrides en temps continu [1]. La principale difficulté réside dans le choix du pas d'échantillonnage. En effet, une commutation d'un mode à un autre peut se produire entre deux échantillons et donc ne sera pas prise en compte après discrétisation. Pour cette raison, on recourt à une identification

de type boîte noire en utilisant les modèles dynamiques affines par morceaux :

$$\begin{aligned} &y(k) = \theta_{1,1}y(k-1) + \cdots + \theta_{1,n_a}y(k-n_a) + \theta_{1,n_a+1}F(k-1) + \cdots \\ &+\theta_{1,n_a+nb}F(k-nb) \quad \text{(mode libre)} \\ &y(k) = \theta_{2,1}y(k-1) + \cdots + \theta_{2,n_a}y(k-n_a) + \theta_{2,n_a+1}F(k-1) + \cdots \\ &+\theta_{2,n_a+nb}F(k-nb) \quad \text{(mode contraint)} \end{aligned} \tag{5.6}$$

où n_a etn_b sont les ordres du système.
Même si la relation entre les deux modèles (5.5) et (5.6) n'est pas évidente, notons que les deux modèles sont des modèles affines par rapport à y et F.

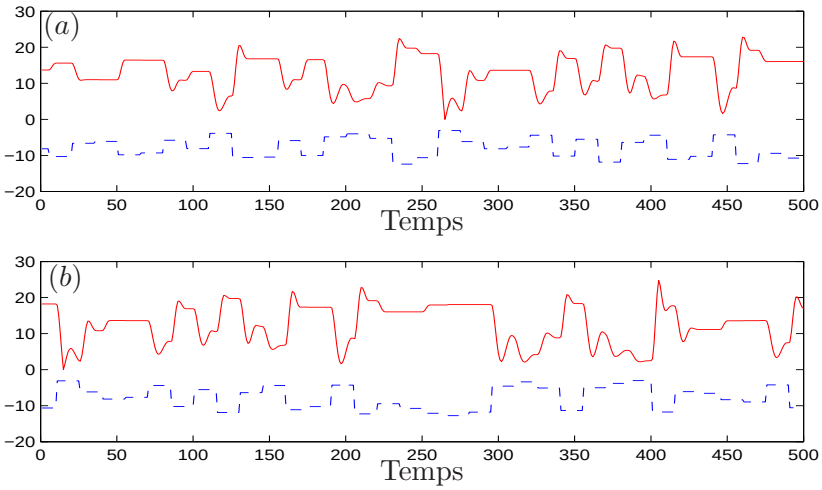


FIGURE 5.8 – (a) Données utilisées pour l'identification, (b) Données utilisées pour la validation (en continu : la sortie réelle du système ; en pointillé : l'entrée du système.)

Un ensemble de 750 échantillons de données a été collecté à partir du dispositif expérimental. Les 500 premières données sont utilisées pour l'identification, et les 500 dernières données sont utilisées pour la validation des paramètres identifiés (figure 5.8). De cette façon, 250 échantillons sont utilisés à la fois pour l'identification et la validation afin de pouvoir comparer les résultats sur les deux parties.

5.3.1 Identification de modèles dynamiques affines par morceaux

Identification de deux modes :
La procédure de classification de données et d'estimation de paramètres (Chapitre 2) utilisant l'approche basée sur la théorie de Dempster-Shafer (Algorithme 2.2) a été appliquée sur les 500 premières données utilisées pour l'identification avec $c = 150$, $n_a = 2$ et $n_b = 1$. Un modèle PWARX de deux sous-modèles affines est identifié. Nous présentons sur la figure 5.9 la sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle

identifié sur les données d'estimation (Fig. 5.9-(a)) et sur les données de validation (Fig. 5.9-(b)). La séquence de modes estimés est également tracée. A partir de la forme de $y(k)$, on peut constater que le mode 1 correspond au mode contraint car il s'agit des valeurs supérieures de $y(k)$ et que le mode 2 correspond au mode libre (partie inférieure de $y(k)$). Comme on peut le voir, la sortie mesurée sur le système réel et la réponse du modèle sont proches. Une certaine différence est cependant visible sur l'intervalle (200,300). Ceci est principalement dû aux effets non linéaires de frottement sec.

n_a,n_b	Notre méthode	La méthode basée sur la technique des K-means [35]	La méthode de l'erreur bornée [8]
2,1	1.18	N/A	N/A
2,2	1.21	1.98	2.15

TABLE 5.1 – Comparaison des moyennes des erreurs quadratiques SSE obtenues par différentes méthodes.

Afin de comparer notre méthode à d'autres méthodes, nous montrons, sur la table 5.1, les moyennes des erreurs quadratiques SSE (calculées sur les données de validation) obtenues par notre méthode et celles obtenues par la méthode basée sur la technique de classification K-means [35] et la méthode de l'erreur bornée [8] [7].
On peut observer que pour $(n_a, n_b) = (2, 2)$, le SSE obtenu par notre méthode est inférieur aux deux autres SSE. Cela signifie que le modèle obtenu par notre méthode reproduit d'une façon plus précise le comportement du système.

Pour estimer les frontières séparatrices délimitant les régions, nous avons appliqué un classifieur SVM binaire en raison de l'existence de seulement deux régions.

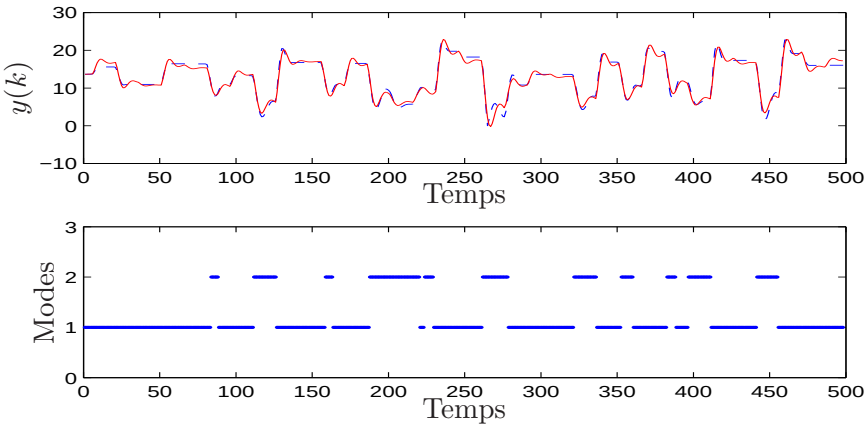
Les vecteurs de paramètres estimés sont donnés par

$$\hat{\theta}_1 = \begin{bmatrix} 1.3877 & -0.6464 & -36.8964 & 0.5318 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} 1.4807 & -0.7075 & -47.5983 & -0.5089 \end{bmatrix},$$

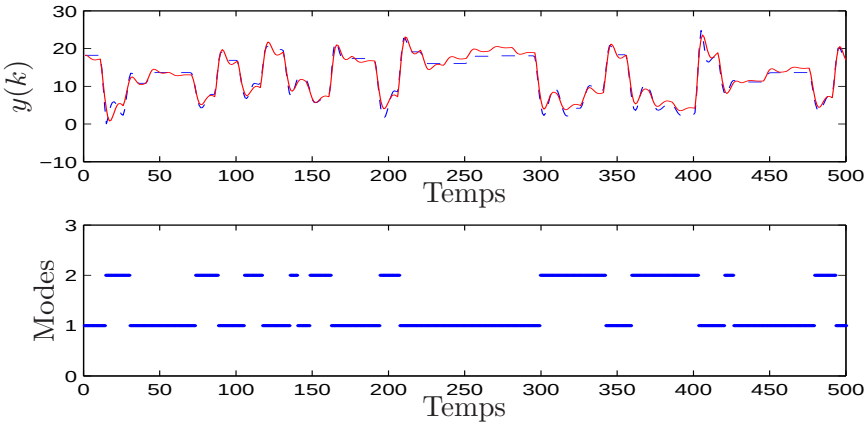
et les hyperplans définissant les deux régions sont donnés par $\hat{H}_1 = \begin{bmatrix} \hat{h} \end{bmatrix}$, $\hat{H}_2 = \begin{bmatrix} -\hat{h} \end{bmatrix}$, où

$$\hat{h} = \begin{bmatrix} 1.3443 & -0.0146 & -1.0967 & -13.6869 \end{bmatrix}.$$

7. Les valeurs de SSE des deux méthodes ont été prises de [68].



(a) Identification



(b) Validation

FIGURE 5.9 – (a)-Identification de deux modes et (b)-Validation du modèle PWA avec $(n_a, n_b, c) = (2, 1, 150)$. La sortie réelle du système (en continu) et la sortie reconstruite à partir du modèle (en pointillé).

Identification de quatre modes :

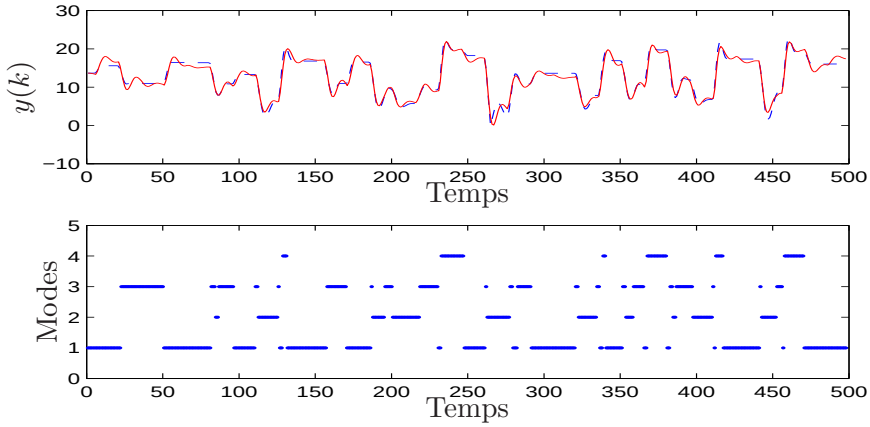
En choisissant $(n_a, n_b, c) = (2, 1, 100)$ et en appliquant l'Algorithme 2.2 sur le même jeu de données, un modèle PWARX de quatre modèles affines est identifié. Notons qu'avec $c = 150$, on a obtenu deux modes (voir le paragraphe ci-dessus) et avec $c = 100$, on obtient quatre modes. La figure 5.10 montre la sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié ainsi que la séquence de modes estimés. Ici, deux des quatre modes estimés correspondent aux modes libre et contraint (respectivement le mode 3 et le mode 4). Les deux autres modes représentent le comportement de la machine aux frontières des deux modes contraint et libre c'est-à-dire la saturation inférieure et la saturation supérieure (respectivement le mode 2 et le mode 1).

Dans ce cas, nous obtenons un SSE de 1.14, qui est inférieur à la valeur trouvée ci-dessus

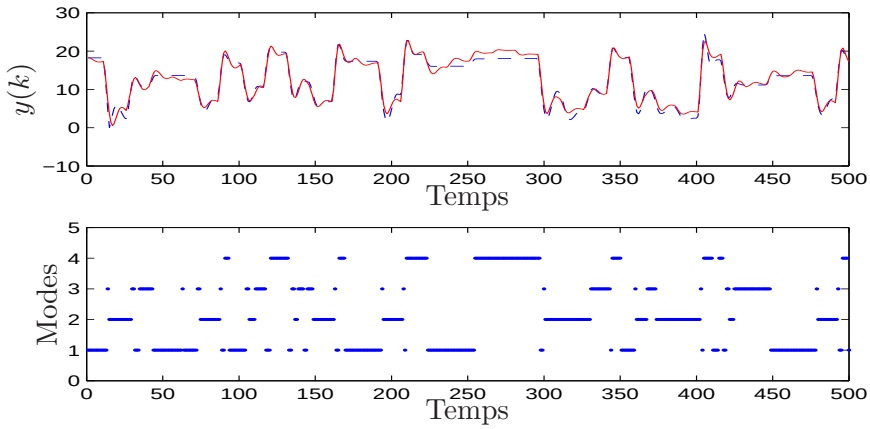
(SSE=1.21) pour deux modes identifiés. Les vecteurs de paramètres identifiés $\hat{\theta}_i$ sont donnés par

$$\hat{\theta}_1 = \begin{bmatrix} 1.5131 & -0.7150 & -32.9637 & 0.0007 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} 1.4260 & -0.6759 & -49.0428 & -0.4732 \end{bmatrix},$$
$$\hat{\theta}_3 = \begin{bmatrix} 1.5168 & -0.7662 & -39.9667 & 0.2442 \end{bmatrix},$$
$$\hat{\theta}_4 = \begin{bmatrix} 1.1747 & 0.5202 & -36.6739 & 2.2990 \end{bmatrix}$$

Les paramètres des hyperplans estimés $\hat{H}_i$ sont donnés par $\hat{H}_1 = \begin{bmatrix} \hat{h}_1 \end{bmatrix}$, $\hat{H}_2 = \begin{bmatrix} -\hat{h}_1^\top & \hat{h}_2^\top \end{bmatrix}^\top$



(a) Identification



(b) Validation

FIGURE 5.10 – (a)-Identification de quatre modes et (b)-Validation du modèle PWA avec $(n_a, n_b, c) = (2, 1, 100)$. La sortie réelle du système (en continu) et la sortie reconstruite à partir du modèle (en pointillé).

et $\hat{H}_3 = \left[-\hat{h}_2^\top \ \hat{h}_3^\top \right]^\top$, où

$$\hat{h}_1 = \left[-30.0422 \ -0.3041 \ 528.5286 \ 283.6267 \right],$$
$$\hat{h}_2 = \left[27.2385 \ -1.4918 \ -8.5364 \ -319.6922 \right],$$
$$\hat{h}_3 = \left[-4.1926 \ 0.1794 \ 0.0456 \ 81.7490 \right].$$

Influence du paramètre c :
Dans ce paragraphe, nous montrons l'influence que peut avoir le paramètre c sur la précision et la complexité du modèle PWA identifié à partir des données entrée-sortie collectées à partir du dispositif expérimental. Pour différentes valeurs de c, nous appliquons les deux algorithmes 2.1 et 2.2 sur le même jeu de données utilisées dans la partie identification. Nous faisons varier le paramètre c dans l'intervalle [70,170] avec un pas égal à 5. Pour chaque valeur de c, le nombre de sous-modèles s, le FIT, le nombre d'itérations et le temps de calcul sont donnés sur la figure 5.11.

Comme prévu, nous constatons que plus c est petit, plus le nombre de sous-modèles est grand. Cependant, le nombre de sous-modèles identifiés par les deux algorithmes pour une valeur de c donnée n'est pas nécessairement le même. Ceci est dû à la nature différente des règles de décision des deux algorithmes. Nous notons qu'à partir de la valeur $c = 140$, un seul sous-modèle a été identifié par l'algorithme 2.1.
A partir de la courbe montrant la variation du FIT en fonction du paramètre c, nous constatons que pour les deux algorithmes 2.1 et 2.2, plus c augmente, plus le FIT diminue et donc plus le modèle devient moins précis. Ceci est cohérent puisque la précision augmente avec le nombre de sous-modèles. Nous constatons également que la précision fournie par l'algorithme 2.2 (basé sur la théorie de Dempster-Shafer) est légèrement meilleure que celle fournie par l'algorithme 2.1 dans ce cas. Le nombre d'itérations quant à lui, augmente légèrement avec l'augmentation du paramètre c pour l'algorithme 2.2. Cependant la variation du nombre d'itérations de l'algorithme 2.1 semble être globalement constante. Une concordance entre l'évolution du nombre d'itérations et le temps de calcul pour toutes les valeurs de c peut être remarquée.

5.3.2 Identification de modèles dynamiques non linéaires par morceaux

Nous avons constaté dans le paragraphe précédent qu'à cause des effets non linéaires du frottement sec de la machine de montage, il existe quelques différences entre la sortie réelle et la sortie reconstruite. Cette différence est bien visible sur l'intervalle (200,300)

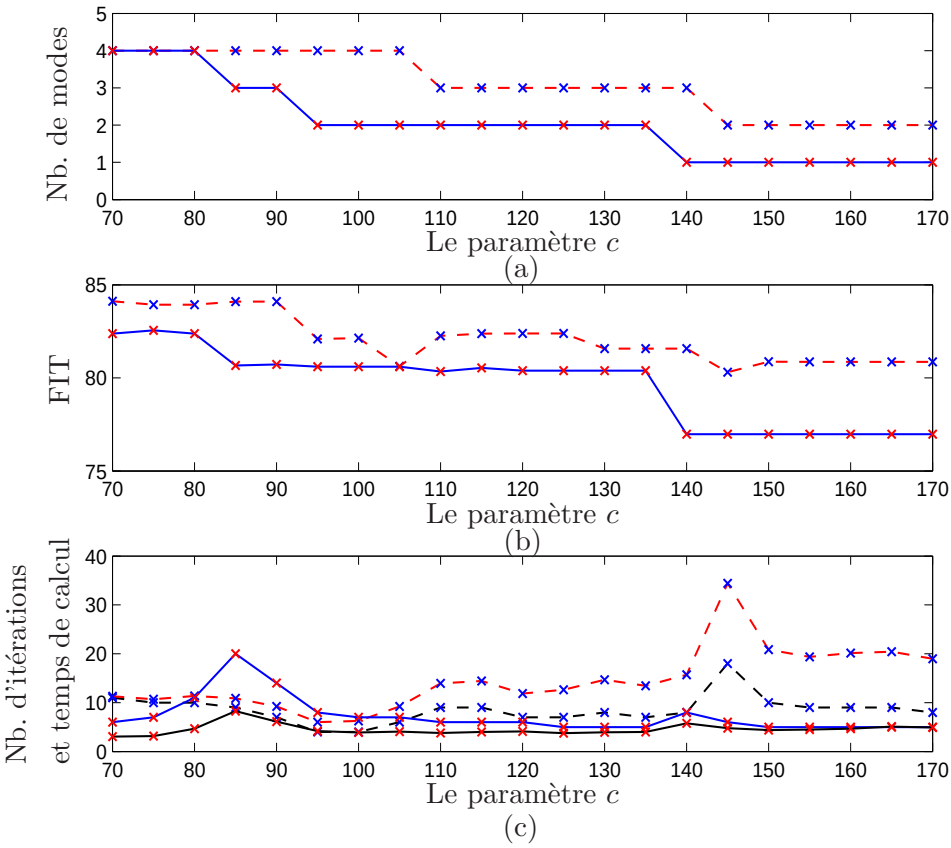
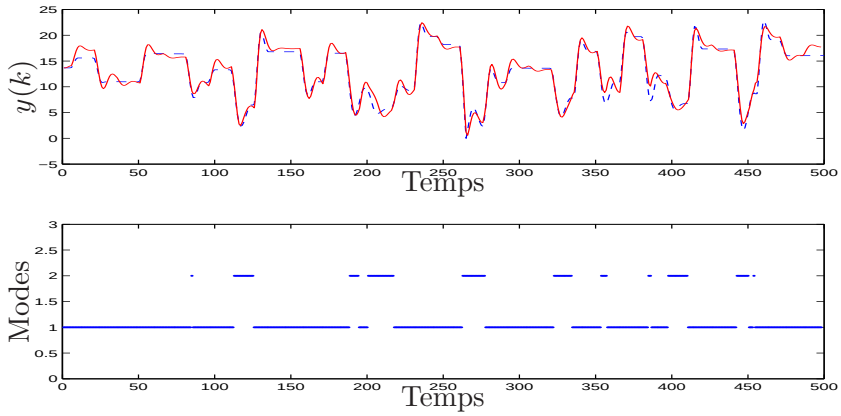


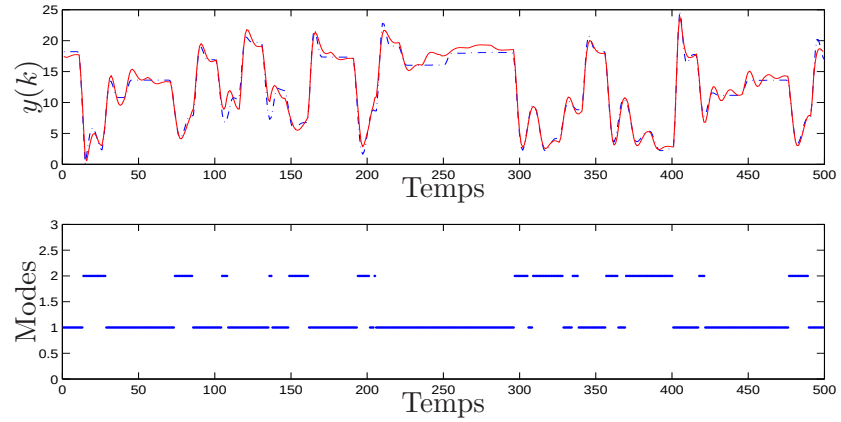
FIGURE 5.11 – Influence du paramètre c sur les résultats fournis par l'algorithme 2.1 (en continu) et par l'algorithme 2.2 (en pointillé) : (a) Nombre de sous-modèles (b) Le FIT (c) Nombre d'itérations (noire) et temps de calcul en secondes (rouge).

sur les données de validation. Pour remédier à cela, la machine de montage est modélisée, dans cette section, par des modèles dynamiques non linéaires par morceaux. Un noyau gaussien est alors utilisé dans la régression par les LSSVM pour l'estimation des fonctions correspondantes aux différents sous-modèles avec $\sigma = 10^{-4}$.
Deux sous-modèles non linéaires ont été identifiés en choisissant $c = 100$, $n_a = 2$ et $n_b = 1$. Nous présentons sur la figure 5.12 la sortie mesurée sur le système réel et la sortie reconstruite à partir du modèle identifié sur les données d'estimation (Fig. 5.12-(a)) et sur les données de validation (Fig. 5.12-(b)). La séquence de modes estimés est également tracée. La sortie mesurée sur le système réel coïncide avec la réponse du modèle avec une précision élevée. De plus, la différence dûe au frottement sec de la machine qui était visible dans le cas de l'identification par les PWA est très réduite dans ce cas. La moyenne des erreurs quadratiques calculée sur les données de validation est SSE= 0.64. Cette valeur représente la moitié de la valeur trouvée dans le cas de l'identification par les PWA (pour $s = 2$) qui

était SSE=1.18.



(a) Identification



(b) Validation

FIGURE 5.12 – (a)-Identification et (b)-Validation du modèle PWN avec $(n_a, n_b, c) = (2, 1, 100)$. La sortie réelle du système (en continu) et la sortie reconstruite à partir du modèle (en pointillé).

Influence du paramètre de la fonction noyau :

Dans ce paragraphe, nous étudions l'influence que peut avoir le paramètre de la fonction noyau sur la qualité de la sortie reconstruite. Nous choisissons ici la fonction noyau Gaussienne et nous faisons varier son paramètre σ. Pour chaque valeur de σ, nous calculons la moyenne des erreurs quadratiques SSE calculée entre la sortie mesurée sur le système réel et la sortie reconstruite à partir des modèles non linéaires par morceaux. Le nombre d'itérations est également donné.

Le résultat de ce calcul est représenté sur le tableau 5.2. Nous constatons pour ce cas que le nombre d'itérations moyen est 5. La moyenne des erreurs quadratiques SSE aug-

Le paramètre σ	10^{-4}	10^{-3}	5×10^{-3}	10^{-2}	5×10^{-2}	10^{-1}	5×10^{-1}
Nombre d'itérations	6	5	5	4	5	4	7
SSE	0.64	1.12	1.13	1.17	1.19	1.21	4.95

TABLE 5.2 – SSE et nombres d'itérations en fonction du paramètre σ.

mente avec le paramètre σ. Le meilleur SSE est obtenu pour $\sigma = 10^{-4}$. Cependant, le SSE reste acceptable jusqu'à la valeur $\sigma = 10^{-1}$. Après cette valeur, la qualité d'estimation se dégrade considérablement. L'influence du paramètre de la fonction noyau sur les performances de l'estimation a été le sujet d'étude de plusieurs auteurs (voir par exemple [65]). En général le choix de ce paramètre dépend de la densité des données et du degré de la non linéarité. Par exemple, une très forte non linéarité locale nécessite dans le cas du noyau Gaussien une valeur de σ relativement faible.

5.4 Identification récursive de modèles dynamiques affines par morceaux

Dans cette section, nous précédons à l'identification récursive des systèmes hydrographiques à surface libre et de la machine de montage de composants sur circuit imprimé par des modèles dynamiques affines par morceaux. L'algorithme 4.1 est alors appliqué aux trois ensembles de données utilisés précédemment. Les deux premiers ensembles concernent les données récoltées à partir du système hydrographique tandis que les données du troisième ensemble sont récoltées à partir de la machine de montage. Comme nous l'avons vu dans la section précédente, les modes identifiés constituant le système hydrographique restent actifs pendant un certain temps avant de commuter (voir figures 5.3 et 5.5). Nous choisissons pour la modélisation des données de ce système les paramètres suivants : un temps de séjour minimum égal à $\tau = 40$, un paramètre c égal à 30. Le paramètre θ_{seuil} est pris égal à 4 pour le premier ensemble de données et égal à 14 pour le deuxième. Les résultats de l'identification récursive des deux ensembles sont respectivement représentés sur la figure 5.15 et 5.13. Sur chaque figure, la sortie réelle du système superposée à la sortie reconstruite à partir du modèle ainsi que la séquence des modes estimés sont présentées. Un FIT égal à 96.29 a été obtenu pour le premier ensemble de données et 95.22 pour le deuxième ensemble.

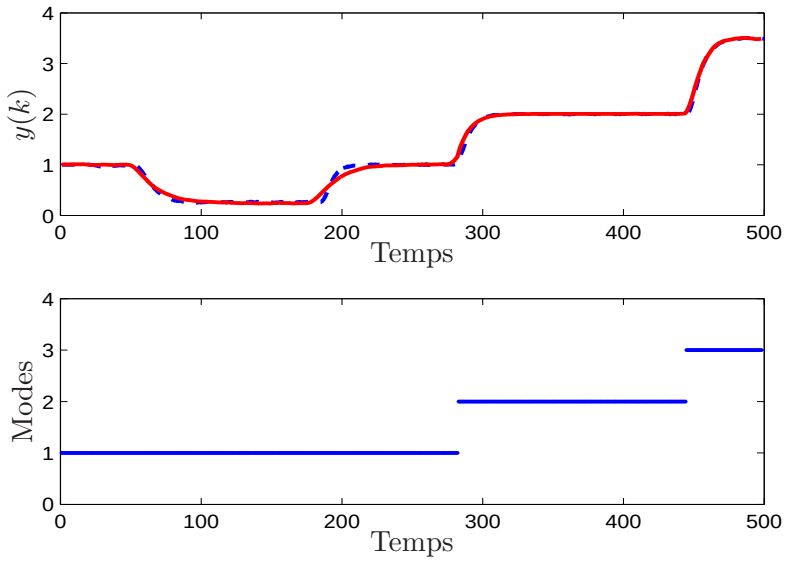


FIGURE 5.13 – Identification récursive du système hydrographique avec $\tau = 40$, $c = 30$ et $\theta_{seuil} = 14$. (Haut) La sortie réelle du système (en pointillé), la sortie reconstruite à partir des modèles PWA (en continu) . (Bas) La séquence des modes estimés.

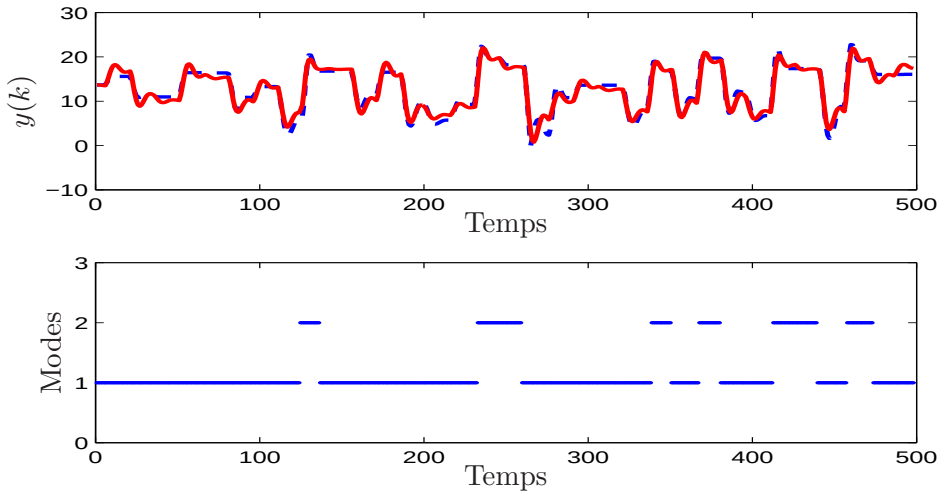


FIGURE 5.14 – Identification récursive de la machine de montage avec $\tau = 10$, $c = 9$ et $\theta_{seuil} = 2$. (Haut) La sortie réelle du système (en pointillé), la sortie reconstruite à partir des modèles PWA (en continu) . (Bas) La séquence des modes estimés.

Les vecteurs de paramètres identifiés sont donnés par

$$\hat{\theta}_1 = \begin{bmatrix} 1.4806 & -0.5315 & -0.0343 & 0.0873 & -0.0028 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} 1.1437 & -0.2285 & 0.0105 & 0.0564 & 0.0773 \end{bmatrix}.$$

pour le premier ensemble et par

$$\hat{\theta}_1 = \begin{bmatrix} 1.5157 & -0.5433 & -0.0219 & 0.0504 & -0.0005 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} 0.8585 & 0.0242 & -0.0039 & 0.0156 & 0.2118 \end{bmatrix},$$
$$\hat{\theta}_3 = \begin{bmatrix} 1.4911 & -0.5510 & 0.0804 & 0.0138 & -0.1201 \end{bmatrix},$$

pour le deuxième ensemble.
Contrairement à ceux du système hydrographique, les modes de la machine de montage de composants sur circuit imprimé et plus particulièrement le mode libre restent actifs pendant un temps bref avant de commuter (voir figure 5.9). Pour tenir compte de cela, nous choisissons pour appliquer l'algorithme 4.1, un temps de séjour minimum réduit $\tau = 10$. Le paramètre c est pris égal à 9. Nous rappelons que c doit toujours être inférieur à τ pour pouvoir calculer le degré d'appartenance des données aux sous-modèles. Le paramètre θ_{seuil} est choisi égal à 2. La sortie reconstruite à partir du modèle identifié superposée à la sortie du système ainsi que la séquence des modes estimés sont présentées sur la figure 5.14. Un FIT égal à 77.95 a été obtenu dans ce cas.
Les vecteurs de paramètres identifiés sont donnés par

$$\hat{\theta}_1 = \begin{bmatrix} 1.4966 & -0.7221 & -36.5434 & 0.0537 \end{bmatrix},$$
$$\hat{\theta}_2 = \begin{bmatrix} 1.2506 & -0.5571 & -35.1767 & 1.6807 \end{bmatrix}.$$

Influence du paramètre θ_{seuil} :
Comme nous l'avons constaté dans le chapitre précédent, pour une valeur donnée de c, le paramètre θ_{seuil} influe sur le nombre de sous-modèles identifiés récursivement et donc la précision du modèle. Nous rappelons que ce paramètre qui représente le seuil de confiance, permet de décider sur la création de nouveaux sous-modèles. Pour montrer cette influence sur des systèmes réels, nous avons appliqué l'algorithme 4.1 sur les trois ensembles de données considérés précédemment pour différentes valeurs de θ_{seuil}. Pour chaque valeur de θ_{seuil}, nous calculons le FIT et le nombre de sous-modèles identifiés. Les résultats de l'influence du paramètre θ_{seuil} sur l'identification récursive des trois ensembles sont respectivement rapportés sur les figures 5.16, 5.17 et 5.18. Nous constatons que plus θ_{seuil} augmente, plus le nombre de sous-modèles est grand. Ce qui est tout à fait normal puisque avec l'augmentation de θ_{seuil}, un degré d'appartenance d'une observation à classer se trouvant inférieur au seuil de confiance génère la création d'un nouveau mode. D'après les courbes du FIT, nous remarquons que la précision augmente avec l'augmentation de nombre de sous-modèles.

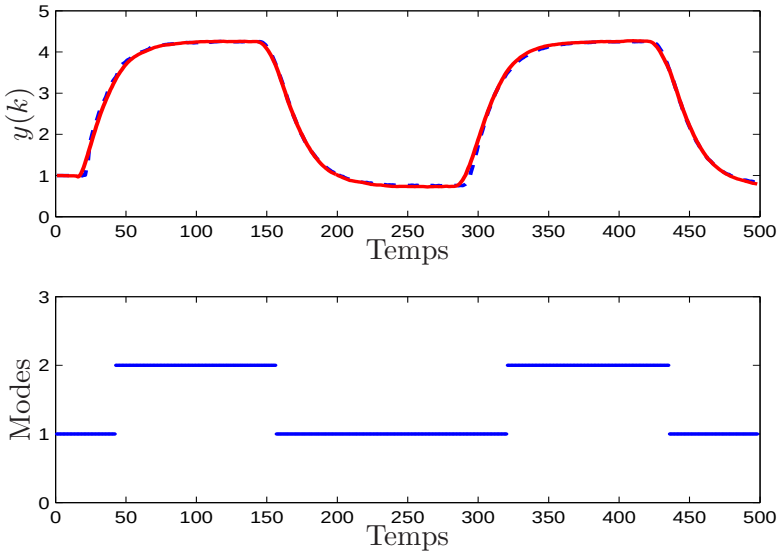


FIGURE 5.15 – Identification récursive du système hydrographique avec $\tau = 40$, $c = 30$ et $\theta_{seuil} = 4$. (Haut) La sortie réelle du système (en pointillé), la sortie reconstruite à partir des modèles PWA (en continu) . (Bas) La séquence des modes estimés.

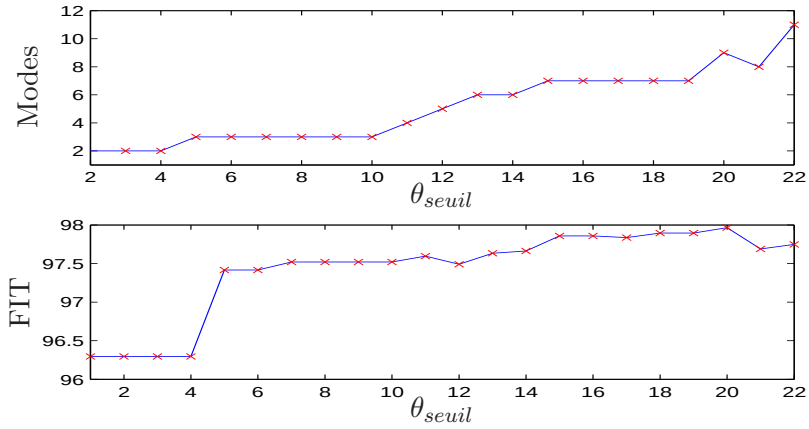


FIGURE 5.16 – Influence du paramètre θ_{seuil} sur l'identification du système hydrographique (Haut) Nombre de sous-modèles en fonction de θ_{seuil}. (Bas) Le FIT en fonction de θ_{seuil}.

5.5 Segmentation temporelle de vidéo par l'estimation de sous-modèles dynamiques

La segmentation temporelle de la vidéo en scènes est très utile pour la navigation et la gestion des données audiovisuelles. Par exemple, pour gérer les archives de télévision, les services responsables ont besoin d'outils automatiques permettant de segmenter

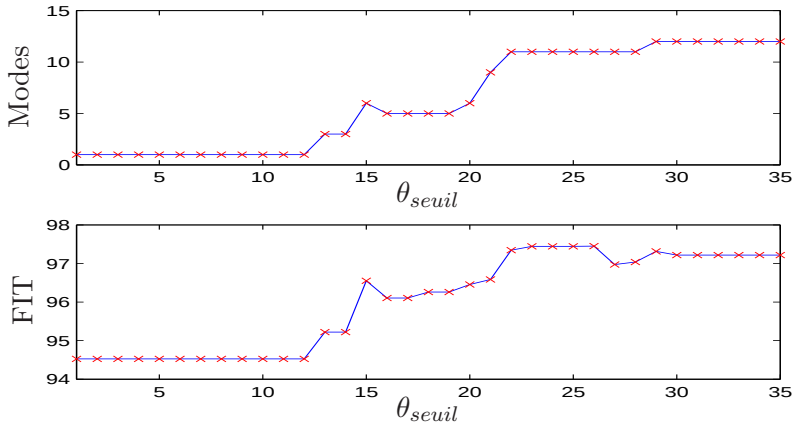


FIGURE 5.17 – Influence du paramètre θ_{seuil} sur l'identification du système hydrographique (Haut) Nombre de sous-modèles en fonction de θ_{seuil}. (Bas) Le FIT en fonction de θ_{seuil}.

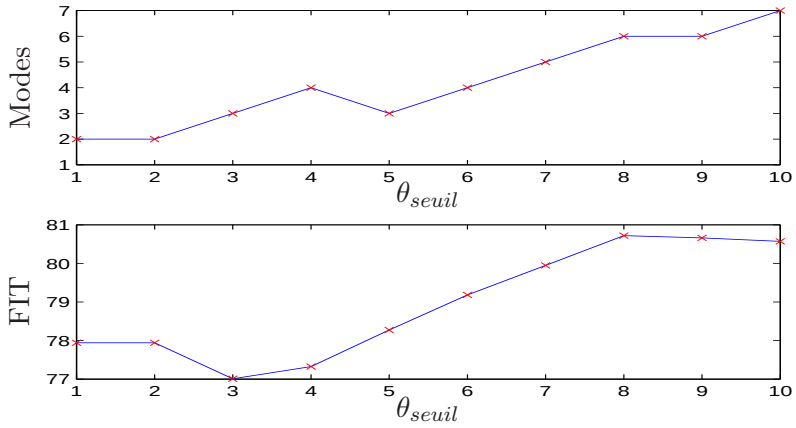


FIGURE 5.18 – Influence du paramètre θ_{seuil} sur l'identification de la machine de montage (Haut) Nombre de sous-modèles en fonction de θ_{seuil}. (Bas) Le FIT en fonction de θ_{seuil}.

un flux d'images en ses différentes prises de vues, scènes ou programmes, afin d'accéder rapidement à ceux-ci et de leur associer des informations pertinentes. La segmentation temporelle de la vidéo est obtenue en détectant les changements de scènes dans le flux vidéo. Chaque scène est un segment d'images visuellement cohérents. Chaque scène peut être modélisée par un modèle composé de plusieurs images ayant un contenu proche au sens d'un certain critère. Il s'agira par exemple d'un dialogue, d'une interview dans un journal télévisé, etc. Le problème de la segmentation en scènes reste délicat et son utilisation pour l'indexation et la recherche d'information en est pour l'instant à ses débuts. On trouve dans la littérature plusieurs travaux qui se sont intéressés à ce problème. Des états de l'art sur le sujet peuvent être trouvés par exemple dans [51], [13]. Très récemment quelques articles comme [84] [70] [69] et [44] ont appliqué des techniques d'identification

des systèmes hybrides [71] à la vision et en particulier à la segmentation spatio-temporelle de la vidéo. En nous basant sur ces techniques originales et prometteuses, nous proposons d'exploiter l'identification de systèmes linéaires commutants comme outils pour la segmentation temporelle de la vidéo en différentes scènes. Le modèle commutant obtenu est tel que chaque sous-modèle correspond à une scène.

Il existe deux types de transition entre les scènes [51] : brusque et progressive. Dans les transitions brusques, les changements sont instantanés d'une scène à une autre. Dans ce cas, la transition est plus simple à détecter. Les transitions progressives se produisent lorsque certains effets cinématographiques sont introduits, tels que le fondu d'image (transition de montage laissant voir une série d'images disparaître en diminuant leur luminosité jusqu'avoir une image sombre) ou le fondu enchainé (technique de transition progressive entre deux plans en diminuant la luminosité du premier et en augmentant la luminosité du second). Ces transitions sont plus difficiles à détecter que les transitions brusques.

Étant donnée une séquence d'images $\left\{I(t) \in \mathbb{R}^{l\times c},\ t = 1, \ldots, T\right\}$, la segmentation temporelle consiste à regrouper ces images en différentes scènes ou événements. Ce problème de segmentation peut être vu comme un problème de modélisation de systèmes linéaires commutants (Switched AutoRegressive SAR) où chaque sous-modèle correspondra à une scène. Pour cela, nous faisons l'hypothèse que les intensités des pixels des images obéissent à des modèles SAR. Chaque image $I(t)$ est une matrice de dimension $(l \times c)$ contenant les intensités lumineuses de tous les pixels. Chaque matrice $I(t)$ est alors transformée en un vecteur $y(t)$ de dimension $(lc \times 1)$ par concaténation de tous les vecteurs colonnes de la matrice $I(t)$. Ainsi, en prenant en compte l'hypothèse précédente, on peut écrire :

$$y(t) = \sum\nolimits_{j=1}^{n} a_j(\sigma_t) y(t-j) \tag{5.7}$$

où $\sigma_t \in \{1, \ldots, s\}$ est l'état discret (s est le nombre de scènes).

Si on pose $b_i = (a_n(i), \ldots, a_1(i), 1)^\top$ et $x_t = (y(t-n), \ldots, y(t-1), -y(t))^\top$ alors pour tout t, il existe i tel que

$$b_i^\top x_t = 0. \tag{5.8}$$

Le problème de la segmentation revient donc à regrouper les données x_t en s (s étant inconnu) sous-modèles. Chaque sous-modèle (ou scène) est défini par son vecteur normal b_i.

La procédure de segmentation basée sur l'estimation de sous-modèles locaux (l'algorithme 2.2) a été appliquée sur trois types de séquences vidéo. Toutes les séquences vidéo sont disponibles sur le site web "open video project" http ://www.open-video.org/. Par souci d'uniformité, chaque vidéo est convertie en format AVI avec une résolution de

240×320 à l'aide des fonctions standards de MATLAB. Comme le nombre de pixels est grand, les images sont d'abord projetées sur les trois premières composantes principales (ACP) [32].

5.5.1 Détection de transitions brusques

La première séquence vidéo est intitulée "Worlds Smaller than Saturn" et comporte cinq scènes différentes. Cette vidéo présente une interview avec des astronomes de l'Observatoire Keck à Mauna Kea, Hawaii. Dans cette première vidéo, la commutation d'une scène à une autre se fait d'une façon brusque. La procédure de classification de données et d'estimation de paramètres (Algorithme 2.2) a été appliquée avec $c = 50$. La figure 5.19 montre les résultats de segmentation de cette séquence. Cinq différentes scènes ont été identifiées. Un échantillon d'images de chaque scène ainsi que l'étiquette de chaque image en fonction du temps sont présentées sur cette figure.

Dans ce cas, la segmentation par notre approche est correcte car la commutation corres-

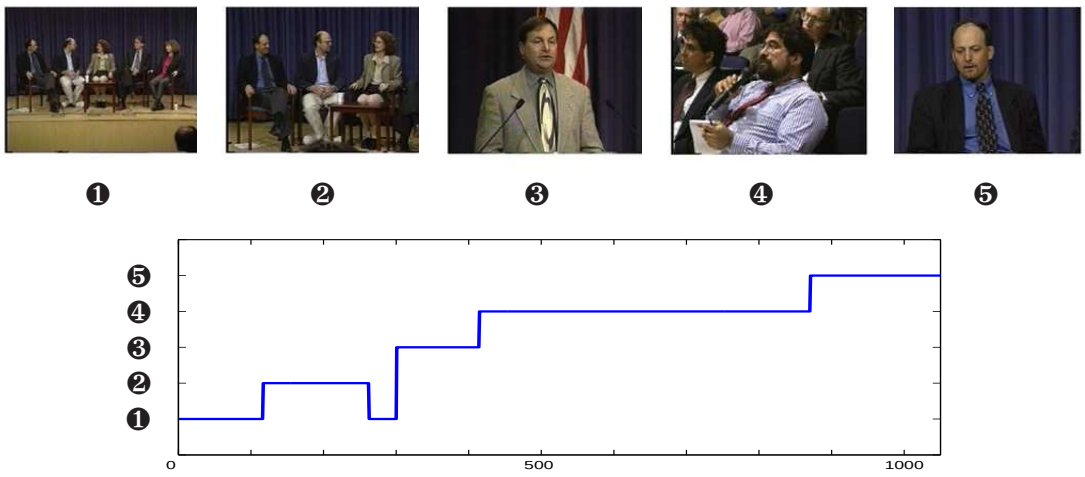


FIGURE 5.19 – Échantillon d'images des différentes scènes et labels des images en fonction du temps de la séquence "Worlds Smaller than Saturn".

pond exactement au changement de scènes. Les paramètres b_i correspondant aux différents sous-modèles sont :

$$b_1^\top = \begin{bmatrix} -0.09 & -0.06 & -256.16 & 1 \end{bmatrix}, b_2^\top = \begin{bmatrix} 0.29 & 0.26 & -393.85 & 1 \end{bmatrix},$$
$$b_3^\top = \begin{bmatrix} -0.90 & -1.80 & 891.26 & 1 \end{bmatrix}, b_4^\top = \begin{bmatrix} 0.25 & -0.18 & -830.73 & 1 \end{bmatrix},$$
$$b_5^\top = \begin{bmatrix} 0.06 & -0.91 & -222.66 & 1 \end{bmatrix}.$$

5.5.2 Détection de transitions progressives

La seconde séquence est intitulée "Hurricane Force - A Coastal Perspective". Elle comporte six différentes scènes. La technique du fondu enchainé est utilisée dans cette séquence. La figure 5.20 montre un fondu enchainé entre deux plans. Cet effet implique la présence de nombreux points de chevauchement entre les différents sous-modèles dans l'espace de régression.

FIGURE 5.20 – Transition progressive : fondu enchainé.

Nous avons appliqué l'algorithme 2.2 sur cette séquence vidéo avec $c = 45$. Nous montrons sur la figure 5.21 des échantillons d'images des six différentes scènes identifiées par notre algorithme avec les étiquettes de toutes les images en fonction du temps. Malgré la présence des effets du fondu enchainé dans cette vidéo, notre algorithme a réussi à détecter les changements de scènes.

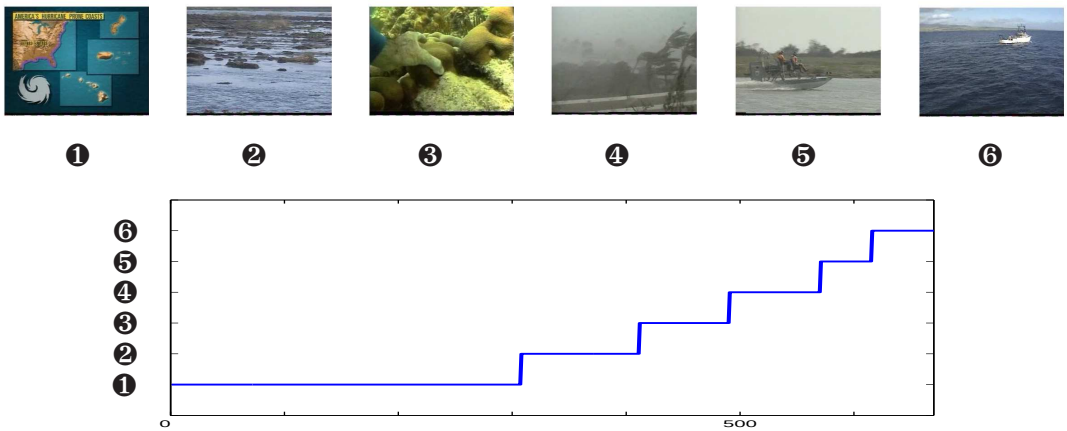


FIGURE 5.21 – Échantillon d'images des différentes scènes et labels des images en fonction du temps de la séquence "Hurricane Force - A Coastal Perspective".

Les paramètres b_i correspondant aux différents sous-modèles sont :

$$b_1^\top = \left[1.592.80 - 175.93\ 1\right], b_2^\top = \left[-0.112\ -0.67\ 2120.60\ 1\right],$$
$$b_3^\top = \left[0.400.12 - 339.35\ 1\right], b_4^\top = \left[0.231.13509.76\ 1\right],$$
$$b_5^\top = \left[0.005\ -1.07\ 3462.7\ 1\right], b_6^\top = \left[1.13\ -0.12\ -1490.0\ 1\right].$$

La troisième séquence est intitulée "The society raffles" est un court-métrage filmé avec un même arrière plan. Contrairement aux deux autres séquences, cette vidéo est prise par une même caméra en continu. Dans cette vidéo, la transition d'une scène à une autre se fait d'une façon progressive. Dans ce cas, il est difficile de localiser précisément, en terme de date, l'instant réel de commutation. En fait, la notion de commutation dans ce cas se rapporte à l'interprétation de la scène. Les scènes doivent être segmentées en fonction du sens sémantique de la prise de vue. L'algorithme 2.1 a été appliqué avec $c = 40$ et a permis de distinguer six différentes scènes. Les résultats de segmentation sont représentés sur la figure 5.22. Des images échantillons des différentes scènes et les étiquettes de toutes les images sont montrées sur cette figure. Chaque scène identifiée est différente (au sens sémantique) des autres.

Les résultats présentés sont satisfaisants et démontrent le potentiel de notre approche.

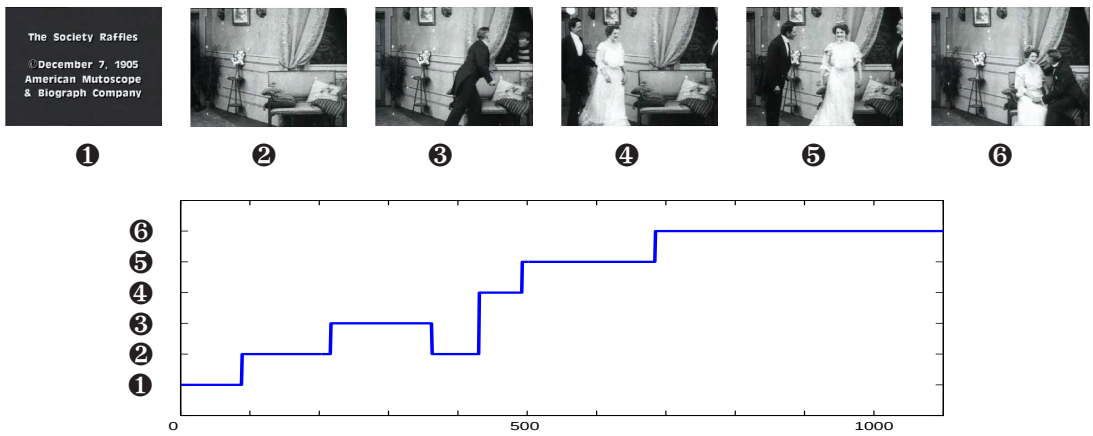


FIGURE 5.22 – Échantillon d'images des différentes scènes et labels des images en fonction du temps de la séquence "The society raffles".

Dans ce cas, les paramètres b_i correspondant aux différents sous-modèles sont :

$$b_1^\top = \begin{bmatrix} 0.19 & -0.00 & 0.32 & 1 \end{bmatrix}, b_2^\top = \begin{bmatrix} 0.07 & 0.11 & -496.45 & 1 \end{bmatrix},$$
$$b_3^\top = \begin{bmatrix} 0.49 & 0.88 & 393.47 & 1 \end{bmatrix}, b_4^\top = \begin{bmatrix} 0.17 & -0.27 & -112.72 & 1 \end{bmatrix},$$
$$b_5^\top = \begin{bmatrix} 0.35 & 1.56 & -815.30 & 1 \end{bmatrix}, b_6^\top = \begin{bmatrix} -0.04 & 0.36 & -1.42 & 1 \end{bmatrix}.$$

Influence du paramètre c sur la segmentation temporelle :
Nous étudions, dans ce paragraphe, l'influence du paramètre c sur le nombre de scènes identifiées par l'algorithme 2.2. Ce dernier a été appliqué sur les données issues des deux séquences vidéo : séquence 1 "Worlds Smaller than Saturn" et séquence 3 "The society raffles" pour différentes valeur de c. Nous faisons varier le paramètre c dans l'intervalle [10,200] avec un pas égal à 10. Le nombre de scènes, le nombre d'itérations ainsi que le temps de calcul sont rapportés sur la figure 5.23.

Comme cela a été vu précédemment, nous constatons que plus c est petit, plus le nombre de sous-modèles (ici scènes) est grand avec une évolution monotone, et ceci pour les deux séquences vidéo. Concernant la séquence 1 où la transition est brusque, nous remarquons que dans l'intervalle [50,90], le nombre de scènes identifiées est correcte et vaut 5. Avec l'augmentation de c, ce nombre a diminué à 4 dans tout l'intervalle [100,160]. Ceci est dû au fait que notre algorithme a fusionné les deux scènes 1 et 2 (voir figure 5.19) suite à la similarité qui existe entre elles. Pour la séquence 3, comme nous l'avons indiqué avant, le nombre de scènes dépend de l'interprétation de la scène et de son sens sémantique. Le nombre de scènes identifiées vaut 6, 4 et 3 respectivement dans les intervalles [40,70], [90,130] et [140,170]. Le nombre d'itérations est compris entre 3 et 7. Il vaut 3 pour la séquence 1 sur tout l'intervalle [60,200]. Le temps de calcul augmente légèrement avec l'augmentation du paramètre c.

5.6 Classification automatique de visages sous différents seuils d'éclairage par l'estimation de sous-modèles locaux

La base de données "Yale Face Database B" [38] (http ://cvc.yale.edu/projects/ yalefacesB/yalefacesB.html) contient 5760 images de 10 sujets vus sous 576 conditions de vision (9 poses (angles de vision) x 64 conditions d'éclairage). Dans notre application, nous n'avons considéré que la variation d'éclairage pour la classification automatique des images de la face frontale (position 0) de 7 sujets présentés sur la figure 5.25.

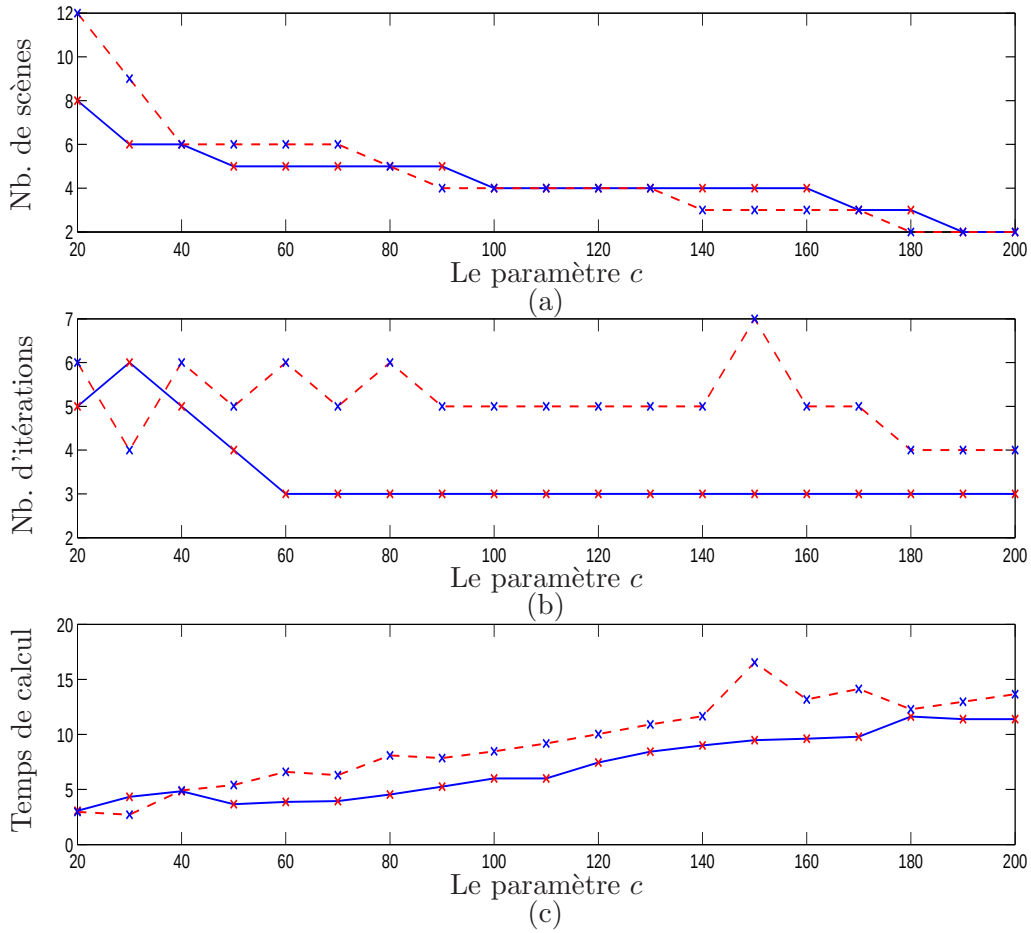


FIGURE 5.23 – Influence du paramètre c sur la segmentation temporelle de la séquence 1 (en continu) et de la séquence 3 (en pointillé) : (a) Nombre de scènes (b) Nombre d'itérations (c) Temps de calcul en secondes.

Les images de la base de données ont été capturées en utilisant un appareil d'éclairage construit à cet effet (voir Figure 5.24). Cette installation est équipée de 64 ordinateurs flashs contrôlés. Les 64 images d'un sujet particulier dans une position spécifique ont été acquises à une fréquence de 30 images/seconde. Environ 2 secondes ont été suffisantes pour effectuer 64 images pour chaque sujet. Durant ces 2 secondes il n'y a qu'un changement minime (invisible) de la position de la tête et de l'expression du visage.
Comme l'indique [41], l'ensemble des images d'un visage suivant un seul angle de vision et sous différentes conditions d'éclairage peut être approché par un sous-espace de faible dimension dans l'espace des images. Ainsi, la classification des visages peut être basée sur l'estimation de sous-modèle linéaire correspondant à chaque sous-espace.

FIGURE 5.24 – Appareil d'éclairage de la base de données "Yale Face Database B" [38].

Comme le nombre de pixels de chaque image est très grand, les images sont d'abord redimensionnées à 30 x 40 pixels puis transformées en des vecteurs de dimension 1200. Ensuite, une analyse en composantes principales (ACP) [32] est employée pour la réduction de la dimensionnalité. Nous représentons sur la figure 5.26 les données issues d'une ACP d'ordre 3 des 7 sujets. Les données de chaque sujet sont proches géométriquement et appartiennent à un même hyperplan.

Les données correspondant à un sujet spécifique peuvent être modélisées par un modèle affine. Cependant, vu le nombre de sujets considérés, les régions de validité des différents sous-modèles (après ACP) se chevauchent. Voilà pourquoi, nous n'allons considérer, dans un premier temps, que trois sujets qui sont les sujets 3, 4 et 7. Les régions qui correspondent aux modèles de ces trois sujets peuvent alors être séparées par des séparateurs non linéaires. Dans un deuxième temps, nous considérons les 7 sujets tous ensemble. Nous identifions les modèles dynamiques correspondant aux différents sujets sans nous soucier des régions correspondantes.

Sujets 3, 4 et 7 :
Dans ce cas, une analyse en composantes principales d'ordre 6 a été utilisée. L'algorithme 2.2 a été appliqué avec $c = 5$ sur la moitié des données pour une classification de données et une estimation des vecteurs de paramètres. L'algorithme 2.2 a permis de partitionner

FIGURE 5.25 – Échantillons d'images des sujets 1 à 7 sous différents seuils d'éclairage.

les données en 3 classes et d'estimer les vecteurs de paramètres qui sont donnés par :

$$\theta_1^\top = \begin{bmatrix} 0.1041 & -0.5446 & 0.4133 & 0.6942 & 0.5592 & -497.1862 \end{bmatrix},$$
$$\theta_2^\top = 10^3 \begin{bmatrix} 0.0011 & 0.0025 & -0.0015 & 0.0012 & -0.0007 & -3.2167 \end{bmatrix},$$
$$\theta_3^\top = 10^4 \begin{bmatrix} 0.0003 & 0.0002 & -0.0000 & 0.0002 & -0.0002 & -1.2637 \end{bmatrix}.$$

Après analyse des résultats, nous avons constaté que chaque classe des données partitionnées correspond bien à un sujet. Toutes les étiquettes estimées coïncident avec les vraies étiquettes. Une fois ces données regroupées selon leurs sous-modèles respectifs, les frontières séparatrices non-linéaires des régions correspondantes sont estimées.

L'autre moitié des données est utilisée pour la validation des modèles identifiés. Pour

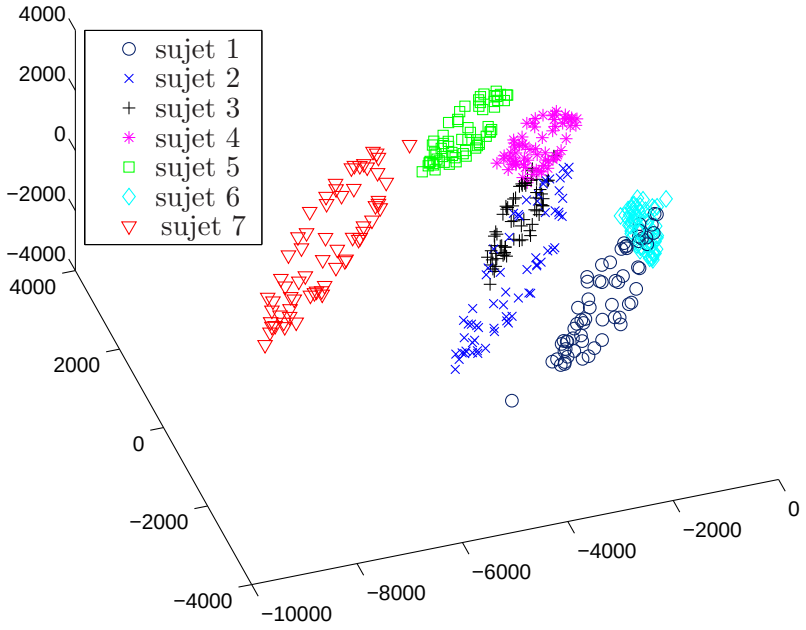


FIGURE 5.26 – Représentation des données issues d'une ACP d'ordre 3 des 7 sujets.

	σ	500	800	3000
Classification hors ligne	#sv	61, 70, 73	32, 36, 38	7, 7, 8
Validation sans adaptation	#sv	61, 70, 73	32, 36, 38	7, 7, 8
	Précision	58.823	76.470	100
Validation avec adaptation	#sv	114, 126, 142	56, 67, 75	10, 12, 16
	Précision	99.019	100	100

TABLE 5.3 – Nombre de vecteurs de support et précision de la classification et de la validation pour différentes valeurs de σ.

montrer l'avantage du caractère adaptatif de nos procédures, la validation est effectuée de deux façons. Dans le premier cas où aucune adaptation n'est effectuée, les données sont étiquetées à partir des fonctions de décision définissant les régions sans que ces fonctions séparatrices ni que les vecteurs de paramètres ne soient mis à jour. Dans le deuxième cas, les vecteurs de paramètres ainsi que les fonctions des frontières séparatrices sont adaptées après chaque incorporation d'une nouvelle donnée.

Pour adapter les frontières séparatrices des régions, nous utilisons le classifieur multi-

Le paramètre c	5	10	15	20
Précision	96.21	92.85	83.61	79.41

TABLE 5.4 – Précision en fonction du paramètre c.

classe M-SVMID en choisissant un noyau Gaussien avec différentes valeurs du paramètre σ. Les résultats de la validation sont rapportés sur le tableau 5.3. Dans ce tableau, nous montrons le nombre de vecteurs de support ainsi que la précision calculée pour différentes valeurs du paramètre σ de la fonction noyau. La précision est définie par :

$$\text{Précision} = 100 \times \frac{\text{Nombre de données} - \text{Nombre de données mal classées}}{\text{Nombre de données}}$$

Nous constatons que quelle que soit la valeur de σ, la précision trouvée dans le cas de la validation avec adaptation de paramètres est bien meilleure que celle trouvée sans adaptation de paramètres. Ceci montre que l'adaptabilité des procédures proposées corrige les faux labels (dans le cas de la validation sans adaptation) des images.

Tous les sujets à la fois :
Dans ce cas, une analyse en composantes principales d'ordre 6 a été utilisée. L'algorithme 2.2 a été appliqué, sur la moitié des données, avec $c = 10$. Sept modèles locaux ont été identifiés. L'autre moitié est utilisée pour la validation avec adaptation des vecteurs de paramètres en ligne (pour la reconnaissance de visage). Nous nous basons, pour l'étiquetage en ligne des données, sur la mesure de similarité définie par l'équation (2.13). Le modèle retenu est celui qui correspond au maximum de cette mesure. Ce modèle est mis à jour par la méthode des moindres carrés récursifs.
Il est à noter que le système considéré ici n'est pas un système à temps de séjour, c'est-à-dire que les sujets ne sont pas traités dans l'ordre. Cela justifie notre démarche qui consiste à effectuer d'abord une modélisation hors ligne sur la moitié de données. Cette modélisation nous permet de disposer d'une base d'apprentissage qui sera enrichie au fur et à mesure dans le temps.
La validation est effectuée sur l'autre moitié des données, pour différentes valeurs de c, tout en adaptant les vecteurs de paramètres après l'incorporation de données. Nous montrons sur le tableau 5.4, la précision calculée pour différentes valeurs de c. Nous constatons que plus c est petit, meilleure est la précision. Ceci est dû au chevauchement de certains sous-modèles.

Les vecteurs de paramètres finaux sont donnés par :

$$\theta_1^\top = \begin{bmatrix} -0.39 & -1.27 & 2.33 & 2.60 & -1.25 & 44.92 \end{bmatrix},$$
$$\theta_2^\top = \begin{bmatrix} 0.2074 & -0.2567 & -0.3302 & -0.2609 & 0.4954 & 226.35 \end{bmatrix},$$
$$\theta_3^\top = 10^3 \begin{bmatrix} -0.0009 & -0.0008 & 0.0018 & 0.0020 & -0.0004 & -3.39 \end{bmatrix},$$
$$\theta_4^\top = 10^3 \begin{bmatrix} -0.0006 & -0.0008 & -0.0009 & -0.0004 & -0.0004 & 1.75 \end{bmatrix},$$
$$\theta_5^\top = 10^3 \begin{bmatrix} -0.0006 & -0.0007 & -0.0011 & -0.0006 & -0.0003 & 2.84 \end{bmatrix},$$
$$\theta_6^\top = 10^3 \begin{bmatrix} -0.0022 & 0.0017 & -0.0009 & -0.0008 & -0.0008 & -3.98 \end{bmatrix},$$
$$\theta_7^\top = 10^4 \begin{bmatrix} -0.0002 & 0.0002 & -0.0001 & 0.0001 & 0.0002 & -1.25 \end{bmatrix}.$$

Cela justifie la possibilité de modéliser chaque individu malgré une large plage de variation de l'éclairage. Cependant dans ce cas plus complexe, il'est impossible de discerner les modèles des individus selon leurs régions de validité.

5.7 Conclusion

Ce chapitre a été consacré à la validation expérimentale des algorithmes développés. Nous avons montré, à travers différentes applications, les performances et la faisabilité des approches proposées. Nos approches ont été tout d'abord appliquées à l'identification de systèmes hydrauliques à surface libre. Puis, elles ont été appliquées à la modélisation d'une machine de montage de composants électroniques sur circuit imprimé. Nous avons pu démontrer via ces deux applications que les algorithmes 2.1 et 2.2 permettent d'obtenir une modélisation satisfaisante et ceci avec un temps de calcul correct. Ensuite, une identification récursive du système hydraulique et de la machine de montage a été effectuée par l'algorithme 4.1. Les résultats obtenus pour ces cas sont également bons.

Nos méthodes ont été ensuite étendues au domaine de la vision par ordinateur. Nous avons ainsi présenté une nouvelle méthode de segmentation temporelle de la vidéo en différentes scènes basée sur une estimation de sous-modèles locaux linéaires. Ensuite nous avons présenté une application qui concerne la classification automatique et la reconnaissance de visages sous différents seuils d'éclairage par l'estimation de sous-modèles dynamiques locaux. Les résultats présentés sont satisfaisants et démontrent le potentiel de nos approches.

Conclusion et perspectives

1 Conclusion

Ces dernières années, l'identification de modèles PWA a bénéficié d'une attention croissante, motivée par le développement d'outils avancés pour l'analyse, la vérification et le contrôle des systèmes hybrides, et l'équivalence entre les modèles PWA et plusieurs classes de systèmes hybrides. Dans cette thèse, nous avons proposé de nouvelles procédures pour l'identification de modèles PWA à partir des données entrée-sortie. Nous avons également proposé une approche d'identification récursive des modèles PWA et PWN variant dans le temps.

Le premier chapitre de ce mémoire a été consacré à un état de l'art sur l'identification des systèmes dynamiques hybrides. Nous avons fait le point sur les méthodes d'identification des modèles PWA qui existent dans la littérature. Nous avons ainsi montré leurs avantages et inconvénients. Nous avons constaté que la principale difficulté dans la construction des modèles PWA est que l'estimation des sous-modèles est très rarement dissociée du problème de classification des données, à savoir, l'attribution de chaque donnée au sous-modèle qui l'a générée. Cela justifie notre démarche basée sur le couplage des outils de classification et d'identification. En effet, nous avons développé de nouvelles méthodes d'identification basées sur des techniques innovantes de classification non supervisée combinées avec des techniques de régression dans le but de séparer les données selon leurs sous-modèles respectifs et d'estimer ainsi les paramètres associés à ces sous-modèles. L'approche de classification que nous avons proposée est inspirée de l'algorithme de classification des k-plus proches voisins « k-ppv ». Dans notre démarche, chaque plus proche voisin d'un vecteur donné fournit une information sur l'affectation de ce vecteur. La classification des données a été ensuite modifiée en introduisant une modélisation des connaissances incertaines par la théorie de Dempster-Shafer. Les procédures proposées dans le chapitre 2 ne nécessitent que la connaissance des ordres du système. En plus, un seul paramètre de réglage doit être fourni par l'utilisateur.

Nous avons ensuite procédé à l'estimation de la loi de commutation des modèles PWA.

Cette loi est définie par une partition du domaine de régression. Chaque sous-modèle est valide sur une région bien définie de cette partition. Pour estimer les paramètres de ces régions, nous avons tout d'abord utilisé des classifieurs binaires de type « un-contre-reste » et « un-contre-un ». Nous avons constaté que ce genre de classifieur ne permet pas en général d'obtenir une partition complète du domaine de régression. Nous avons alors proposé un classifieur multi-classe permettant de séparer les régions en considérant simultanément l'ensemble de toutes les données et en recherchant directement un séparateur linéaire par morceaux.

Nous avons ensuite considéré des modèles qui peuvent évoluer dans le temps. Ces évolutions concernent les paramètres des sous-modèles et des régions. Nous avons développé, pour ce genre de modèles, un algorithme générique d'identification récursive. Cet algorithme est doté de procédures d'adaptation lui permettant de prendre en compte des changements et des modifications des paramètres des modèles dans le temps et donc de suivre leur évolution. Ces procédures concernent l'adaptation des paramètres de chaque sous-modèle linéaire où non linéaire mais aussi l'adaptation des paramètres des régions. L'adaptation des paramètres des sous-modèles linéaires est effectuée par un algorithme de moindres carrés récursifs à fenêtre glissante (FW-RLS). Les paramètres des sous-modèles non linéaires, quant à eux, sont mis à jour grâce à un algorithme de régression par des LS-SVM récursifs (LS-SVMR). Enfin, la mise à jour des paramètres des frontières séparatrices des régions dans le temps est assurée par un nouveau classifieur incrémental et décrémental multi-classe à vecteurs de support (MID-SVM) que nous avons développé durant ce travail de thèse.

La dernière partie de ce mémoire a été réservée à la validation expérimentale. Plusieurs résultats expérimentaux ont montré la capacité et les performances des approches développées. Nous avons tout d'abord modélisé un système hydraulique à surface libre puis une machine de montage de composants électroniques sur circuit imprimé. Nous avons également appliqué nos approches à la segmentation temporelle de vidéos en différentes scènes et à la classification automatique et la reconnaissance de visages sous différents seuils.

2 Perspectives

Au niveau des perspectives, certains points restent à améliorer et à analyser dans les travaux futurs. Ces points sont résumés dans les paragraphes suivants :

– La nécessité de trouver des règles pour la sélection automatique du paramètre c permettant d'automatiser complètement la procédure d'identification des algorithmes

2.1 et 2.2. La sélection du paramètre c doit tenir compte de la densité des données dans l'espace de régression, du nombre total de données, du nombre minimal de données dans un sous-modèle, etc. Il serait également intéressant de trouver une relation de dépendance entre le paramètre c et le nombre de sous-modèles obtenus s. Cette tâche paraît plus complexe car d'après les résultats expérimentaux, nous avons constaté que cette relation varie d'un système à un autre.

- Un deuxième point à développer concerne la prise en compte de sous-modèles ayant des différents ordres. Dans les modèles considérés dans nos travaux, les ordres des différents sous-modèles sont tous égaux. En réalité, dans le cas général, ces ordres peuvent varier d'un sous-modèle à un autre. Cela rend en fait la tâche de classification de données plus complexe. En effet, la dimension des données à regrouper ne sera plus la même. Une mesure de similarité appropriée doit être alors développée. D'autre part, il est également important de dériver des procédures d'estimation des ordres des sous-modèles.
- L'analyse de convergence, cette tâche consiste à trouver des conditions théoriques dans lesquelles la convergence peut être garantie. L'étude des propriétés de convergence peut présenter quelques difficultés quand il s'agit d'algorithmes itératifs, comme c'est le cas pour les algorithmes 2.1 et 2.2. Dans ce cas, il est nécessaire de poser quelques hypothèses (par exemple borner le niveau de bruit, etc) afin d'atteindre cet objectif. Les conditions théoriques de convergence peuvent par exemple être inspirées de [34].
- Un autre point à développer concerne l'algorithme 4.1. Il s'agit de contourner la contrainte de temps de séjour. En fait, cette contrainte a été introduite en vue de réduire les erreurs d'affectation de données aux différents modes dans un contexte non stationnaire (modèles évoluant dans le temps). En l'absence de la contrainte de temps de séjour, une solution consisterait par exemple à initialiser de façon aléatoire les paramètres de s (fixe et connu à l'avance) sous-modèles, puis à affecter les données une après l'autre aux sous-modèles tout en adaptant leurs paramètres. L'affectation de données est basée sur un critère d'appartenance spécifique (voir par exemple [4]).
- Le développement de modules complémentaires permettant d'assurer le suivi en ligne des différents modes composant le système dans le but d'effectuer le diagnostic et le pronostic de ce système. Le suivi de la dynamique d'évolution entre les modes (détection de défaillance) ou la caractérisation des dérives des paramètres d'un mode seront ainsi rendus possible grâce à ces modules.
- Dans la partie validation expérimentale, nous n'avons considéré que la segmentation temporelle de vidéos. Il est souhaitable d'étendre ce travail pour traiter le cas de la segmentation spatio-temporelle de vidéos. Dans ce cas, la loi de commutation dépend

du temps et des coordonnées des pixels. La segmentation spatio-temporelle permet de fournir plus de détails concernant la dynamique de la scène et ainsi améliore la modélisation des objets dynamiques (formes en déplacement) au sein d'une scène donnée.

Bibliographie

[1] M. Çamlibel, W. Heemels et J. Schumacher : Consistency of a time-stepping method for a class of piecewise-linear networks. *IEEE Transactions on Circuits and Systems : Fundamental Theory and Applications*, 49(3):349–357, 2002.

[2] L. Bako : *Contribution à l'identification de systèmes dynamiques hybrides*. Thèse de doctorat, Université des sciences et Technologies de Lille - Lille 1, 2008.

[3] L. Bako, K. Boukharouba, E. Duviella et S. Lecoeuche : Switched affine models for describing nonlinear systems. *In IFAC Conference on Analysis and Design of Hybrid Systems, Zaragoza, Spain*, 2009.

[4] L. Bako, K. Boukharouba, E. Duviella et S. Lecoeuche : A recursive identification algorithm for switched linear/affine models. *Nonlinear Analysis :Hybrid Systems*, 5(2):242–253, 2010.

[5] L. Bako et R. Vidal : Algebraic identification of switched MIMO ARX models. *In Hybrid Systems : Control and Computation, St Louis MO, USA*, 2008.

[6] R. Batruni : A multilayer neural network with piecewise-linear structure and back-propagation learning. *IEEE Transactions on Neural Networks*, 2(3):395–403, 1991.

[7] A. Bemporad, G. Ferrari-Trecate et M. Morari : Observability and controllability of piecewise affine and hybrid systems. *IEEE Transactions on Automatic Control*, 45(10):1864–1876, 2000.

[8] A. Bemporad, A. Garulli, S. Paoletti et A. Vicino : A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, 50(10):1567–1580, 2005.

[9] A. Bemporad et M. Morari : Control of systems integrating logic, dynamics and constraints. *Automatica*, 35(3):407 – 427, 1999.

[10] A. Bemporad, J. Roll et L. Ljung : Identification of hybrid systems via mixed-integer programming. *In 40th IEEE Conference on Decision and Control, Orlando, Fl, USA*, 2001.

[11] K. P. Bennett et O. L. Mangasarian : Multicategory discrimination via linear programming. *Optimization Methods and Software*, 3:27 – 39, 1994.

[12] D. S. Bernstein : *Matrix Mathematics : Theory, Facts, and Formulas with Application to Linear Systems Theory*. Princeton University Press, 2005.

[13] J. S. Boreczky et L. A. Rowe : Comparison of video shot boundary detection techniques. *In Storage and Retrieval for Image and Video Databases*, 1996.

[14] B. E. Boser, I. M. Guyon et V. N. Vapnik : A training algorithm for optimal margin classifiers. *In the 5th annual ACM workshop on computational learning theory*, 1992.

[15] K. Boukharouba, L. Bako et S. Lecoeuche : Identification of piecewise affine systems based on Dempster-Shafer theory. *In IFAC Symposium on System Identification, Saint Malo, France*, 2009.

[16] K. Boukharouba, L. Bako et S. Lecoeuche : Incremental and decremental multicategory classification by support vector machines. *In IEEE International Conference on Machine Learning and Applications, Miami Beach, FL, USA*, 2009.

[17] K. Boukharouba, L. Bako et S. Lecoeuche : Segmentation temporelle de la vidéo par l'estimation de sous-modèles dynamiques locaux. *In XVII Rencontres de la Société Francophone de Classification, Saint-Denis de la Réunion*, 2010.

[18] K. Boukharouba, L. Bako et S. Lecoeuche : Temporal video segmentation using a switched affine model identification technique. *In IEEE International Conference on Image Processing Theory, Tools and Applications, Paris, France*, 2010.

[19] K. Boukharouba, E. Duviella, L. Bako et S. Lecoeuche : Modélisation de la dynamique des systèmes hydrauliques à surface libre par l'approche des systèmes hybrides. *In Sixième Conférence Internationale Francophone d'Automatique, Nancy, France*, 2010.

[20] K. Boukharouba, E. Duviella, L. Bako et S. Lecoeuche : Multimodeling vs piecewise affine modeling for the identification of open channel systems. *In IFAC Large Scale Systems : Theory and Applications, Villeneuve dAscq, France*, 2010.

[21] S. Boyd et L. Vandenberghe : *Convex Optimization*. Cambridge University Press, March 2004.

[22] E. Bredensteiner et K. Bennett : Multicategory classification by support vector machines. *Computational Optimization and Applications*, 12:53–79, 1999.

[23] L. Breiman : Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, 39(3):999 – 1013, 1993.

[24] G. Cauwenberghs et T. Poggio : Incremental and decremental support vector machine learning. *In Advances in Neural Information Processing Systems, Vancouver, British Columbia, Canada*, 2001.

[25] C. Cortes et V. Vapnik : Support-vector networks. *In Machine Learning*, p. 273–297, 1995.

[26] N. Cristianini et J. Shawe-Taylor : *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, March 2000.

[27] B. De Schutter : Optimal control of a class of linear hybrid systems with saturation. *SIAM Journal on Control and Optimization*, 39(3):835–851, 2000.

[28] T. Denoeux : A k-nearest neighbor classification rule based on Dempster-Shafer theory. *IEEE Transactions on Systems, Man and Cybernetics*, 25(05):804–813, 1995.

[29] A. J. V. der Schaft et J. M. Schumacher : Complementarity modelling of hybrid systems. *IEEE Transactions on Automatic Control*, 43:483–490, 1998.

[30] F. Ding et Y. Xiao : A finite-data-window least squares algorithm with a forgetting factor for dynamical modeling. *Applied Mathematics and Computation*, 186:184–192, 2007.

[31] A. Doucet, A. Logothetis et V. Krishnamurthy : Particle filters for state estimation of Jump Markov Linear systems. *IEEE Transactions on Signal Processing*, 49(3):613–624, 2001.

[32] R. Duda, P. Hart et D. Stork : *Pattern classification.* Wiley, New York. 2nd edition, 2000.

[33] R. O. Duda et P. E. Hart : *Pattern Classification and Scene Analysis*. New York : Wiley, 1973.

[34] G. Ferrari-Trecate et M. Muselli : Single-linkage clustering for optimal classification in piecewise affine regression. *In IFAC Conference on the Analysis and Design of Hybrid Systems, Saint Malo, France*, 2003.

[35] G. Ferrari-Trecate, M. Muselli, D. Liberati et M. Morari : A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205 – 217, 2003.

[36] J. H. Friedman : Another approach to polychotomous classification. Rap. tech., Department of Statistics, Stanford University, 1996.

[37] E. F. Gad, A. F. Atiya, S. Shaheen et A. El-Dessouki : A new algorithm for learning in piecewise-linear neural networks. *Neural Networks*, 13(4-5):485–505, 2000.

[38] A. Georghiades, P. Belhumeur et D. Kriegman : From few to many : Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[39] W. P. M. H. Heemels, B. De Schutter et A. Bemporad : Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.

[40] W. P. M. H. Heemels, J. M. Schumacher et S. Weiland : Linear complementarity systems. *SIAM Journal of Applied Mathematics*, 60:1234–1269, 2000.

[41] J. Ho, M. H. Yang, J. Lim, K. C. Lee et D. Kriegman : Clustering apperances of objects under varying illumination conditions. *In IEEE Conference on Computer Vision and Pattern Recognition, Madison, WI, USA*, 2003.

[42] C. Hsu et C. Lin : A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425, 2002.

[43] X. Hu et L. Xu : A comparative study of several cluster number selection criteria. *In Intelligent Data Engineering and Automated Learning*, vol. 2690 de *Lecture Notes in Computer Science*, p. 195–202. Springer Berlin / Heidelberg, 2003.

[44] K. Huang, A. Wagner et Y. Ma : Identification of hybrid linear time-invariant systems via subspace embedding and segmentation. *In IEEE Conference on Decision and Control, Atlantis, Paradise Island, Bahamas*, 2004.

[45] P. Julian, A. Desages et O. Agamennoni : High level canonical piecewise linear representation using a simplicial partition. *IEEE Transactions on Circuits and Systems, Fundamental Theory and Applications*, 46(4):463–480, 1999.

[46] P. M. Julian : *A High Level Canonical Piecewise Linear Representation : Theory and Applications*. Thèse de doctorat, Departamento de Ingenieria Electrica, Universidad Nacional del Sur, Argentina, 1999.

[47] A. Juloski : *Observer design and identification methods for hybrid systems*. Thèse de doctorat, Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands, 2004.

[48] A. Juloski, W. Heemels et G. Ferrari-Trecate : Data-based hybrid modelling of the component placement process in pick-and-place machines. *Control Engineering Practice*, 12(10):1241–1252, 2004.

[49] A. Juloski, S. Wieland et W. Heemels : Bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, 50(10):1520 – 1533, 2005.

[50] S. Knerr, L. Personnaz et G. Dreyfus. : Single-layer learning revisited : a stepwise procedure for building and training a neural network. *Neurocomputing : Algorithms, Architectures and Applications*, F68:41–50, 1990.

[51] I. Koprinska et S. Carrato : Temporal video segmentation : A survey. *Signal Processing : Image Communication*, 16:477–500., 2001.

[52] U. Kreβel : Pairwise classification and support vector machines. *In B. Schölkopf, C. J. C. Burges, & A. J. Smola (Eds.), Advances in kernel methods : Support vector learning. Cambridge, MA : MIT Press*, 1999.

[53] H. W. Kuhn et A. W. Tucker : Nonlinear programming. *In Proceedings of 2nd Berkeley Symposium. Berkeley : University of California Press*, 1951.

[54] F. Lauer : *From Support Vector Machines to Hybrid System Identification*. Thèse de doctorat, Centre de Recherche en Automatique de Nancy, Université Henri Poincaré - Nancy 1, France, 2008.

[55] F. Lauer et G. Bloch : A new hybrid system identification algorithm with automatic tuning. *In IFAC World Congress, COEX, Korea, South*, 2008.

[56] F. Lauer et G. Bloch : Switched and piecewise nonlinear hybrid system identification. *In 11th International Conference on Hybrid Systems : Computation and Control, St. Louis, MO, USA*, 2008.

[57] F. Lauer, R. Vidal et G. Bloch : A product-of-errors framework for linear hybrid system identification. *In IFAC Symposium on System Identification, Saint Malo, France*, 2009.

[58] Y. Lee, Y. Lin et 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(465):67–81, 2004.

[59] X. Litrico et D. Georges : Robust continuous-time and discrete-time flow control of a dam-river system. (i) modelling. *Applied Mathematical Modelling*, 23:809–827, 1999.

[60] L. Ljung : *System identification : theory for the user*. Prentice-Hall, Upper Saddle River, NJ, 1999.

[61] L. Ljung et T. Söderström : *Theory and Practice of Recursive Identification*. MIT Press, Cambridge, 1983.

[62] Y. Ma et R. Vidal : Identification of deterministic switched arx systems via identification of algebraic varieties. *In Hybrid systems computation and control, Zurich, Switzerland*, 2005.

[63] M. C. Medeiros, M. G. C. Resende et A. Veiga : Piecewise linear time series estimation with grasp. Rap. tech., AT&T Labs Research, Florham Park, NJ 07932 USA, 1999.

[64] M. C. Medeiros, A. Veiga et M. G. C. Resende : A combinatorial approach to piecewise linear time series analysis. Rap. tech., AT&T Labs Research, Florham Park, NJ 07932 USA, 1999.

[65] C. Men et W. Wang : Selection of gaussian kernel parameter for svm based on convex estimation. *In Advances in Neural Networks*, vol. 5263 de *Lecture Notes in Computer Science*, p. 709–714. Springer Berlin / Heidelberg, 2008.

[66] M. Milanese et A. Vicino : Optimal estimation theory for dynamic systems with set membership uncertainty : an overview. *Automatica*, 27(6):997–1009, 1991.

[67] H. Nakada, K. Takaba et T. Katayama : Identification of piecewise affine systems based on statistical clustering technique. *Automatica*, 41:905–913, 2005.

[68] H. Niessen, A. Juloski, G. Ferrari-Trecate et W. Heemels : Comparison of three procedures for the identification of hybrid systems. *In International Conference on Control Applications, Taipei , Taiwan*, 2004.

[69] N. Ozay, C. Lagoa et M. Sznaier : Robust identification of switched affine systems via moments-based convex optimization. *In 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, Shanghai, China*, 2009.

[70] N. Ozay, M. Sznaier, C. M. Lagoa et O. I. Camps : A sparsification approach to set membership identification of a class of affine hybrid systems. *In 47th IEEE Conference on Decision and Control, Cancun, Mexico*, 2008.

[71] S. Paoletti, A. Juloski, G. Ferrari-Trecate et R. Vidal : Identification of hybrid systems : a tutorial. *European Journal of Control*, 513(2-3):242 – 260, 2007.

[72] M. Pontil et A. Verri : Properties of support vector machines. *Neural Computation*, 10:955 – 974, 1997.

[73] P. Pucar et J. Sjöberg : On the hinge-finding algorithm for hinging hyperplanes. *IEEE Transactions on Information Theory*, 44(3):1310 – 1319, 1998.

[74] J. Ragot, G. Mourot et D. Maquin : Parameter estimation of switching piecewise linear systems. *In the 42nd IEEE Conference on Decision and Control, Maui, Hawaii, USA*, 2003.

[75] J. Roll : *Local and piecewise affine approaches to system identification.* Thèse de doctorat, Departement of Electrical Engineering, Linköping University, 2003.

[76] K. Sentz : *Combination of Evidence in Dempster-Shafer Theory.* Thèse de doctorat, Systems Science and Industrial Engineering Department. Thomas J. Watson School of Engineering and Applied Science. Binghamton University, 2002.

[77] G. Shafer : A mathematical theory of evidence. *In Princeton University Press, Princeton, N.J.*, 1976.

[78] P. Smets et R. Kennes : The transferable belief model. *Artificial Intelligence*, 66:191–234, 1994.

[79] E. D. Sontag : Nonlinear regulation : The piecewise linear approach. *IEEE Transactions on Automatic Control*, 26:346–357, 1981.

[80] J. Suykens et J. Vandewalle : Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

[81] V. Vapnik : The nature of statical learning theory. *In New York : Springer-Verlag*, 1995.

[82] R. Vidal : Identification of PWARX hybrid models with unknown and possibly different orders. *In IEEE American Control Conference, Boston MA, USA*, 2004.

[83] R. Vidal, A. Chiuso et S. Soatto : Observability and identifiabilty of jump linear systems. *In IEEE Conference on Decision and Control, Sydney, Australia*, 2002.

[84] R. Vidal, S. Soatto et A. Chiuso. : Applications of hybrid system identification in computer vision. *In European Control Conference, Kos, Greece*, 2007.

[85] R. Vidal, S. Soatto, Y. Ma et S. Sastry : An algebraic geometric approach to the identification of a class of linear hybrid systems. *In 42nd IEEE Conference on Decision and Control, Maui, Hawaii, USA*, 2003.

[86] J. Weston et C. Watkins : Multi-class support vector machines. Rap. tech., Royal Holloway, University of London, Department of Computer Science, 1998.

A

A.1 Recherche d'un hyperplan séparateur linéaire

Nous allons présenter dans cette annexe la méthode générale de construction du séparateur linéaire optimal entre deux régions, d'abord dans le cas de données linéairement séparables, ensuite dans le cas de données non linéairement séparables.

A.1.1 Cas de données linéairement séparables

Soit un ensemble de données $D = \{\varphi(1), \ldots, \varphi(N)\}$. Le problème de la classification linéaire consiste à trouver des paramètres w et b tels que

$$\begin{array}{ll} w^\top \varphi(i) + b > 0 & \text{si} \quad \varphi(i) \in \Re_1 \\ w^\top \varphi(i) + b < 0 & \text{si} \quad \varphi(i) \in \Re_2. \end{array} \tag{A.1}$$

La fonction signe$(w^\top \varphi(i) + b)$ est appelée classifieur. Ce classifieur divise l'espace des données en deux demi-espaces correspondant chacun à une région. Cette séparation est réalisée par l'hyperplan $H(w, b)$ défini par l'équation $w^\top \varphi(i) + b = 0$.

Les deux régions $\Re_1$ et $\Re_2$ sont dites linéairement séparables ssi il existe (w, b) tel que : $\forall \varphi(i) \in \Re_1 : w^\top \varphi(i) + b > 0$ et $\forall \varphi(i) \in \Re_2 : w^\top \varphi(i) + b < 0$. Dans ce cas, il est possible qu'il existe une multitude d'hyperplans valides permettant de séparer les données des deux régions (voir figure A.1). L'objectif serait alors de trouver l'hyperplan optimal qui permettra de classer au mieux les nouvelles données.

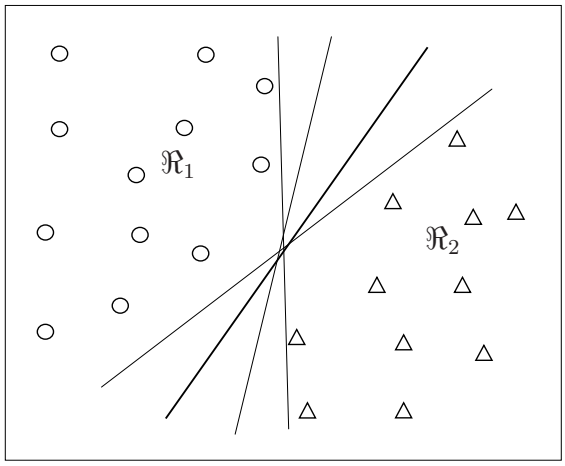


FIGURE A.1 – Hyperplans séparateurs linéaires entre deux régions.

Formellement, cela revient à chercher un hyperplan tel que la distance entre cet hyperplan et la donnée qui lui est la plus proche soit maximale. Cette distance est appelée « marge »de séparation entre $\Re_1$ et $\Re_2$. Sans perte de généralité, la recherche des hyperplans est restreinte à la classe des hyperplans dits « sous forme canonique »[81], où les paramètres w et b sont contraints par l'équation

$$\min_i \left| w^\top \varphi(i) + b \right| = 1. \tag{A.2}$$

Cela veut dire que la norme du poids w doit être égale à l'inverse de la distance entre le point de donnée le plus proche à l'hyperplan et l'hyperplan.
La distance $d(H(w,b);\varphi(i))$ entre le point de donnée $\varphi(i)$ et l'hyperplan $H(w,b)$ est donnée par :

$$d(H(w,b);\varphi(i)) = \frac{\left| w^\top \varphi(i) + b \right|}{\|w\|}. \tag{A.3}$$

L'hyperplan optimal est alors celui qui maximise une marge ρ sous la contrainte de l'équation (A.1) (figure A.2). La marge ρ est alors donnée par :

$$\begin{aligned} \rho &= \min_{\varphi(i)\in\Re_1} d(H(w,b);\varphi(i)) + \min_{\varphi(i)\in\Re_2} d(H(w,b);\varphi(i)) \\ &= \min_{\varphi(i)\in\Re_1} \frac{\left| w^\top \varphi(i)+b \right|}{\|w\|} + \min_{\varphi(i)\in\Re_2} \frac{\left| w^\top \varphi(i)+b \right|}{\|w\|} \\ &= \frac{1}{\|w\|} + \frac{1}{\|w\|} \\ &= \frac{2}{\|w\|}. \end{aligned} \tag{A.4}$$

Maximiser la marge de séparation entre les classes $\Re_1$ et $\Re_2$ par un hyperplan "canonique" [81], revient donc à minimiser la norme euclidienne du vecteur w. Il faut donc,

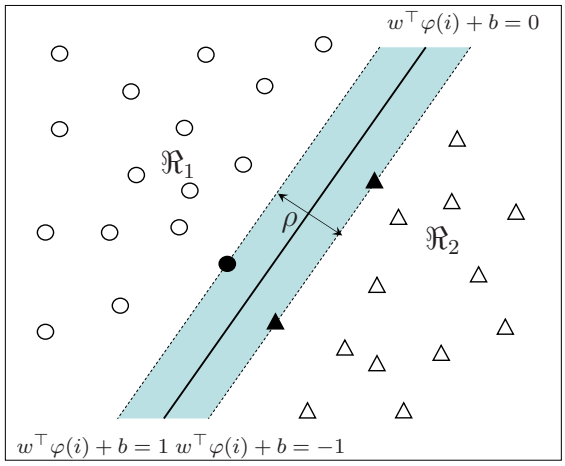


FIGURE A.2 – L'hyperplan séparateur linéaire optimal entre deux régions.

pour cela, résoudre le problème d'optimisation sous contraintes suivant :

$$\begin{aligned} &\min_{w,b} \tfrac{1}{2}\|w\|^2 \\ &s.c. \quad z_i\left[w^\top\varphi(i)+b\right] > 1, \quad \forall\varphi(i)\in\Re_1\cup\Re_2 \end{aligned} \tag{A.5}$$

où $z_i = 1$ si $\varphi(i)\in\Re_1$ et $z_i = -1$ si $\varphi(i)\in\Re_2$.

Le Lagrangien du problème (A.5) est donné par l'expression suivante :

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N}\alpha_i\left(z_i\left[w^\top\varphi(i)+b\right]-1\right). \tag{A.6}$$

Au point-selle, la dérivée du Lagrangien par rapport aux variables primaires doit s'annuler. Ceci s'écrit :

$$\begin{aligned} \frac{\partial L}{\partial w} = 0 \quad &\Rightarrow \quad w = \sum_{i=1}^{N}\alpha_i z_i\varphi(i) \\ \frac{\partial L}{\partial b} = 0 \quad &\Rightarrow \quad \sum_{i=1}^{N}\alpha_i z_i = 0 \end{aligned} \tag{A.7}$$

Par ailleurs, il est démontré (conditions de Karush-Kuhn-Tucker) que seuls les points qui sont sur les hyperplans frontière ($w^\top\varphi(i)+b=\pm1$) jouent un rôle. Ces points pour lesquels les multiplicateurs de Lagrange sont non nuls sont appelés *vecteurs de support* par Vapnik. Ces vecteurs se placent géométriquement comme les plus proches de l'hyperplan optimal qui sépare les deux classes.

Par la suite, la solution du problème d'optimisation est équivalente à celle du problème

dual suivant :

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j z_i z_j \varphi(i)^\top \varphi(j) \\ &sc. \quad \sum_{i=1}^{N} \alpha_i z_i = 0 \\ &\quad \alpha_i \geq 0, \quad i = 1, \ldots, N \end{aligned} \tag{A.8}$$

Une fois la solution optimale du problème (A.8) obtenue, le vecteur de poids w ainsi que le bais b de l'hyperplan à marge maximale recherché s'écrivent :

$$\begin{aligned} w &= \sum_{i=1}^{N} \alpha_i z_i \varphi(i) \\ b &= -\tfrac{1}{2} \left(w^\top \varphi(i_1^*) + w^\top \varphi(i_{-1}^*) \right) \end{aligned} \tag{A.9}$$

où $\varphi(i_1^*)$ et $\varphi(i_{-1}^*)$ sont des vecteurs supports quelconques respectivement de la région $\Re_1$ et de la région $\Re_2$.

A.1.2 Données bruitées et relaxation

L'hypothèse selon laquelle les données sont linéairement séparables est fondamentale dans la résolution du problème (A.5). En effet, il suffit qu'une observation des deux régions viole les contraintes pour que ce problème n'ait plus de solution. La figure A.3 montre une telle situation. Pour tenter de résoudre ce problème, une première idée simple consiste à relâcher les contraintes dans le but d'autoriser quelques erreurs de classification.

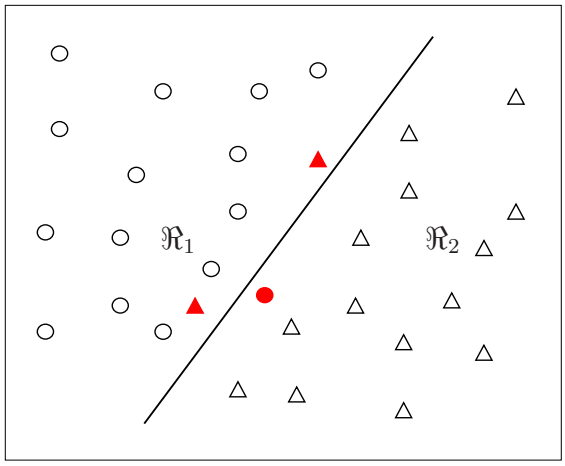


FIGURE A.3 – Deux régions non linéairement séparables.

Dans le cas où les données ne sont pas linéairement séparables, l'hyperplan optimal est celui qui sépare les données des deux régions avec le minimum d'erreurs, et donc celui qui satisfait les contraintes suivantes :

– la distance entre les données bien classées et l'hyperplan doit être maximale,

– la distance entre les données mal classées et l'hyperplan doit être minimale.

Une technique dite des variables ressorts (en anglais : slack variables) [25] [26] permet en effet de chercher un hyperplan séparateur faisant le moins d'erreurs possible. Pour cela, on modifie les contraintes en les relâchant par des variables ressorts ξ_i.

Le problème (A.5) devient alors :

$$\begin{aligned} &\min_{w,b} \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \\ &s.c. \quad z_i\left[w^\top\varphi(i)+b\right] > 1-\xi_i, \quad \forall\varphi(i)\in\Re_1\cup\Re_2 \\ &\qquad \xi_i \geq 0, \quad i=1,\ldots,N. \end{aligned} \tag{A.10}$$

où $z_i = 1$ si $\varphi(i) \in \Re_1$ et $z_i = -1$ si $\varphi(i) \in \Re_2$.

Cela signifie qu'on cherche à maximiser la marge tout en autorisant pour chaque contrainte une erreur positive ξ_i, la plus petite possible. Le paramètre C, dit paramètre de régularisation, est une constante positive fixée à l'avance qui permet de contrôler l'importance de l'erreur que l'on s'autorise par rapport à la taille de la marge. Plus C est important, moins d'erreurs sont autorisées.

La forme duale du problème (A.10) est obtenue de la même façon que pour le cas séparable, elle est donnée par :

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{N}\alpha_i - \tfrac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j z_i z_j \varphi(i)^\top\varphi(j) \\ &sc. \quad \sum_{i=1}^{N}\alpha_i z_i = 0 \\ &\qquad 0 \leq \alpha_i \leq C, \quad i=1,\ldots,N \end{aligned} \tag{A.11}$$

Dans ce cas, les multiplicateurs de Lagrange α_i sont majorés par le paramètre C. Si les données sont linéairement séparables et si C est suffisamment grand, alors on peut montrer que les problèmes (A.8) et (A.11) deviennent équivalents.

B

B.1 Expressions des vecteurs Γ, B et des matrices $\mathscr{L}$, $\mathscr{J}$ et $\mathscr{K}$ de l'équation (4.40)

Vecteur Γ :

$$\Gamma = \left[\gamma_c^{1,pq}, \ldots, \gamma_c^{k,pq}, \ldots, \gamma_c^{s,pq}\right]^\top.$$

Vecteur B :

$$\mathrm{B} = \left[\beta^{12,pq}\kappa^{12,c}, \ldots, \beta^{mn,pq}\kappa^{mn,c}, \ldots, \beta^{s(s-1),pq}\kappa^{s(s-1),c}\right]^\top.$$

où $\kappa^{mn,c} = [\kappa(\varphi(c), sv_1^{mn}), \ldots, \kappa(\varphi(c), sv_{s^{mn}}^{mn})]$, avec s^{mn} est le nombre de vecteurs de support définissant la frontière qui sépare $\Re_m$ de $\Re_n$.

Matrice $\mathscr{L}$:

$$\mathscr{L} = [L_1, \ldots, L_k, \ldots L_s]^\top$$

où $L_k = \left[l_1^{12,k}, l_2^{12,k}, \ldots, l_1^{1s,k}, l_s^{1s,k}, l_2^{23,k}, \ldots, l_i^{ij,k}, l_j^{ij,k}, \ldots, l_s^{(s-1)s,k}\right]^\top$.

avec $l_k^{ij,k} = \mathbf{1}_{s_i^{ij}}^\top$ et $l_{i\neq k}^{ij,k} = -\mathbf{1}_{s_i^{ij}}^\top$, $\mathbf{1}_{s_i^{ij}} = [1, \ldots, 1]^\top \in \mathbb{R}^{s_i^{ij}}$.

s_i^{ij} est le nombre de vecteurs de support appartenant à $\Re_i$ et contribuant à la définition de la frontière qui sépare $\Re_i$ de $\Re_j$.

Matrice $\mathscr{J}$:

$$\mathscr{J} = \left[J^{12}, \ldots, J^{1s}, J^{23}, \ldots J^{mn}, \ldots, J^{(s-1)s}\right]^\top$$

avec $J^{mn} = \left[\underset{1}{0} \cdots 0 \underset{m}{\mathbf{1}_{s_{mn}}} 0 \cdots 0 \quad \underset{n}{-\mathbf{1}_{s_{mn}}} 0 \cdots \underset{s}{0}\right]^\top$

Matrice $\mathscr{K}$:

$$\mathscr{K} = \left[K^{12}, \ldots, K^{1K}, K^{23}, \ldots K^{mn}, \ldots, K^{(s-1)s}\right]^\top.$$

avec $K^{mn} = \left[\beta_1^{12,mn}\kappa_1^{12,mn}, \beta_2^{12,mn}\kappa_2^{12,mn}, \ldots, \beta_1^{1s,mn}\kappa_1^{1s,mn}, \beta_s^{1s,mn}\kappa_s^{1s,mn}, \ldots, \beta_i^{ij,mn}\kappa_i^{ij,mn}, \ldots, \beta_s^{(s-1)s,mn}\kappa_s^{(s-1)s,mn}\right]$.

où $\kappa_i^{ij,mn}$ est la matrice de Gram calculée entre les vecteurs de support sv_l^{ij}, $l = \left\{1, \ldots, s_i^{ij}\right\}$ et les vecteurs de support sv_l^{mn}, $l = \{1, \ldots, s^{mn}\}$

Résumé

Les travaux de cette thèse portent sur l'identification des systèmes dynamiques hybrides. Nous nous intéressons plus précisément à l'identification d'une classe particulière des systèmes hybrides qui est la classe des modèles dynamiques affines par morceaux (PWA). Nous faisons tout d'abord un état de l'art sur l'identification des modèles PWA. Nous proposons ensuite de nouvelles méthodes d'identification des modèles PWA basées sur des techniques innovantes de classification non supervisée combinées avec des techniques de régression pour grouper les données selon leurs sous-modèles respectifs. Puis nous procédons à l'estimation des régions de validité des sous-modèles. Nous proposons ensuite un algorithme récursif d'identification de modèles dynamiques affines par morceaux et de modèles dynamiques non linéaires par morceaux dont les paramètres des sous-modèles et des régions peuvent varier dans le temps. Une technique de régression par des LS-SVM récursifs permettant l'adaptation des fonctions de régression est ainsi proposée. L'adaptation des paramètres des régions de validité est assurée par un nouvel algorithme de classification incrémentale et décrémentale multi-classe à vecteurs de support. La dernière partie de ce travail est consacrée à la validation de nos méthodes sur des exemples réels. Nous appliquons nos méthodes à l'identification d'un système hydraulique à surface libre puis à la modélisation d'une machine de montage de composants électroniques sur circuit imprimé. Nous montrons aussi comment la segmentation temporelle de vidéos en différentes scènes peut être effectuée en se basant sur une estimation de sous-modèles linéaires locaux.

Abstract

In this thesis, we consider the identification of a special class of hybrid systems which is the class of PieceWise Affine (PWA) systems from input-output data. The identification of PWA models is a challenging problem. It involves the estimation of both the parameters of the affine sub-models, and the coefficients of the hyperplanes defining the partition of the state + input set. First, we give an overview of the different approaches available in the literature for the identification of PWA systems. Then, we propose new methods for identifying PWA models from data. The solution includes the estimation of the number of sub-models, the identification of the parameter vectors that describe the different sub-models and the determination of the bounding hyperplanes of the polyhedral regions associated with the sub-models. After this, we propose a recursive algorithm for identifying PieceWise Affine systems and PieceWise nonlinear systems where the parameters of the sub-models and the regions can vary over time. A recursive LS-SVM technique is proposed for recursive updating of the parameters of each sub-model. The adaptation of the parameters of the regions is ensured by an online multi-category support vector classifier. The last part of this work is devoted to the validation of our methods on real examples. We apply our methods to the identification of a hydraulic system and a pick-and-place machine. We also show how the temporal segmentation of video into different shots can be performed, based on the estimation of local linear sub-models.