Sur les modèles flous adaptatifs dynamiques

Mariela Cerrada Lozada

▶ **To cite this version:**

Mariela Cerrada Lozada. Sur les modèles flous adaptatifs dynamiques. Automatique / Robotique. INSA de Toulouse, 2003. Français. NNT : . tel-00010013

HAL Id: tel-00010013
https://theses.hal.science/tel-00010013

Submitted on 31 Aug 2005

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

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

THÈSE

Préparée au
Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

En vue de l'obtention du
Doctorat de l'Institut National des Sciences Appliquées de Toulouse

Specialité
Systèmes Automatiques

par
Mariela CERRADA-LOZADA
MSc. en Ingénierie de Commande de l'Université des Andes au Vénézuela

SUR LES MODÈLES FLOUS ADAPTATIFS DYNAMIQUES

Rapporteurs :

Rosalba LAMANNA de ROCCO	Professeur à l'Université Simón Bolívar, Caracas-Vénézuela
Pierre-Yves GLORENNEC	Professeur à l'INSA de Rennes

Examinateurs :

Eliézer COLINA	Professeur à l'Université des Andes, Mérida-Vénézuela
Joseph AGUILAR-MARTIN	Directeur de Recherche au CNRS

Directeurs de thèse

André TITLI	Professeur à l'INSA de Toulouse
Jose AGUILAR-CASTRO	Professeur à l'Université des Andes, Mérida-Vénézuela

REMERCIEMENTS

Cette thèse a été développée avec le support du programme français-venézuélien PCP "Optimisation et Intégration des Processus" et dans le cadre du contrat de cotutelle entre l'Institut National des Sciences Appliquées de Toulouse (I.N.S.A.T.) et l'Université des Andes (U.L.A) au Vénézuela. Je remercie l'ensemble des personnes responsables de ce programme.

J'adresse mes remerciements à Messieurs G. AUTHIE, Professeur à l'Université Paul Sabatier, Responsable de l'Ecole Doctorale "Systèmes Automatiques" et L. CASTEX, Professeur et Directeur de l'I.N.S.A.T., d'avoir retenu ma candidature.

Je remercie la Direction du Laboratoire d'Analyse et d'Architecture des Systèmes du Centre National de Recherche Scientifique (LAAS-CNRS) ainsi que Monsieur J. AGUILAR-MARTIN, Responsable du groupe DISCO du LAAS, de m'avoir accueillie pendant mes séjours périodiques en France.

J'exprime mes sincères remerciements à Monsieur A. TITLI, Professeur à l'I.N.S.A.T., de m'avoir acceptée comme doctorante au LAAS. Je remercie son soutien, son orientation et ses conseils. Son expérience et sa connaissance ont contribué à ma formation scientifique.

Je remercie Monsieur J. AGUILAR-CASTRO, Professeur à l'U.L.A, de m'avoir acceptée comme doctorante et pour sa disponibilité et son orientation pendant la réalisation de cette thèse.

Mes remerciements aussi à mes camarades du groupe : Elvia, Elsa et Miguel, pour leur amitié pendant mes séjours au LAAS.

Je remercie spécialement mon ami Juan pour son amitié et son soutien pendant ces années.

J'en profite pour remercier le personnel technique et administratif du LAAS de faire de celui-ci un agréable lieu de travail.

Antonio, Cristian et Sabrina,... les mots ne suffisent pas : Je vous adore, vous êtes ma vie.

Table des matières

Table des figures

Liste des tableaux

INTRODUCTION

La recherche de méthodes pour le développement d'une représentation précise des processus réels est un aspect important dans la modélisation floue, en particulier, les méthodes qui utilisent la connaissance disponible sur les processus.

La premier méthode, appelée "approche directe", utilise la connaissance des experts pour la définition d'une base de règles SI-ALORS, avec les entrées du processus réel dans l'antécédent des règles et les sorties dans la partie conséquence des règles [1]. Le développement de ces modèles flous est basé sur la définition d'une partition non disjointe et floue du domaine du problème ; en utilisant un mécanisme d'inférence, il est possible d'obtenir la sortie, ou un ensemble de sorties, du modèle à partir d'une entrée ou un ensemble d'entrées. Dans ce cas-ci, la précision du modèle flou est dépendante de la qualité de la connaissance : une connaissance pauvre donne lieu à un modèle flou avec une performance insuffisante. Effectivement, une connaissance experte n'est pas toujours suffisante pour obtenir un " bon" modèle dans cette approche directe et l'on a intérêt à compléter cette démarche par l'utilisation d'une autre "connaissance" que constituent les données entrée-sortie sur un système.

Ainsi, inspirés de la théorie classique des systèmes et, récemment, des développements des réseaux des neurones, des méthodes de modélisation utilisant les observations quantitatives disponibles sur le processus ont été proposées pour déterminer la structure et les paramètres du modèle comme une procédure d'identification des systèmes [2, 3]. L'identification de la structure du modèle vise à la détermination des variables d'entrée, de sortie et de leurs relations (la structure des règles floues), le nombre des règles et la partition des ensembles floues. L'identification des paramètres comprend la définition des fonctions d'appartenance.

Aussi, a-t-on pu voir, même si ce n'est pas nécessaire, sous la forme de systèmes dits "neuro-flous !", l'utilisation de réseaux de neurones artificiels pour capturer le mécanisme d'inférence floue dans un réseau de neurones et des algorithmes d'optimisation ("apprentissage")associés pour résoudre le problème d'identification des paramètres des modèles flous, en utilisant les données numériques disponibles [4, 5] : les modèles flous obtenus ont un comportement d'approximateur universel et peuvent apprendre en utilisant différentes méthodes et la connaissance sur le processus peut être incorporée dans le modèle flou.

Dans tous les cas, sans passer par la représentation neuronale, par un choix judicieux du mécanisme d'inférence, on peut donner au modèle flou une forme fonctionnelle, analytique, sur laquelle on peut appliquer des algorithmes d'ajustement des paramètres.

Le succès de ces méthodes a été acquis ces dernières années [6] et les modèles flous qui utilisent des algorithmes d'apprentissage ont été appelés *Modèles Flous Adaptatifs* (MFA) [7]. Deux approches de base ont inspiré de nombreux travaux de recherche autour de l'ajustement des MFA : d'une part, la méthode ANFIS [2] et, d'autre part, la méthode FALCON [3], toutes

les deux proposant des algorithmes hors ligne et en ligne pour améliorer les performances des MFA.

Les algorithmes basés sur la méthode du gradient sont très utilisés pour l'ajustement des paramètres des MFA [7, 8]. Cependant, si ces méthodes sont utilisées hors-ligne, après le processus d'apprentissage, les modèles flous sont des modèles statiques et il est possible d'obtenir des ensembles flous voisins qui sont disjoints et, comme conséquence, les règles floues ne sont pas suffisamment activées ou, même, restent inactives [9].

D'un autre côté, les modèles flous ont été utilisés pour la modélisation des systèmes dynamiques. Classiquement, l'utilisation de modèles flous en identification dynamique consiste en la définition des vecteurs d'état $\mathbf{x(k)}$, d'entrée $\mathbf{u(k)}$ et de sortie $\mathbf{y(k)}$ comme variables flous, où k est le temps échantillonné [10, 11]. La dépendance par rapport au temps des ensembles flous, bien qu'importante dans les applications de commande, n'apparaît pas dans les applications classiques où les modèles flous "statiques" peuvent ne pas être adéquats pour la modélisation des systèmes dynamiques, variant avec le temps, et en présence de perturbations. La dynamique d'un système peut être prise en considération en utilisant des fonctions d'appartenance dépendantes du temps. Dans [12], les premières idées sur les fonctions d'appartenance dépendantes du temps sont introduites en faisant une extension des définitions classiques d'un ensemble flou F en un ensemble flou dépendant du temps.

Dans ce contexte, notre travail considère trois aspects importants de la modélisation floue :

- L'incorporation de fonctions d'appartenance variant avec le temps.
- La proposition d'un *Modèle Flou Adaptatif Dynamique* (MFAD).
- La proposition d'un algorithme d'apprentissage en ligne pour l'ajustement des paramètres du modèle flou.

L'approche présentée dans ce travail prend en compte la dynamique des variables du système en introduisant, dans les fonctions d'appartenance d'un modèle flou, la valeur moyenne et la variance des valeurs d'entrée et de sortie du modèle, disponibles au temp t. De cette manière, les ensembles flous se déplacent sur le domaine de discours des variables, en fonction de ces valeurs de la moyenne et de la variance échantillonnées et la possibilité d'obtenir des ensembles flous disjoints peut être minimisée.

Le modèle flou a, donc, des fonctions d'appartenance paramétrées, dynamiques, dont les paramètres sont ajustés par une méthode d'apprentissage. La propriété dynamique du modèle flou proposé est un atout pour résoudre les problèmes de commande de systèmes variant avec le temps, par exemple [13, 6].

Dans un premier temps, on utilise la méthode du gradient comme algorithme d'apprentissage hors ligne pour démontrer la capacité du MFAD à traiter des tâches de prédiction / identification de systèmes invariants et variants avec le temps. Cependant, bien que l'usage des algorithmes hors-ligne basés sur la méthode du gradient aient donné de bons résultats [2, 14, 15, 16], ce type d'apprentissage hors ligne est dépendant de la qualité et de la quantité des données historiques. D'où la proposition d'algorithmes d'apprentissage en ligne appréciés pour l'ajustement des modèles flous dans des environnements dynamiques.

Partant de l'importance de la variable temps dans les problèmes de commande, on propose un algorithme d'identification en ligne, basé sur les idées de l'apprentissage par renforcement, pour l'ajustement du MFAD utilisé comme identificateur. Le problème d'apprentissage par renforcement a été largement travaillé après le rapport de R. Sutton [17], en particulier

dans les années 1990. On peut trouver un nombre important de références sur les aspects théoriques et les applications qui rapportent des résultats intéressants en environnements dynamiques et sur la commande des systèmes [18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28]. Le problème de l'apprentissage par renforcement vise la recherche d'une fonction qui prédit la valeur d'une action. Dans ce travail, le problème d'ajustement des modèles flous est formulé comme un problème de prédiction où *les actions* sont l'ajustement des paramètres et *la valeur de l'action* est donnée par la capacité du modèle flou à offrir une bonne identification de l'environnement.

Suite à la proposition de ce type de modèle MFAD et aux algorithmes d'ajustement associés, on teste la capacité de prédiction des MFAD sur un problème de commande prédictive. Pour résoudre ce problème, on a besoin d'utiliser un modèle de prédiction aux instants futurs et une séquence de signaux de commande est calculée en minimisant une fonction de performance. L'usage des modèles flous comme modèle de prédiction a fourni des résultats importants [29, 30, 31]. Dans cette application, on utilise le MFAD comme modèle de prédiction, ajusté hors ligne par la méthode du gradient. Le signal de commande est determiné en minimisant un critère qui utilise la prédiction fournie par le MFAD avec un algorithme utilisant l'apprentissage par renforcement et fonctions de pénalisation.

Finalement, on aborde l'application de la modélisation floue proposée aux problèmes de diagnostic et de supervision.

Vu les aspects traités dans cette thèse, ce mémoire est organisé de la manière suivante : le chapitre 1 concerne la problématique générale de la modélisation et de l'identification des systèmes. La première partie, contenant les chapitres 2, 3 et 4, présente les aspects de la modélisation floue développés dans cette thèse, tels que la motivation, le modèle flou adaptatif dynamique et les exemples illustrant la performance de ce type de modèles dans l'identification des systèmes non-linaires variant avec le temps. Les chapitres 5,6, et 7 de la deuxième partie, abordent les problèmes d'application des modèles flous adaptatifs dynamiques pour la commande prédictive d'un processus de fermentation et pour le diagnostic consideré comme une tâche du niveau de supervision. Finalement, le dernier chapitre propose une conclusion générale et des perspectives sur des futurs travaux toujours sur la modélisation floue adaptative dynamique développée pendant ce travail de recherche.

Chapitre 1

La problématique générale de la modélisation et de l'identification

Cette thèse concerne un aspect lié à la caractérisation des systèmes dynamiques : la construction d'un modèle par identification. Le *modèle* d'un système décrit l'interaction entre ses variables (entrées, sorties, perturbations) et pour le caractériser, on peut faire appel à des formalismes mathématiques avec des grades différents de complexité, selon les besoins.

Dans certains cas, les systèmes sont décrits en utilisant des tableaux numériques : on parle de *modèles graphiques*. Par exemple, dans le cas de systèmes linéaires, la représentation graphique de leurs réponses indicielle, impulsionelle ou fréquentielle, sont très utilisées pour différentes raisons (conception de contrôleurs, par exemple) [32, 33]. D'autres caractéristiques non-linéaires, par exemple celles des vannes, sont bien décrites par des modèles graphiques.

Dans certaines applications, on a besoin d'expressions mathématiques données par des équations aux différences ou différentielles. Dans ce cas, on parle de *modèles mathématiques* [34]. Ce type de modèles est très utilisé au niveau des applications en engénierie et on dispose des outils de simulation pour valider de tels modèles.

Finalement, il existe des systèmes complexes où le modèle est un logiciel composé d'autres sous-modèles inter-connectés, de tableaux numériques, etc. Dans ce cas, il n'est pas possible de fusionner les modèles en une seule expression mathématique, mais l'interconnexion de l'ensemble des sous-modèles décrit le système global.

1.1 Modélisation et identification

En général, un modèle est défini à partir des observations du comportement du système. Ces observations son obtenues par différentes expériences sous différentes situations et environnements. Un modèle mathématique peut être obtenue par deux chemins : d'un côté, on peut profiter de la connaissance de certaines propriétés du système, propriétés qui sont connues pendant le temps, par exemple des lois physiques, chimiques, etc., et proposer un modèle à travers l'union des lois décrivant les sous-systèmes : on parle, alors, de *modélisation* et on n'a pas besoin d'expérimentation sur le système. A nouveau, pour les systèmes com-

plexes, on peut fusionner, par logiciel, les différents sous-systèmes pour arriver à proposer le modèle global.

Le deuxième chemin est de construire le modèle à partir d'une experimentation et de l'observation du système : on parle de *l'identification des systèmes*. Dan ce cas, les entrées et sorties du processus son analysées afin d'inférer un modèle (mathématique, graphique, etc).

Cependant, quel que soit le chemin suivi, on ne va pas arriver à proposer un modèle décrivant, exactement, le système réel. En conséquence, l'acceptation d'un modèle est guidée par des considérations liées à son utilisation, plutôt qu'à sa précision (évidemment, il existe des applications où le niveau de précision requis est primordial).

1.1.1 La procédure d'identification

La proposition d'un modèle, en utilisant les procédures d'identification des systèmes, comprendre trois éléments de base [13] :

1. **Un ensemble de données d'observation.**
   Ces données, en général associées aux variables d'entrée et de sortie du système, sont enregistrées pendant une expérimentation conçue par l'utilisateur : il décide le type de signal d'entrée, quelles variables seront mesurées et quand prendre les échantillons, afin d'obtenir des données contenant une information représentative du système.
2. **Un ensemble de structures ou types de modèles**
   Le choix d'une structure de modèle est la décision la plus difficile de la procédure d'identification et l'intervention de l'intuition ou de la connaissance a priori sur le système peut aider pour le choix du type de modèle. Il existe des structures linéaires ou non-linéaires, paramétriques sans sens physique, où les paramètres sont ajustés pour affiner la relation entre les données entrée-sortie enregistrées. Ce type de modèle est appelé modèle "boîte noire". Si les paramètres du modèle ont un sens physique, alors, on les appelle modèle type "boîte grise". En général, la structure d'un modèle est une transformation mathématique, paramétrique, des entrées et sorties passées, dénotées par le vecteur Z^{t-1}, vers l'espace de sortie du système :

   $$y_e(t|\theta) = g(\theta, Z^{t-1}) \tag{1.1}$$

   où θ dénote un vecteur de dimension fini des paramètres et y_e est la sortie estimée par le modèle.
3. **L'ajustement du modèle et sa validation**
   Ce dernier aspect est guidé en utilisant les données enregistrées et une méthode d'identification particulière. La méthode d'identification ne dépend pas, a priori, de la structure de modèle et différentes approches ont été développées. Soulignons une procédure en particulière : l'approche d'identification basée sur l'erreur de prédiction. Avec cette approche, le vecteur des paramètres θ est obtenu tel que :

   $$\theta_N = argmin_{\theta \in D_\theta} V_N(\theta, Z^N) \tag{1.2}$$

   où

$$V_N(\theta, Z^N) = \frac{1}{N}\sum_{t=1}^{N} l(\epsilon(t,\theta), \theta, t) \tag{1.3}$$

Dans les équations ci-dessus, D_θ dénote un domaine des valeurs possibles de θ, Z^N est un ensemble de N paires de données entrée-sortie, V_N est une fonction objectif, $\epsilon(t,\theta)$ dénote l'erreur de prédiction $y(t) - y_e(t|\theta)$, $y(t)$ étant la sortie réelle du système, et l est une *norme* pour l'erreur de prédiction. Ce problème de minimization (1.2) peut être résolu par des méthodes classiques comme la méthode des Moindres Carrés, la méthode de Probabilité Maximale, etc. Dans [13], on peut trouver plus de détails sur des méthodes classiques d'ajustement de modèles.

Finalement, l'évaluation de la qualité d'un modèle est basée, en général, sur sa performance à reproduire les données de validation disponibles (étape dite de "cross-validation").

1.2 Les structures de modèles

Dans les paragraphes suivants, on rappelle les structures de modèles linéaires et non-linéaires les plus connues.

1.2.1 Modèles linéaires

Recensons, ici, les structures les plus connues de modèles linéaires invariants avec le temps, dans l'espace entrée-sortie, et décrits par des fonctions de transfert. En général, ces structures sont du type "boîte noire" et sont représentées mathématiquement selon :

$$y(t) = G(q)u(t) + H(q)e(t) \tag{1.4}$$

où $G(q) = \sum_{k=1}^{\infty} g(k)q^{-k}$, $H(q) = 1 + \sum_{k=1}^{\infty} h(k)q^{-k}$ et $e(t)$ est une perturbation.

Mentionnons les structures générales suivantes [13] :

1. *Régression linéaire*, donnée par l'équation :

$$y_e(t|\theta) = B(q)u(t) + [1 - A(q)]\, y(t) = \phi^T(t)\theta \tag{1.5}$$

où

$$\phi(t) = [-y(t-1) \ \ldots \ -y(t-n_a) \ u(t-1) \ \ldots \ u(t-n_b)]^T \tag{1.6}$$

et $A(q) = 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}$, $B(q) = b_1 q^{-1} + \ldots + b_{n_b} q^{-n_b}$. D'habitude, le vecteur ϕ est appelé *vecteur de régression* et le vecteur des paramètres θ contient les coefficients des polynômes A et B.

2. *Régression pseudo-linéaire*, donné par l'équation suivante :

$$y_e(t|\theta) = B(q)u(t) + [1 - A(q)]\, y(t) + [C(q) - 1]\, [y(t) - y_e(t|\theta)] = \phi^T(t, \theta)\theta \quad (1.7)$$

où

$$\phi(t, \theta) = [-y(t-1) \ \ldots \ -y(t-n_a) \ u(t-1) \ \ldots \ u(t-n_b) \ \epsilon(t-1, \theta) \ \ldots \ \epsilon(t-n_c, \theta)]^T \quad (1.8)$$

introduisant l'erreur de prédiction $\epsilon(t, \theta) = y(t) - y_e(t|\theta)$. Les polynômes A et B conservent la forme du modèle de régression et $C(q) = 1 + c_1 q^{-1} + \ldots + c_{n_c} q^{-n_c}$. Le vecteur des paramètres θ contient les coefficients des polynômes A, B et C.

Ce type de modèle est semblable à celui de la régression mais avec un effet non-linéaire de θ dans le vecteur ϕ ; on l'appelle, donc, *pseudo-linéaire*. Ce modèle suit la structure du modèle appelé ARMAX (Auto Regressive Moving Average with eXternal input, en anglais) [13].

Dans les modèles du type *régression pseudo-linéaire*, on peut trouver la structure appelée *erreur de sortie*, donnée par l'équation suivante :

$$y_e(t|\theta) = \frac{B(q)}{F(q)} u(t) = \phi^T(t, \theta)\theta \quad (1.9)$$

où

$$\phi(t, \theta) = [u(t-1) \ \ldots \ u(t-n_b) \ -w(t-1, \theta) \ \ldots \ w(t-n_f, \theta)]^T \quad (1.10)$$

et $F(q) = 1 + f_1 q^{-1} + \ldots + f_{n_f} q^{-n_f}$. Le vecteur des paramètres θ contient les coefficients des polynômes B et F et $w(t-k, \theta) = y_e(t-k|\theta), k = 1, \ldots, n_f$.

D'autres structures, utilisant différentes relations entre les entrées et les sorties, sont obtenues comme une variation des modéles du type ARMAX.

1.2.2 Modèles non-linéaires

Dans ce cas, les modèles d'identification tentent de reproduire une relation du type suivant :

$$\begin{aligned} x(t+1) &= f(t, x(t), u(t), w(t); \theta) \\ y(t) &= h(t, x(t), u(t), v(t); \theta) \end{aligned} \quad (1.11)$$

où $w(t)$ et $v(t)$ sont des séquences de variables aléatoires indépendantes, x est un vecteur d'état et θ est un vecteur de paramètres inconnus.

Bien que l'obtention d'un modèle d'identification n'est pas, dans ce cas, un problème évident, ces modèles ont la forme (1.1) où g est une fonction non-linéaire, du type "boîte

noire", des données passées. Reprenant le concept de régression, on peut caractériser la fonction g de la manière suivante :

$$g(Z^{t-1}, \theta) = g(\phi(t), \theta) \tag{1.12}$$

où $\phi(t) = \phi(Z^{t-1}, \theta)$. En conséquence, le choix de la transformation non-linéaire (1.12) implique le choix :

- du vecteur de régression $\phi(t)$ à partir des entrées et sorties passées.
- de la transformation non-linéaire $g(\phi, \theta)$ de l'espace de régression vers l'espace du sortie.

Habituellement, le vecteur de régression est choisi comme dans le cas linéaire : à partir des mesures passées, sorties estimées passées et erreurs de prédiction passées. Donc, le problème intéressant est d'obtenir la transformation non-linéaire $g(\phi, \theta)$.

Pour résoudre ce problème, on mentionne, par exemple, l'utilisation des *fonctions de base*, approche qui conduit à proposer la formule non-linéaire suivante :

$$g(\phi, \theta) = \sum_{k=1}^{n} \alpha_k g_k(\phi) \tag{1.13}$$

où $\theta = [\alpha_1 \ \dots \ \alpha_n]^T$ et g_k sont des fonctions de base du type développement en série de Taylor, séries de Fourier, Ondelettes, etc. Pour plus de détails, voir [13].

D'un autre côté, les réseaux de neurones et les modèles flous sont devenus très commodes pour réaliser cette transformation non-linéaire $g(\phi, \theta)$. Les réseaux de neurones sont des modèles du type "boîte noire" car leurs paramètres n'ont pas de sens physique [35, 36]. Par contre, la conception d'un modèle flou prend en compte les relations entre les variables du modèle et la valeur de ses paramètres. On introduit, ci-dessous, les modèles flous et leur procédure d'identification.

1.2.3 Modèles flous

Un modèle flou est un modèle linguistique basé sur des règles de type SI-ALORS [8, 37]. Ce modèle utilise des opérateurs logiques et des valeurs floues pour ses variables, ce qui permet de gérer l'information d'un processus tout comme l'homme le fait. De cette manière, et comme une extension de la définition d'un modèle classique, un modèle flou permet d'avoir une répresentation des caractéristiques essentielles d'un système en utilisant la théorie de la Logique Floue [1].

Un modèle flou est basé sur une partition linguistique des valeurs de ses variables. Ces valeurs linguistiques sont décrites par des ensembles flous caractérisés par une fonction d'appartenance qui donne une niveau d'appartenance d'une valeur numérique, de cette variable, à l'ensemble flou. Cela permet d'associer une valeur numérique à une valeur linguistique avec un certain "degré" entre 0 (aucune appartenance) et 1 (appartenance maximale)

Les modèles flous comportent deux types de représentation (voir [8] pour plus de détails) :

- **Modèles du type Mamdani**, avec des règles linguistiques de la forme :

$$SI\ x_i\ est\ A_i\ et \dots et\ x_n\ est\ A_n\ ALORS\ y\ est\ B \tag{1.14}$$

où $x_i, i = 1, \ldots, n$ sont des variables d'entrée et y est une variable de sortie. A_i et B sont des valeurs linguistiques décrites par des ensembles flous. Ce type de modèle est, tout à fait, une expression qualitative du système utilisant un langage naturel.

– **Modèles du type Tagaki-Sugeno-Kang (TSK)**, avec des règles de la forme :

$$SI\ x_i\ est\ A_i\ et \ldots et\ x_n\ est\ A_n\ ALORS\ y = f(x_i, \ldots, x_n; \Phi) \tag{1.15}$$

où f est une fonction linéaire (en général) ou non-linéaire et Φ est un vecteur de paramètres. Ce type de modèle est une combinaison entre modélisation floue et non-floue.

Ainsi, on arrive à une description du processus par un modèle flou, par l'obtention d'une valeur numérique de la sortie à partir des valeurs numériques en entrée, ceci par une procédure qui comprend trois étapes [8] : transformation des valeurs numériques d'entrée en des valeurs linguistiques, mise en fonctionnement du mécanisme d'inférence logique et, finalement, transformation de la valeur linguistique de sortie en une valeur numérique.

Modèles flous et identification

En partant de la procédure décrite dans la section 1.1.1, la structure d'un modèle flou comprend le choix :

– du type de modèle à utiliser : Mamdani ou TSK.
– du type de fonctions d'appartenance pour les ensembles flous décrivant une valeur linguistique.
– du type d'opérateurs logiques de connexion entre les variables.

Ensuite, l'ajustement du modèle traite de :

– l'ajustement des paramètres :
  – des fonctions d'appartenance.
  – de la fonction de sortie dans le cas des modèles de type TSK.
– l'ajustement du nombre des règles.

Remarque 1 *Sous certaines considérations, un modèle floue peut être conçu pour arriver à une expression comme celle donnée dans (1.13).*

1.3 Conclusion

Dans ce chapitre, on a donné une vision globale du problème de modélisation et l'identification des systèmes. En général, le problème se focalise sur le choix d'une expression mathématique donnant la sortie d'un système à partir de l'information contenue dans certaines variables du système (variables d'entrée, de sortie pasées, d'état). Dans le cas de la modélisation non-linéaire, on peut utiliser de nouveaux outils issus de l'intelligence artificielle tels les modèles flous ou les réseaux de neurones. En particulier, les modèles flous ont la capacité de gérer une information linguistique et numérique issue de la combinaison des connaissances qualitative et quantitative sur le système.

Pour ce qui est des modèles flous, on se concentre sur les modèles de type Mamdani, en proposant une structure dynamique pour les fonctions d'appartenance et en développant une tâche d'identification consistant en l'ajustement, hors ligne et en ligne, des paramètres de ces fonctions d'appartenance.

I° PARTIE

Modélisation floue

Dans cette partie, on précise les contributions sur la modélisation floue développées dans ce travail : d'une part, la proposition de Modèles Flous Adaptatifs Dynamiques (MFAD) et, d'une autre part, la proposition d'un algorithme d'identification en ligne basé sur l'apprentissage par renforcement. Dans le premier chapitre, on fournit les aspects théoriques nécessaires au développement de ce travail : la logique floue temporelle, la modélisation floue et les modèles flous adaptatifs classiques. Ensuite, on rappelle les aspects théoriques sur l'apprentissage par renforcement. Dans le deuxième chapitre, on présente les principaux résultats obtenus sur l'identification des fonctions non linéaires variant avec le temps : on précise les performances du MFAD en utilisant la méthode du gradient comme algorithme d'apprentissage hors ligne, puis en utilisant l'algorithme en ligne basé sur l'apprentissage par renforcement.

Chapitre 2

État de l'art et motivation

Dans ce chapitre, on fournit les aspect théoriques nécessaires au développement de ce travail : la logique floue temporelle, la modélisation floue et les modèles flous adaptatifs classiques. Ensuite, on rappelle les aspects théoriques sur l'apprentissage par renforcement.

2.1 La logique floue temporelle

La plupart des applications en intelligence artificielle ont besoin de prendre en considération le temps comme variable d'intérêt pour la prise de décision. Par exemple, les temps ou les dates faisables et admissibles pour l'initialisation ou l'arrêt de tâches sont des éléments importants dans les applications de planification. En ce sens, la logique floue, comme outil puissant de représentation de la connaissance, offre un cadre conceptuel pour le traitement de la connaissance temporelle.

Dans [38], sont présentés les concepts sur l'intégration du temps comme une variable floue qui peut être une date, une période de temps ou un intervalle de temps. A partir de ces concepts, on considère plusieurs méthodes de raisonnement selon la caractérisation floue de ces variables. L'importance de l'approche présentée dans [38] est de pouvoir utiliser l'information disponible sur l'occurrence des événements passés ou futurs, décrite en langage naturel avec des expressions telles que "avant","après", "autour de".

Selon [38], la connaissance disponible sur une *date* est représentée par un ensemble flou $F(t)$, qui dépend de la variable *temps*; la certitude sur l'occurrence d'un événement à une *date* donnée est déterminée par le niveau d'appartenance de telle *date* à l'ensemble flou. Dans la figure 2.1, on illustre ce concept en montrant une *date floue* autour de $t = 3$, ayant lieu entre $t = 0$ et $t = 6$. La fonction d'appartenance montrée dans la figure 2.2 caractérise une *date floue* dont la connaissance est concentrée entre $t = 2$ et $t = 3$.

Avec ce concept de *date*, il est possible de manipuler des propositions P comme : *La période de pluie commence début Mai*; la variable floue *date* fait référence au "temps de démarrage de la période de pluie" et "début Mai" est un ensemble de dates en concordance avec cet événement.

Bien que les concepts développés dans [38] formalisent les idées sur la manipulation de l'information temporelle imprécise et incertaine, le travail présenté dans cette thèse se rapproche plus de la notion de variable *temps* en logique floue présentée dans [12] où est soulignée la

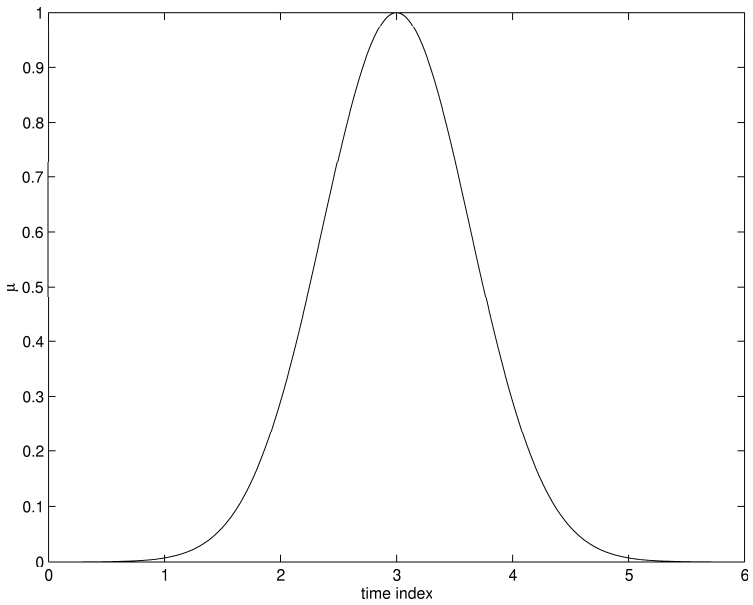

FIG. 2.1. Une date floue autour $t = 3$.

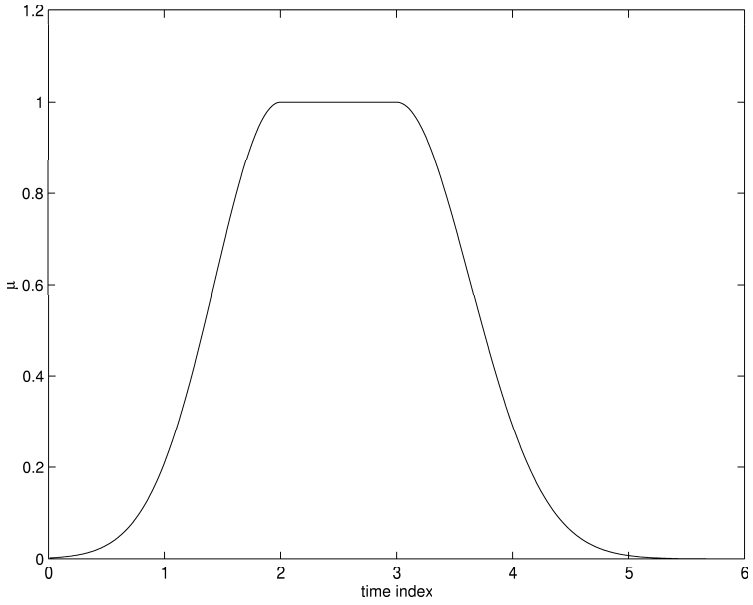

FIG. 2.2. Une date floue avec une haute certitude entre $t = 2$ et $t = 3$.

dépendance au temps des opérations logiques contenues dans la procédure d'inférence floue, inférence qui permet le traitement de l'information floue liée à la variable *temps* et aux valeurs numériques des mesures sur les variables du système considéré. En particulier, dans [12] les auteurs introduisent un ensemble flou dépendant du temps, $F(t)$, dont la fonction d'appartenance dépend du temps, $\mu_F(t, x)$. L'aspect numérique fait référence à *la valeur* qu'une variable floue, X, peut avoir et l'aspect temporel fait référence au *temps* où cette variable peut avoir cette valeur. Les deux informations peuvent s'exprimer à travers une proposition P de la forme : *"La température est environ 30° C autour de 5 :05 hrs."* [39].

Dans ce qui suit, on présente, donc, l'aspect le plus important pour le développement de cette thèse, à savoir les ensembles flous dépendant du temps [12].

2.1.1 Des ensembles flous dépendant du temps

Une ensemble flou est défini par :

$$F = \{(x, \mu_F(x))\} \tag{2.1}$$

où x est un élément de l'espace X et $\mu_F(x)$ est la fonction d'appartenance de x à l'ensemble flou F. En prenant en compte la variable temps, on peut avoir au temps t_1 un ensemble flou $F(t_1)$, au temps t_2 un ensemble flou $F(t_2)$ et, en général, on peut avoir un ensemble flou $F(t)$ dépendant du temps.

Chaque ensemble flou a une function d'appartenance, donc, pour les ensembles flous $F(t_1)$ et $F(t_2)$, on dénote $\mu_{F(t_1)}(x)$ et $\mu_{F(t_2)}(x)$ les fonctions d'appartenance. En conséquence, on obtient, par chaque ensemble flou, la description suivante [12] :

$$\begin{aligned} F(t_1) &= \{\frac{\mu_1(t_1)}{x_1}, \ldots, \frac{\mu_n(t_1)}{x_n}\} \\ F(t_2) &= \{\frac{\mu_1(t_2)}{x_1}, \ldots, \frac{\mu_n(t_2)}{x_n}\} \end{aligned} \tag{2.2}$$

où x_i sont des éléments sur X, $\frac{\mu_i(t_j)}{x_i}$, $j = 1, 2$, dénote, en notation de logique floue [8], la valeur d'appartenance de x_i à $F(t_j)$, et, de façon générale, pour $F(t)$, extension d'un ensemble flou F :

$$\begin{aligned} F(t) &= \{(x, \mu_{F(t)}(x))\} \\ F(t) &= \{\frac{\mu_1(t)}{x_1}, \ldots, \frac{\mu_n(t)}{x_n}\} \end{aligned} \tag{2.3}$$

où x appartient au domaine du discours de l'ensemble flou F et $\mu_{F(t)}(x)$ est la fonction d'appartenance de F. Le domaine de t peut être continu ou discret. La projection sur le plan (μ, t), pour un x donné, est appelée *fonction d'appartenance dynamique*. En conséquence, le degré d'appartenance de x est dépendant du temps. La figure 2.3 visualise la définition des fonctions d'appartenance dynamiques.

Les opérations classiques sur les ensembles flous se généralisent, notamment les union, intersection, complémentation, avec les opérateurs suivants :

$$\begin{aligned} \mu_{A(t)\cap B(t)}(x) &= min[\mu_{A(t)}(x), \mu_{B(t)}(x)] \\ \mu_{A(t)\cup B(t)}(x) &= max[\mu_{A(t)}(x), \mu_{B(t)}(x)] \\ \mu_{\overline{A}(t)}(x) &= 1 - \mu_{A(t)}(x) \end{aligned} \tag{2.4}$$

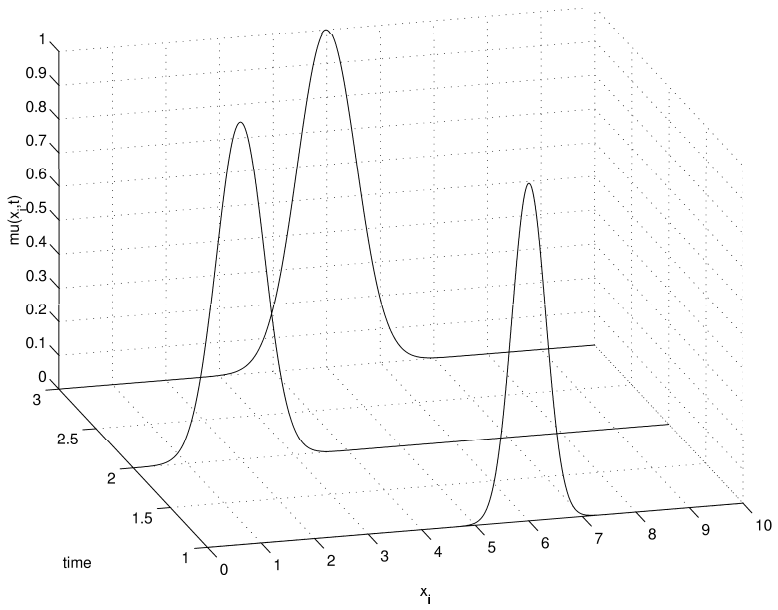

FIG. 2.3. Fonctions d'appartenance dynamiques d'un ensemble flou défini sur la variable x_i.

2.2 Modèles flous linguistiques et adaptatifs

La première approche pour la proposition de modèles flous est l'approche linguistique promue au début par Zadeh [40] et développée, ensuite, par de nombreux autres auteurs, voir par exemple [41, 42, 43, 44].

En termes généraux, un modèle flou linguistique est composé de M règles floues SI-ALORS de la forme [8, 7] :

$$R^{(l)} : SI\ x_1\ est\ F_1^l\ \ AND... AND\ \ x_n\ est\ F_n^l\ ALORS\ \ y\ \ is\ G^l \tag{2.5}$$

où x_i sont des variables linguistiques d'entrée définies sur un domaine de discours U_i. La variable de sortie y est définie sur un domaine de discours V. F_i^l et G^l sont des ensembles flous sur U_i et V, respectivement, $(i = 1, ..., n)$ et $(l = 1, ..., M)$.

La procédure pour obtenir une valeur de sortie à partir des valeurs d'entrée est appelée *le mécanisme d'inférence floue*. En général, les variables d'entrée et de sortie du modèle ont des valeurs réelles ; on a donc besoin d'un mécanisme d'inférence floue avec fuzzification et défuzzification. On rappelle que la procédure de fuzzification par singleton flou permet de caractériser une valeur numérique $x^* \in X$ par un ensemble flou A défini comme un singleton flou avec un niveau d'appartenance $\mu_A(x) = 0$ quelle que soit la valeur x sur X, sauf pour la valeur $x = x^*$ pour laquelle on a $\mu_A(x) = 1$. D'un autre côté, on rappelle que la procédure de défuzzification permet de choisir une valeur particulière la plus représentative d'un ensemble flou de sortie obtenue à partir d'un ensemble de règles activées. Les méthodes de défuzzification les plus connues sont la méthode de la moyenne du maximum et la méthode de la moyenne des centres. Les mécanismes d'inférence floue sont très connues ; pour plus de détails, voir [8].

Le modèle flou linguistique susmentionné est un système basé sur la connaissance et la précision de l'information donnée par ce modèle dépend, donc, de la qualité de la connais-

sance et du mécanisme de raisonnement flou. Dans les applications en ingénierie, l'obtention de règles linguistiques à partir de la connaissance humaine peut être une tâche difficile, en particulier dans le cas des processus complexes. Cependant, les données obtenues par les sources d'information naturelles des processus (les capteurs) sont numériques et doivent être utilisées afin de proposer un meilleur modèle.

Devant ces besoins, il faut avoir des mécanismes qui permettent d'incorporer des informations linguistiques et numériques, pour construire des modèles flous linguistiques en utilisant les idées d'identification des systèmes. Les modèles flous qui utilisent des mécanismes d'identification sont appellés *Modèles Flous Adaptatifs (MFA)*. En général, les travaux sur les MFA sont basés sur la transcription analytique d'un modèle flou linguistique en un réseau de neurones et sur l'utilisation des algorithmes d'apprentissage afférents. Plusieurs approches dans ce sens sont reportées dans [15, 2, 3, 45].

Soulignons l'intérêt, pour ce type de modéle, d'obtenir une description analytique d'un système flou donnée par le lemme suivant :

Lema 1 *Le système de logique floue (2.5), avec un mécanisme d'inférence utilisant la méthode de fuzzification par singleton flou, la méthode de défuzzification de la moyenne des centres et des fonctions d'appartenance de gauss pour les ensembles flous des variables flous d'entrée, a la description analytique suivante :*

$$y(\mathbf{X}) = \frac{\sum_{\mathbf{l=1}}^{\mathbf{M}} \gamma^{\mathbf{l}} \left(\prod_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{exp} \left[-\frac{\left(\mathbf{x_i} - \alpha_{\mathbf{i}}^{\mathbf{l}}\right)^{\mathbf{2}}}{\beta_{\mathbf{i}}^{\mathbf{l}}} \right] \right)}{\sum_{\mathbf{l=1}}^{\mathbf{M}} \left(\prod_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{exp} \left[-\frac{\left(\mathbf{x_i} - \alpha_{\mathbf{i}}^{\mathbf{l}}\right)^{\mathbf{2}}}{\beta_{\mathbf{i}}^{\mathbf{l}}} \right] \right)} \tag{2.6}$$

où $\gamma^l \in \Re$ est le centre de l'ensemble flou G^l associé à la sortie y dans la règle l (on suppose que $\mu_G^l(\gamma^l) = 1$) ; $\alpha_i^l \in \Re$ et $\beta_i^l \in \Re, (\beta_i^l > 0)$ sont, respectivement, la moyenne et la variance de la fonction d'appartenance de gauss de l'ensemble flou F_i^l associé à la variable x_i dans la règle l. $\mathbf{X} = (\mathbf{x_1}\ \mathbf{x_2}\ ...\ \mathbf{x_n})^{\mathbf{T}}$ dénote le vecteur de variables d'entrée x_i.

Preuve. Voir [7], p. 25.

Cette représentation d'un modèle flou permet d'envisager des algorithmes d'apprentissage pour un ajustement des paramètres γ^l, α_i^l and β_i^l. Dans [7], l'ajustement de ces paramètres est realisé en utilisant la méthode du gradient. Néanmoins, l'utilisation de la méthode du gradient peut ne pas être adéquate pour l'obtention d'un bon modèle flou. Mentionnons un point souligné dans [9] : après l'ajustement des paramètres, les règles floues du modèle simplifié (2.5) peuvent rester inactives quelle que soit l'entrée (voir figure 2.4). D'autres points sont aussi discutés en détail dans [9].

Dans [46], une approche pour améliorer l'activation des règles floues est presentée, en proposant une base de connaissance utilisant le même ensemble flou pour une variable dans l'ensemble des règles. Cette proposition permet d'obtenir une base de règles semblable à celles obtenues en utilisant un tableau de Mamdani (voir figure 2.5). De cette manière, le nombre de paramètres ajustables est considérablement réduit.

Les contraintes ci-dessus, rattachées aux MFA classiques qui utilisent des algorithmes d'ajustement basés sur la méthode du gradient, peuvent être attenuées ou supprimées, peut-être, par la définition de fonctions d'appartenance dynamiques captant le comportement

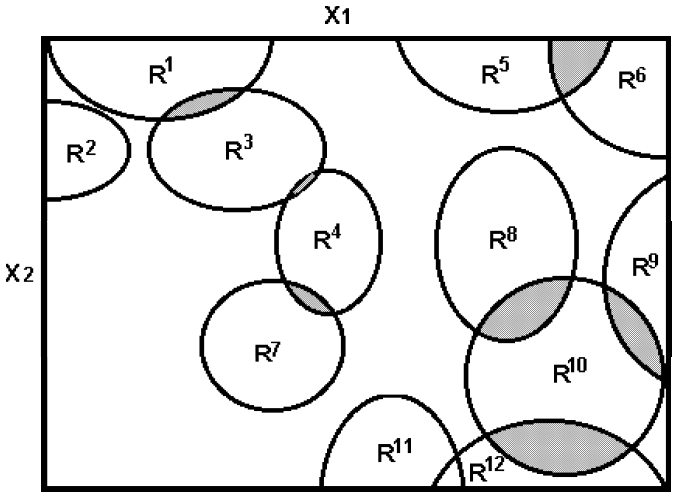

FIG. 2.4. Régions non couvertes par une base de règles flous.

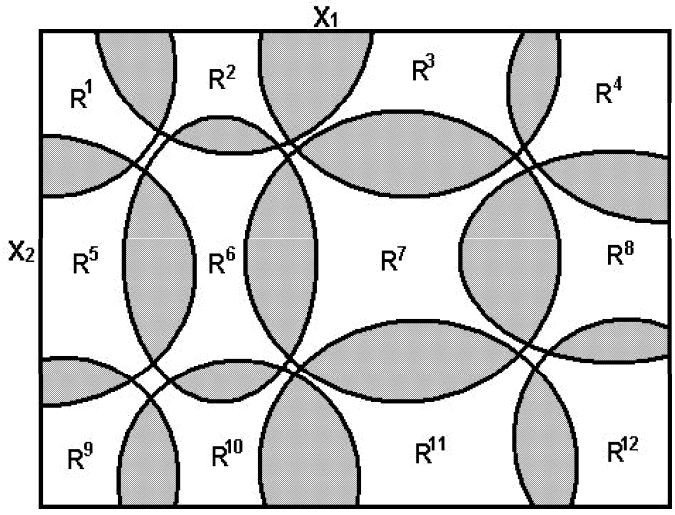

FIG. 2.5. Base de règles de type Mamdani.

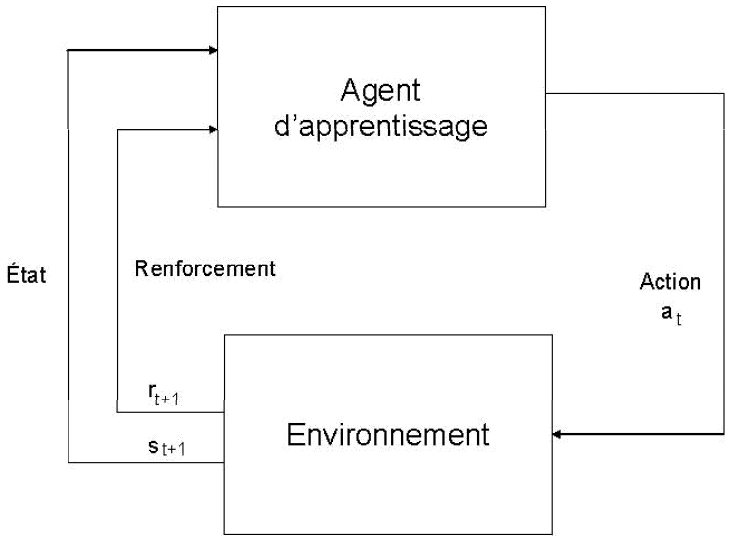

FIG. 2.6. Interaction Agent-Environnement.

temporel du système réel. Une proposition dans ce sens est l'apport essentiel de cette thèse.

2.3 L'apprentissage par renforcement et différences temporelles

L'apprentissage par renforcement (AR) fait référence au problème d'apprentissage basé sur l'essai et l'erreur pour arriver à un objectif global [18]. Dans un environnement dynamique, un *agent d'apprentissage* reçoit des récompenses ou des pénalités, selon sa relation avec l'environnement, afin d'apprendre les bonnes actions. Les problèmes qui utilisent l'apprentissage par renforcement sont ceux pour lesquels l'agent ne sait pas, a priori, ce qu'il doit faire. Ainsi, l'agent doit découvrir une politique d'action pour maximiser le *gain attendu*, lequel est une fonction spécifique de la séquence de récompenses ou pénalités que l'agent a reçu de l'environnement. Ces récompenses ou pénalités dépendent de la réalisation de bonnes ou mauvaises actions. Spécifiquement, agent d'apprentissage et environnement inter-réagissent à chaque temps t (discret ou continu) : l'agent reçoit une représentation de l'environnement appelée *l'état* S_t et, sur la base de cette information, choisit une *action* a_t. Un pas de temps plus tard, en conséquence de l'action prise, l'agent reçoit une *signal de renforcement ou récompense* r_{t+1}. Dans la figure 2.6, on montre l'interaction agent-environnement.

La somme de tous les renforcements qu'on espère recevoir sur un intervalle de temps fini et futur est appelée le *gain attendu* et est défini par l'équation suivante :

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + ...r_{t+m} = \sum_{k=0}^{m} r_{t+k+1} \tag{2.7}$$

Les observations sur chaque intervalle sont appelées *une séquence* et, à chaque pas, l'objectif est de maximiser le gain attendu sur toute la séquence.

Sur un horizon de temps infini, on introduit le concept *d'escompte* sur le gain attendu : l'agent d'apprentissage doit apprendre une politique d'action qui permet de maximiser le *gain ponderé*, défini par l'équation suivante :

$$R_t = r_{t+1} + \mu r_{t+2} + \mu^2 r_{t+3} + ... = \sum_{k=0}^{\infty} \mu^k r_{t+k+1} \tag{2.8}$$

où μ, $0 \leq \mu \leq 1$, est un paramètre appelé *taux d'escompte* qui détermine la pondération au temps t de la valeur des futurs renforcements. Par exemple, les problèmes d'optimisation de coût à horizon infini, où les conséquences des actions au temps t sur le futur ne sont pas connues, peuvent être considérés comme un problème d'AR défini par l'équation (2.8).

Trois méthodes de base sont rapportés dans la littérature pour résoudre le problème d'AR : la programmation dynamique, les méthodes de Monte Carlo (très appliquées dans le cas discret) et la méthode d'apprentissage par différences temporelles (DT) (intéressante pour le cas continu) [18].

L'apprentissage basée sur la méthode DT est une méthode d'apprentissage proposée pour résoudre le problème de prédiction en prenant en compte la différence (erreur) entre deux estimées d'affilée données par une fonction de prédiction $P(x, \theta)$ où x est un vecteur de données disponibles et θ est un vecteur des paramètres ajustables au cours du temps. Selon la méthode DT, la loi d'ajustement du vecteur θ de la fonction de prédiction P est donnée par l'équation suivante [19] :

$$\theta_{t+1} = \theta_t + \eta(P(x_{t+1}, \theta_t) - P(x_t, \theta_t)) \frac{\partial P(x_t, \theta_t)}{\partial \theta} \tag{2.9}$$

où x_t et x_{t+1} sont des vecteurs de données disponibles au temps t et $t+1$, θ_t est le vecteur des paramètres ajustables et $0 \leq \eta \leq 1$ est le taux d'apprentissage. Le terme entre parenthèses est appelé la *différence temporelle* et la loi d'ajustement est appelée, d'habitude, dans la littérature, *l'algorithme TD*.

On peut voir la problème de AR comme un problème de prédiction où l'objectif est la prédiction du gain donné par les équations (2.7) ou (2.8).

Soit $\hat{R}_t$ la prédiction au temps t de R_t. À partir de l'équation (2.8), on a :

$$R_t = r_{t+1} + \mu R_{t+1} \tag{2.10}$$

Vu le fait que la valeur réelle de R_{t+1} n'est pas disponible au temps t, en substituant R_{t+1} par sa valeur estimée $\hat{R}_{t+1}$ dans (2.10), l'erreur de prédiction entre la valeur réelle de R_t et sa valeur estimée $\hat{R}_t$ est défini comme :

$$\Delta = R_t - \hat{R}_t = r_{t+1} + \mu \hat{R}_{t+1} - \hat{R}_t \tag{2.11}$$

laquelle est une différence temporelle entre les valeurs estimées de R. La valeur du renforcement r_{t+1} doit être définie afin d'obtenir au temps $t+1$ la meilleure estimation de R_t en utilisant l'information disponible. Une "meilleure estimation", dans le problème d'apprentissage par renforcement, signifie l'optimisation de la valeur de $\hat{R}_t$. En changeant la notation

de $\hat{R}$ par P et en substituant la différence temporelle dans (2.9) par celle-là définie dans (2.11), on obtient la loi d'actualisation du vecteur des paramètres θ suivante :

$$\theta_{t+1} = \theta_t + \eta(r_{t+1} + \mu P(x_{t+1}, \theta_t) - P(x_t, \theta_t))\frac{\partial P(x_t, \theta_t)}{\partial \theta} \tag{2.12}$$

L'agent d'apprentissage qui utilise l'équation (2.12) pour l'ajustement de ses paramètres est caracterisé dans la littérature comme l'agent *Critique Heuristique Adaptatif* [17]. Les propositions qu'on peut trouver dans les références sur la fonction de prédiction P de l'algorithme TD portent sur une fonction linéaire par rapport au vecteur de paramètres ajustables θ_t, définie par l'équation suivante [18] :

$$P(x_t, \theta_t) = \theta_t x_t \tag{2.13}$$

où x_t est le vecteur de données disponibles au temps t. On parle, alors, d'Algorithme *TD linéaire.*

2.3.1 Le méthode DT et l'algorithme du gradient

La méthode DT peut être vue comme une descente du gradient par rapport aux paramètres ajustables de la fonction de prédiction P. L'objectif est de minimiser une mesure d'erreur, dénotée par $J(\theta)$, sur l'espace des paramètres, par l'algorithme suivant :

$$\Delta\theta_t = -\alpha \nabla_\theta J(\theta) \tag{2.14}$$

où α est le taux d'apprentissage et $\nabla_\theta J(\theta)$ est le gradient de $J(\theta)$ par rapport à θ.

Une mesure de l'erreur est définie d'habitude par l'équation suivante :

$$J(\theta, x) = (E\{z|x\} - P(x, \theta))^2 \tag{2.15}$$

où $E\{z|x\}$ dénote la valuer attendue de z, valeur réelle, à partir de la connaissance d'un vecteur de données disponibles x.

En utilisant (2.15) dans (2.14), chaque incrément sur le vecteur des paramètres est déterminé par l'équation suivante :

$$\Delta\theta_t = 2\alpha(E\{z|x_t\} - P(x_t, \theta))\nabla_\theta P(x_t, \theta) \tag{2.16}$$

La valeur de $E\{z|x_t\}$ n'est pas connue et doit être estimée. Si on fait une approximation de $E\{z|x_t\}$ par $P(x_{t+1}, \theta)$, c'est à dire, la prédiction suivante, on a l'algorithme DT défini en (2.9).

Dans [21, 22, 24, 25, 18], on peut trouver plus de détails sur la manière de résoudre le problème de AR en utilisant la méthode DT, et aussi des applications. Soulignons que la mise en oeuvre de cette méthode peut se faire en ligne, de façon incrémentale, ce qui constitue un énorme avantage.

Dans cette thèse, on utilise la méthode DT pour proposer un algorithme d'ajustement en ligne des paramètres dans un problème global d'apprentissage par renforcement.

2.4 Conclusion

Dans ce chapitre, on a présenté les aspects théoriques point de départ de notre travail. En particulier, on s'est intéressé à développer l'idée présentée dans [12] sur les fonctions d'appartenance dynamiques. Suite au besoin de l'ajustement des paramètres des fonctions d'appartenance pour l'identification des systèmes, à partir de données entrées-sorties, on a trouvé que le paradigme de l'apprentissage par renforcement fournissait des idées attractives pour la proposition d'algorithmes en ligne, afin d'obtenir des modèles d'identification ajustés en temps réel, selon l'évolution du système.

Dans le prochain chapitre, on présente notre contribution principale : le développement d'un modèle flou adaptatif avec des fonctions d'appartenance dynamiques ajustées, en ligne, par une technique de renforcement.

Chapitre 3

Le modèle flou adaptatif dynamique

Dans ce chapitre on précise les contributions sur la modélisation floue développées dans ce travail : d'une part, dans la première partie, la proposition des Modèles Flous Adaptatifs Dynamiques (MFAD) et, d'une autre part, dans la deuxième partie, la proposition d'un algorithme en ligne basé sur l'apprentissage par renforcement. Dans la troixième partie, on présente les principaux résultats obtenus sur l'identification de fonctions non linéaires variant avec le temps : on précise les performances du MFAD en utilisant la méthode du gradient comme algorithme d'apprentissage hors ligne puis, en utilisant l'algorithme en ligne basé sur l'apprentissage par renforcement

3.1 Structure du MFAD

Les modèles flous statiques, comme les MFA présentés dans le chapitre précédent, peuvent ne pas être adéquats pour la modélisation des systèmes variant avec le temps ou en présence de perturbations internes et externes agissant sur le système réel. La dynamique doit être pris en compte dans la modélisation floue en utilisant des fonctions d'appartenance dépendantes du temps (voir la section (2.1.1)). En utilisant cette idée, on peut proposer des modèles flous avec des fonctions d'appartenance variant avec le temps, ce qui conduit aux Modèles Flous Adaptatifs Dynamiques (MFAD). Ce MFAD peut s'adapter aux changements de comportement temporel des variables du système. La caractéristique dynamique des fonctions d'appartenance peut minimiser la faiblesse déjà mentionnée, associée à l'activation des règles floues dans les MFA utilisant des algorithmes d'apprentissage basés sur la méthode du gradient.

Dans cette section, on propose un modèle flou avec des fonctions d'appartenance dynamiques. Le modéle est obtenu à partir de l'extension du MFA donné par l'équation (2.6) en un MFA avec α, β et γ fonctions paramétriques dépendant du temps.

On donne, ci- après, les définitions les plus importantes pour notre travail.

Définition 1 *Une fonction d'appartenance est dynamique si sa structure ou ses paramétres changent avec le temps.*

Définition 2 *Soit $F(t)$ un ensemble flou dépendant du temps sur l'univers du discours d'une variable floue X. La fonction d'appartenance dynamique de Gauss $\mu_F(x,t)$ qui caractérise l'ensemble flou $F(t)$ est définie comme :*

$$\mu_F(x,t) = exp\left[-\frac{(x\ -\ \alpha(\mathbf{v},t))^2}{\beta(\mathbf{w},t)}\right] \tag{3.1}$$

où x est une valeur numérique sur l'univers du discours de X, $\alpha(\mathbf{v},t)$ et $\beta(\mathbf{w},t)$ sont des fonctions dépendant du temps, $\mathbf{v}$ et $\mathbf{w}$ sont des vecteurs de paramètres.

Définition 3 *Soit $G(t)$ un ensemble flou dépendant du temps caractérisé par une fonction d'appartenance quelconque définie sur l'univers du discours d'une variable floue Y. Le centre de la fonction d'appartenance dynamique $\mu_G(y,t)$ qui caractérise l'ensemble flou $G(t)$ est défini par une fonction dépendante du temps $\gamma(\mathbf{u},t)$ telle que $\mu_G(\gamma(\mathbf{u},t))$ atteint sa valeur maximale. On suppose que $\mu_G(\gamma(\mathbf{u},t)) = 1$ (ensemble normalisé).*

Définition 4 *Un modèle flou est dynamique si ses fonctions d'appartenance sont dynamiques.*

En conséquence, l'expression analytique du MFAD, basée sur l'équation (2.6), est définie par le lemme suivant :

Lema 2 *Le système de logique floue (2.5), avec des fonctions d'appartenance dynamiques pour les ensembles flous F_i^l et G^l, est décrit par l'expression analytique suivante :*

$$y(\mathbf{X},\mathbf{t}) = \frac{\sum_{\mathbf{l=1}}^{\mathbf{M}} \gamma^{\mathbf{l}}(\mathbf{u^l},\mathbf{t}) \left(\prod_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{exp}\left[-\frac{\left(\mathbf{x_i}\ -\ \alpha_{\mathbf{i}}^{\mathbf{l}}(\mathbf{v_i^l},\mathbf{t})\right)^{\mathbf{2}}}{\beta_{\mathbf{i}}^{\mathbf{l}}\ (\mathbf{w_i^l},\mathbf{t})}\right]\right)}{\sum_{\mathbf{l=1}}^{\mathbf{M}} \left(\prod_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{exp}\left[-\frac{\left(\mathbf{x_i}\ -\ \alpha_{\mathbf{i}}^{\mathbf{l}}(\mathbf{v_i^l},\mathbf{t})\right)^{\mathbf{2}}}{\beta_{\mathbf{i}}^{\mathbf{l}}\ (\mathbf{w_i^l},\mathbf{t})}\right]\right)} \tag{3.2}$$

où $\mathbf{X}$ dénote le vecteur des variables d'entrée x_i ; t est le temps ; $\mathbf{u}^l$, $\mathbf{v}_i^l$ et $\mathbf{w}_i^l$ sont des vecteurs de paramètres ajustables.

Preuve. Identique à celle du lemme 1.

Une fois que le MFAD est défini, les fonctions $\alpha_i^l(\mathbf{v}_i^l,t)$, $\beta_i^l\ (\mathbf{w}_i^l,t)$ et $\gamma^l(\mathbf{u}^l,t)$ doivent être précisées. Dans ce travail, ces fonctions sont choisies en prenant en compte leur signification dans l'équation (3.2).

Définition 5 *Soit $x_i(t_j)$, $i = 1,...,n$, les valeurs des variables d'entrée du MFAD au temps $t = t_j$. La structure dynamique pour la fonction $\alpha_i^l(\mathbf{v}_i^l,t_j)$ est donnée par l'équation suivante :*

$$\alpha_i^l(\mathbf{v}_i^l,t_j) = v_i^l * \overline{x}_i(t_j) \tag{3.3}$$

où :

$$\overline{x}_i(t_j) = \frac{\sum_{k=j-\delta_1}^{j}(x_i(t_k))}{\delta_1+1},\quad \delta_1 \in \aleph \tag{3.4}$$

et $v_i^l \in \Re$ est un paramètre ajustable.

Définition 6 *Soit $x_i(t_j)$, $i = 1, ..., n$, les valeurs des variables d'entrée du MFAD au temps $t = t_j$. La structure dynamique pour la fonction $\beta_i^l(\mathbf{w}_i^l, t_j)$ est donnée par l'équation suivante :*

$$\beta_i^l(\mathbf{w}_i^l, t_j) = w_i^l * (\sigma_i^2(t_j) + \epsilon); \quad \epsilon \in \aleph \tag{3.5}$$

où :

$$\sigma_i^2(t_j) = \frac{\sum_{k=j-\delta_1}^{j}(x_i(t_k) - \overline{x}_i(t_k))^2}{\delta_1 + 1}, \quad \delta_1 \in \aleph \tag{3.6}$$

$w_i^l \in \Re, (w_i^l > 0)$ est un paramètre ajustable et $\overline{x}_i(t_k)$ est donnée par l'équation (3.4).

Définition 7 *Soit $y(t_j)$ la valeur de la variable de sortie obtenue par le MFAD à partir des valeurs d'entrées $x_i(t_j)$. En prenant en compte la signification de la fonction $\gamma^l(u^l, t_j)$ dans l'équation (3.2), on propose la structure suivante :*

$$\gamma^l(u^l, t_j) = u^l * \overline{y}(t_j) \tag{3.7}$$

où

$$\overline{y}(t_j) = \frac{\sum_{k=j-\delta_2}^{j-1}(y(t_k))}{\delta_2}, \quad \delta_2 \in \aleph \tag{3.8}$$

L'équation (3.4) est la valeur moyenne des observations précédentes de la variable d'entrée x_i sur l'intervalle $[t_j - \delta_1, t_j]$. L'équation (3.6) est la valeur de la variance des valeurs $x_i(t_k)$ par rapport à $\overline{x}_i(t_k)$, sur l'intervalle $[t_j - \delta_1, t_j]$. L'équation (3.8) est la moyenne des δ_2 valeurs précédentes de la variable de sortie y jusqu'au temps $t_{(j-1)}$.

L'expression proposée dans (3.3) permet l'ajustement des fonctions d'appartenance de Gauss de l'ensemble flou F_i^l autour de la moyenne $\overline{x}_i(t_j)$, pondérée par le paramètre v_i^l. L'équation(3.5) laisse place à l'ajustement de la pente de cette fonction d'appartenance autour $\sigma_i^2(t_j)$, pondérée par le paramètre w_i^l. ϵ est un biais constant pour prévenir une indétermination numérique dans l'équation (3.2) lorsque $\sigma_i^2(t_j)$ s'approche de zéro. L'équation (3.7) permet l'ajustement du centre de l'ensemble flou de sortie G^l autour $\overline{y}(t_j)$, pondérée par le paramètre u^l.

Remarque 2 *Ces définitions permettent d'avoir une idée très claire sur les valeurs de paramètres u^l, v_i^l et w_i^l : ils doivent être autour de 1 pour arriver à des ensembles flous centrés autour de la moyenne des variables correspondantes et fonction de la variance de ces mêmes variables. Cependant, il faut faire de l'attention : des valeurs très proches de 1 peuvent donner lieu à une partition des ensembles flous très proches aussi, ce qui minimise la caractérisation linguistique du modèle (problème de "granularité").*

La figure 3.1 montre des fonctions d'appartenance de Gauss à moyenne et variance variables sur un domaine de discours $[0, 10]$ (voir aussi la figure 2.3) : les fonctions d'appartenance en continu sont proposées au temps t_1, les fonctions d'appartenance en tirets sont proposées au temps t_2 et les lignes en pointillés montrent les fonctions d'appartenance au temps t_3.

La structure proposée pour les fonctions $\alpha_i^l(\mathbf{v}_i^l, t)$, β_i^l $(\mathbf{w}_i^l, t)$ et $\gamma^l(\mathbf{u}^l, t)$ sera très utile pour le nouveau modèle flou en vue de l'identification de systèmes variant avec le temps parce qu'il utilise, au temps t, la valeur moyenne et la variance des variables, sur une fenêtre finie du temps récent.

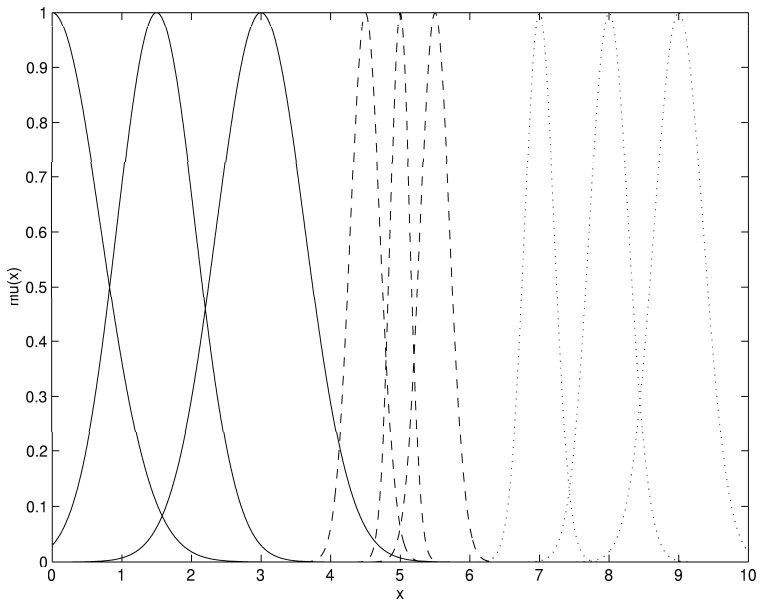

FIG. 3.1. Fonctions d'appartenance dynamiques, pour une variable x, aux temps t_1,t_2,t_3.

3.2 Ajustement des paramètres du MFAD

Il existe des méthodes variées pour l'optimisation des paramètres d'un MFA (supervisées, non supervisées, des méthodes classiques, des méthodes basées sur des techniques émergentes, etc.).

Dans ce travail, on utilise, d'une part, la méthode classique du gradient pour l'ajustement hors ligne des paramètres du MFAD et, d'une autre part, on propose un algorithme en ligne basé sur l'apprentissage par renforcement pour l'ajustement de ces mêmes paramètres.

3.2.1 Algorithme d'apprentissage hors ligne en utilisant la méthode du gradient

Soit E la fonction d'erreur quadratique moyenne donnée par l'équation (3.9), définie sur un ensemble de données d'apprentissage $\{(\mathbf{X}(t_j), y(t_j)),\ j = 1, ..., N\}$:

$$E = \frac{1}{T}\sum_{j=1}^{T}\frac{1}{2}(y_e(t_j) - y(t_j))^2 \tag{3.9}$$

où $y(t_j)$ est la sortie du système et $y_e(t_j)$ est la sortie estimée par le MFAD (3.2), au temps t_j. En utilisant cette fonction d'erreur et la méthode du gradient, on propose les lois d'ajustement de paramètres données par les équations ci-dessous :

$$u^l(K+1) = u_p^l(K) - \rho_u \frac{\partial E}{\partial u^l}\bigg|_K \tag{3.10}$$

$$v_i^l(K+1) = v_i^l(K) - \rho_v \frac{\partial E}{\partial v_q^l}\bigg|_K \tag{3.11}$$

$$w_i^l(K+1) = w_i^l(K) - \rho_w \frac{\partial E}{\partial w_i^l}\bigg|_K \tag{3.12}$$

où K est l'indice d'itération dans la phase d'apprentissage et ρ_u, ρ_v and ρ_w sont les taux d'apprentissage.

En développant les équations précédentes, on obtient :

$$\left.\frac{\partial E}{\partial u^l}\right|_K = \frac{1}{T}\sum_{j=1}^{T}\left[\frac{(y_e(t_j)-y(t_j))}{d(t_j)}h^l(t_j)\frac{\partial \gamma^l(t_j)}{\partial u^l}\right]\Bigg|_K \tag{3.13}$$

$$\left.\frac{\partial E}{\partial v_i^l}\right|_K = \frac{1}{T}\sum_{j=1}^{T}\left[2\frac{(y_e(t_j)-y(t_j))}{d(t_j)}(\gamma^l(t_j)-y_e(t_j))h^l(t_j)\frac{\left(x_i\ -\ \alpha_i^l(t_j)\right)}{\left(\beta_i^l(t_j)\right)}\frac{\partial \alpha_i^l(t_j)}{\partial v_i^l}\right]\Bigg|_K \tag{3.14}$$

$$\left.\frac{\partial E}{\partial w_i^l}\right|_K = \frac{1}{T}\sum_{j=1}^{T}\left[\frac{(y_e(t_j)-y(t_j))}{d(t_j)}(\gamma^l(t_j)-y_e(t_j))h^l(t_j)\frac{\left(x_i\ -\ \alpha_i^l(t_j)\right)^2}{\left(\beta_i^l(t_j)\right)^2}\frac{\partial \beta_i^l(t_j)}{\partial w_i^l}\right]\Bigg|_K \tag{3.15}$$

où :
$d(t_j) = \sum_{l=1}^{M} h^l(t_j)$

$h^l(t_j) = \prod_{i=1}^{n} exp\left[-\frac{\left(x_i\ -\ \alpha_i^l(t_j)\right)^2}{\beta_i^l\ (t_j)}\right]$

En substituant (3.13), (3.14) et (3.15) respectivement dans (3.10), (3.11) et (3.12), les lois d'ajustement sont définies. Dans le cas particulier des fonctions proposées dans (3.3), (3.5) et (3.7), elles sont :

$$\frac{\partial \alpha_i^l(t_j)}{\partial v_i^l} = \overline{x_i}(t_j) \tag{3.16}$$

$$\frac{\partial \beta_i^l(t_j)}{\partial w_i^l} = (\sigma_i^2(t_j)+\epsilon) \tag{3.17}$$

$$\frac{\partial \gamma^l(t_j)}{\partial u^l} = \overline{y}(t_j) \tag{3.18}$$

3.2.2 Algorithme d'apprentissage en ligne basé sur l'apprentissage par renforcement

On présente, maintenant, l'algorithme d'apprentissage en ligne développé dans ce travail. En particulier, on reprend les propositions de base de l'apprentissage par renforcement, utilisant la méthode des différences temporelles, comme une extension de l'algorithme $TD(\lambda)$ linéaire, proposé par Sutton dans [18], pour le problème de modélisation floue.

Partant de l'utilisation de l'algorithme $TD(\lambda)$ classique pour le problème de modélisation floue, l'algorithme qu'on propose présente quelques différences importantes par rapport à cet algorithme classique :

– Le MFAD paramétrique est incorporé dans la fonction de prédiction de telle manière que l'ajustement de la fonction de prédiction implique l'ajustement du modèle flou. En conséquence, la fonction de prédiction est une fonction non linéaire par rapport aux paramètres à ajuster.

– La valeur du signal de renforcement est définie à partir de la différence temporelle et de l'erreur d'identification.

Pour présenter l'algorithme TD non linéaire proposé, on définit, d'abord, la fonction de prédiction associée à l'objectif du modèle floue qui est d'estimer un comportement réel (identification floue). On définit, ensuite, le signal de renforcement en prenant en compte le problème de prédiction du gain ponderé R_t et la fonction de prédiction qui a été proposée.

1. **La fonction de prédiction**
   Il faut rappeler que la fonction de prédiction de l'algorithme TD dans ce qu'on appelle *Algorithme TD linéaire* (voir la section 2.3), est une fonction linéaire par rapport au vecteur de paramètres ajustables

   Dans ce travail, on propose une fonction de prédiction non linéaire introduite de manière naturelle afin d'ajuster les paramètres du MFAD. L'agent d'apprentissage est utilisé pour prédire la performance du MFAD dans l'identification floue du système à modéliser. De cette manière, la fonction de prédiction est définie comme une fonction dépendant de *l'erreur d'identification* :

$$e(x_t, \theta_t) = y(t) - y_e(x_t, \theta_t) \tag{3.19}$$

   où $y(t)$ dénote la variable de sortie réelle, au temps t, et $y_e(x_t, \theta_t)$ la variable de sortie estimée par le MFAD (3.2), en utilisant les valeurs disponibles du vecteur de paramètres θ_t et du vecteur d'information x_t, au temps t.

   Remarque 3 *Le vecteur des paramètres θ_t fait référence aux paramètres u^l, v_i^l et w_i^l du MFAD.*

   Remarque 4 *L'information contenue dans le vecteur x_k fait référence à la moyenne $\overline{x}(k)$, à la variance $\sigma^2(k)$, à la moyenne $\overline{y}(k)$ et aux données d'entrée $x_i(k)$ du MFAD disponibles au temps $t = k$.*

   Soit P_t la fonction de prédiction non linéaire que l'on propose :

$$P(x_t, \theta_t) = \frac{1}{2} \sum_{k=K}^{t} (\mu\lambda)^{t-k} e^2(x_k, \theta_t) \tag{3.20}$$

   avec $e(t, \theta_t)$ décrit par l'équation (3.19) et $0 < \mu, \lambda < 1$.
   Cette fonction de prédiction permet de prendre en compte toutes les erreurs obtenues jusqu'à K pas de temps en arrière, en utilisant la valeur des paramètres au temps t et l'information disponible au temps k.

   Remarque 5 *Le paramètre μ conserve le sens qui lui ait donné dans l'équation (2.8)*

   Remarque 6 *Le paramètre λ pondère l'influence des erreurs passées.*

   Remarque 7 *Le critère de choix des valeurs des paramètres μ et λ est simple : ces valeur doivent permettre de capter les mesures les plus importantes pour les propriétés courantes du système. En conséquence, des valeurs autour de 1 correspondent à un bon choix.*

En développant la dérivée partielle de (3.20), on obtient l'équation suivante :

$$\frac{\partial P(x_t,\theta_t)}{\partial \theta} = \sum_{k=K}^{t} (\mu\lambda)^{t-k} e(x_k,\theta_t)\Psi(k,\theta_t) \tag{3.21}$$

où

$$\Psi(k,\theta_t) = \frac{\partial e(x_k,\theta_t)}{\partial \theta} \tag{3.22}$$

En substituant l'équation (3.21) dans l'équation (2.12), qui fait référence au problème d'AR, on a la loi suivante d'ajustement des paramètres :

$$\theta_{t+1} = \theta_t + \rho(r_{t+1} + \mu P(x_{t+1},\theta_t) - P(x_t,\theta_t)) \sum_{k=K}^{t} (\mu\lambda)^{t-k} e(x_k,\theta_t)\Psi(k,\theta_t) \tag{3.23}$$

où $\sum_{k=K}^{t} (\mu\lambda)^{t-k} e(k,\theta_t)\Psi(k,\theta_t)$ est la *trace d'éligibilité* qui garde une mémoire des erreurs d'identification du passé [18]. L'équation (3.23) correspond à la famille des algorithmes $TD(\lambda)$ presentée dans [18].
A partir de (3.20), on fait l'extension de $P(x_t,\theta_t)$ vers $P(x_{t+1},\theta_t)$ de la manière suivante :

$$\begin{aligned} P(x_{t+1},\theta_t) &= \frac{1}{2}\sum_{k=1}^{t+1} (\mu\lambda)^{t+1-k} e^2(x_k,\theta_t) \\ &= \frac{1}{2}\left[e^2(x_{t+1},\theta_t) + \sum_{k=1}^{t} (\mu\lambda)^{t+1-k} e^2(x_k,\theta_t) \right] \\ &= \frac{1}{2} e^2(x_{t+1},\theta_t) + \mu\lambda \left[\frac{1}{2}\sum_{k=1}^{t} (\mu\lambda)^{t-k} e^2(_k,\theta_t) \right] \\ &= \frac{1}{2} e^2(x_{t+1},\theta_t) + \mu\lambda P(x_t,\theta_t) \end{aligned} \tag{3.24}$$

En substituant (3.24) dans (3.23), on a la loi d'ajustement suivante :

$$\theta_{t+1} = \theta_t + \rho(r_{t+1} + \mu(\frac{1}{2} e^2(x_{t+1},\theta_t) + \mu\lambda P(x_t,\theta_t)) - P(x_t,\theta_t)) \sum_{k=1}^{t} (\mu\lambda)^{t-k} e(x_k,\theta_t)\Psi(k,\theta_t) \tag{3.25}$$

2. **La signal de renforcement**
Dans le problème de prédiction du gain pondéré R_t, on espère, à chaque itération, que la valeur estimée de $\hat{R}_t$ se rapproche de sa valeur réelle R_t. C'est équivalent à espèrer que $P(x_t,\theta_t)$ approche $r_{t+1} + \mu P(x_{t+1},\theta_t)$. Cette condition est déduite de l'équation (2.11). Sur la base de la fonction de prédiction proposée, qui est la somme pondérée de l'erreur d'identification au carré, $e^2(t)$, la signal de renforcement r_{t+1} doit être choisi tel que :

$$0 \leq r_{t+1} + \mu P(x_{t+1},\theta_t) < P(x_t,\theta_t) \tag{3.26}$$

D'un autre côté, si le modèle flou est ajusté correctement, on espère que :

$$0 < P(x_{t+1}, \theta_t) < P(x_t, \theta_t) \tag{3.27}$$

Le signal de renforcement r_{t+1} est donc choisi pour satisfaire la condition (3.26), sur la base de la condition espérée (3.27), de la manière suivante :

$$r_{t+1} = \begin{cases} r_{t+1} = 0 & si \quad P(x_{t+1}, \theta_t) < P(x_t, \theta_t) \\ r_{t+1} = -\frac{1}{2}\mu e^2(x_{t+1}, \theta_t) & si \quad P(x_{t+1}, \theta_t) > P(x_t, \theta_t) \end{cases} \tag{3.28}$$

En conséquence, l'erreur d'identification introduite dans la fonction de prédiction $P(x_{t+1}, \theta_t)$, selon l'équation (3.24), est refusée par le signal de renforcement si $P(x_{t+1}, \theta_t) > P(x_t, \theta_t)$.

3.3 Exemples illustratifs

L'identification des systèmes est un problème important pour les application réelles, particulièrement la commande des processus. La logique floue a été largement utilisée dans ce domaine et de bons résultats ont été obtenus [6, 4, 5, 10, 11]. Cependant, les modèles flous statiques peuvent ne pas être adéquats dans le cas de dynamiques variant avec le temps ou en présence de perturbations internes et externes importantes. Dans ces cas-ci, le MFAD proposé dans cette thèse a une utilité importante comme identificateur.

Les trois exemples qui suivent illustrent les performances du MFAD pour l'identifications des systèmes non linéaires variant avec le temps, en utilisant un apprentissage hors ligne (basé sur la méthode du gradient), puis en ligne (basé sur l'apprentissage par renforcement) des paramètres du modèle flou. Les premier et deuxième exemples correspondent à des modèles dont une partie non linéaire inconnue va être identifiée par un MFAD ; le troisième exemple correspond à une fonction totalement inconnue et le MFAD est utilisé pour construire totalement la relation entrée-sortie.

Dans tous les cas, l'équation (3.8) est utilisée en prenant $\delta_2 = 1$, ce qui correspond à $\overline{y}(t_j) = y(t_{j-1})$. En conséquence, le centre de l'ensemble flou de sortie G^l dépend de la dernière valeur disponible de la sortie réelle y. Les équations (3.4) et (3.6) utilisent $\delta_1 = 4$.

L'algorithme d'apprentissage supervisé, hors ligne, pour l'ajustement des paramètres du MFAD est mis en oeuvre selon les équations (3.10), (3.11) et (3.12) ; l'algorithme d'apprentissage en ligne est mis en oeuvre avec l'équation (3.25). De plus, le taux d'apprentissage ρ dans la loi d'ajustement peut être constant ou avoir un comportement décroissant selon l'équation suivante :

$$\rho(K) = \frac{\rho(K-1)}{\eta + \rho(K-1)}, \quad 0 < \eta < 1 \tag{3.29}$$

où K est l'itération courante. On espère, ainsi, un ajustement précis des paramètres à chaque itération. Habituellement, $\rho(0)$ est une valeur proche de 1 ; de plus, $\rho(K) \to 1 - \eta$ lorsque $K \to \infty$. Dans le cas d'un apprentissage en ligne, K coincide avec le temps courant t.

Afin d'obtenir les données d'apprentissage, dans le cas de l'utilisation des algorithmes d'ajustement des paramètres hors ligne, on choisit un signal d'entrée du type PRBS (Pseudo

Random Binary Signal, en anglais), habituellement utilisé pour le développement de modèles par identification [13].

3.3.1 Critères d'évaluation de la performance du MFAD

– **Ajustement des paramètres hors ligne**

Dans la phase d'apprentissage, la précision de prédiction des modéles flous est évaluée sur la base de la valeur de la racine carrée de l'erreur quadratique moyenne (Root Mean Squared Error (RMSE), en anglais), donné par l'équation suivante :

$$RMSE(K) = \sqrt{\frac{\sum_{n=1}^{N}(y(n) - y_e(\theta(K), n))^2}{N}} \tag{3.30}$$

où N est le nombre total des données de l'ensemble d'apprentissage et K est l'itération courante. θ est le vecteur des paramètres du modèle flou.
Dans la phase de validation, le RMSE est calculé en utilisant l'équation suivante :

$$RMSE = \sqrt{\frac{\sum_{t=1}^{T_f}(y(t) - y_e(\theta, t))^2}{N_{T_f}}} \tag{3.31}$$

où t est la variable temps et T_f est l'horizon de simulation ; N_{T_f} est le nombre total d'échantillons utilisés sur T_f. L'erreur d'identification relative $e_r(t) = \frac{(y(t)-y_e(\theta,t))}{y(t)}$, normalisée sur l'intervalle $[0, 1]$, est utilisée, aussi, comme une mesure de précision du modèle flou dans la phase de validation.
Pendant les expériences d'apprentissage hors ligne, les résultats obtenus en utilisant le MFAD sont comparés avec ceux obtenus en utilisant le MFA classique, donné par (2.6), α, β et γ étant des paramètres ajustables par la méthode du gradient. Dans ce cas, les équations (3.10), (3.11) et (3.12) sont utilisées en prenant :

$$\frac{\partial \alpha_i^l(t_j)}{\partial v_i^l} = 1 \tag{3.32}$$

$$\frac{\partial \beta_i^l(t_j)}{\partial w_i^l} = 1 \tag{3.33}$$

$$\frac{\partial \gamma^l(t_j)}{\partial u^l} = 1 \tag{3.34}$$

– **Ajustement des paramètres en ligne**

La performance du MFAD est mesurée par l'équation (3.31). Les expériences d'apprentissage en ligne sont mises en comparaison avec les résultats obtenus par l'algorithme d'apprentissage hors ligne, en utilisant une mesure de pourcentage d'erreur en prenant comme référence la valeur du RMSE obtenu dans le cas hors ligne. Cette mesure est donnée par l'équation suivante :

$$\%RMSE = \frac{RMSE \;\; en \;\; ligne}{RMSE \;\; hors \;\; ligne} * 100 \tag{3.35}$$

A partir de l'équation précédente, on peut calculer le *pourcentage d'amélioration* de la performance d'un algorithme par rapport à l'autre.

Finalement, dans les deux cas d'apprentissage hors ligne et en ligne, on valide la performance des modèles flous en présence d'une perturbation interne. Cette perturbation peut signifier, au niveau des applications d'ingénierie, la présence d'une faute ou un changement des valeurs habituelles des paramètres du système par changement de la région d'opération, par exemple. L'objectif est, alors, de visualiser le comportement des modèles obtenus dans ce cas.

3.3.2 Exemple 1

Dans cette exemple, le système est décrit par l'équation aux différences suivante :

$$y(k+1) = 0.3y(k) + 0.6y(k-1) + g[u(k)] \tag{3.36}$$

où $g[u(k)] = a(k)\sin(\pi u(k)) + 0.3\sin(3\pi u(k)) + 0.1\sin(5\pi u(k))$ et $a(k)$ est un paramètre variant avec le temps décrit par $a(k) = 0.6 + 0.3\sin(2\pi k/50)$.

La partie inconnue $g[u(k)]$ est estimée en utilisant un modèle flou. On souhaite $g_e[u(k)] \approx g[u(k)]$, où $g_e[u(k)]$ est donné par le modèle flou. Il faut remarquer que la partie inconnue $g[u(k)]$ dépend seulement de la variable d'entrée $u(k)$; par conséquent, la variable d'entrée du modèle flou est $x_1(k) = u(k)$. Cette exemple a été développé dans [2, 7, 47] mais seulement dans sa version invariante (a=cte).

Un ensemble de 250 couples de données d'apprentissage $\{(\underline{X}(t_j), y(t_j)),\ j = 1, ..., 250\}$ a été obtenu à partir d'une entrée PRBS appliquée au système, avec les niveaux $[-2.5, 2.5]$ et une longueur maximale de $L = 18$.

1. **Résultats expérimentaux en utilisant un MFA**

   Plusieurs cycles ont été realisés dans la phase d'apprentissage jusqu'à convergence de l'erreur d'identification, avec des valeurs différentes des taux d'apprentissage ρ_α ,ρ_β, ρ_γ et du nombre de règles M. Les valeurs initiales des paramètres ont été choisies de façon aléatoire comme suggéré dans [7] : $\alpha_i^l \in [-1, 1]$, $\beta_i^l \in [0, 0.03]$ et $\gamma^l \in [-5, 5]$.

   Le tableau 3.1 résume le RMSE obtenu dans la phase d'apprentissage et la figure 3.2 montre les résultats les plus importants avec $M = 10, 20, 30, 40$. Les valeurs de convergence du RMSE sont identiques dans tous les cas et la performance finale, dans cette phase-ci, n'est pas améliorée en utilisant d'autres valeurs des taux d'apprentissage.

TAB. 3.1
RMSE POUR L'ENSEMBLE D'APPRENTISSAGE, SELON LE NOMBRE DE RÈGLES M, EN UTILISANT UN MFA (EXEMPLE 1).

M	No. de paramètres	ρ_α	ρ_β	ρ_γ	RMSE
10	30	0.7	0.3	0.7	0.134
20	60	0.5	0.3	0.5	0.13
30	90	0.7	0.3	0.7	0.13
40	120	0.7	0.3	0.7	0.134

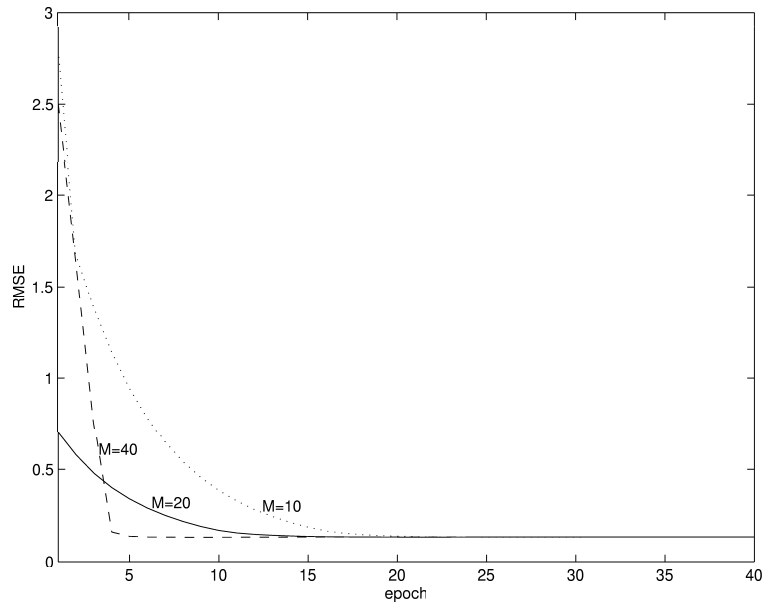

FIG. 3.2. Courbes d'évolution du RMSE pour la phase d'apprentissage en utilisant un MFA, en fonction du nombre de règles (Exemple 1).

Chaque modèle obtenu dans la phase d'apprentissage a été évalué en utilisant la signal d'entrée $u(k) = u_1(k)$ suivant :

$$u_1(k) = \begin{cases} \sin(2\pi k/250) & si \quad 1 < k < 250 \quad et \quad 501 < k < 700 \\ 3 + (0.5\sin(2\pi k/250) + 0.5\sin(2\pi k/25)) & si \quad 251 < k < 500 \end{cases} \tag{3.37}$$

Le tableau 3.2 résume le RMSE pour les données de validation. Basé sur ce résultat, le MFA, avec $M = 10$, a la meilleure performance.

TAB. 3.2
RMSE POUR LES DONNÉES DE VALIDATION, SELON LE NOMBRE DE RÈGLES M, EN UTILISANT UN MFA (EXEMPLE 1).

M	RMSE
10	0.36
20	0.85
40	0.42

La figure 3.3 montre l'erreur relative d'identification en utilisant le MFA, avec $M = 10$. Une autre validation a été faite, en prenant $a(k) = 0$, $k > 400$, afin d'évaluer le comportement du MFA face à une perturbation interne. Dans ce cas-ci, l'erreur d'identification relative est montrée, figure 3.4 : on a obtenu $RMSE = 0.3882$.
Au vu des résultats précédents, on constate que la vitesse de convergence (figure 3.2) et la performance du modèle dans la phase d'apprentissage (tableau 3.1) est adéquate, mais la performance du MFA dans la phase de validation, par rapport au RMSE, pourrait être améliorée (voir l'erreur d'identification relative dans les figures 3.3 et 3.4). Dans les expérimentations suivantes, on utilise le MFAD pour l'identification de $g[u(k)]$.

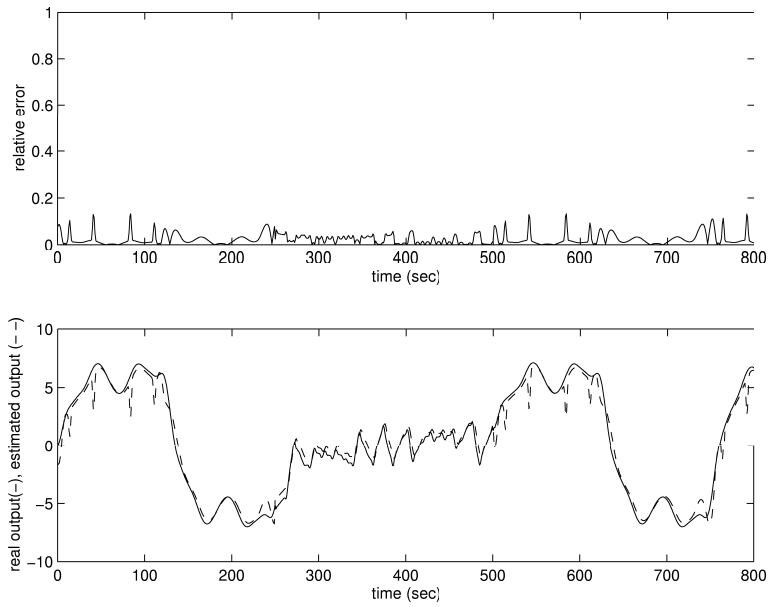

FIG. 3.3. Erreur d'identification relative, en utilisant le MFA, avec $u(k) = u_1(k)$, pour $M = 10$ (Exemple 1).

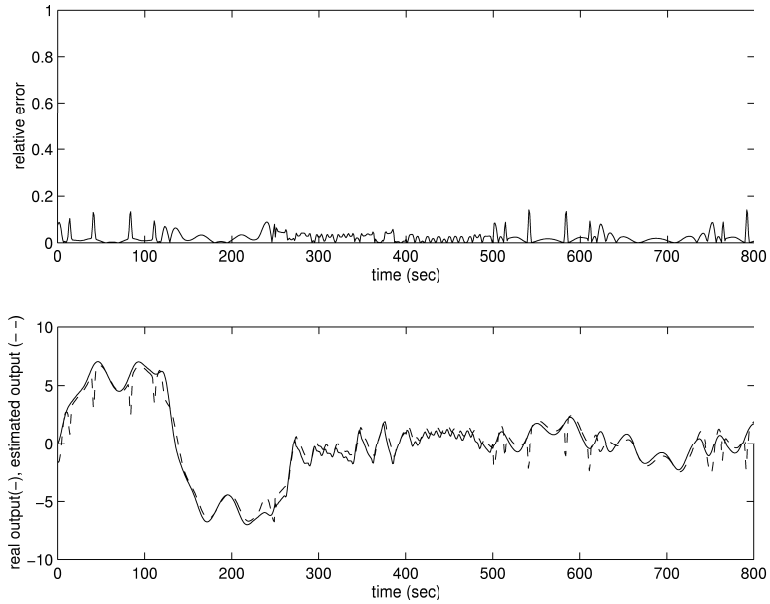

FIG. 3.4. Erreur d'identification relative, en utilisant le MFA, avec $u(k) = u_1(k)$ et $a(k) = 0$, $k > 400$ (Exemple 1).

2. **Résultats expérimentaux en utilisant le MFAD et l'apprentissage hors ligne**

   Le même ensemble d'apprentissage utilisé pour les expérimentations précédentes a été repris pour l'apprentissage d'un MFAD. Les valeurs initiales des paramètres u^l, v_i^l et w_i^l ont été choisies d'une manière aléatoire sur l'intervalle $[0, 1]$ avec, de plus, $\epsilon = 0.01$.

   Sur le tableau 3.3, on montre les résultats les plus importants dans la phase d'apprentissage. Les valeurs des taux d'apprentissage $\rho_u = \rho_v = 0.7$, $\rho_w = 0.3$ ont été retenues après de nombreuses séances d'essai et erreur.

   Comme pour l'utilisation du MFA, la valeur de convergence du RMSE est la même

TAB. 3.3
RMSE POUR L'ENSEMBLE D'APPRENTISSAGE, SELON LE NOMBRE DE RÈGLES M, EN UTILISANT UN MFAD (EXEMPLE 1).

M	No. des paramètres	RMSE
6	18	0.1718
8	24	0.1714
10	30	0.1717
15	45	0.1736

dans tous les cas et la figure 3.5 montre les courbes de performance.

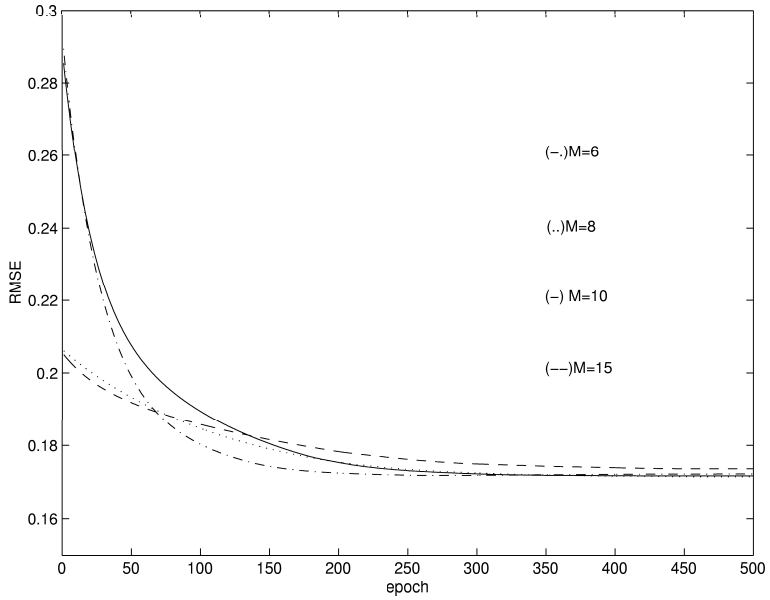

FIG. 3.5. Courbes d'évolution du RMSE pour la phase d'apprentissage, en utilisant un MFAD (Exemple 1).

Les MFAD obtenus dans la phase d'apprentissage ont été évalués en utilisant la signal de validation $u(k) = u_1(k)$, défini par l'équation (3.37). Les valeurs du RMSE obtenues pour les données de validation sont résumées sur la table 3.4. Basé sur les résultats obtenus dans la phases de validation, le MFAD, avec $M = 15$, a la meilleure performance, mais les modèles avec $M = 6$, ou $M = 8$, ont aussi une performance adéquate, tant dans la phase d'apprentissage que dans la phase de validation.

La figure 3.6 montre l'erreur d'identification relative, pour les données de validation, en utilisant le MFAD, avec $M = 6$. La performance du MFAD, en présence d'une perturbation interne, en prenant $a(k) = 0$, $k > 400$, est montrée dans la figure 3.7. Dans ce cas-ci, on obtient $RMSE = 0.0769$.

Afin de comparer les performances du MFA et du MFAD que l'on propose, la table 3.5 résume les résultats pour le meilleur modèle flou, dans les deux cas. Bien que le MFA ait un meilleur RMSE dans la phase d'apprentissage, l'erreur d'identification dans la phase de validation est améliorée par le MFAD, même quand peu de règles sont utilisées par le MFAD. On peut dire que la fonction non linéaire est très bien approchée par le MFAD.

TAB. 3.4
RMSE POUR LES DONNÉES DE VALIDATION, SELON LE NOMBRE DE RÈGLES M, EN UTILISANT UN MFAD (EXEMPLE 1).

M	RMSE
6	0.1010
8	0.1006
10	0.1276
15	0.0955

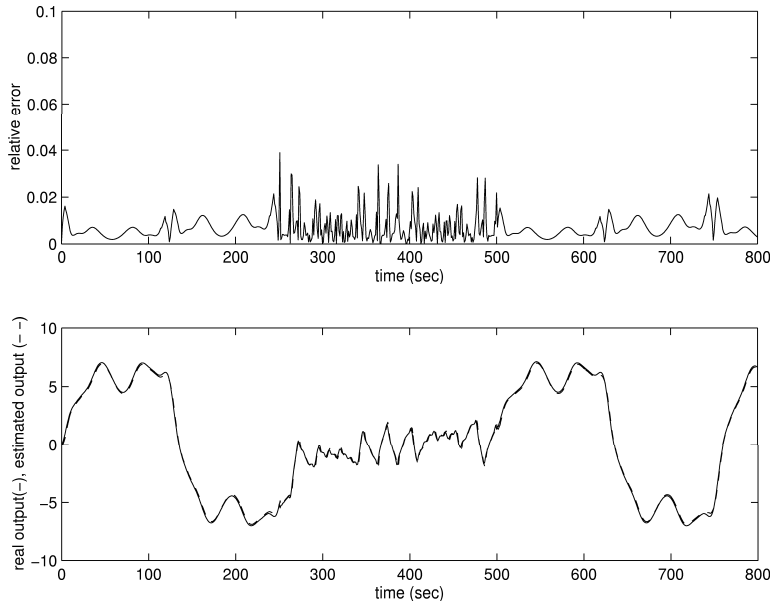

FIG. 3.6. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_1(k)$ (Exemple 1).

3. **Résultats expérimentaux en utilisant le MFAD et l'apprentissage en ligne**

   Dans cette partie, on utilise l'algorithme d'apprentissage en ligne donné par l'équation (3.25) pour l'ajustement des paramètres du MFAD, avec $\lambda = \mu = 0.9$. Le taux d'apprentissage ρ, dans cette équation, est mis à jour par l'équation (3.29) avec $\eta = 0.01$, à chaque itération du temps $t = k$.

   Pour ces expériences, on valide la performance de cet algorithme en utilisant, d'une part, l'entrée $u(k) = u_1(k)$ décrite par l'équation (3.37) et, d'autre part, en appliquant une perturbation interne en prenant $a(k) = 0$ pour $k > 400$.

   Afin de comparer les résultats actuels avec ceux obtenus en utilisant un apprentis-

TAB. 3.5
MEILLEURS RÉSULTATS POUR LES DONNÉES D'APPRENTISSAGE ET VALIDATION EN UTILISANT LE MFA ET LE MFAD (EXEMPLE 1).

Modèle	M	RMSE (phase d'apprentissage)	RMSE (phase de validation)	RMSE (perturbation interne)
MFA	10	0.134	0.36	0.3882
MFAD	6	0.1718	0.1006	0.0769

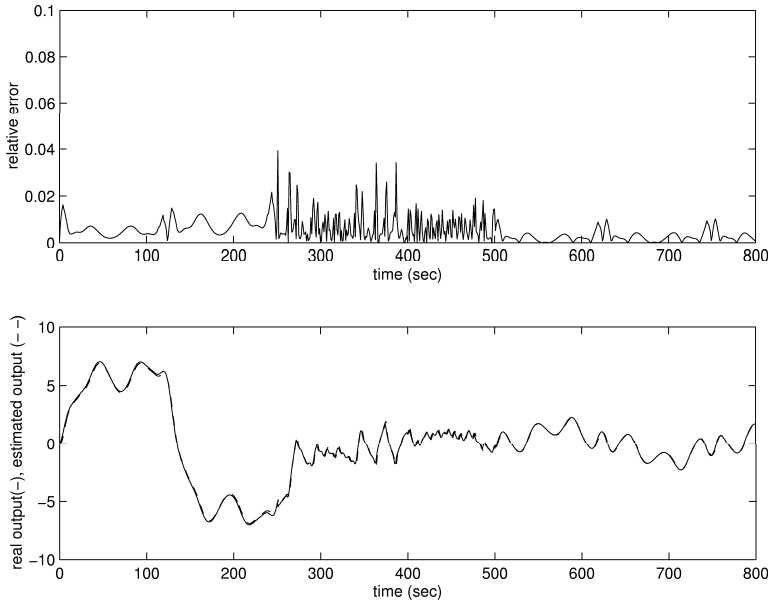

FIG. 3.7. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_1(k)$ et $a(k) = 0$, $k > 400$ (Exemple 1).

sage hors ligne, on prend les RMSE obtenus par l'ajustement en ligne et on calcule le pourcentage d'amélioration, via l'équation (3.35), par rapport aux RMSE obtenus dans la phase de validation, pour le cas hors ligne (donnés dans le tableau 3.4). On présente dans le tableau 3.6 les RMSE obtenus en utilisant l'apprentissage en ligne et le pourcentage d'amélioration susmentionné. L'espace blanc indique qu'aucune simulation hors ligne n'a été faite pour ce nombre de règles.

TAB. 3.6
RMSE EN UTILISANT LE MFAD AVEC $u(k) = u_1(k)$ ET L'APPRENTISSAGE EN LIGNE (EXEMPLE 1).

M	RMSE	% d'amelioration	RMSE (perturbation interne)
6	0.0834	17.46	0.0756
8	0.0769	23.56	0.0690
10	0.0875	31.43	0.0808
12	0.0930		0.0857

Le tableau précédent montre qu'une bonne performance du MFAD est atteinte avec $M = 8$ et, pour tous les nombres de règles, la performance du MFAD est améliorée en utilisant l'apprentissage en ligne. On observe, aussi, que la meilleure performance obtenue en utilisant l'apprentissage hors ligne, avec $M = 15$ (voir le tableau 3.4), est ameliorée en utilisant l'algorithme en ligne mais avec un nombre de règles plus petit.

La figure 3.8 montre l'erreur relative d'identification en utilisant le MFAD, avec $M = 8$. Une autre validation, avec $M = 8$, a été faite en prenant la perturbation interne déjà décrite et la performance correspondante du MFAD est présentée, figure 3.9. Dans ce cas, on obtient une amélioration du 10.27% par rapport au RMSE obtenu en utilisant l'algorithme hors ligne, avec $M = 6$ (voir la partie (2) de cet exemple).

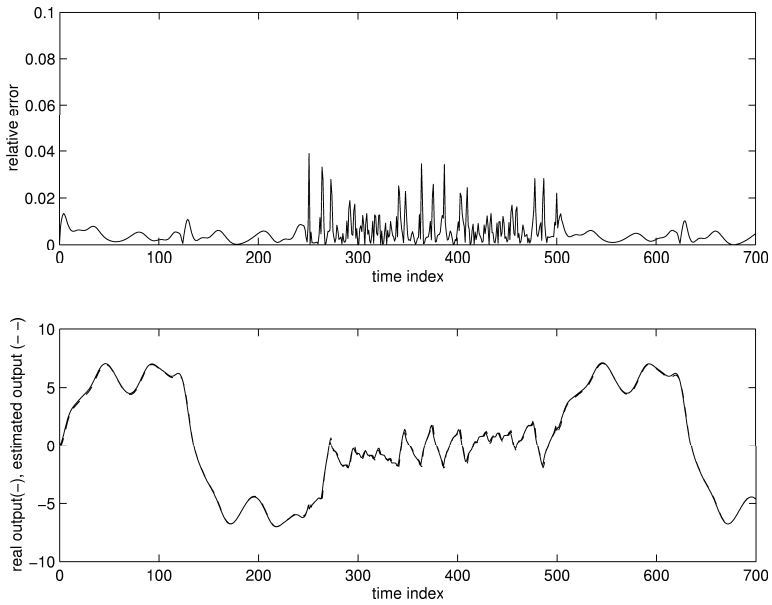

FIG. 3.8. Erreur d'identification relative, en utilisant le MFAD et l'AR, avec $u(k) = u_1(k)$ (Exemple 1).

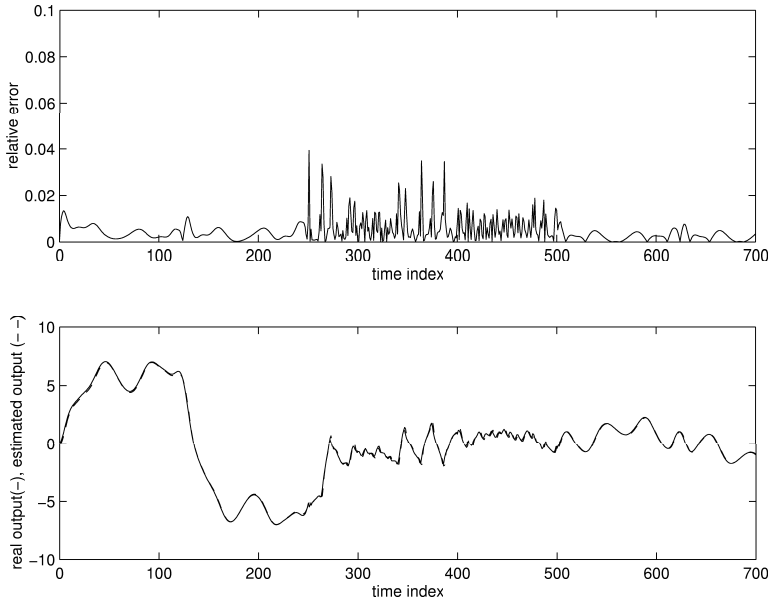

FIG. 3.9. Erreur d'identification relative, en utilisant le MFAD et l'AR , avec $u(k) = u_1(k)$ et la perturbation interne (Exemple 1).

Basé sur les pourcentages d'amélioration constatés pour l'usage de l'algorithme en ligne, par rapport aux résultats obtenus en utilisant un ajustement du MFAD hors ligne, on peut dire qu'un tel algorithme peut être mis en oeuvre en ligne, en espérant des bonnes performances au niveau des valeurs du RMSE.

3.3.3 Exemple 2

La fonction que l'on va, maintenant, chercher à identifier est décrite par l'équation aux différences suivante :

$$y(k+1) = g[y(k), y(k-1)] + u(k) \tag{3.38}$$

où

$$g[y(k), y(k-1)] = \frac{y(k)y(k-1)[y(k)+a(k))]}{1+y^2(k)+y^2(k-1)} \tag{3.39}$$

et $a(k) = 2.5 + 2.5sen(2\pi k/50)$ est un paramètre variant avec le temps. On suppose que la fonction $g[y(k), y(k-1)]$ est inconnue et doit être approchée par une fonction $g_e[y(k), y(k-1)]$ en utilisant un modèle flou. Dans ce cas-ci, la fonction $g_e[y(k), y(k-1)]$ dépend seulement de la variable de sortie y. Donc, les variables d'entrée du modèle flou sont $x_1(k) = y(k)$ et $x_2(k) = y(k-1)$.

Un ensemble de 250 couples de données d'apprentissage $\{(\underline{X}(t_j), y(t_j)),\ j = 1, ..., 250\}$ a été obtenu à partir d'une entrée PRBS appliquée au système, avec les niveaux $[-0.5, 0.5]$ et une longueur maximale de $L = 18$.

1. **Résultats expérimentaux en utilisant un MFA**

   Des expériences, comme celles développées dans l'exemple précédent, ont aussi été développées pour cet exemple. Plusieurs cycles d'apprentissage ont été parcourus jusqu'à la convergence du RMSE, avec plusieurs valeurs des taux d'apprentissage ρ_α, ρ_β, ρ_γ et du nombre de règles M. Les valeurs initiales des paramètres α_i^l, β_i^l et γ^l ont été choisies de façon aléatoire sur l'intervalle $[0, 1]$. Après la phase d'apprentissage, les MFA obtenus ont été utilisés en simulation avec, comme entrée, le signal de validation donnée par l'équation suivante :

$$u_2(k) = \begin{cases} \sin(2\pi k/25) & si \quad 1 < k < 100 \quad et \quad 501 < k < 700 \\ 3 + (0.5\sin(2\pi k/250) + 0.5\sin(2\pi k/25)) & si \quad 101 < k < 500 \end{cases} \tag{3.40}$$

   Le tableau 3.7 résume les résultats les plus importants dans les phases d'apprentissage et validation. Dans le cas $M = 30$, les taux d'apprentissage ont été calculés en utilisant l'équation (3.29).

   Sur la base de ces résultats, le MFA avec $M = 20$ affiche la meilleure performance. Dans la figure 3.10, on montre les courbes d'évolution du RMSE obtenu dans la phase d'apprentissage ; la performance de ce modèle flou est présentée dans la figure 3.11.

   En prenant $a(k) = 0$, $k > 400$, la performance du MFA avec cette perturbation interne est présentée dans la figure 3.12. Dans ce cas-ci, on obtient $RMSE = 0.7908$.

   Par référence à ces résultats, bien qu'on ait atteint la convergence de l'erreur dans la phase d'apprentissage en utilisant le MFA, la performance du modèle proposé avec $M = 20$, dans la phase de validation (avec et sans perturbation interne), est peu satisfaisante (voir la figure 3.11). Dans tout les cas, le RMSE est très grand et le MFA

TAB. 3.7
RMSE POUR LES DONNÉES D'APPRENTISSAGE ET VALIDATION, SELON LE NOMBRE DE RÈGLES M, EN UTILISANT UN MFA (EXEMPLE 2).

M	No. des paramètres	ρ_α	ρ_β	ρ_γ	RMSE (apprentissage)	RMSE (validation)
10	50	0.5	0.5	0.5	0.1533	0.6036
20	100	0.7	0.3	0.7	0.1444	0.5630
30	150	$\rho_\alpha(0) = 1$ $\eta = 0.9$	$\rho_\beta(0) = 1$ $\eta = 0.9$	$\rho_\gamma(0) = 1$ $\eta = 0.9$	0.1594	-
40	200	$\rho_\alpha(0) = 1$ $\eta = 0.9$	$\rho_\beta(0) = 1$ $\eta = 0.9$	$\rho_\gamma(0) = 1$ $\eta = 0.9$	0.1764	0.7388

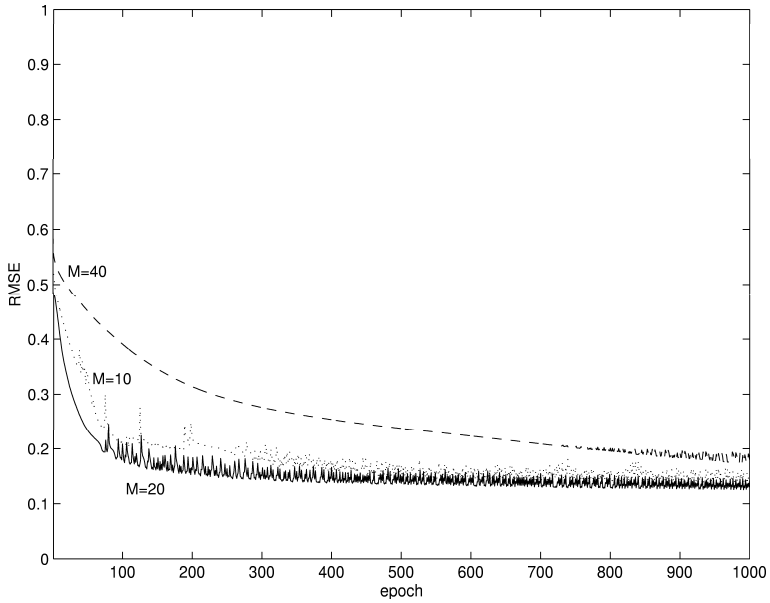

FIG. 3.10. Courbes d'évolution du RMSE pour la phase d'apprentissage, en utilisant un MFA (Exemple 2).

ne peut pas suivre la sortie réelle en présence de changements soudains sur le signal d'entrée ou de perturbation interne (voir la figure 3.12). On va maintenant essayer d'améliorer ces résultats en utilisant un MFAD.

2. **Résultats expérimentaux en utilisant le MFAD et l'apprentissage hors ligne**

   Les mêmes données d'apprentissage utilisées dans les expériences précédentes ont été reprises dans la phase d'apprentissage du MFAD. Les valeurs initiales des paramètres u^l, v_i^l et w_i^l ont été choisies aléatoirement sur l'intervalle $[0.9, 1.1]$, et $\epsilon = 0.1$. Les cycles d'apprentissage ont été limités à 1000 et 2000 itérations et les taux d'apprentissage ont été calculés selon l'équation (3.29) avec $\rho_u(0) = \rho_v(0) = \rho_w(0) = 1$ et $\eta = 0.8$.

   Sur le tableau 3.8, on montre les résultats les plus importants dans les phases d'apprentissage et validation, en utilisant $u(k) = u_2(k)$ comme signal d'entrée. La figure 3.13 montre les courbes d'évolution du RMSE jusqu'à 2000 itérations.

   Selon les résultats présentés sur le tableau 3.8, le RMSE dans la phase de validation n'est pas amélioré, même si on accomplit un grand nombre d'itérations dans la phase

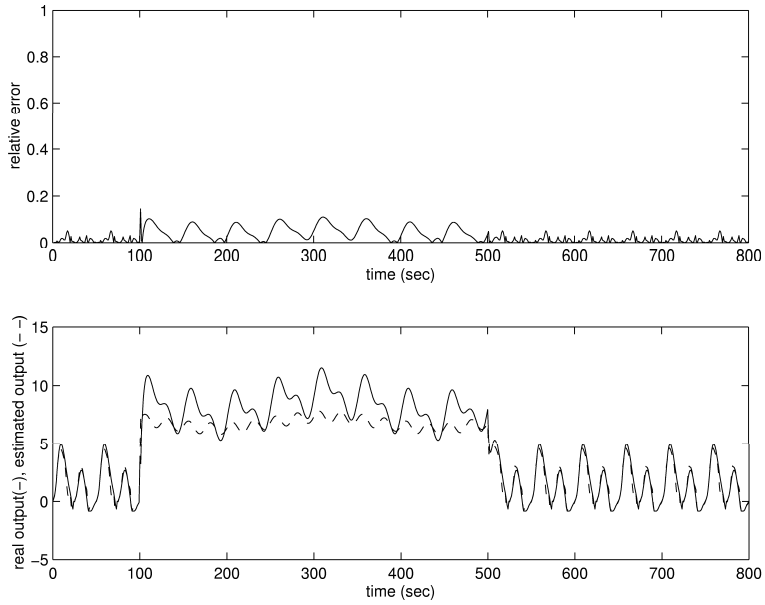

FIG. 3.11. Erreur d'identification relative, en utilisant le MFA, avec $u(k) = u_2(k)$ (Exemple 2).

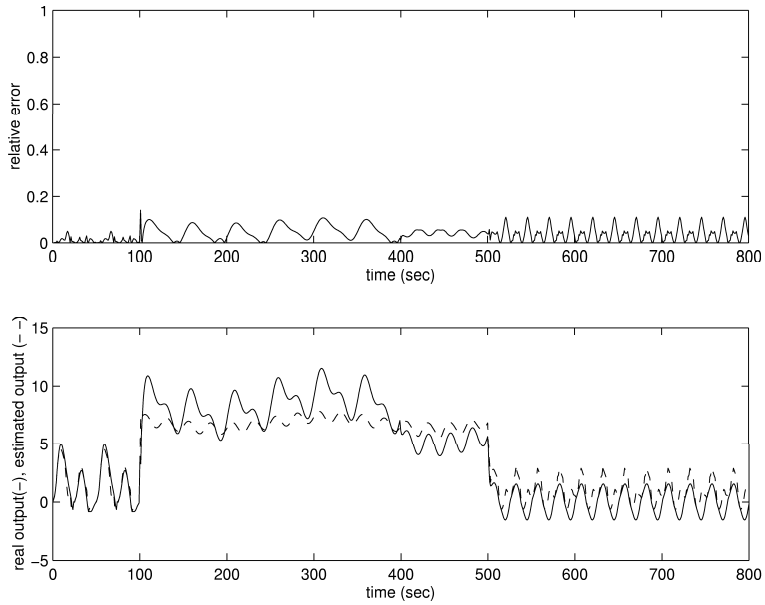

FIG. 3.12. Erreur d'identification relative, en utilisant le MFA, avec $u(k) = u_2(k)$ et $a(k) = 0$, $k > 400$ (Exemple 2).

d'apprentissage, ou si la convergence de l'erreur n'est pas atteinte. Ce fait permet de dire que la capacité de généralisation du MFAD peut être atteinte avant la convergence de l'erreur dans la phase d'apprentissage. Les MFAD, avec $M = 8$ ou $M = 20$, ont une performance satisfaisante dans la phase d'apprentissage et, de plus, tous les résultats expérimentaux sont meilleurs que ceux obtenus en utilisant le MFA déjà présenté.

La figure 3.14 montre l'erreur d'identification relative pour les données de validation, en utilisant le MFAD, avec $M = 8$, et les paramètres obtenus au bout de 1000 itérations. La performance du MFAD, pour une perturbation interne, en prenant $a(k) = 0$, $k > 400$, est présentée dans la figure 3.15. Dans ce cas-ci, on obtient $RMSE = 0.0952$.

TAB. 3.8
RMSE POUR LES DONNÉES D'APPRENTISSAGE ET VALIDATION, SELON LE NOMBRE DE RÈGLES M, EN UTILISANT UN MFAD (EXEMPLE 2).

M	No. des paramètres	Itérations	RMSE (phase d'apprentissage)	RMSE (phase de validation)
8	40	1000	0.2130	0.1476
8	40	2000	0.2008	0.1824
10	50	1000	0.2051	0.1618
10	50	2000	0.1912	0.1933
20	100	1000	0.2325	0.1590
20	100	2000	0.2045	0.2222
40	200	1000	0.2545	0.2414
40	200	2000	0.2471	0.2199

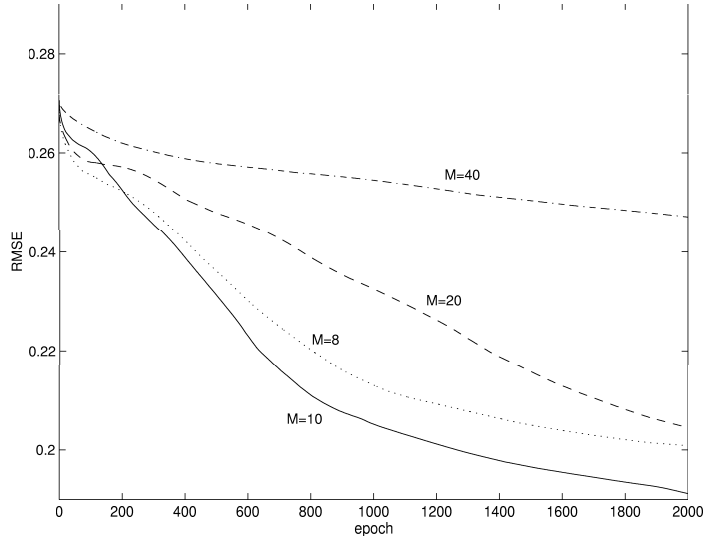

FIG. 3.13. Courbes d'évolution du RMSE pour la phase d'apprentissage, en utilisant un MFAD (Exemple 2).

Dans les deux figures 3.14 et 3.15, on observe que l'erreur d'identification relative est acceptable. Il existe des sauts des valeurs de l'erreur juste lors du changement de la valeur d'entrée ($k = 100$ et $k = 500$ dans la figure 3.14) ou de la présence de la perturbation interne ($k = 400$ dans la figure 3.15). Néanmoins, après ces sauts, l'erreur relative revient à des valeurs convenables.

La comparaison entre les tableaux 3.7 et 3.8 permet d'affirmer que la performance du MFAD, dans les deux phases d'apprentissage et validation, est meilleure que celle obtenue en utilisant le MFA, même si la convergence de l'erreur d'apprentissage n'est pas atteinte.

3. **Résultats expérimentaux en utilisant le MFAD et l'apprentissage en ligne**
Comme pour l'exemple 1, on valide la performance de l'algorithme en ligne donnée par l'équation (3.25) avec l'entrée $u(k) = u_2(k)$ décrite par l'équation (3.40) et en présence

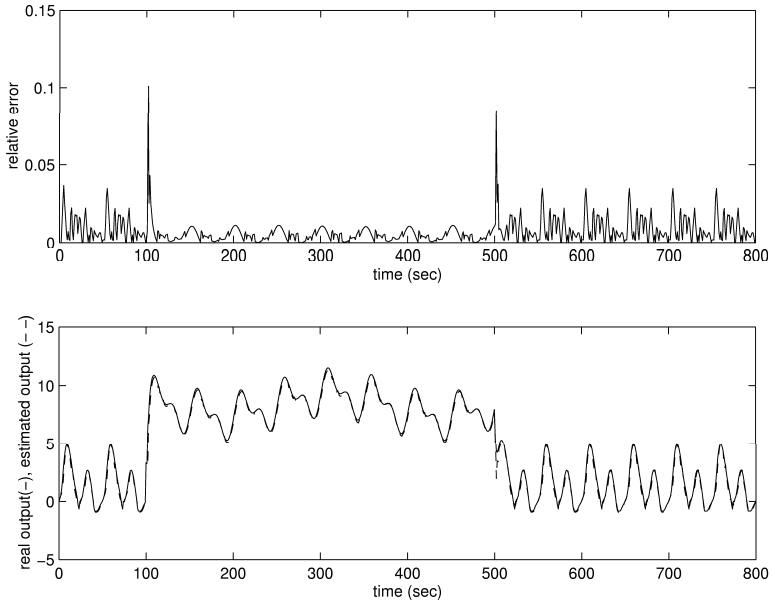

FIG. 3.14. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_2(k)$ (Exemple 2).

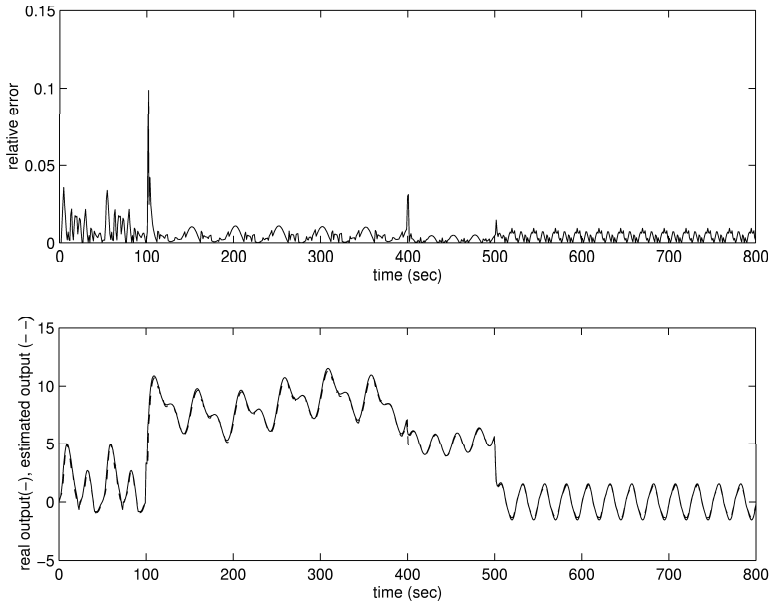

FIG. 3.15. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_2(k)$ et $a(k) = 0$, $k > 400$ (Exemple 2).

de la perturbation interne qui consiste à prendre $a(k) = 0$, $k > 400$. Dans ce cas, on prend $\lambda = \mu = 0.9$ dans la loi d'ajustement en ligne. Le taux d'apprentissage ρ est mis à jour par l'équation (3.29), avec $\eta = 0.9$, à chaque itération du temps $t = k$.

Les valeurs initiales des paramètres ont été prises sur l'intervalle $[0.8, 1.2]$, un peu plus loin des valeurs retenues pour l'apprentissage hors ligne. L'objectif est de comparer la performance de l'algorithme en ligne en partant de valeurs initiales des paramètres différentes autour de 1 (voir remarque 2). Cette performance est comparée à celle obtenue dans le cas hors ligne, sur la base de l'amélioration du RMSE. Le tableau ci-dessous donne cette comparaison. L'espace blanc indique qu'aucune simulation hors

ligne n'a été faite pour ce nombre de règles.

TAB. 3.9
RMSE EN UTILISANT $u(k) = u_2(k)$ ET L'APPRENTISSAGE EN LIGNE. (EXEMPLE 2).

M	RMSE	% d'amelioration	RMSE (perturbation interne)
6	0.1110		0.0838
8	0.1285	12.94	0.1084
10	0.1327	17.99	0.1044
15	0.1069		0.0860
20	0.1398	12.07	0.1056

Le tableau montre qu'une bonne performance du MFAD est atteinte avec $M = 15$, et, pour tous les nombres de règles, la performance du MFAD est améliorée avec l'algorithme en ligne. On observe, aussi, qu'aucune valeur de RMSE en utilisant l'apprentissage hors ligne (voir le tableau 3.8) n'est proche des valeurs du RMSE obtenues avec l'algorithme en ligne proposé.

Bien que la meilleure valeur du RMSE soit obtenue avec $M = 15$, on considère le MFAD, avec $M = 6$, pour montrer que la performance pour ce nombre de règles est toujours satisfaisante et l'effort de calcul plus petit pour une mise en oeuvre en ligne, puisque le nombre de règles est réduit. La figure 3.16 montre l'erreur relative d'identification en utilisant le MFAD, avec $M = 6$. Dans la figure 3.17, on montre une autre validation de la performance du MFAD en prenant la perturbation interne déjà mentionnée après $k = 400$. Dans ce cas-ci, on obtient une amélioration du 11.97% par rapport au RMSE obtenu en utilisant l'algorithme hors ligne, avec $M = 8$ (voir la partie (2) de cet exemple).

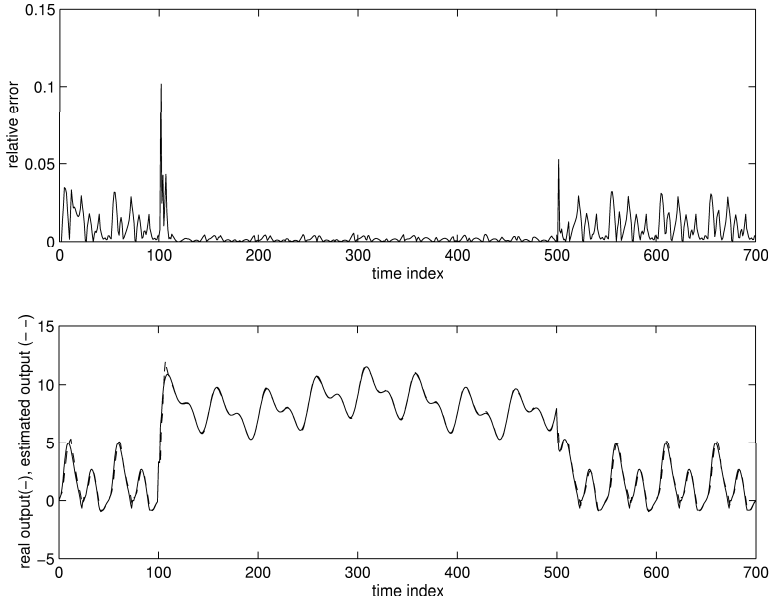

FIG. 3.16. Erreur d'identification relative, en utilisant le MFAD et l'AR, avec $u(k) = u_2(k)$ (Exemple 2).

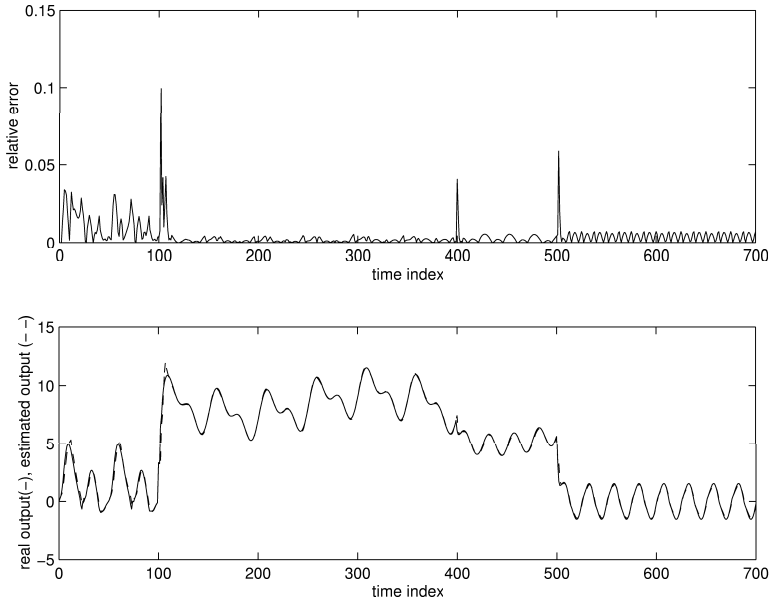

FIG. 3.17. Erreur d'identification relative, en utilisant le MFAD et l'AR , avec $u(k) = u_2(k)$ et la perturbation interne (Exemple 2).

À nouveau, comme dans la partie 2 de cet exemple, on observe, dans les figures 3.16 et 3.17, des sauts dans la valeur de l'erreur d'identification juste lors du changement du signal d'entrée et de la perturbation interne, mais la valeur de l'erreur reste acceptable. Sur la base de ces améliorations du RMSE obtenues par l'utilisation de l'algorithme en ligne, par rapport aux résultats acquis en utilisant un ajustement du MFAD hors ligne, on peut dire que cet algorithme permet une bonne identification en ligne de modèles non linéaires.

3.3.4 Exemple 3

Soit la fonction aux différences non linéaire, variant avec le temps, suivante :

$$y(k+1) = g[y(k), y(k-1), y(k-2), u(k), u(k-1)] \tag{3.41}$$

où

$$g[.] = \frac{y(k)y(k-1)y(k-2)u(k-1)(y(k-2)-1)+u(k)}{a(k)+y(k-2)^2+y(k-1)^2} \tag{3.42}$$

avec $a(k) = 1 + 0.1\sin(2\pi k/50)$.

La fonction estimée $y_e(k+1)$ est définie comme :

$$y_e(k+1) = g_e[y(k), y(k-1), y(k-2), u(k), u(k-1)] \tag{3.43}$$

où la fonction $g_e[.]$ est donnée par un modèle flou. Dans cet exemple, la fonction inconnue $g_e[.]$ dépend des variables d'entrée et sortie. En conséquence, les variables d'entrée du modèle flou sont $x_1(k) = y(k)$, $x_2(k) = y(k-1)$, $x_3(k) = y(k-2)$, $x_4(k) = u(k)$ et $x_5(k) = u(k-1)$.

Pour ce cas, deux ensembles de données d'apprentissage ont été obtenus en utilisant comme entrée du système un signal PRBS avec les niveaux $[-0.5, 0.5]$ et une longueur maximale de $L = 18$. Le premier ensemble contient 500 couples de données d'apprentissage et le deuxième ensemble 250 couples de données.

1. **Résultats expérimentaux en utilisant un MFA**

   Les mêmes expériences que précédemment ont été reprisés en utilisant le premier ensemble de données d'apprentissage et les cycles d'apprentissage ont été arrêtés à 3000 itérations. Les taux d'apprentissage ont été définis dynamiquement, en utilisant l'équation (3.29), avec $\rho_\alpha(0) = \rho_\beta(0) = \rho_\gamma(0) = 1$ et $\eta = 0.9$, pour $M = 20$ et $M = 30$, et des valeurs constantes sur $\rho_\alpha = \rho_\beta = \rho_\gamma = 0.5$, pour $M = 10$. Les valeurs initiales des paramètres α_i^l, β_i^l et γ^l du modèle flou ont été choisies aléatoirement sur l'intervalle $[0, 1]$. Après la phase d'apprentissage, les modèles flous proposés ont été évalués en utilisant le signal de validation donné par l'équation suivante :

$$u_3(t) = \begin{cases} \sin(2\pi k/250) & si \quad 1 < k < 500 \quad et \quad 801 < k < 1000 \\ 1.5 + (0.8\sin(2\pi k/250) + 0.2\sin(2\pi k/25)) & si \quad 501 < k < 800 \end{cases} \tag{3.44}$$

   Sur le tableau 3.10, on résume les résultats les plus importants obtenus dans les phases d'apprentissage et validation.

Tab. 3.10
RMSE pour les données d'apprentissage et validation, selon le nombre de règles M, en utilisant un MFA (Exemple 3).

M	No. des paramètres	RMSE (phase d'apprentissage)	RMSE (phase de validation)
10	110	0.0886	0.1709
20	220	0.0772	0.1443
40	440	0.0734	0.0655

   En prenant en compte ces résultats, le MFA avec $M = 40$ a la meilleure performance. Dans la figure 3.18, on montre les courbes du RMSE lors de la phase d'apprentissage et la figure 3.19 présente la performance de ce modèle flou, avec l'entrée de validation (3.44).

   La performance du MFA, avec une perturbation interne, a été évaluée en prenant le paramètre variant avec le temps $a(k) = 5$ pour $k > 450$. Dans la figure 3.20, on observe l'erreur d'identification relative. Dans ce cas-ci, on obtient $RMSE = 0.0969$ et le MFA a failli en précision de prédiction, après $k = 450$.

   Le deuxième ensemble de données d'apprentissage, qui contient 250 couples d'entrée-sortie seulement, a été pris pour réaliser une nouvelle phase d'apprentissage du MFA. Dans ce cas, avec certaines valeurs des taux d'apprentissage, le RMSE pour $M = 30$, est au-dessus de 0.20. La valeur de convergence du RMSE, obtenue avec 500 couples des données d'apprentissage (cas précedent), n'a été atteinte qu'après 10000 cycles d'apprentissage, et 250 couples des données. En plus, le RMSE est plus grand dans la

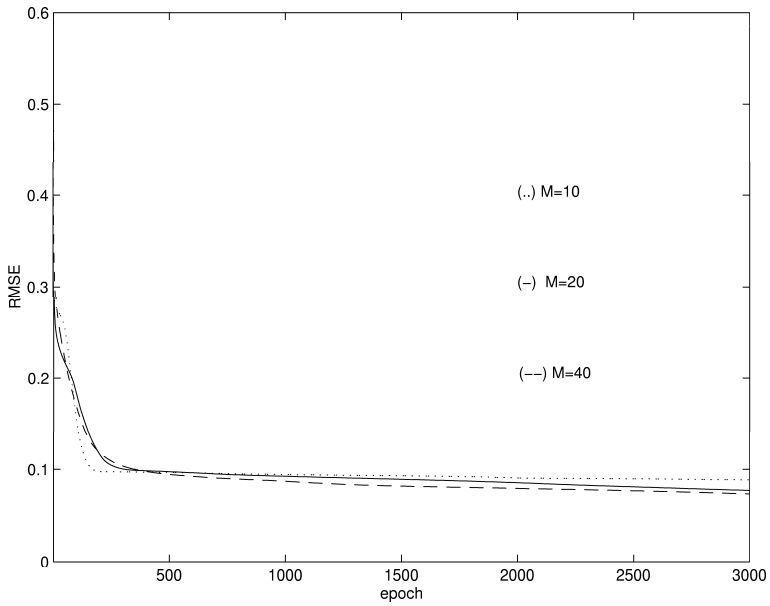

FIG. 3.18. Courbes d'évolution du RMSE pour la phase d'apprentissage, en utilisant un MFA (Exemple 3).

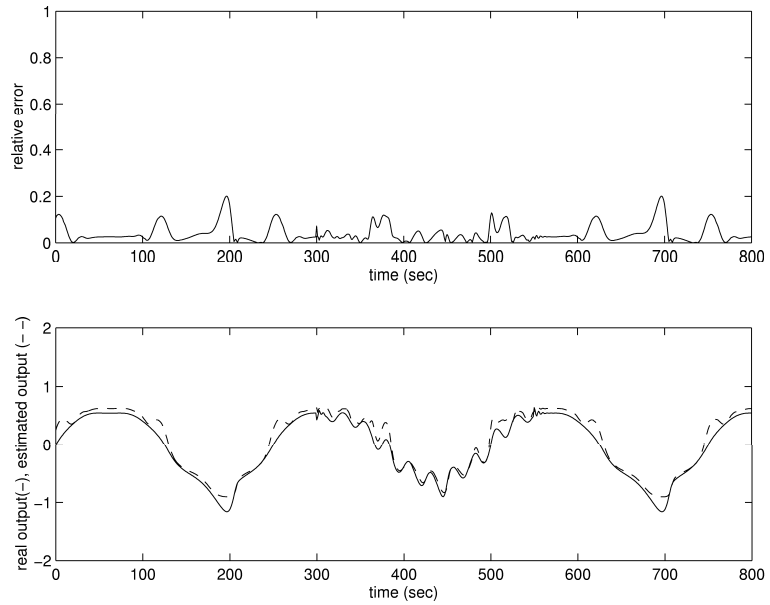

FIG. 3.19. Erreur d'identification relative, en utilisant le MFA, avec $u(k) = u_3(k)$ (Exemple 3).

phase de validation par rapport au cas précédent (le RMSE est au-dessus de 0.25, en utilisant 250 couples de données et 10000 itérations).

Sur la base de ces résultats, on peut conclure que la performance du MFA, avec $M = 40$, est adéquate mais qu'on a besoin d'une quantité certaine de données d'apprentissage ; sinon, on peut arriver à des valeurs considérables du RMSE. Dans les expériences suivantes, on montre la capacité du MFAD à surmonter les difficultés rencontrées avec le MFA classique.

2. **Résultats expérimentaux en utilisant le MFAD et l'apprentissage hors ligne**
   Pour ces expériences, on a pris le deuxième ensemble de données d'apprentissage avec

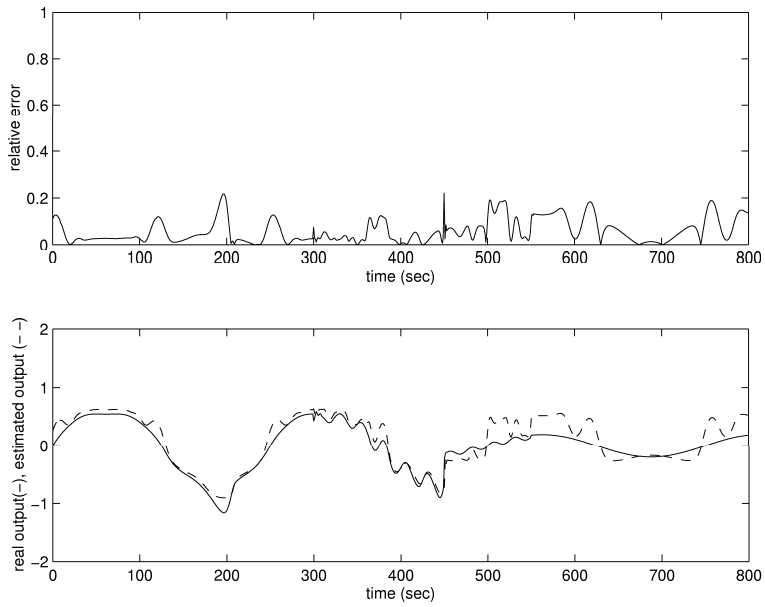

FIG. 3.20. Erreur d'identification relative, en utilisant le MFA, avec $u(k) = u_3(k)$ et $a(k) = 5$, $k > 450$ (Exemple 3).

250 couples d'entrée-sortie. Pour la phase d'apprentissage, les valeurs initiales des paramètres u^l, v_i^l et w_i^l ont été choisies toujours aléatoirement sur l'intervalle $[0, 1]$, avec $\epsilon = 0.1$.

Le tableau 3.11 montre les résultats les plus importants, après une phase d'apprentissage très intensive. Les cycles d'apprentissage ont été arrêtés à 500 et 1000 itérations et les taux d'apprentissage ont été pris à $\rho_u = \rho_v = 0.7$ et $\rho_w = 0.3$, pour $M = 8, 10, 15, 20$. On présente, aussi, les résultats obtenus dans la phase de validation, avec l'entrée $u(k) = u_3(k)$. La figure 3.21 montre les courbes du RMSE, lors de la phase d'apprentissage.

TAB. 3.11
RMSE POUR LES DONNÉES D'APPRENTISSAGE ET VALIDATION, SELON LE NOMBRE DE RÈGLES M, EN UTILISANT UN MFAD (EXEMPLE 3).

M	No. des paramètres	No. des itérations	RMSE (phase d'apprentissage)	RMSE (phase de validation)
8	88	500	0.1401	0.0156
8	88	1000	0.1318	0.0378
10	110	500	0.1678	0.0837
10	110	1000	0.1476	0.1101
15	165	500	0.1566	0.0308
15	165	1000	0.1393	0.0566
20	220	500	0.1607	0.1334
20	220	1000	0.1273	0.1209

Selon les résultats présentés sur le tableau 3.11, le MFAD, avec $M = 8$, a une performance adéquate. La figure 3.22 montre l'erreur d'identification relative en utilisant l'entrée $u(k) = u_3(k)$, avec $M = 8$. Dans la figure 3.23, on observe la performance

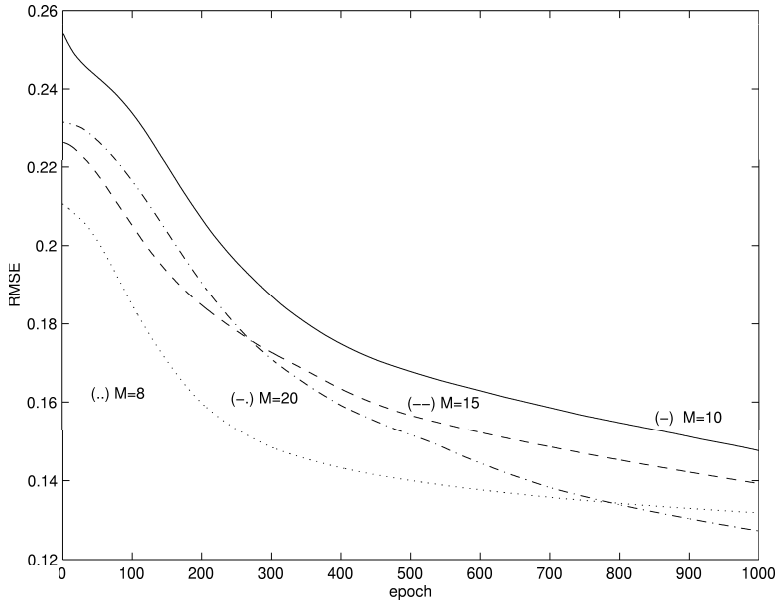

FIG. 3.21. Courbes d'évolution du RMSE pour la phase d'apprentissage, en utilisant un MFAD (Exemple 3).

du MFAD, dans le cas d'une perturbation interne qui consiste à fixer $a(k) = 5$ pour $k > 450$. Dans ce cas, $RMSE = 0.0130$ et le MFAD a une performance adéquate, même avec le système perturbé.

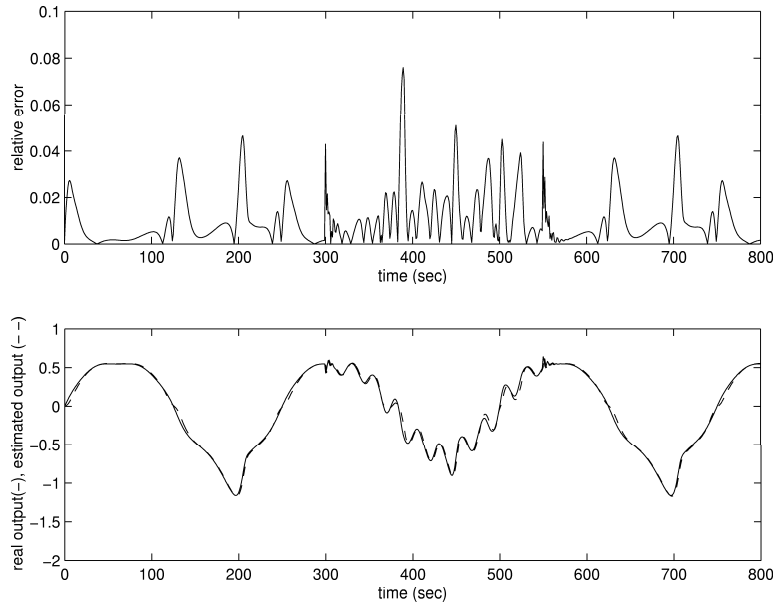

FIG. 3.22. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_3(k)$ (Exemple 3).

La comparaison des tableaux 3.10 et 3.11 permet d'observer que le RMSE dans la phase d'apprentissage, en utilisant le MFA, est améliorée par le MFAD proposé. De plus, selon les valeurs du RMSE obtenues, la capacité de généralisation du MFAD, dans la phase de validation, est meilleure que celle obtenue en utilisant le MFA, qui plus est avec un nombre d'itérations moindre. On peut remarquer, aussi, que ce résultat est

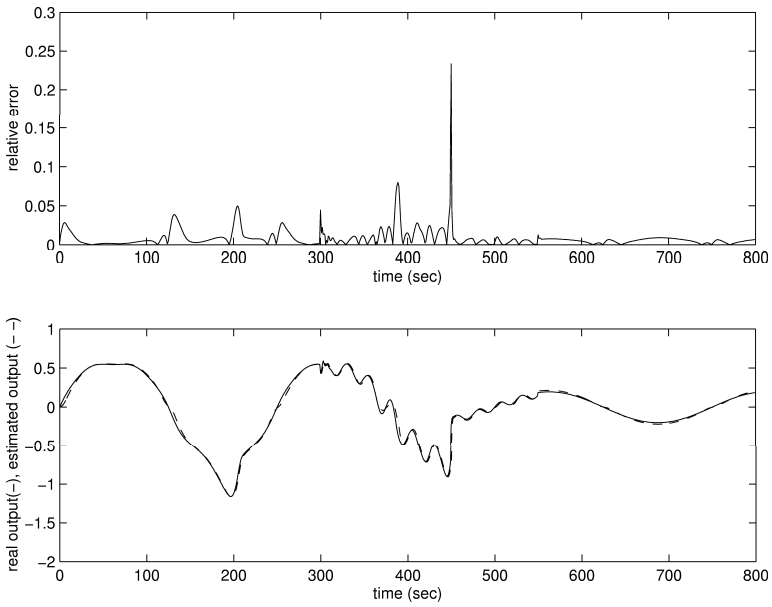

FIG. 3.23. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_3(k)$ et $a(k) = 5$, $k > 450$ (Exemple 3).

obtenu à partir d'un ensemble réduit de données d'apprentissage et un faible nombre de règles, par rapport au cas du MFA.

3. **Résultats expérimentaux en utilisant le MFAD et l'apprentissage en ligne**

   Dans ce cas-ci, on utilise l'algorithme d'apprentissage en ligne donné par l'équation (3.25), avec $\lambda = \mu = 0.9$ et le taux d'apprentissage ρ mis à jour par l'équation (3.29), avec $\eta = 0.01$ à chaque itération du temps $t = k$.

   La performance de l'algorithme en ligne est mise en exergue en utilisant l'entrée $u(k) = u_3(k)$ décrite par l'équation (3.44) et en présence de la perturbation interne déjà mentionée.

   Les valeurs initiales des paramètres ont été choisies sur l'intervalle $[0.5, 1.5]$, un peu différentes des valeurs retenues pour l'apprentissage hors ligne. L'objectif est de comparer la performance de l'algorithme en ligne en partant d'autres valeurs initiales des paramètres. Ci-dessous, un tableau comparatif, pareil à ceux établis pour les exemples 1 et 2, est presenté. Les pourcentages négatifs indiquent que la performance du MFAD n'est pas améliorée par l'usage de l'algorithme d'apprentissage en ligne.

TAB. 3.12
RMSE EN UTILISANT $u(k) = u_3(k)$ ET L'APPRENTISSAGE EN LIGNE. (EXEMPLE 3).

M	RMSE	% d'amelioration	RMSE (perturbation interne)
8	0.0323	-51.7	0.0247
10	0.0339	59.50	0.0276
15	0.0339	-9.14	0.0340
20	0.0205	84.63	0.0134

Le tableau montre que la meilleure performance obtenue par l'algorithme d'apprentissage en ligne du MFAD est atteinte avec $M = 20$ mais, cette valeur du RMSE est

obtenue avec un nombre de règles plus faible en utilisant le MFAD et un apprentissage hors ligne (voir les résultats hors ligne sur le tableau 3.11). En conséquence, on observe sur le tableau 3.12 des valeurs négatives de pourcentage d'amélioration pour deux valeurs du nombre de règles. Ces résultats peuvent indiquer qu'on a besoin, dans certains cas, d'un nombre de règles plus grand dans le cas de l'apprentissage en ligne, pour obtenir des résultats semblables à ceux obtenus en utilisant l'apprentissage hors ligne. D'un autre côté, on souligne qu'il est possible d'obtenir des résultats peu satisfaisants au début d'un apprentissage en ligne, ce que peut avoir une incidence sur la valeur du RMSE.

Les figures suivantes, montrant l'erreur relative, peuvent être plus explicatives sur ces résultats :

La figure 3.24 montre l'erreur relative d'identification en utilisant le MFAD, avec $M = 20$, et la figure 3.25 montre la performance du MFAD avec la perturbation interne appliquée après $k = 450$. Dans ce cas, le RMSE obtenu par l'algorithme d'apprentissage en ligne reste assez proche de celui obtenu dans le cas hors ligne, avec $M = 8$. Par comparaison entre ces figures et celles de l'apprentissage hors ligne (figures 3.22 et 3.23), on observe qu'il existe une erreur importante dans k=200,450,700 dans le cas de l'apprentissage en ligne. Dans la section suivante, on va étudier la sensibilité de l'algorithme en ligne vis-à-vis des conditions initiales des paramètres et on va essayer d'améliorer cet erreur via le changement des conditions initiales.

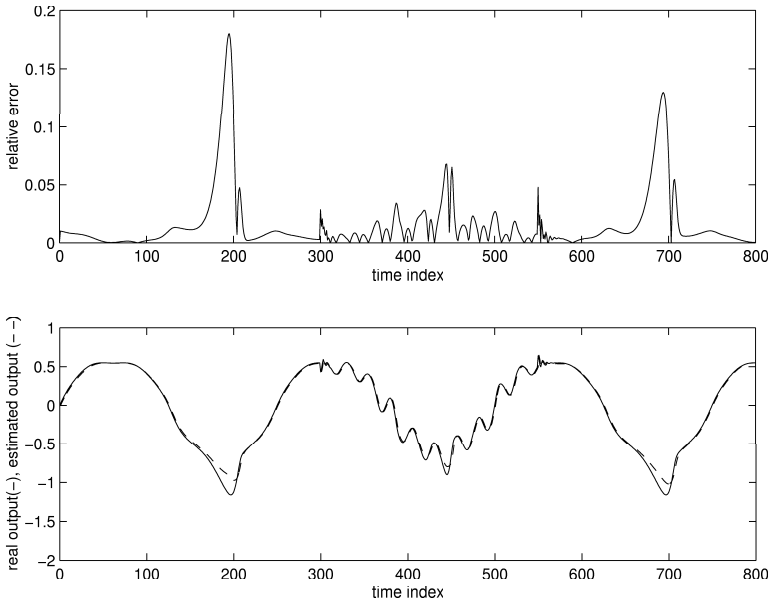

FIG. 3.24. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_3(k)$ (Exemple 3).

Un bilan sur les résultats obtenus dans cet exemple permet de dire que la performance du MFAD, en utilisant l'algorithme en ligne, est bonne malgré les résultats obtenus au niveau des valeurs du RMSE qui ne sont pas toujours améliorées par rapport aux résultats obtenus en utilisant un MFAD et un ajustement des paramètres hors ligne. On a déjà mentionné, que la valeur importante du RMSE peut être une conséquence des erreurs locales significatifs mais, globalement, les résultats obtenus, avec un apprentissage en ligne, sont tout à fait satisfaisants.

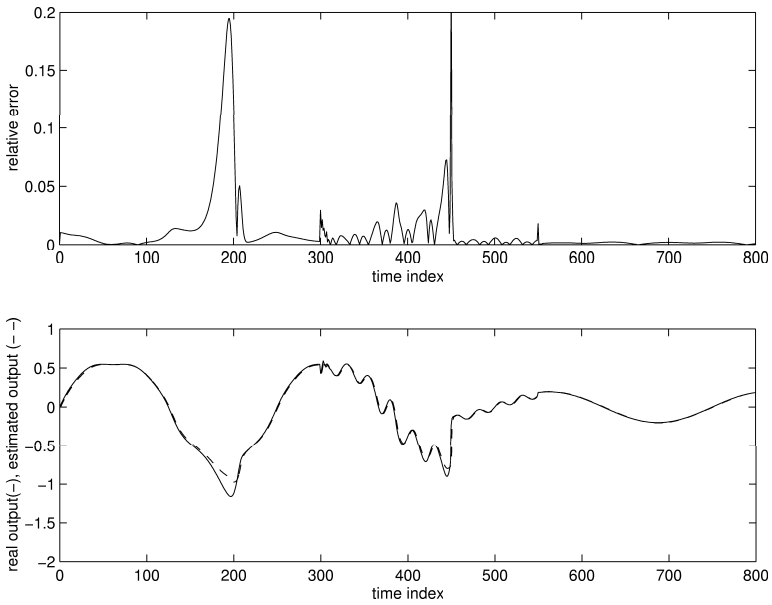

FIG. 3.25. Erreur d'identification relative, en utilisant le MFAD, avec $u(k) = u_3(k)$ et la perturbation interne (Exemple 3).

3.4 Étude de sensibilité de l'algorithme d'apprentissage en ligne

En vue de l'utilisation de l'algorithme d'apprentissage reposant sur l'AR proposé dans cette thèse, des indicateurs numériques, obtenus à partir des simulations, sont fournis pour la mesure de la sensibilité de cet algorithme vis-à-vis des paramètres μ, λ, et du taux d'apprentissage ρ.

De plus, il faut étudier la sensibilité de l'algorithme vis-à-vis du nombre de règles M et des valeurs initiales des paramètres u^l, v_i^l et w_i^l du MFAD. Pour cette analyse expérimentale, on prend seulement les résultats obtenus pendant le traitement de l'exemple 3.

Enfin, des indicateurs graphiques sur le niveau de complexité des calculs sont fournis afin d'estimer les temps de calcul.

1. **Sensibilité vis-à-vis des paramètres μ, λ, ρ**

   Il a été déjà mentionné que le paramètre λ pondère l'influence des erreurs de prédiction passées et le paramètre μ détermine la pondération au temps t de la valeur des futurs renforcements. Il est sugeré, donc, de prendre des valeurs autour de 1, même pour le choix de la valeur de η, dans l'équation (3.29), pour la variation dynamique du taux d'apprentissage ρ.

   Ci-dessous, on présente un tableau avec les valeurs du RMSE pour différentes valeurs des paramètres et différents nombres de règles, avec l'entrées $u(k) = u_3(k)$, pour l'exemple 3. Un taux de variation, par rapport à la meilleure valeur du RMSE obtenue pour chaque valeur de M, est calculé pour mesurer la sensibilité vis-à-vis du nombre de règles.

TAB. 3.13
SENSIBILITÉ VIS-À-VIS DES PARAMÈTRES μ, λ, η (EXEMPLE 3).

M	η	μ	λ	RMSE	taux de variation
8	0.01	0.9	0.9	0.0323	1
8	0.01	0.5	0.5	0.0384	1.19
8	0.01	0.1	0.1	0.0411	1.27
8	0.9	0.9	0.9	0.0500	1.54
10	0.01	0.9	0.9	0.0339	1
10	0.01	0.5	0.5	0.0366	1.08
10	0.01	0.1	0.1	0.0378	1.12
10	0.9	0.9	0.9	0.0414	1.22
15	0.01	0.9	0.9	0.0339	1
15	0.01	0.5	0.5	0.0365	1.08
15	0.01	0.1	0.1	0.0373	1.10
15	0.9	0.9	0.9	0.0392	1.16
20	0.01	0.9	0.9	0.0205	1
20	0.01	0.5	0.5	0.0219	1.07
20	0.01	0.1	0.1	0.0223	1.09
20	0.9	0.9	0.9	0.0225	1.10

Partant de ces valeurs des taux de variation montrées dans la table 3.13, on peut remarquer que :

- La valeur du taux de variation reste presque identique dans tous les cas. Il n'y a pas de variations considérables pour des valeurs différentes des paramètres μ et λ, lorsque la valeur de η est fixée.
- Par contre, en prenant comme valeur de référence la meilleure valeur du RMSE obtenue pour chaque nombre de règles, la variation de cette valeur est plus sensible au changement du taux d'apprentissage ρ, à travers la variation du paramètre η.

Remarque 8 *Ces résultats rejoignent les idées générales sur les valeurs adéquates de μ et λ suggérées dans les références bibliographiques. Par contre, les résultats obtenus sur la variation de la valeur du taux d'apprentissage ρ, à travers la valeur de η, ne sont pas décisifs au moment de faire une conclusion sur la robustesse de l'algorithme en ligne.*

2. **Sensibilité vis-à-vis du nombre de règles** M

A partir des résultats présentés dans le tableau 3.13 ci-dessus, on remarque que les valeurs du RMSE, pour différentes valeurs de μ, λ et η, restent très proches jusqu'à $M = 15$ et le RMSE est amélioré avec $M = 20$. Ce résultat permet de penser que pour chaque cas particulier, il existe un nombre de règles "optimal" pour la mise en oeuvre de l'algorithme d'apprentissage en ligne.

3. **Sensibilité vis-à-vis des conditions initiales du MFAD**

En reprenant les résultats obtenus dans la partie (3) de l'exemple 3 (voir le tableau 3.12), on a développé d'autres expériences afin de voir la performance de l'algorithme d'apprentissage en ligne, en partant d'autres conditions initiales, pour le même temps de simulation. De plus, d'autres expériences ont été developpées en utilisant un autre signal d'entrée $u(k) = u_4(k)$, donné par l'équation (3.45), et en introduisant une modification sur le terme variant avec le temps $a(k) = a_1(k) = 1 + 0.3\sin(2\pi k/10)$ dans la fonction (3.41).

$$u_4(t) = \begin{cases} \sin(2\pi k/250) & si \quad 1 < k < 500 \quad et \quad 801 < k < 1000 \\ 1.5 + (0.5\sin(2\pi k/250) + 0.5\sin(2\pi k/25)) & si \quad 501 < k < 800 \end{cases} \tag{3.45}$$

Le tableaux 3.14 ci-dessous montre des résultats comparatifs entre les performances du MFAD, en partant de conditions initiales différentes, en visualisant le pourcentage de variation semblable à celui défini par l'équation (3.35). Ce tableau reprend quelques résultats déjà montrés dans le tableau 3.12 et joints aux résultats des nouvelles expériences.

Tab. 3.14
Sensibilité vis-à-vis des conditions initiales (CI) (Exemple 3).

M	Entrée	RMSE (CI ∈ [0.5 1.5])	RMSE (CI ∈ [0 1])	% de variation
8	u_3	0.0323	0.0392	17.60
8	u_4	0.0475	0.0499	4.81
20	u_3	0.0205	0.0492	58.33
20	u_4	0.0361	0.0587	38.5

A partir de ce tableau, on peut voir que la performance du MFAD peut se dégrader, sur le même temps de simulation, selon l'intervalle où ont été placées les conditions initiales ; les figures suivantes (avec $M = 20$) expliquent cette dégradation.

En effet, on peut remarquer, sur la figure 3.26, qu'il existe au début un erreur importante, mais la convergence de l'erreur d'identification est atteinte après $k = 200$. D'un autre côté, bien que, globalement, la valeur du RMSE lorsque CI ∈ [0.5 1.5] soit adéquate et qu'il n'existe pas d'erreur trop grande au début de la simulation (figure 3.27), les erreurs pour $k = 200$ et $k = 700$ sont significatives. La comparaison entre les figures 3.26 et 3.27 montre que, apparemment, les conditions initiales sur l'intervalle [0.5 1.5] sont très proches des valeurs optimales et l'atout de l'algorithme d'apprentissage en ligne n'est pas utilisé.

Dans la figure 3.28, on montre la performance du MFAD en utilisant l'algorithme d'apprentissage en ligne, avec le signal d'entrée $u(k) = u_4(t)$ et CI ∈ [0 1]. On peut voir une bonne performance du MFAD après $k = 200$.

A partir de ces résultats, on pense qu'il est possible de trouver de meilleures performances en partant de certaines valeurs des conditions initiales, même si la performance globale, mesurée avec la valeur du RMSE, peut être améliorée en partant d'autres intervalles pour les conditions initiales.

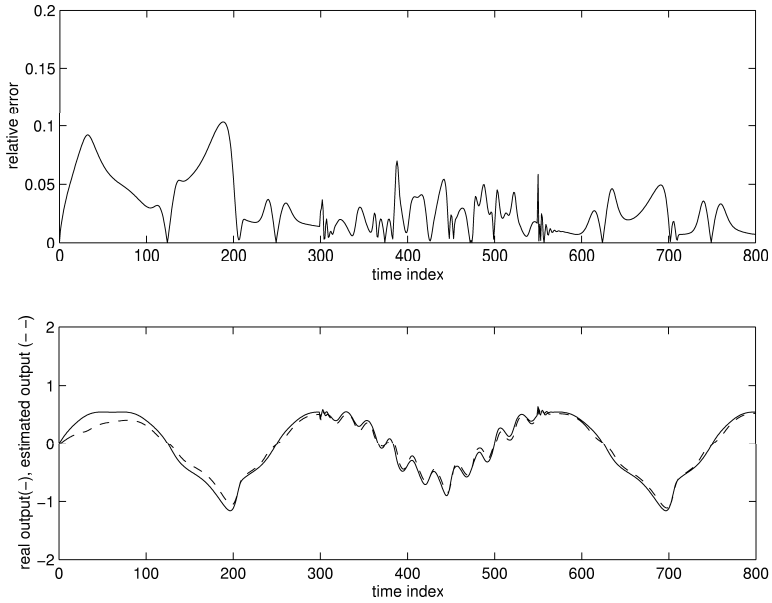

FIG. 3.26. Erreur d'identification relative, en utilisant le MFAD, avec CI $\in$ [0 1] (Exemple 3).

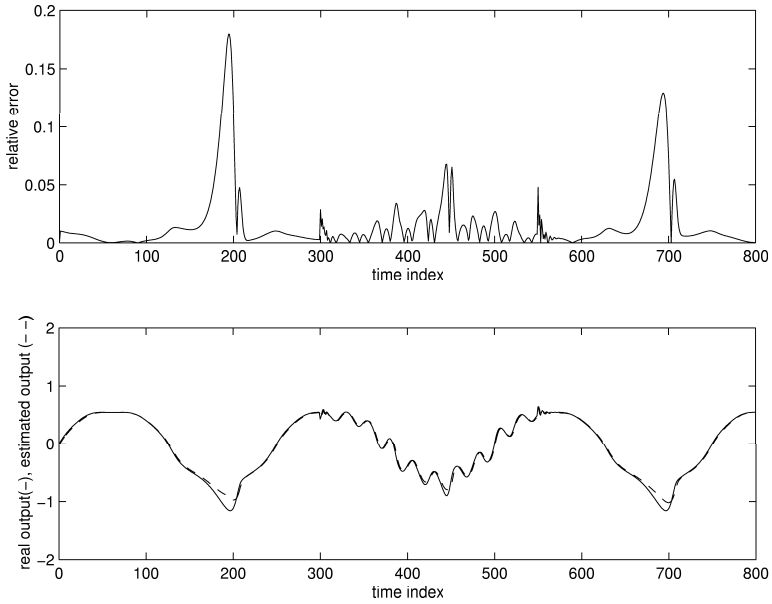

FIG. 3.27. Erreur d'identification relative, en utilisant le MFAD, avec CI $\in$ [0.5 1.5] (Exemple 3).

Remarque 9 *Les résultats obtenus sur la vitesse de convergence du RMSE, avec l'algorithme en ligne, ne permettent pas de conclure sur la sensibilité des résultats vis-à-vis des conditions initiales.*

3.4.1 Indicateurs graphiques sur le niveau de complexité des calculs

La complexité des calculs d'un mécanisme d'inférence floue est mesurée par la quantité de FLIPS (Fuzzy Logical Inference Per Second, en anglais) utilisées pendant l'application,

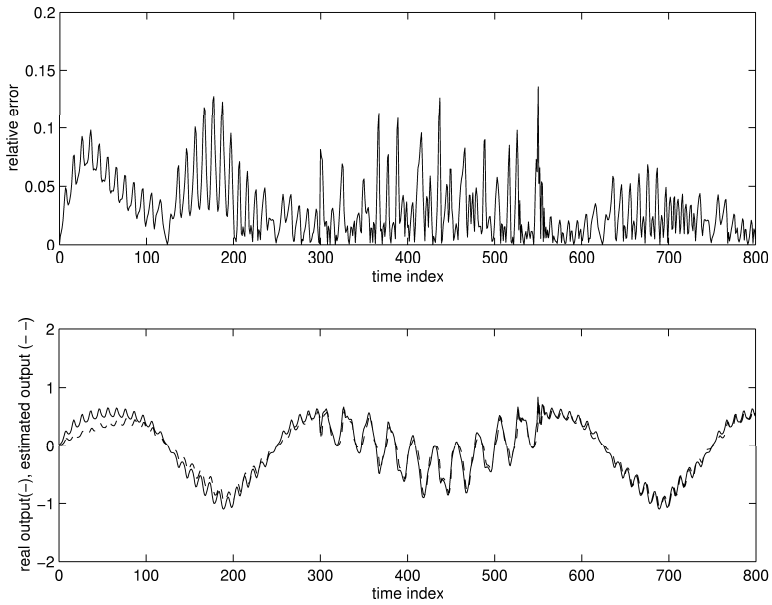

FIG. 3.28. Erreur d'identification relative, en utilisant le MFAD, avec CI $\in [0\ 1]$ et $u(k) = u_4(k)$ (Exemple 3).

lorsqu'on implante l'algorithme sur un processeur adéquat, c'est à dire, un processeur flou. Lors de la mise en oeuvre, par simulation, de l'algorithme avec MatLab ©, on mesure la complexité des calculs par la quantité de FLOPS (Floating Point Operations, en anglais) [1], utilisée pour le calcul de l'inférence donnée par l'équation (3.2).

Dans les figures 3.29 et 3.30, on présente des indicateurs graphiques qui montrent le nombre de FLOPS utilisées, selon le nombre de règles et de variables d'entrée du MFAD. Ces graphiques sont obtenues à partir des expériences développées dans la section 3.3.

Sur ces figures, on peut voir que le nombre de FLOPS reste raisonnable, même pour le cas le plus complexe traité dans cette thèse ($2.3e10^4$ FLOPS sont mis à exécution pour $n = 5$ et $M = 20$). A partir de ces graphiques, il est simple d'inférer le nombre de FLOPS qui sera utilisé en fonction du nombre de règles et de variables d'entrée.

3.5 Conclusion

Dans ce chapitre, on a présenté la contribution principale de ce travail : la proposition d'un modèle flou avec des fonctions d'appartenance dynamiques. La performance de ce modèle en identification de systèmes variant avec le temps a été comparée avec celle obtenue en utilisant des modèles flous adaptatifs classiques, avec ajustement hors ligne des paramètres des fonctions d'appartenance. Suite à cette proposition de modèle flou dynamique, on a conçut un algorithme d'identification en ligne basé sur l'AR : les résultats obtenus sur des exemples illustratifs ont démontré l'efficacité de la méthode en utilisant cet algorithme.

Dans le chapitre suivant, on discute en détail les résultats les plus importants obtenus dans la première partie de ce travail de recherche.

[1] Cette quantité est obtenue par une commande disponible sur MatLab ©.

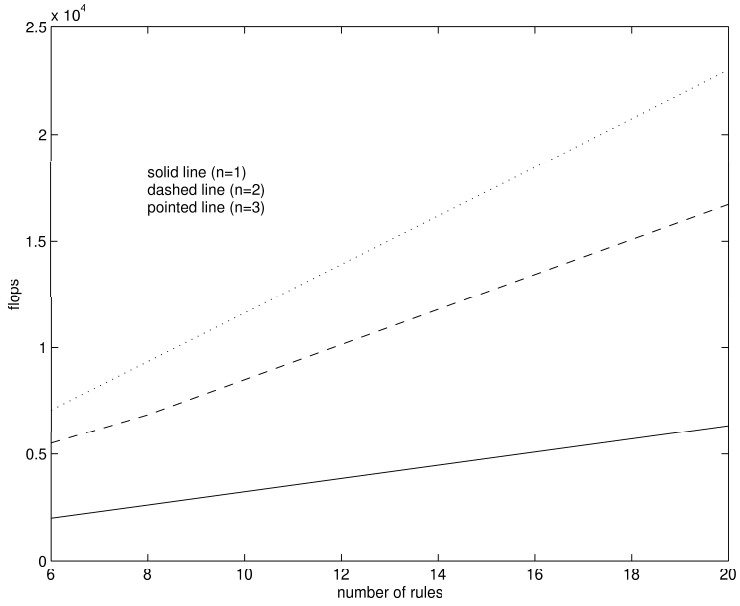

FIG. 3.29. Indicateurs graphiques sur le niveau de complexité des calculs, selon le nombre de règles M.

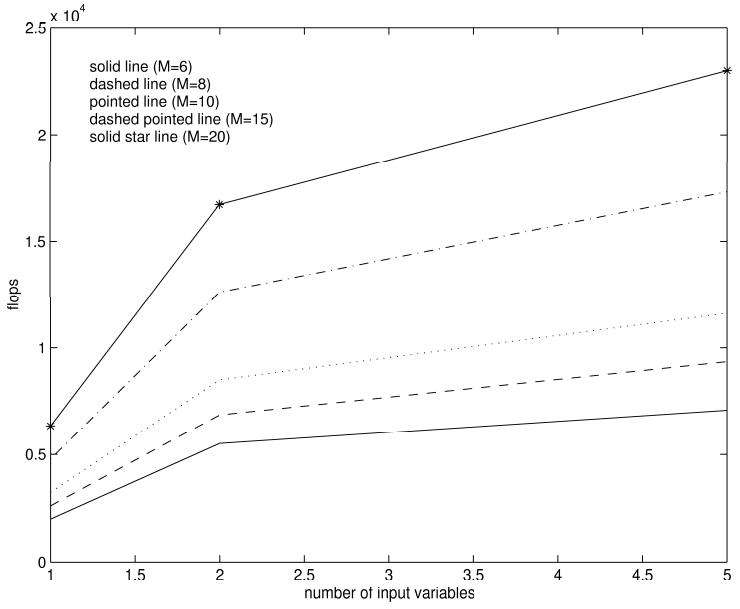

FIG. 3.30. Indicateurs graphiques sur le niveau de complexité des calculs, selon le nombre de variables d'entrée n.

Chapitre 4

Conclusions de la partie I

Les nouvelles approches de modélisation floue permettent de reculer des limitations pratiques rencontrées dans la modélisation floue adaptative classique, ce qui explique l'engouement qu'elles suscitent dans le domaine de la logique floue.

Dans cette thèse, on a présenté une approche de modélisation floue, adaptative, dynamique. Cette approche repose sur l'introduction, dans les fonctions d'appartenance, du comportement temporel des variables du système réel. Des fonctions paramétriques, qui caractérisent les fonctions d'appartenance dynamiques, sont définies à partir de la moyenne et de la variance des valeurs disponibles des variables du système.

Des algorithmes d'apprentissage hors ligne et en ligne ont été utilisés pour l'ajustement des paramètres du modèle floue dynamique. Cependant, même si l'apprentissage hors ligne est mis en oeuvre, la structure proposée pour le Modèle Floue Adaptatif Dynamique (MFAD) lui permet de s'adapter en ligne aux changements de la dynamique du système réel à modéliser. Cette propriété est très utile dans le cas de dynamiques variant avec le temps ou, même, lors de la présence de perturbations internes ou internes importantes.

Les exemples illustratifs, présentés précédemment, sur l'identification de fonctions variant avec le temps, montrent la capacité des MFAD dans de telles tâches.

1. **Conclusion sur l'algorithme hors ligne**

   Lors de l'identification hors ligne, avec la méthode du gradient, le MFAD montre une meilleure performance que le MFA par rapport :
   - au nombre plus faible de règles floues utilisées par le MFAD. Cette propriété permet de diminuer la complexité des calculs lors de sa mise en oeuvre.
   - aux valeurs du RMSE dans la phase de validation.
   - à la capacité de généralisation à partir de peu de données et peu d'itérations d'apprentissage.
   - à la performance, dans le cas de perturbation interne, sur la base de la valeur du RMSE.
   - à la performance, dans le cas de changements imprévus sur le signal d'entrée.

   Tous ces aspects sont en faveur du MFAD et correspondent à des propriétés importantes d'un modéle d'identification adapté aux applications pratiques.

2. **Conclusion sur l'algorithme en ligne**

   Pour ce qui est de l'algorithme en ligne basé sur l'apprentissage par renforcement, la contribution repose sur la proposition d'une fonction de prédiction non linéaire introduite de manière naturelle afin d'ajuster les paramètres du MFAD. L'agent d'apprentissage est utilisé pour prédire la performance du MFAD dans l'identification floue du système à modéliser. De cette manière, la fonction de prédiction est définie comme une fonction dépendant de l'erreur d'identification.

   Au sujet de la sensibilité de l'algorithme en ligne, on remarque que :
   - Il n'y a pas de variations considérables de la valeur du RMSE pour des valeurs différentes des paramètres μ et λ, en fixant la valeur de η.
   - la variation du RMSE est plus significative avec le changement du taux d'apprentissage ρ, via la variation du paramètre η.
   - la variation du RMSE pour différentes valeurs de μ, λ et η reste stable à partir d'un certain nombre de règles.
   - la performances du MFAD, à partir d'un temps t, peut être améliorée en choisissant les conditions initiales dans un certain intervalle.
   - la complexité des calculs, en prenant en compte le nombre de FLOPS réalisées, selon le nombre de règles et de variables d'entrée, reste raisonnable.

Finalement, à partir de la comparaison des performances obtenues par le MFAD en utilisant l'apprentissage hors ligne et en ligne, on peut remarquer que, globalement, les résultats au niveau de la valeur du RMSE, au sujet de l'utilisation de l'algorithme en ligne, sont améliorés par rapport à ceux obtenus par l'usage de l'algorithme d'apprentissage hors ligne.

II° PARTIE

Applications de la modélisation floue et de l'apprentissage par renforcement

Dans cette partie, on utilise nos contributions sur la modélisation floue dynamique et l'apprentissage par renforcement pour développer une application de commande prédictive. En particulier, le MFAD est utilisé comme modèle de prédiction et un algorithme d'optimisation basé sur l'apprentissage par renforcement est développé pour proposer la loi de commande.
D'abord, on rappelle les aspects méthodologiques généraux nécessaires pour le développement de l'application de commande prédictive et on présente, ensuite, notre proposition pour résoudre ce problème de commande. Finalement, on discute l'utilisation de la modélisation floue dynamique dans les problèmes de supervision et diagnostic.

Chapitre 5

Aspects méthodologiques préliminaires

Dans ce chapitre, on présente les aspects méthodologiques de base pour le développement d'une application de commande prédictive, utilisant le MFAD et l'AR. Dans la première partie, on rappelle les aspects généraux de la commande prédictive et le schéma d'AR utilisé ; on présente, ensuite, la méthode de résolution des problèmes de Programation Non Linéaire (PNL) soumis à des contraintes, algorithme de base pour l'algorithme de commande prédictive développé dans cette thèse.

5.1 Schéma de commande par modèle interne

Un aspect important dans les problèmes de commande est la capacité du contrôleur à compenser les perturbations, les bruits et les erreurs de modélisation. Dans la commande en boucle fermée, ces éléments se visualisent par la présence d'une erreur entre la signal de référence et la sortie du processus. Le schéma de Commande par Modèle Interne (CMI) est une façon de compenser cette erreur [48]. En général, ce schéma est composé de trois parties, figure 5.1 :

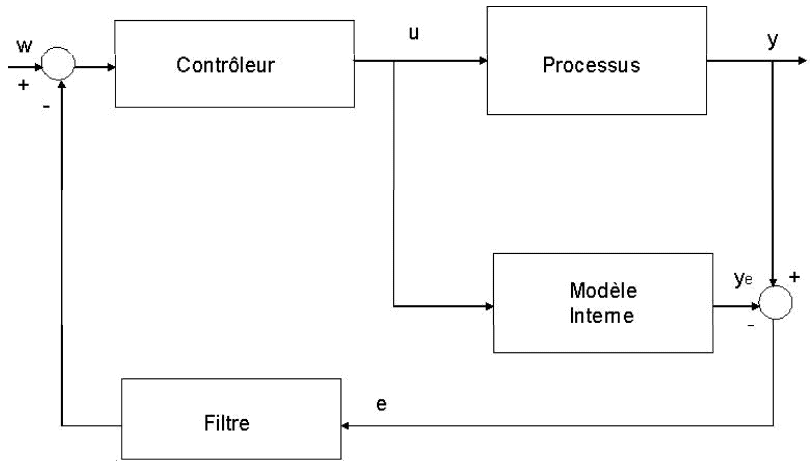

FIG. 5.1. Schéma d'une CMI.

1. Un modéle interne qui prédit la sortie du processus.
2. Un filtre de rétroaction
3. Un contrôleur basé sur le modèle

L'objectif du modèle interne est d'informer, via le calcul d'une erreur entre le processus et son modèle, sur l'existence de performances non souhaitées. Cette erreur est traitée dans un filtre de rétroaction et retranchée du signal de référence afin de calculer une action de commande rejetant l'effet des erreurs.

Avec un modèle parfait, la CMI a la capacité de supprimer l'effet des perturbations additives sur le processus, les bruits et les erreurs de modélisation. Le filtre de rétroaction permet de rejeter les bruits de mesures, donnant un effet de stabilisation, en diminuant le gain de boucle pour les hautes fréquences ; dans le cas de processus non linéaires, ce filtre doit être conçu de façon empirique.

Dans le paragraphe ci-dessous, on montre que la commande prédictive fait partie, de manière générale, de la commande par modèle interne.

5.2 La commande prédictive

La commande prédictive des processus basée sur le modèle (Model Based Predictive Control (MBPC), en anglais) n'est pas une stratégie spécifique. Il s'agit d'un ensemble de méthodes de commande qui ont été développées autour de certaines idées fondamentales [49, 48] :

- L'utilisation d'un modèle pour prédire la sortie du processus aux temps futurs (on dit "sur un horizon de prédiction").
- Le calcul d'une séquence de commande minimisant une fonction de performance.
- L'application d'une stratégie à horizon glissant dans laquelle la séquence de commande est calculée à chaque instant d'échantillonnage ; après, seulement, le premier signal de commande est envoyé au processus et les autres sont oubliés. [50].

De nombreux algorithmes de commande prédictive ont été développés et leurs différences sont basées sur les types de modèles de prédiction utilisés pour représenter le processus et les bruits et sur la fonction de performance à minimiser. La caractéristique la plus intéressante de la commande prédictive est qu'il s'agit d'une méthodologie ouverte, basée sur certains principes fondamentaux, ce qui permet de faire de nouvelles propositions pour la résolution de problèmes concrets de commande, en respectant simplement ces principes.

Les avantages fournis par une stratégie de commande prédictive peuvent être recensés comme suit :

- On peut développer la stratégie de commande en utilisant une connaissance limitée sur le processus à commander et sans qu'un type bien précis de modèle soit imposé.
- La prise en compte des retards fait partie de l'approche.
- La commande "feedforward" est introduite de manière naturelle pour la compensation des perturbations.
- Les cas mutivariables peuvent être traités assez facilement.
- Le traitement des contraintes est simple, d'un point de vue conceptuel.

– Cette approche est très utile lorsque les signaux de référence sont connus.
– La mise en oeuvre de la loi de commande est simple et le temps de calcul n'est pas prohibitif.

Malgré ces avantages, il faut souligner un inconvénient majeur lié à l'optimisation de la fonction performance : si la dynamique de système ne change pas, la loi de commande peut être calculée à l'avance mais, dans le cas de commandes adaptatives, le calcul de la loi doit se faire à chaque instant d'échantillonnage. Néanmoins, prenant en compte la puissance courante des processeurs actuels, ce "désavantage" est, de moins en moins, pénalisant.

Le problème de l'optimisation, dans la commande prédictive, peut se faire en utilisant différentes méthodes, indépendamment du modèle, mais l'élément le plus important est d'avoir un modèle de prédiction approprié car la performance de la loi de commande est très dépendante des différences entre le processus et son modèle.

5.2.1 Stratégie générale de la commande prédictive

De manière générale, la loi de commande prédictive est obtenue à partir de la méthodologie suivante :

1. Prédire les sorties futures du processus sur l'horizon de prédiction défini, en utilisant le modèle de prédiction. On dénote par $y_e(t+k|t)$, $k = 1, \ldots, Hp$, les sorties prédites et par Hp l'horizon de prédiction. Ces sorties sont dépendantes des valeurs de sorties et d'entrées du processus à commander connues jusqu'au temps t.
2. Calculer la séquence de signaux de commande, dénoté par $u(t+k|t)$, $k = 0, \ldots, Hc-1$, en minimisant un critère de performance afin de mener la sortie du processus vers une sortie de référence. On dénote par $w(t+k)$, $k = 1, \ldots, Hp$, cette sortie de référence et par Hc l'horizon de commande avec $Hc \leq Hp$. Comme $Hc \leq Hp$, on prend $u(t+Hp-1) = u(t+Hc-1)$ pour tout $Hc < k < Hp$; cela veut donc dire que le signal de commande reste constant après $k = Hc$ jusqu'à la fin de l'horizon de prédiction Hp. D'habitude, le critère de performance à minimiser est un compromis entre une fonction quadratique des erreurs entre $y_e(t+k|t)$ et $w(t+k)$ et un coût de l'effort de commande. Par ailleurs, la minimisation d'une telle fonction peut être soumise à des contraintes sur l'état et plus généralment (et plus facilement) à des contraintes sur la commande.
3. Le signal de commande $u(t|t)$ est envoyé au processus tandis que les autres signaux de commande sont oubliés. Au temps $t+1$, on acquiert la sortie réelle $y(t+1)$ et on recommence au premier pas de la méthodologie.

La figure 5.2 illustre cette méthodologie et sa mise en oeuvre utilise la structure de base montrée dans la figure 5.3. Les deux blocs fondamentaux à remarquer sur cette figure sont le modèle et l'optimiseur. Le modèle doit être capable de "capturer" la dynamique du processus, de prédire les sorties futures de manière précise et sa mise en oeuvre doit être facile ; l'optimiseur fournit les actions de commande. En présence de contraintes, la solution est obtenue via des algorithmes itératifs, avec plus de temps de calcul, évidemment.

Au niveau des applications, la MBPC est devenue très populaire, pour la commande de processus chimiques, pour la simplicité de son algorithme, en utilisant des modèles de prédiction

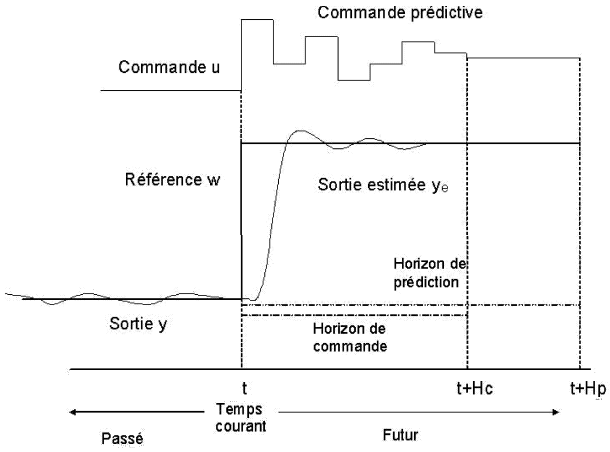

FIG. 5.2. Stratégie générale de la commande prédictive .

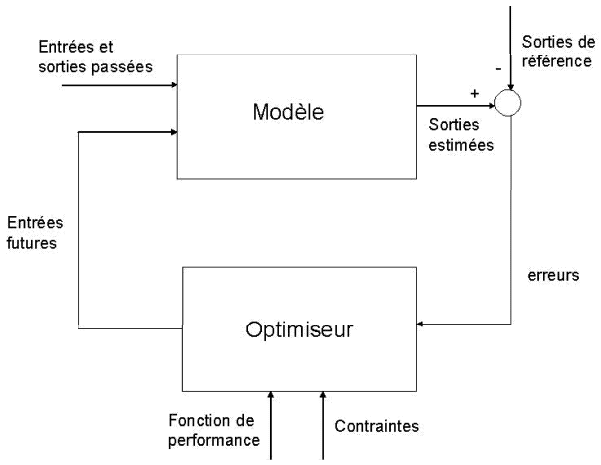

FIG. 5.3. Structure de base de la commande prédictive .

simples [51, 52] mais, aussi, dans le cas des systèmes multivariables avec contraintes. En général, la MBPC peut être utilisée sur des processus avec des retards, à phase non minimum ou instables. La commande adaptative a fourni, aussi, des idées pour le développement de stratégies à partir de modèles entrée-sortie. La MBPC a été, aussi, développée avec des modèles Espace d'état, ce qui permet l'usage de formalismes connus et sa généralisation à des cas complexes [53].

Bien qu'on ait déjà remarqué les avantages et le succès de la MBPC, les premiers travaux ne permettaient pas d'afficher des résultats généraux sur la stabilité et la robustesse, dans le cas d'un horizon de temps fini. Actuellement, de tels résultats existent, tant au niveau de la stabilité [54, 55, 56] que de la robustesse [48, 57].

5.2.2 Les éléments de la MBPC

La stratégie générale de la commande prédictive, repose sur trois éléments fondamentaux :

- **Le modèle de prédiction** est l'élément le plus important de la commande prédictive. Il doit capturer complètement la dynamique du processus sans être complexe, plutôt intuitif, tout en permettant une analyse théorique.
- **La fonction objectif** ou fonction de performance J, dont la minimisation a pour but de minimiser la différence entre la sortie y future et un signal de référence w. On peut, en plus, pénaliser l'effort de la commande pour attendre cet objectif et avoir des contraintes sur les valeurs possibles de la sortie y et du signal de commande u.
- **La procédure d'optimisation** fait référence à la méthode utilisée pour minimiser la fonction de performance J. Sous certaines considérations (critère quadratique, modèle de prédiction linéaire et sans contraintes), on peut trouver une solution analytique du problème mais, en général, il faut utiliser des méthodes itératives d'optimisation.

Les modèles de prédiction

Le modèle de prédiction a pour objectif de prédire la sortie aux instants futurs du temps. Il existe de nombreuses manières d'obtenir un tel modèle de prédiction : modèles linéaires, non linéaires, modèles dans l'espace d'état ou modèles entrée-sortie. On mentionne, ci dessous, les modèles les plus utilisés [49, 48] :

1. *Modèle basé sur la réponse impulsionelle ou modèle de convolution*, donné par l'équation suivante :

$$y_e(t+k|t) = \sum_{i=1}^{N} h_i u(t-i) = H(z^{-1})u(t+k|t) \tag{5.1}$$

où $H = h_1 z^{-1} + h_2 z^{-2} + \ldots + h_N z^{-N}$ et z^{-1} est l'opérateur de retard. La valeur de $h_i, i = 1, \ldots, N$ coincide avec la valeur de la sortie y à chaque instant d'échantillonnage lorsqu'une entrée impulsionelle unitaire est appliquée au processus.

Ce modèle est très utile pour les applications industrielles avec des processus stables : il est très intuitif et on n'a pas besoin d'information a priori sur le processus. Néanmoins, la valeur de N doit être grande (autour de 50 ou 60) ce qui arrive, en conséquence, à calculer N paramètres du pôlynome H.

2. *Modèle basé sur la réponse indicielle*, donné par l'équation suivante :

$$y_e(t+k|t) = \sum_{i=1}^{N} g_i \Delta u(t+k-i|t) \tag{5.2}$$

où g_i est la valeur de la sortie y, à chaque instant d'échantillonnage, en réponse à une entrée en échelon et le terme $\Delta u(t+k-i|t)$ est calculé comme une extension de l'équation définissant la variation de la commande $\Delta u(t) = u(t) - u(t-1)$.

Ce modèle est inspiré du modèle précédent, en changeant simplement le signal d'entrée, avec les mêmes avantages et inconvénients.

3. *Fonction de transfert*, donnée par l'équation suivante :

$$y_e(t+k|t) = \frac{B(z^{-1})}{A(z^{-1})} u(t+k|t) \tag{5.3}$$

où A et B sont des polynômes d'ordre n_a et n_b, respectivement.

Ce modèle est valide pour des processus instables ; on n'a pas besoin de beaucoup de paramètres pour les polynômes A et B, mais il faut avoir quelque information pour choisir l'ordre de ces polynômes.

4. *Modèle dans l'espace d'état*, supposé linéaire et discret, donné par l'équation suivante :

$$y_e(t+k|t) = Q\left[M^k x(t) + \sum_{i=1}^{k} M^{i-1} N u(t+k-i|t)\right] \tag{5.4}$$

où

$$x(t+1) = Mx(t) + Nu(t-1) \tag{5.5}$$

Dans ces équations, x dénote l'état, M et N sont des matrices du système. En utilisant ce modèle, on peut facilement généraliser au cas multivariable ; cependant, on a besoin d'un observateur si les états ne sont pas disponibles.

5. *Modèles non linéaires.* L'utilisation de modèles de prédiction linéaires peut ne pas être efficace pour des processus avec des non-linéarités sévères. En conséquence, il peut être intéressant d'envisager des modèles non-linéaires tels les réseaux de neurones ou les modèles flous. Il existe de nombreux travaux sur l'utilisation des modèles neuronaux et flous dans la commande préditive, voir par exemple, [58, 59, 29, 30, 31].

Quant à nous, nous allons envisager l'utilisation de notre MFAD comme modèle de prédiction non linéaire.

La fonction de performance

La forme générale de cette fonction de performance, est donnée par l'équation suivante :

$$J(N_1, Hc, Hp) = E\{\sum_{k=N_1}^{Hp} \delta(k)\left[y(t+k|t) - w(t+k)\right]^2 + \sum_{k=1}^{Hc} \lambda(k)\left[\Delta u(t+k-1)\right]^2\} \tag{5.6}$$

Le deuxième terme de l'équation (5.6) est là pour tenir compte de l'effort de commande lors de l'objectif de minimisation de l'erreur sur la sortie, mais quelques méthodes ne le prennent pas en compte. Les valeurs de N_1 et Hp définissent l'horizon de prédiction, c'est à dire l'intervalle du temps pendant lequel on veut suivre le signal de référence ; si on prend $N_1 \neq 1$, alors, on n'accorde pas d'importance aux erreurs aux premiers instants de l'horizon. Souvent, $N_1 = 1$, notamment en l'absence de retard dans le système. Les paramètres $\delta(k)$ et $\lambda(k)$ pondèrent les termes d'erreur et de coût de commande : ils peuvent être des valeurs constantes ou avoir un comportement décroissant dans le temps, de façon exponentielle.

De plus, la minimisation de la fonction de performance (5.6) peut être soumise à des contraintes pour satisfaire des limitations physiques sur les actionneurs et capteurs du processus, ou pour des raisons liées à la réalisation des contrôleurs. De façon générale, les contraintes dans le problème de commande prédictive sont du type inégalité :

$$u_{min} \leq u(t) \leq u_{max} \quad \forall t \tag{5.7}$$

$$\Delta u_{min} \leq u(t) - u(t-1) \leq \Delta u_{max} \quad \forall t \tag{5.8}$$

$$y_{min} \leq y(t) \leq y_{max} \quad \forall t \tag{5.9}$$

Dans le cas de commande avec contraintes, la minimisation de la fonction objectif ne peut pas être atteinte de façon explicite et il faut avoir recours à des procédures algorithmiques.

La procédure d'optimisation

On a déjà mentionné que, dans le cas de modèles de prédiction linéaires associés à une fonction de performance quadratique, on peut obtenir une expression analytique de la loi de commande prédictive. Dans ce contexte, la méthode la plus utilisée est la *Commande Prédictive Généralisée* (Generalized Predictive Control (GPC), en anglais) [60] qui utilise un modèle linéaire entrée-sortie et une fonction de performance générique du type (5.6). L'idée de base de cette méthode est le calcul d'une séquence de signaux de commande sur un horizon de commande défini, à partir d'une solution analytique du problème de minimisation de la fonction de performance ; cette solution est obtenue par l'utilisation d'un modèle de prédiction particulier, comme celui, par exemple, donné par l'équation suivante :

$$\mathbf{y} = \mathbf{G}\mathbf{u} + \mathbf{f} \tag{5.10}$$

où :

$$\begin{bmatrix} y_e(t+d+1|t) \\ y_e(t+d+2|t) \\ \vdots \\ y_e(t+d+N|t) \end{bmatrix} \tag{5.11}$$

et

$$\begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N+1) \end{bmatrix} \tag{5.12}$$

f étant un vecteur contenant les valeurs passées de la sortie y et de l'entrée u disponibles au temps t, et $\mathbf{G}$ est une matrice triangulaire inférieure de paramètres connus. Ce modèle (5.10) prend en considération un temps de retard dans le système de d périodes d'échantillonnage et il fournit N prédictions prenant en compte tous les incréments sur les signaux de commande futurs jusqu'au temps de prédiction courant. En substituant les prédictions données par (5.10) dans la fonction objectif (5.6) et en prenant le vecteur de référence $\mathbf{w} = [w(t+d+1) \;\; w(t+d+2) \;\; \ldots \;\; w(t+d+N)]^T$, $N_1 = d+1$, $H_p = d+N$ et $H_c = N$, on obtient la solution analytique suivante, qui donne chaque élément de la séquence de signaux de commande futurs :

$$\mathbf{u} = -\mathbf{H}^{-1} b \tag{5.13}$$

où $\mathbf{H}$ et $\mathbf{b}$ sont des matrice et vecteur, respectivement, contenant une information connue.

De cette manière, la méthode GPC, généralise le problème de commande prédictive par l'utilisation d'un modèle de prédiction entrée-sortie qui permet d'obtenir, en un pas, la séquence d'incréments de la commande à partir du temps t, via la minimisation de la fonction objectif (5.6) quelles que soient les valeurs de H_p et H_c.

Remarque 10 *On peut prendre $H_c \leq N$ afin de minimiser l'effort de calcul mais, alors, on dégrade la qualité de la prédiction donnée par l'équation (5.10).*

La GPC peut s'appliquer aux systèmes unstables et de phase non minimum, ce qui rend cette méthode très populaire au niveau des applications. Pour plus de détails sur cette méthode, voir [49, 48].

D'autres algorithmes particuliers ont été proposés dans le domaine de la commande prédictive. En particulier, en faisant une modification sur le modèle de prédiction (5.3), la loi de commande est obtenue par une approche désignée, en anglais, *Extended Prediction Self Adaptive Control* [49]. L'idée principale de cette approche est de réduire l'horizon de commande à 1 ; donc, l'unique valeur de commande qu'on a besoin de calculer est u(t), via la minimisation de la fonction de performance suivante :

$$\sum_{k=d}^{N} \gamma(k) \left[w(t+k) - P(z^{-1}) y_e(t+k|t) \right]^2 \tag{5.14}$$

Remarquons que la fonction objectif ci-dessus est différente de celle de l'équation générale (5.6). Ici, l'erreur est calculée par rapport au terme de prédiction $P(z^{-1})y_e(t+k|t)$ qu'incorpore, via le polynôme P, les prédictions passées au temps $t+k$.

On mentionne aussi d'autres algorithmes désignés, en anglais, *Extended Horizon Adaptive Control*, *Model Predictive Heuristique Control*, *Dynamic Matrix Control*, utilisant des modèles linéaires entrée-sortie. On peut les trouver dans [49].

Néanmoins, si on utilise des modèles non linéaires pour prédire la sortie du processus, alors, on perd la convexité du problème d'optimisation. C'est, évidemment, un gros désavantage au niveau de l'application en ligne, à cause du temps de calcul que peut prendre chaque pas de l'optimisation pour arriver à une solution admissible. Dans ce cas de commande prédictive non-linéaire, on doit donc utiliser des méthodes d'optimisation non convexe, comme la *Programmation Quadratique Séquentielle* [48] ou des techniques de recherche discrète comme la Programmation Dynamique, la méthode de Séparation et Évaluation Progressive (SEP), les Algorithmes Génétiques [61, 58], etc.

D'autres algorithmes, utilisant des techniques de linéarisation, sont des approches viables pour la commande prédictive non- linéaire. On peut, par exemple, linéariser le modèle non linéaire à chaque instant du temps et utiliser ce modèle dans une méthode ordinaire de commande prédictive [58] ; on peut, aussi, linéariser par rétroaction (Feedback Linearization, en anglais) l'ensemble du système à tout instant. Dans ces cas, on peut revenir à l'utilisation de la Programmation Quadratique [59].

Dans notre travail, on propose un algorithme d'optimisation basé sur l'apprentissage par renforcement pour minimiser la fonction objectif.

Remarque 11 *Après la présentation de la commande prédictive, on peut remarquer que cette stratégie de commande entre dans un schéma de CMI (voir section 5.1, figure 5.1), où le contrôleur utilise un algorithme d'optimisation et le filtre compense les erreurs de modélisation et les perturbations.*

5.3 Algorithme d'apprentissage TD-Gammon

Une des applications les plus connues de l'apprentissage par renforcement est celle développée dans [22] pour l'apprentissage du jeu de backgammon par un réseau de neurones. Dans ce travail, l'auteur propose un algorithme d'apprentissage, appellé *Apprentissage TD-Gammon*, inspiré de l'algorithme d'apprentissage utilisant la méthode TD, donné par l'équation (2.9). Cette équation est reprise par l'algorithme TD-Gammon, mais en substituant à la valeur de $P(x_{t+1}, \theta_t)$ une valeur de renforcement r donnée selon le résultat obtenu dans l'itération d'apprentissage courante. En conséquence, on obtient une loi d'apprentissage donnée par l'équation suivante :

$$\theta_{t+1} = \theta_t + \eta(r - P(x_t, \theta_t))\frac{\partial P(x_t, \theta_t)}{\partial \theta} \tag{5.15}$$

Dan le cas particulier de l'application traitée dans [22], le vecteur des paramètres θ correspond aux poids dans le réseau de neurones donnant la valeur de $P(x_t, \theta_t)$, pour l'apprentissage du jeu. Ce schèma d'apprentissage, mis en oeuvre par l'équation (5.15), a aussi été utilisé dans [28] pour calculer une commande en ligne, basée sur le renforcement par association [18]. Dans les deux applications, le renforcement est donné par deux valeurs discrètes : zéro et une autre valeur constante, définie selon le problème et associées au succès ou échec pendant l'apprentissage.

On va utiliser ce schèma d'apprentissage pour calculer le signal de commande à chaque instant du temps.

5.4 Le problème de programmation non linéaire et les fonctions de pénalisation

On présente, ici, la méthode proposée dans [62] pour résoudre le problème de PNL soumis à des contraintes. Cette méthode repose sur l'utilisation de fonctions de pénalisation qui permettent de déplacer une solution violant les contraintes justement dans le sens négatif du gradient de la fonction de performance à optimiser.

Soit la forme canonique du problème de PNL suivante :

$$\begin{aligned} Max_x h(x) \quad ; \quad & h(x) = h(x_1, x_2, ..., x_n) \\ sous \quad & g_i(x) \leq 0, i = 1, 2, ..., m \end{aligned} \tag{5.16}$$

On suppose que $h(x)$ et $g_i(x)$ sont continues et dérivables. Basée sur la méthode du gradient, si une solution admissible $x(k)$, obtenue à l'itération k, se déplace dans la direction appropriée du gradient $\nabla h(x)$, alors, la fonction de performance $h(x)$ peut être améliorée. Si la nouvelle solution $x(k+1)$ est en dehors du domaine admissible, c'est à dire viole les contraintes, alors, $x(k+1)$ doit se déplacer selon la direction opposée au gradient afin de satisfaire $g_i \leq 0$. Basée sur ce raisonnement, une solution $x(k+1)$ est obtenue à partir d'une solution $x(k)$, selon l'algorithme suivant :

$$x(k+1) = x(k) + \alpha d(x(k)) \tag{5.17}$$

où k est l'indice d'itération, $0 < \alpha < 1$ est le taux d'apprentissage et $d(x(k))$ est définie comme :

$$d(x(k)) = \nabla h(x) - \sum_{i=1}^{m} w_i \nabla g_i(x) \tag{5.18}$$

où $w_i \in \Re$ est un paramètre qui pondère le gradient $\nabla g_i(x)$. Ce paramètre est défini par l'équation suivante :

$$w_i = \begin{cases} 0 & si \quad g_i(x) \leq 0 \\ \delta_i & si \quad g_i(x) > 0 \end{cases} \tag{5.19}$$

$$\delta_i = \frac{1}{g_{max} - g_i(x) + \delta} \tag{5.20}$$

$$g_{max} = max\{g_i(x), i = 1, 2, ..., m\} \tag{5.21}$$

où $\delta \in \Re$ a une valeur positive très petite et g_{max} dénote la valeur maximale des contraintes violées. Selon l'usage des fonctions de pénalisation, on calcule, à nouveau, la fonction de performance $h(x)$ de la manière suivante :

$$h(x) = \begin{cases} h(x) & si \quad g_{max} \leq 0 \\ \frac{h(x)}{M+g_{max}} & si \quad g_i(x) > 0 \end{cases} \tag{5.22}$$

où $M \in \Re$ a une valeur positive très grande. L'étude de la convergence de cet algorithme est présenté en détail dans [62].

5.5 Conclusion

Ce chapitre a présenté les fondements méthodologiques de base pour le développement des applications de commande prédictive. Dans ce schéma de commande, on a repris l'algorithme d'AR pour la résolution des problèmes d'optimisation et on a mis l'accent sur la résolution de problèmes de PNL avec des contraintes via des fonctions de pénalisation. Soulignons que l'algorithme TD-Gammon présenté est différent de celui proposé dans ce travail (section 3.2.2) car il s'agit d'une approche classique d'apprentissage par AR qui a été validé dans d'autres travaux.

Dans le prochain chapitre, on propose un algorithme hybride utilisant l'AR, via l'algorithme TD-Gammon et fonctions de pénalisation, pour la résolution des problèmes de PNL associés à la commande prédictive. Le MFAD, ajusté hors ligne, sera utilisé comme modèle de prédiction.

Chapitre 6

Le MFAD et l'AR dans la commande prédictive

Dans ce chapitre, on montre l'intérêt de la modélisation floue dynamique et de l'apprentissage par renforcement pour la commande prédictive d'un réservoir de fermentation, soumis à des contraintes sur la valeur du signal de commande [61, 63].

Le MFAD, ajusté par un algorithme d'apprentissage hors ligne, est utilisé comme modèle de prédiction. Dans ce cas, on introduit une modification sur la définition de la moyenne et la de la variance proposée dans les équations (3.4) et (3.6). L'apprentissage par renforcement est utilisé pour résoudre le problème d'optimisation, en utilisant un schéma classique de renforcement, ce qui amène à proposer une fonction de performance particulière (voir section 5.3). Étant donné la présence des contraintes dans le problème, on utilise l'approche proposée dans [62] (voir section (5.4)), pour l'optimisation de la nouvelle fonction de performance.

Dans la première partie, on présente la modification faite sur les équations (3.4) et (3.6). Ensuite, dans la deuxième partie, on propose l'algorithme de commande prédictive ; finalement, on montre les résultats obtenus sur la modélisation du processus de fermentation en utilisant le MFAD et la performance de l'algorithme de commande prédictive travaillant avec ce modèle.

6.1 MFAD pour l'identification des systèmes invariants

La définition qu'on a proposée dans le paragraphe (3.1) pour la moyenne et la variance, sur un certain intervalle de temps, est utile dans le cas de l'identification de systèmes variant avec le temps ; cependant, on a obtenu une performance satisfaisante avec le MFAD, pour le cas invariant, en proposant une modification des équations (3.4) et (3.6) prenant en compte toute l'information disponible sur les données historiques depuis $t = 0$ jusqu'au temps courant $t = t_j$. En conséquence, les équations (3.4) et (3.6) deviennent :

$$\overline{x}_i(t_j) = \frac{\sum_{k=0}^{j}(x_i(t_k))}{j+1} \tag{6.1}$$

et

$$\sigma_i^2(t_j) = \frac{\sum_{k=0}^{j}(x_i(t_k) - \overline{x}_i(t_k))^2}{j+1} \tag{6.2}$$

La performance du MFAD en utilisant cette modification a déjà été démontrée dans l'identification de fonctions invariantes [64, 65] et d'un processus variant avec le temps [66]. C'est donc cette structure qu'on va utiliser pour l'identification du processus de fermentation.

6.2 Algorithme de commande prédictive

On va utiliser, ici, une commande prédictive à un pas, c'est à dire que l'on va choisir $Hc = Hp = 1$ (voir la section (5.2)). La loi de commande prédictive est obtenue en résolvant le problème de PNL suivant :

$$\begin{aligned} min_u J(u) &= \frac{1}{2}(w(t+1) - y_e(t+1))^2 \\ &sous \\ u_{min} &\leq u(t) \leq u_{max} \\ \Delta u_{min} &\leq \Delta u(t) \leq \Delta u_{max} \end{aligned} \tag{6.3}$$

où $w(t+1)$ est le signal de référence, $y_e(t+1) = f(u(t), y(t))$ est la sortie estimée par le modèle de prédiction, en utilisant l'information disponible au temps t, et $\Delta u(t) = u(t) - u(t-1)$.

On utilise la méthode présentée dans la section (5.4) pour la résolution du problème (6.3), en changeant ce problème de minimisation en un problème de maximisation, donné par :

$$\begin{aligned} max_u h(u) &= -J(u) \\ sous\ g_i(u) &\leq 0, \quad i = 1, ...I \end{aligned} \tag{6.4}$$

La loi de commande prédictive à un pas est calculée par l'algorithme suivant :

$$u^{k+1}(t) = u(t-1) + \rho^k d^k(u) \tag{6.5}$$

où k est l'indice d'itération pendant un intervalle d'échantillonnage, $0 < \rho < 1$ est le taux d'apprentissage, $d^k(u)$ est défini selon l'équation (5.18) et $\nabla h(u)$ est donné par l'équation suivante :

$$\nabla h(u) = (w(t+1) - y_e(t+1))\frac{\partial y_e(t+1)}{\partial u^k(t)} \tag{6.6}$$

6.2.1 Définition du taux adaptatif par AR

La loi de commande proposée par l'équation (6.5) est dépendante du taux d'apprentissage ρ : selon sa valeur, on n'arrive pas à une solution exploitable vis-à-vis des contraintes pendant la recherche de la solution optimale. Le choix d'une bonne valeur du taux d'apprentissage n'est pas évident ; en conséquence, on propose un algorithme en ligne pour ajuster cette valeur.

Vu la performance des algorithmes basés sur l'AR pour la résolution des problèmes d'optimisation en ligne, au niveau de la commande de processus [26, 28], on formule, à nouveau, un problème de PNL, pour obtenir une loi d'ajustement de ρ pendant le calcul de la commande $u(t)$, via l'équation (6.5). Dans ce cas, on résout un deuxième problème de PNL pour arriver, ensuite, à une valeur améliorée de $u(t)$ et $\Delta u(t)$ respectant les contraintes.

Le nouveau problème de PNL est formulé de la façon suivante :

$$\begin{aligned} & min_{\rho} h(\rho) = \frac{1}{2}(r - J)^2 \\ & sous \quad g_j(\rho), j = 1, ...J \end{aligned} \tag{6.7}$$

où J est la fonction de performance définie pour le problème de PNL (6.3) de recherche de la commande u et r est un signal de renforcement à définir. La résolution du problème ci-dessus, via la méthode présentée dans la section (5.4), conduit à la loi d'ajustement suivante :

$$\rho(j+1) = \rho(j) + \eta d(\rho(j)) \tag{6.8}$$

où j est l'indice d'itération pour l'ajustement de ρ pendant une itération k, au temps t du calcul de u ; $0 < \eta < 1$ est le taux d'apprentissage et $d(\rho)$ est défini selon l'équation (5.18), avec $\nabla h(\rho)$ donné par l'équation suivante :

$$\nabla h(\rho) = -(r - J)(w(t+1) - y_e(t+1)) \frac{\partial y_e(t+1)}{\partial u^k(t)} \frac{\partial u^k(t)}{\partial \rho} \tag{6.9}$$

où $u(t)$ est le résultat de l'équation (6.5) et $\frac{\partial u^k(t)}{\partial \rho} = d^k(u)$.

Avec cette fonction de performance proposée en (6.7), on arrive à respecter les contraintes $g_i(\rho)$; la loi d'ajustement (6.8) est semblable à celle retenue dans les applications classiques utilisant l'AR (voir section 5.3).

6.3 Exemple illustratif

6.3.1 Description du processus

Nous allons appliquer la commande prédictive, par simulation, à un processus de fermentation de laboratoire [61, 63]. Le volume du réservoir est de 40 litres, mais il est rempli, d'ordinaire, avec 25 litres d'eau. Le réservoir est alimenté par un flux d'air constant, fixé par

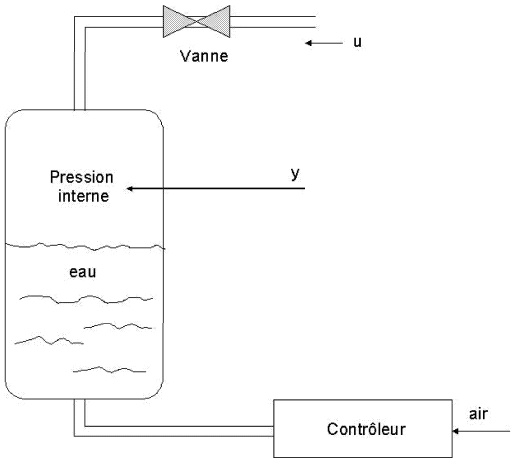

FIG. 6.1. Processus de fermentation.

un contrôleur placé au fond du réservoir. La pression d'air dans le réservoir est commandée par une vanne placée en haut du réservoir. La figure 6.1 donne le schéma du processus qui possède une caractéristique non linéaire en régime permanent à cause de la caractéristique non linéaire de la vanne de commande et une dynamique non linéaire, donnée par l'équation suivante :

$$\frac{dP}{dt} = \frac{1000RT}{22.4V_h}\left[\Psi_g - (\pi R_H^2)\sqrt{\frac{2P_o}{\rho_o K_f}}ln(\frac{P}{P_o})\right] \tag{6.10}$$

où :

R est la constante des gaz ($8.134 Jmol^{-1}K^{-1}$)

T est la température ($305K$)

V_h est le volume du gaz ($0.015m^3$)

Ψ_g est le débit du gaz ($3.75x10^{-4}m^3s^{-1}$)

R_H est le rayon du tube de sortie ($0.0178m$)

P_o est la pression de référence ($1.013x10^5 Nm^{-2}$)

ρ_o est la densité de l'air à l'extérieur ($1.2Kgm^{-3}$)

P est la pression dans le réservoir (Nm^{-2})

K_f est le facteur de friction de la vanne ($Jmol^{-1}$)

Le facteur K_f est une fonction non-linéaire de la position de la vanne, dénotée par u, et de Ψ_g. Sous ces caractéristiques, le processus a une entrée (la position de la vanne) et une sortie (la pression dans le réservoir) ; la constante de temps la plus petite est environ de 45 sec., ce qui permet de prendre un temps d'échantillonnage de 5 sec. Le facteur K_f est donné par l'équation suivante :

$$K_f = Kg * Kv \tag{6.11}$$

avec

$$K_g = -3.538e10 * (\Psi_g)^3 + 4.473e7 * (\Psi_g)^2 - 1.9567e4 * (\Psi_g) + 3.898 \tag{6.12}$$

et

$$K_v = 8.7056{*}u^6 - 787.6{*}u^5 - 1.9388e3{*}u^4 + 2.4454e6{*}u^3 - 7.2568e7{*}u^2 + 1.2393e9{*}u + 1.1610e11; \tag{6.13}$$

où u est un pourcentage de fermeture de la vanne ; en conséquence, $u \in [0 - 100]$. De plus, on a des contraintes sur la valeur de Δu : le changement maximal entre chaque instant d'échantillonnage est de 10 %, c'est à dire : $\Delta u_{min} = -10\%$ et $\Delta u_{max} = 10\%$.

6.3.2 Modélisation du processus par un MFAD

Le modèle de prédiction à un pas utilisé dans cette commande prédictive est obtenu à partir de l'ajustement hors ligne du MFAD. Au vu de la caractéristique invariante dans le temps du processus, on utilise la modification déjà présentée pour la définition des fonctions d'appartenance dynamiques.

Un ensemble de 323 paires de données d'apprentissage $\{(\mathbf{X}(t_j), y(t_j)),\ j = 1, ..., 323\}$, a été obtenu, en prenant en compte les expériences proposées par [63], en utilisant l'entrée $u(t) = u_T(t)$ définie par l'équation (6.14). Dans la figure 6.2, on montre le comportement de processus avec cette entrée.

$$u_T(t) = \begin{cases} 50 + 30\sin(6.28t/1000) + 5\sum_{i=1}^{5}\sin(6.28t/a_i) & si \quad t < 1600 \\ 80 + \sum_{i=1}^{4}\sin(6.28t/b_i) & si \quad t > 1600 \end{cases} \tag{6.14}$$

où $a_i = 10, 50, 100, 250, 500$ et $b_i = 10, 50, 100, 500$.

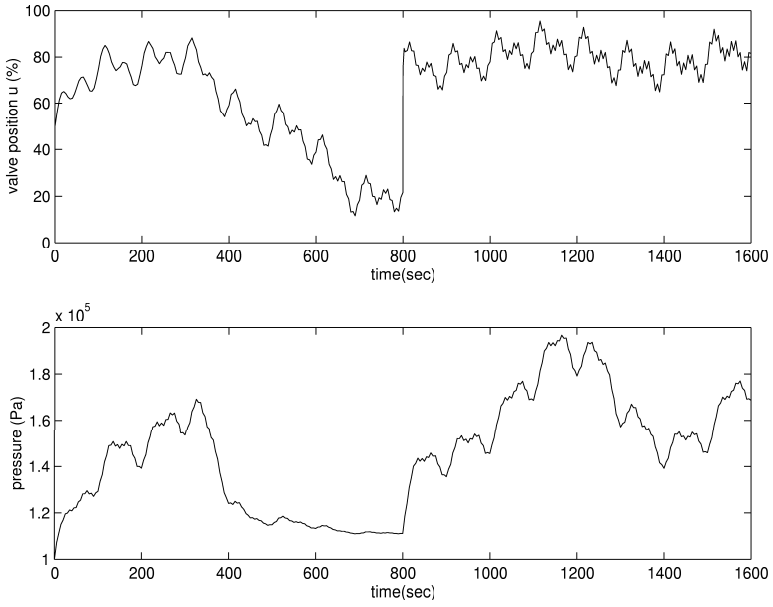

FIG. 6.2. Comportement du processus en réponse à l'entrée u_T.

En utilisant les lois d'ajustement hors ligne proposées dans (3.10), (3.11) et (3.12), on arrive à un modèle de prédiction de la forme $y(t+1) = f(u(t), y(t))$, où $f(u(t), y(t))$ est décrit par un MFAD. Pour tenir compte des amplitudes différentes des variables d'entrée et de sortie du processus de fermentation, on pondère la variation de chaque paramètre u^l, v_i^l et w_i^l du MFAD, par la norme $||.||$ de chaque vecteur de variation des paramètres $\Delta\theta$ associé, dans les règles, aux variables d'entrée $u(t)$ et de sortie $y(t)$ du modèle [28](cf. équation (6.15) pour cette procédure de "normalisation"). De cette manière, on n'a pas besoin de mettre à l'échelle les variables d'entrée et de sortie, tout en gardant des valeurs convenables pour les paramètres du MFAD ; ainsi, pour chaque itération de la phase d'apprentissage, on a :

$$\theta(K+1) = \theta(K) - \rho_\theta \frac{\Delta\theta}{||\Delta\theta||}\bigg|_K \tag{6.15}$$

où $\theta(K+1)$ dénote le vecteur de paramètres de l'ensemble flou dynamique associé aux variables d'entrée ou sortie du MFAD, contenant les valeurs pour chaque règle. Le vecteur $\Delta\theta$ est obtenu en traitant les équations (3.13), (3.14) et (3.15) pour chaque élément du vecteur.

Après de nombreuses séances d'apprentissage, comme celles développées dans la partie modélisation (voir section (3.3)), le meilleur MFAD a $M = 30$ et 90 paramètres ajustables. Ce modèle flou a atteint une précision, dans la phase d'apprentissage, de $RMSE = 909.6689$ et $RMSE = 776.7259$ avec une erreur relative moyenne autour de 0.5% dans la phase de validation, en utilisant l'entrée $u(t) = u_V(t)$, donnée par l'équation (6.16). La performance du MFAD dans la phase de validation est montrée dans la figure 6.3.

$$u_V(t) = 60 + 25\sin(6.28t/250) + 10\sin(6.28t/75) + 3\sin(6.28t/10) \tag{6.16}$$

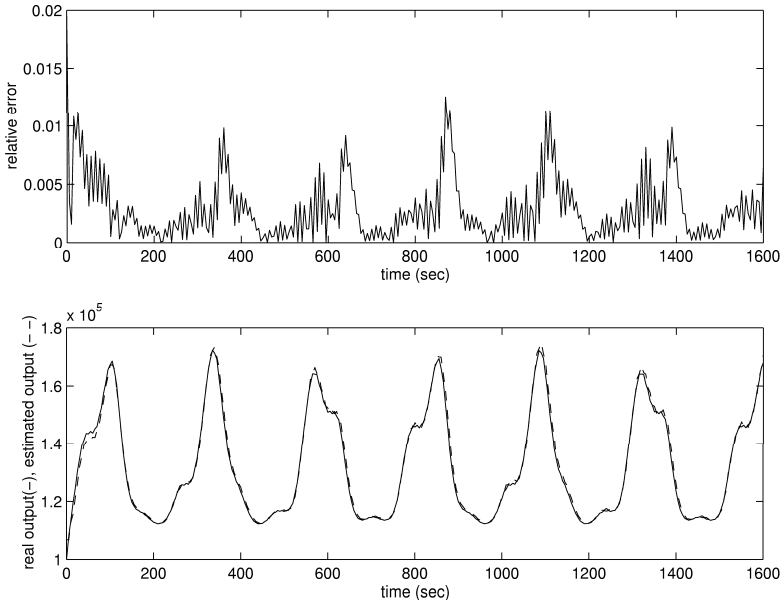

FIG. 6.3. MFAD pour la prédiction de $y(t+1)$.

6.3.3 Commande prédictive

La commande prédictive utilise le schéma de CMI illustré ci-dessous. Il fait intervenir un filtre sur lequel on n'a pas travaillé prenant, tout simplement, celui proposé dans [63] :

$$\frac{F(z)}{E(z)} = \frac{0.1z + 0.9}{z} \tag{6.17}$$

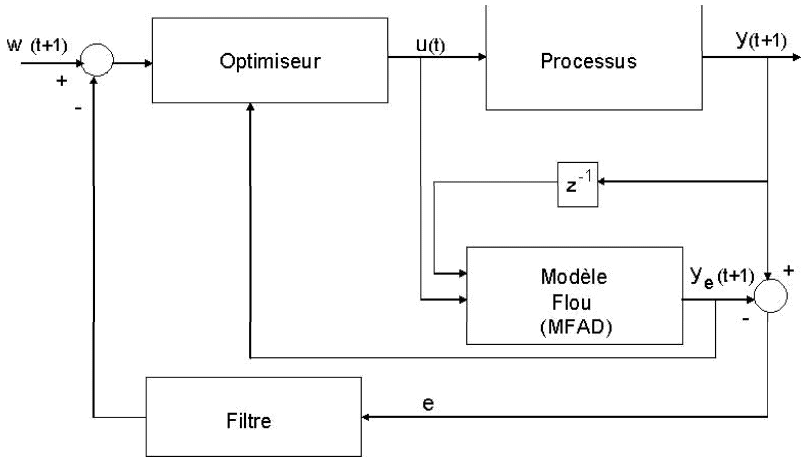

FIG. 6.4. Schéma de commande prédictive.

Pour l'utilisation de l'algorithme proposé dans la section (6.2), les contraintes $g_i(u), i = 1, ..., 4$, sur la commande seront décrites par les inégalités suivantes :

$$\begin{aligned} g_1(u) &= u(t) - 100 \leq 0 \\ g_2(u) &= -u(t) \leq 0 \\ g_3(u) &= \Delta u(t) - 10 \leq 0 \\ g_4(u) &= -\Delta u(t) - 10 \leq 0 \end{aligned} \tag{6.18}$$

en conséquence :

$$\begin{aligned} \nabla g_1(u) &= 1 \\ \nabla g_2(u) &= -1 \\ \nabla g_3(u) &= 1 \\ \nabla g_4(u) &= -1 \end{aligned} \tag{6.19}$$

D'un autre côté, les contraintes $g_j(\rho), i = 1, ..., 6$, sur le taux d'apprentissage ρ sont décrites par les inégalités suivantes :

$$\begin{aligned} g_1(\rho) &= \rho d(u) - 10 \leq 0 \\ g_2(\rho) &= -\rho d(u) - 10 \leq 0 \\ g_3(\rho) &= u(t-1) + \rho d(u) - 100 \leq 0 \\ g_4(\rho) &= -u(t-1) - \rho d(u) \leq 0 \\ g_5(\rho) &= \rho - 1 \leq 0 \\ g_6(\rho) &= -\rho \leq 0 \end{aligned} \tag{6.20}$$

où le terme $d(u)$ est celui donné après satisfaction des contraintes du problème de PNL (6.4). En développant les dérivées partielles, on a :

$$\begin{aligned} \nabla g_1(\rho) &= d(u) \\ \nabla g_2(\rho) &= -d(u) \\ \nabla g_3(\rho) &= d(u) \\ \nabla g_4(\rho) &= -d(u) \\ \nabla g_5(\rho) &= 1 \\ \nabla g_6(\rho) &= -1 \end{aligned} \tag{6.21}$$

La valeur de renforcement, dans l'équation (6.7), est définie par l'équation suivante :

$$r = \begin{cases} 0 & si \quad J \leq 1000000 \\ -10^3 * J & si \quad J > 1000000 \end{cases} \tag{6.22}$$

De cette manière, si on a une erreur $e = 0.5(w - y_e)$ avec un ordre d'amplitude de 10^3, le taux d'apprentissage est minimisé par rapport à une fonction de performance comme celle donnée dans le problème (6.3) ; sinon, on punit cette erreur en donnant un renforcement négatif.

On a pris les valeurs suivantes pour les paramètres de l'algorithme : $M = 10^4$, $\delta = 10^{-2}$ et $\eta = 10^{-5}$, avec une valeur maximale d'itérations $k = 5$ pour chaque problème de PNL associé à l'optimisation de la commande et à l'optimisation du taux ρ. On part d'une valeur initiale de $\rho = 0.02$, au début de chaque itération du problème de PNL associé à la commande.

Remarque 12 *Une itération k dans le problème de PNL de la commande correspond à une certaine quantité d'itérations jusqu'à trouver des valeurs de $u(t)$ et $\Delta u(t)$ satisfaisant les contraintes. De même, pour une itération j dans le cas du problème de PNL associé à la recherche de ρ.*

On présente, dans ce qui suit, les résultats obtenus par simulation. Afin de faire des comparaisons, on montre, d'abord, la performance de la commande prédictive obtenue en utilisant la fonction objectif proposée dans le problème (6.3) avec un taux d'apprentissage ρ constant, mais prenant des valeurs différentes, figure 6.5. Le signal de référence est décrit par une fonction en escalier. Le signal de sortie, en continu, correspond à une valeur $\rho = 0.02$; les signaux tiret et pointillé, correspondent aux valeurs $\rho = 0.03$ et $\rho = 0.04$, respectivement.

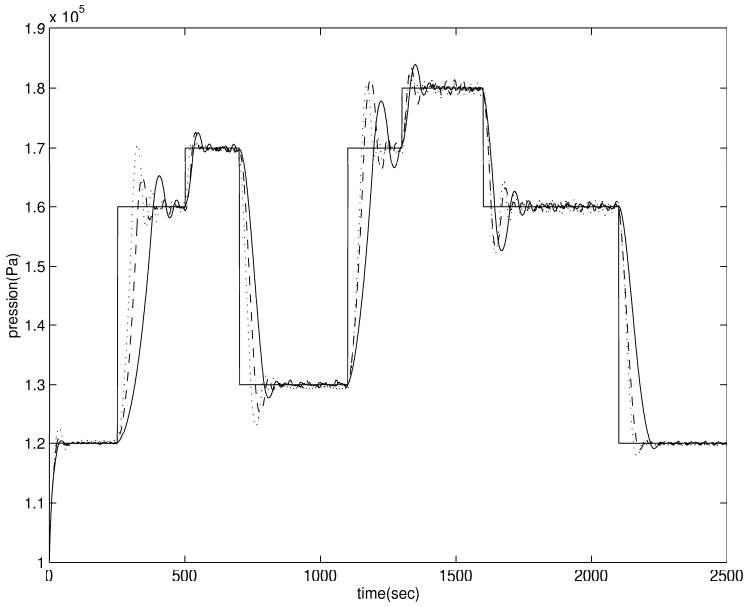

FIG. 6.5. Commande prédictive obtenue pour le problème (6.3).

En prenant ces dernières valeurs du taux ρ, on peut obtenir des réponses plus rapides, mais avec des dépassements importants au début d'un changement de la valeur du signal de référence. En utilisant des valeurs de ρ autour 0.02, on n'arrive pas à suivre la valeur de la référence à certains niveaux.

La figure 6.6 suivante montre la performance de la commande prédictive en utilisant l'algorithme proposé dans cette thèse, avec le taux d'apprentissage ρ ajusté en ligne, par AR.

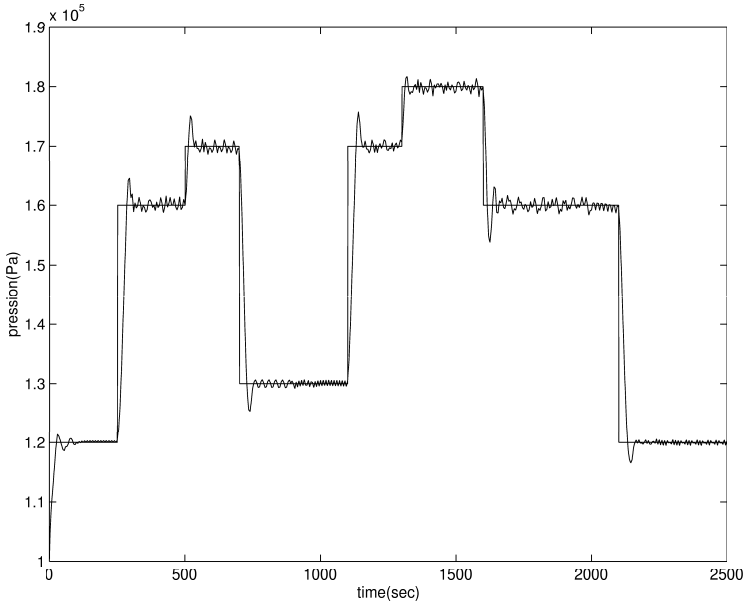

FIG. 6.6. Commande prédictive obtenue en utilisant un taux d'apprentissage adaptatif.

Les comparaisons entre les deux figures 6.6 et 6.6 mettent en évidence la performance de la commande prédictive obtenue avec l'algorithme (6.5) avec une taux adaptatif : on peut

suivre les changements du signal de référence plus vite et sans un dépassement important, par rapport aux performances illustrées dans la figure 6.6.

Un résultat, en utilisant un taux adaptatif et en prenant la fonction de performance donnée dans (6.3), c'est à dire sans renforcement et sous les mêmes conditions de simulation, est illustré dans la figure 6.7 : on peut remarquer que, malgré un taux adaptatif aussi (sortie en continue), on n'arrive pas à obtenir la performance obtenue en introduisant l'AR dans la fonction de performance définie par l'équation (6.7) (sortie pointillée).

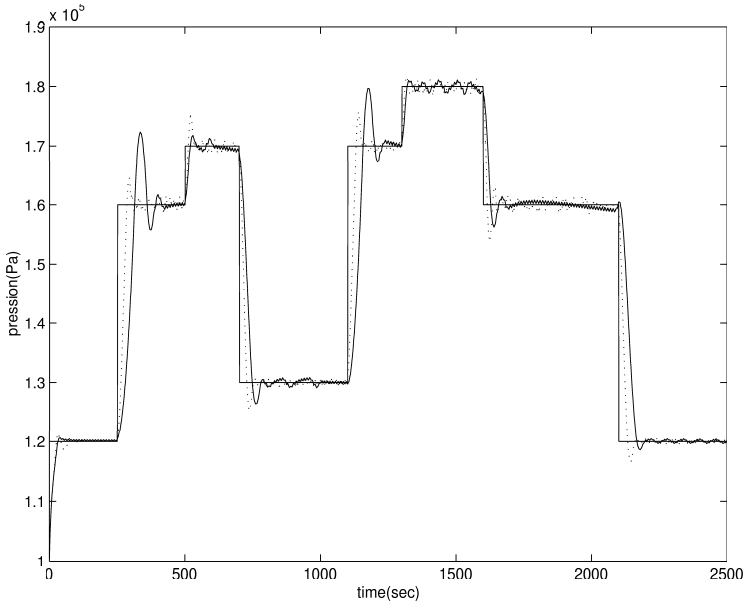

FIG. 6.7. Commande prédictive obtenue en utilisant un taux d'apprentissage adaptatif avec et sans AR.

Sur la base des résultats précédents, on souligne trois points d'intérêt, comme conséquence de l'utilisation d'un taux d'apprentissage ajusté par l'AR :

- L'utilisation de l'AR pour l'ajustement de ρ permet de trouver une meilleure valeur de la commande par rapport à celle obtenue avec le taux adaptatif sans renforcement ou le taux à valeurs constantes.
- Au niveau du régime transitoire, on arrive plus rapidement au signal de référence sans un dépassement important.
- Au niveau du régime permanent, bien qu'on arrive à maintenir la sortie de processus autour de la valeur de référence, il existe un comportement oscillatoire autour de cette valeur. Ce comportement peut indiquer le besoin d'ajuster le modèle de prédiction ou d'augmenter l'horizon de prédiction H_p (en proposant des modèles de prédiction à H_p pas). On rappelle que, dans ces expériences, on a pris un horizon de prédiction $H_p = 1$.

Remarque 13

Bien qu'il existe une preuve de convergence de la méthodologie de base, lorsqu'on utilise des fonctions de pénalisation pour la recherche de la valeur optimale de $u(t)$ minimisant la fonction objectif (voir le travail [62]), il faut trouver les conditions pour assurer la stabilité du système de commande à boucle fermée. Cela conduit, peut être, à trouver une condition additionnelle sur la valeur du taux ρ.

6.4 Conclusion

Ce chapitre a été centré sur la proposition d'un algorithme hybride, basé sur l'utilisation de fonctions de pénalisation et l'AR, pour résoudre le problème d'optimisation de la commande prédictive. Ceci a permis d'améliorer la valeur de la commande, en proposant un taux d'apprentissage ajusté en ligne. Ce taux d'apprentissage est solution d'un autre problème d'optimisation utilisant, aussi, des fonctions de pénalisation. Les résultats obtenus, en simulation, pour la commande d'un processus de fermentation, ont démontré l'efficacité de l'algorithme de commande proposée et du MFAD comme identificateur.

Dans le chapitre suivant, on avance que l'information contenue dans les fonctions d'appartenance du MFAD, peut être utilisée pour des tâches de diagnostic et supervision, comme une autre application possible du MFAD.

Chapitre 7

Le MFAD dans la supervision et le diagnostic

L'automatisation est un facteur important qui permet d'augmenter la performance des procédés industriels. La mise en oeuvre des systèmes d'automatisation vise, en principe, à la surveillance des différentes variables d'opération, de manière dynamique, et au calcul, en partant de cette information, des signaux de commande. Ainsi, les systèmes d'automatisation ont besoin, d'une part, de systèmes d'information pour s'assurer que l'information est bien utilisée dans les différents secteurs d'opération et, d'autre part, de systèmes de commande numérique permettant et assurant un bon comportement des processus, tout ceci avec le but de satisfaire la demande en produits de meilleure qualité, de diminuer les temps et les coûts de production.

Cependant, l'automatisation devient plus importante si on la pense de façon intégrale [67] : la structure d'intégration doit permettre le flux d'information à tous les niveaux de la production (gestion administrative, opérations, etc.) sur l'état des processus, des produits obtenus et toute l'information importante autour des aspects de la production. En conséquence, les besoins du secteur industriel demandent une procédure de gestion globale, intégrant les systèmes de commande et les systèmes de gestion administratives, pour la prise de décisions. Une des architectures la plus connue au niveau des schémas d'automatisation est celle hiérarchisée en cinq niveaux : Processus, Commande Locale, Supervision, Planification et Gestion Administrative, chacun représentant une partie de l'organisation industrielle avec des tâches particulières.

Ce chapitre vise à l'utilisation de modèles flous au niveau supervision, dans lequel la présence d'opérateurs humains est indispensable afin d'agir lorsqu'une dégradation de la performance du processus devient importante. Dans la première partie, on présente la perspective générale du problème de supervision et du diagnostic, comme tâche particulière ; dans la deuxième partie, on souligne l'intérêt de la modélisation flou dynamique dans les tâches qui concernent ce niveau.

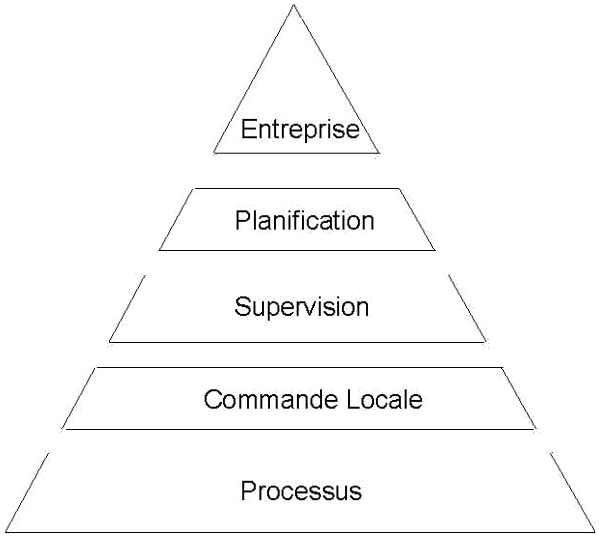

FIG. 7.1. La hiérarchie en automatisation integrée.

7.1 Le problème de supervision et de diagnostic

La partie commande et supervision, dans le cadre de l'automatisation integrée, sont des aspects très développés ces dernières années [33, 68, 69]. Ils sont basés, d'un côté, sur la proposition de modèles qui, en principe, sont une description mathématique des lois physiques décrivant les processus et, d'un autre côté, sur la proposition de schémas de commande qui, en général, supposent le système stationnaire autour d'un point de fonctionnement. Classiquement, l'ensemble modèle-commande est sujet à des hypotheses très restrictives sur le processus, modèles très raffinés, de bon fonctionnement des composants matériels, faisant l'hypothèse, souvent, de stationarité des procédés (d'invariance avec le temps). En cas de panne, la performance du processus peut être dégradée de façon importante et, en général, les sources des pannes ne sont pas modélisables de manière exhaustive.

Toutes ces considérations appellent au développement de systèmes de surveillance, de détection des fautes, de diagnostic et de prise de décision, tous en cohabitation au niveau supervision, dans l'architecture présentée dans la figure 7.1, où l'opérateur a la position hiérarchique la plus élevée. En fait, on parle de systèmes de supervision comme de systèmes d'aide à l'opérateur. Pour accomplir les tâches de supervision, comme celles de diagnostic et de prise de décision, on a besoin de l'analyse de telle information sur un intervalle de temps qui peut être très petit.

Bien que la surveillance des variables (état) du processus par des systèmes informatiques, dont l'interface graphique opérateur-système, est une tâche de base au niveau de la supervision, une surcharge d'information peut s'avérer néfaste s'il n'existe pas de systèmes de traitement de cette information pour aider l'opérateur à réagir au moment précis. D'ailleurs, les systèmes conventionnels de supervision sont plus des détecteurs d'alarmes que des systèmes de traitement d'alarmes réalisant l'analyse et le diagnostic d'une situation anormale.

Le but d'un système de supervision n'est pas de fonctionner à boucle fermée, pour remplacer l'opérateur, mais d'aider ce dernier à comprendre les données, ce qui conduit à disposer d'une capacité de raisonnement pour accomplir les tâches suivantes [70] :

– Synthèse de l'information.

- Traitement de l'information temporelle.
- Mise à jour des informations du processus arrivant au temps actuel.
- Traduction des données numériques, imprécises, incertaines en symboles interprétables par l'opérateur.

La performance d'un système de supervision peut être mesurée par sa capacité à distinguer entre fonctionnements normal et degradé afin de [71] :

- Redéfinir le point de consigne pour les algorithmes de commande au niveau local (régulateurs).
- Planifier la séquence d'opération sur le processus.
- Évaluer les conditions d'opération sur les installations des unités de production.
- Traiter les fautes : détection, diagnostic (localisation et caractérisation), démarrage des tâches de maintenance (préventives ou correctives).
- Prédire les comportements du processus selon les différents scénarios du moment.

D'un autre côté, le système de surveillance des variables, dont les tâches d'estimation d'état, permet de valider les résultats du raisonnement ci-dessus.

Les perspectives au niveau de la supervision sont très variées, sur les techniques informatiques ou les développements théoriques sur les modèles de raisonnement, dans le domaine de l'intelligence artificielle [72]. Plus en détail, on peut mentioner des perspectives de supervision au niveau [70] :

1. Des boucles de commande locales. C'est la tâche la plus simple et, partant de l'existence de bons régulateurs, elle est limitée à la validation des capteurs et/ou actionneurs, la redundance physique et analytique et la reconciliation des données pour obtenir une base de données valide. D'un autre côté, la tâche de diagnostic au niveau local s'appuie sur des méthodes statistiques ou sur des méthodes de l'intelligence artificielle telles que les systèmes experts temps réel, utilisant, par exemple, des outils comme la logique floue. Ces systèmes travaillent de manière synchrone avec le système d'acquisition.
2. La gestion des fautes au niveau global. Elle concerne une grande quantité d'information et doit suivre, de manière générale, les étapes d'un problème de diagnostic, figure 7.2 [73] :
   - Détecter un état anormal.
   - Localiser et caractériser la faute.
   - Trouver les causes.
   - Prendre des actions correctives au niveau de la commande et de la maintenance.

   Cependant, cette gestion n'est pas facile car il persiste des problèmes compliqués comme ceux liés aux caractéristiques des données. Ces données sont de nature variée : imprécises, incomplètes, non homogènes, dépendantes du contexte, temporelles. Cette dernière caractéristique "temporelle" assure une transition continue entre le fonctionnement normal et celui degradé au lieu d'une transition abrupte de style "saut". En conséquence, l'objectif global du système de supervision est d'aller de la reconnaissance des fautes à l'action.

On peut donc voir qu'il existe plusieurs types de raisonnement au niveau de la supervision qui visent à fournir à l'opérateur une aide au processus de détection-décision et on arrive, en conséquence, au besoin de modèles efficaces pour décrire toutes les situations possibles. On a dèjá mentionné l'usage de modèles mathématiques, très adéquats pour l'objectif de

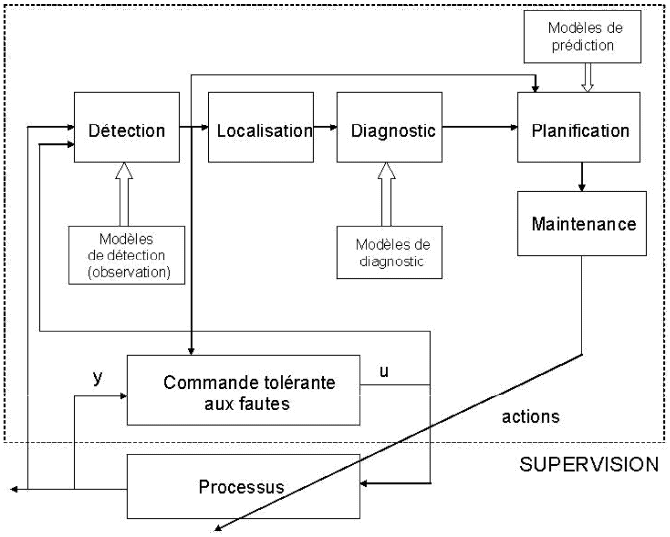

FIG. 7.2. Structure générale d'une commande supervisée.

commande ; mais, de tels modèles peuvent donner des résultats numériques sans qu'on puisse leur attribuer une valeur explicative (modèles de type "boîte noire", tels que les réseaux de neurones, par exemple). Ce fait devient un handicap au moment de développer le raisonnement. En ce sens, les systèmes basés sur des règles flous peuvent conduire à des modèles capables d'incorporer, en partie, un raisonnement semblable à celui des opérateurs humains.

7.2 Le MFAD dans la supervision et le diagnostic

Des architectures de supervision et diagnostic utilisant des modèles flous sont dèjá connues [74, 75, 76, 77], mais le point important est la proposition d'un bon modèle, ou modèles, permettant une vue globale des procédés. Un bon modèle doit [70] :

- Être bâti sur des informations ayant un sens physique.
- Rendre compte des phénomènes principaux sans être une transcription précise.
- Permettre de visualiser des informations non numériques.
- Être utilisable pour chacun des raisonnements associés aux différentes tâches du niveau supervision (voir section 7.1).
- Être basé sur la notion de causalité, afin de pouvoir remonter aux causes et déterminer les effets des phénomènes.
- Contenir une représentation explicite du temps pour savoir à quel moment un problème est susceptible d'arriver et déterminer le raisonnement approprié.

Quelques-unes de ces caractéristiques sont présentes dans les MFAD proposés. Partant de la définition donnée aux fonctions α, β et γ (voir equations (3.3), (3.5) et (3.7)), on n'a pas besoin de mettre à l'échelle les amplitudes originelles des variables d'entrée et de sortie, permettant, aussi, de préserver le sens physique des variables du modèle, tout en gardant des valeurs convenables pour les paramètres u^l, v_i^l et w_i^l (voir remarque 2). Par ailleurs, il est très connu qu'un modèle flou est composé d'un ensemble de règles linguistiques SI-ALORS dans lesquelles la notion de causalité est bien présente. Dans le cas du MFAD, cette notion est aussi incorporée dans la définition des fonctions α, β et γ : la variation des

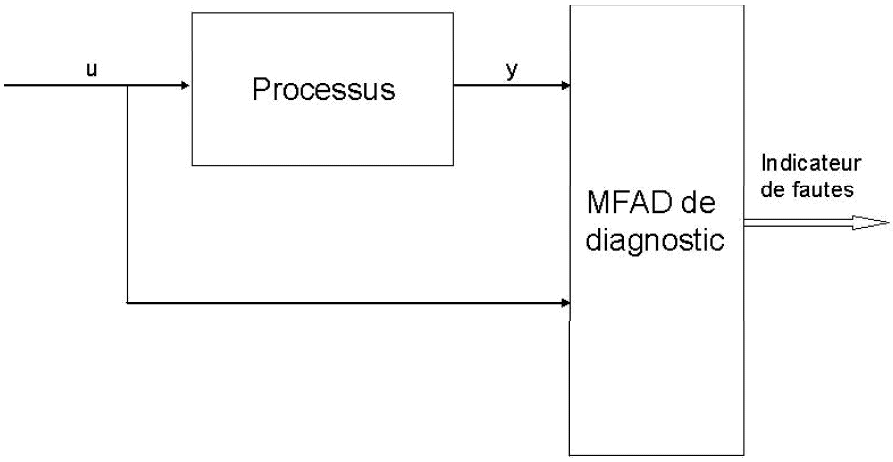

FIG. 7.3. Schéma de diagnostic via un MFAD.

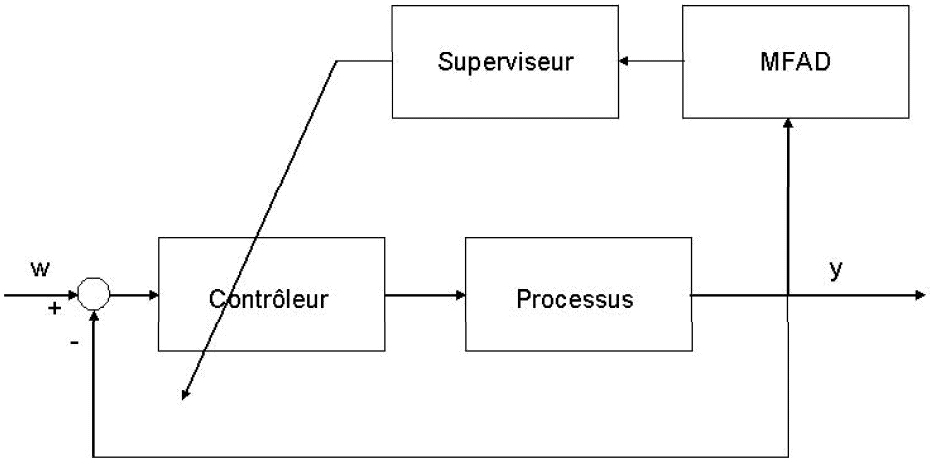

FIG. 7.4. Schéma de Supervision via un MFAD.

valeurs de la moyenne et de la variance des variables d'entrée du MFAD, visualisées dans α et β, sont associées à des variations sur les valeurs de γ. Finalement, bien qu'on n'ait pas une représentation explicite du temps dans le MFAD, la notion temporelle incorporée dans la définition des fonction α, β et γ permet d'avoir une mémoire temporelle en prenant en compte une quantité d'information au temps passé (voir à nouveau les équations (3.3), (3.5) et (3.7)).

A partir des caractéristiques susmentionnées, on peut se poser la question suivante : Comment peut-on utiliser le MFAD dans les tâches de supervision et diagnostic ? A priori, on peut dire que le MFAD peut être utilisé dans les schémas classiques pour ces tâches en donnant une idée sur l'évolution de la dynamique du processus, grâce à sa capacité à enregistrer une mémoire temporelle dans les fonctions α, β et γ, ce qui permet, au niveau de supervision, de prendre une décision. De cette manière, le MFAD peut être utilisé comme modèle de détection ou diagnostic, pour proposer des indicateurs de fautes (figure 7.3), et pour évaluer la performance du processus afin d'offrir une base d'information pour la prise de décision sur la commande au niveau de supervision (figure 7.4).

Dans ce qui suit, on reprend l'exemple 2 (fonction non-linéaire variant avec le temps) traitée dans la section 3.3 et le cas du processus de fermentation traité dans la section 6.3.

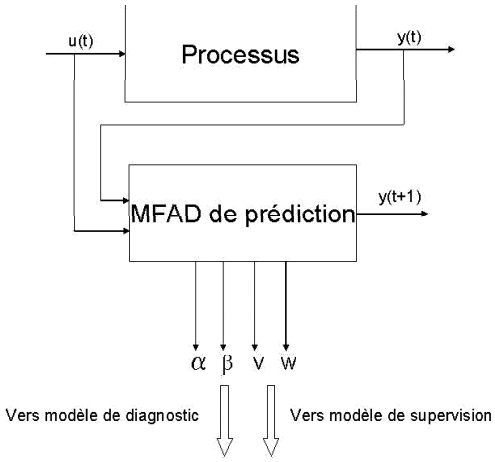

FIG. 7.5. Information contenue dans le MFAD comme identificateur.

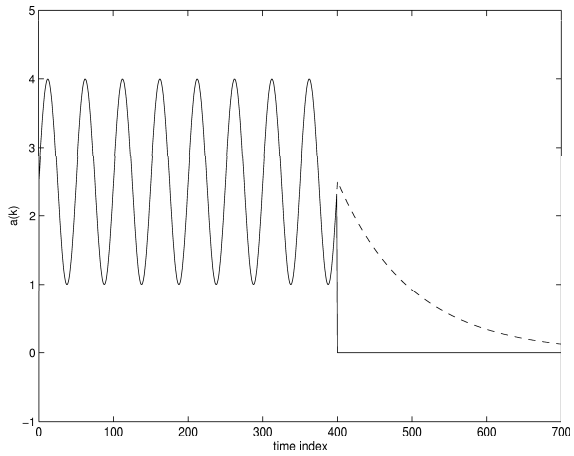

FIG. 7.6. Fautes sur le terme $a(k)$.

Dans les deux exemples, on simule les réponses à des perturbations différentes et on suit l'évolution des fonctions d'appartenance α et β, asociées aux variables d'entrée du MFAD dans le schéma d'identification (figure 7.5), afin de détecter l'existence d'une faute à partir de l'examen de la variation de ces ensembles flous. L'information contenue dans α et β peut être utilisée à des fins de diagnostic ou supervision, selon les schémas des figures 7.3 et 7.4.

Comme indicateur global, nous choisissons la *norme*, à instant d'échantillonnage, des matrices $\alpha(\mathbf{t})$ et $\beta(\mathbf{t})$. Le composant (i, j) de ces matrices correspond à la valeur de cette fonction associée à la variable x_i dans la règle j [1].

7.2.1 Cas 1 : Fonction variant avec le temps

Dan ce cas, on simule deux types de fautes après $k = 400$ (figure 7.6) : une abrupte (courbe continue) et une autre continue (courbe en pointillés), sur la valeur du terme $a(k)$ de la fonction. On rappelle, aussi, qu'il existe des changements sur la valeur de l'entrée à $k = 100$ et $k = 500$ (voir équation (2.5)).

[1] Ces matrices sont, en général, des matrices non carrées et on prend $norm(MM^T)$ où M désigne une matrice.

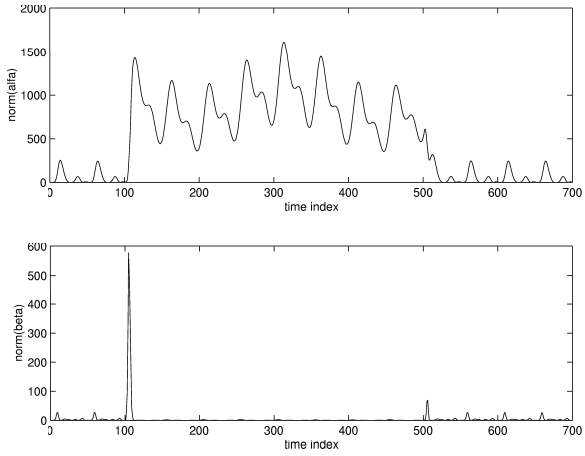

FIG. 7.7. Evolution de la norme des matrices α et β (Cas 1, condition normale).

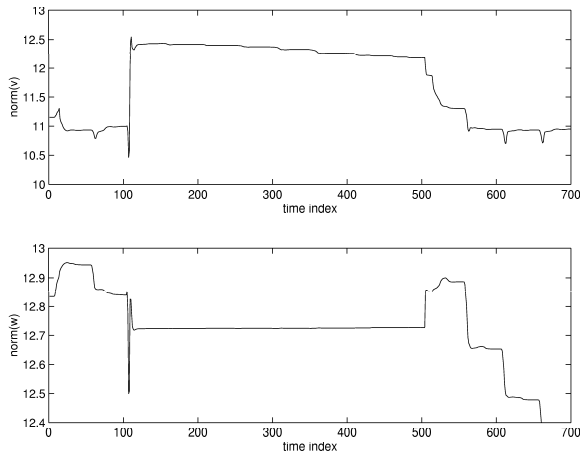

FIG. 7.8. Evolution de la norme des matrices $\mathbf{v}$ et $\mathbf{w}$ (Cas 1, condition normale).

On montre ci-dessous, l'évolution des normes des matrices retenues ; de plus, en vue de l'ajustement en ligne des paramètres v_i^l et w_i^l des fonctions α et β, on montre, aussi, l'évolution des normes associées aux matrices $\mathbf{v(t)}$ et $\mathbf{w(t)}$.

D'abord, on présente dans les figures 7.7 et 7.8, l'évolution des fonctions d'appartenance et paramètres sous des "conditions normales", c'est à dire sans faute.

Ensuite, on visualise l'évolution des normes dans le cas de la faute abrupte (figures 7.9 et 7.10), et dans le cas de la faute continue (figures 7.11 et 7.12).

Par rapport au fonctionnement normal, on peut faire les remarques le suivantes :

- Dans le cas de la faute abrupte, dont le changement de l'entrée, il existe des variations importante tant de l'évolution des normes des matrices α et β que de celles des paramètres :
  - L'évolution des paramètres est plus sensible à ces changements : on peut les visualiser dans la figure 7.10, justement à $k = 100$ et $k = 500$, pour le changement de l'entrée, et à $k = 400$ pour la faute abrupte.
  - Bien que l'on puisse espérer des changements au niveau des fonctions α et β (on rappelle qu'elles sont dépendantes de la moyenne et la variance), on ne visualise des variations importantes sur l'évolution de β, que pour le changement à l'entrée (voir

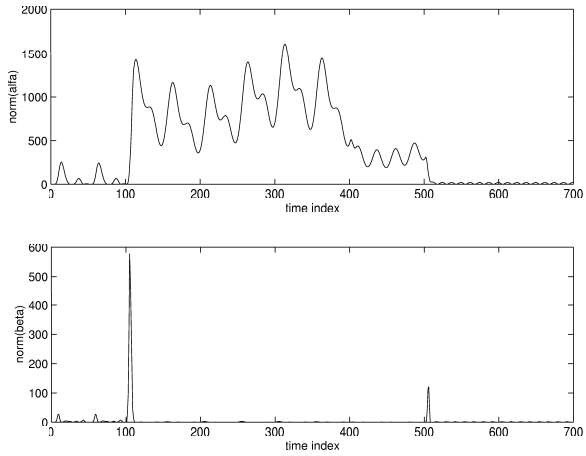

FIG. 7.9. Evolution de la norme des matrices α et β (Cas 1, faute abrupte).

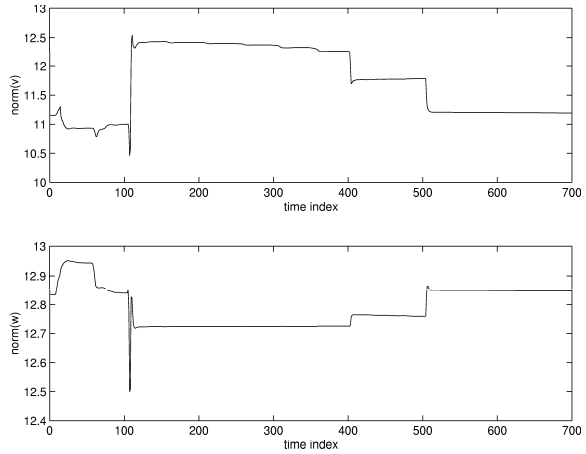

FIG. 7.10. Evolution de la norme des matrices $\mathbf{v}$ et $\mathbf{w}$ (Cas 1, faute abrupte).

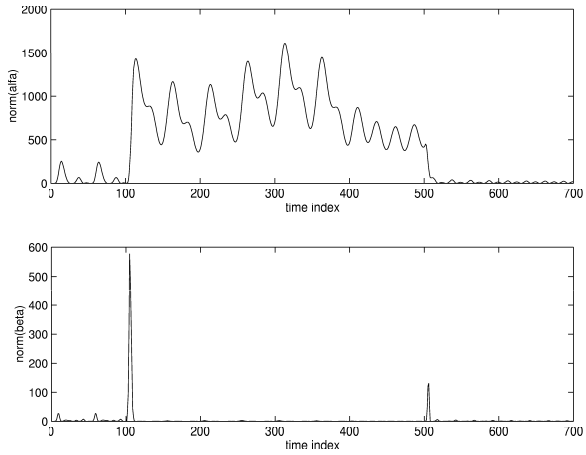

FIG. 7.11. Evolution de la norme des matrices α et β (Cas 1, faute continue).

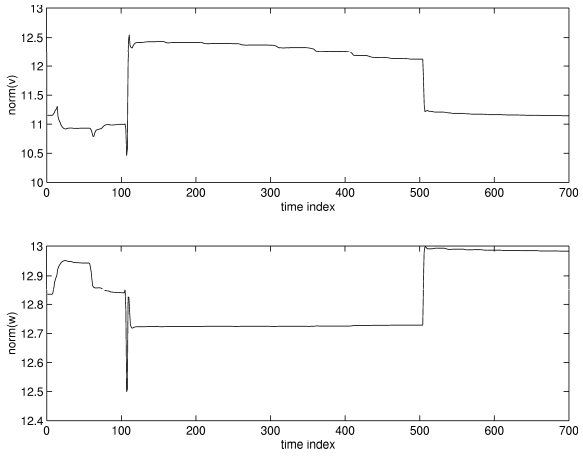

FIG. 7.12. Evolution de la norme des matrices **v** et **w** (Cas 1, faute continue).

figure 7.9).
  - Par contre, l'évolution de la fonction α est perturbée de façon importante tant pour le changement à l'entrée que pour la faute abrupte.
- Dans le cas de la faute continue, il existe une évolution différente des paramètres après $k = 500$, changement à l'entrée et faute continue, par rapport aux cas de fonctionnement normal et de faute abrupte. On rappelle que la faute continue devient faute abrupte autour $k = 700$; on a donc des informations différentes entre $k = 400$ et $k = 700$, ce qui peut aider à spécifier le type de perturbation sur le système (interne ou à l'entrée).
- Au niveau de l'évolution de α, on aperçoit un changement différent entre $k = 400$ et $k = 500$, par rapport à celui obtenu pour la faute abrupte, cet intervalle étant celui pendant lequel commence la dégradation du fonctionnement du système, à cause d'une faute.

Sur la base de ces résultats, on observe qu'il est possible d'obtenir des informations sur le fonctionnement du système à partir de l'évolution tant des fonctions α que des paramètres **v** et **w**, lorsqu'ils sont ajustés en ligne. Cette information peut être analysée par un classifieur de fautes en utilisant le schéma de la figure 7.3 où le classifieur peut être un MFAD. Soulignons que, dans ce cas, l'évolution de la fonction β n'est pas sensible à la faute (perturbation interne).

7.2.2 Cas 2 : Processus de fermentation

On reprend le cas de la commande prédictive d'un processus de fermentation. Ici, on considère un changement abrupt de débit du gaz Ψ_g après $t = 100$, ce qui diminue la valeur du facteur de friction de la vanne K_f (voir équation (6.11)). Considérons l'évolution des fonctions α et β (on rappelle que les paramètres v_i^l et w_i^l sont ajustés hors ligne).

On présente, dans la figure 7.13, la performance du système de commande avec une entrée de référence constante. On montre, ensuite, cette performance, lorsqu'on applique la perturbation (figure 7.14), et l'évolution des normes des matrices α et β (figures 7.15 et 7.16) [2].

[2]La période d'échantillonnage est de 5 sec.

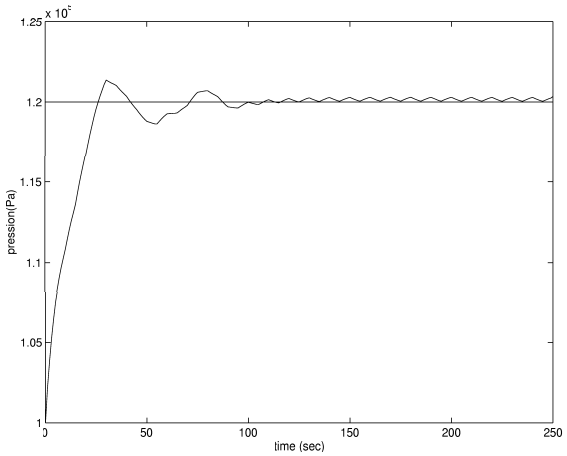

FIG. 7.13. Evolution de la sortie commandée (Cas 2, sans faute).

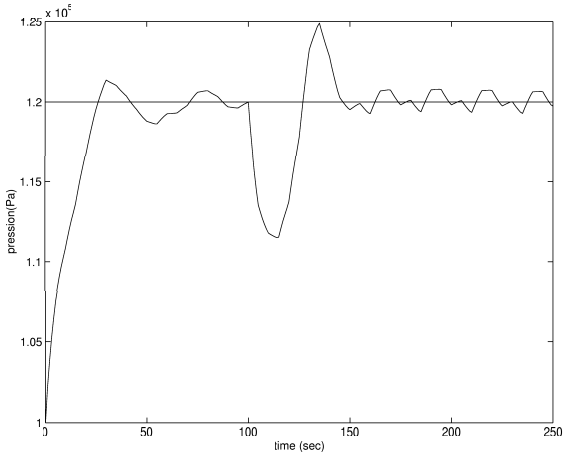

FIG. 7.14. Evolution de la sortie commandée (Cas 2, avec faute).

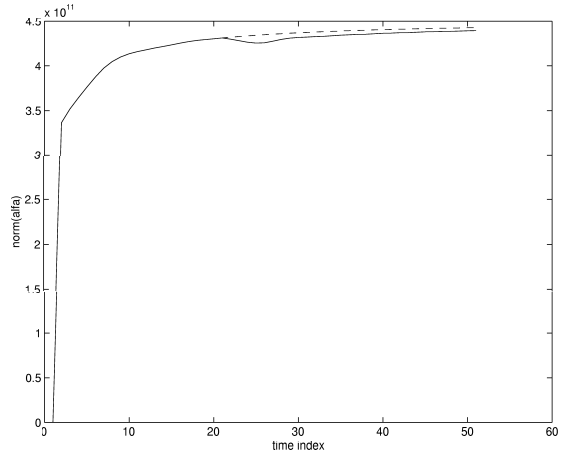

FIG. 7.15. Evolution de la norme de la matrice α (Cas 2, avec et sans faute).

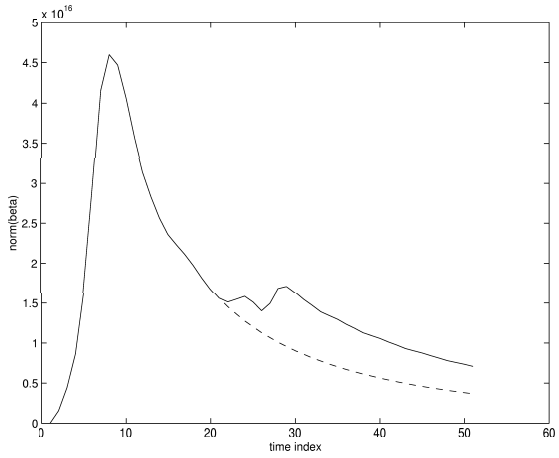

FIG. 7.16. Evolution de la norme de la matrice β (Cas 2, avec et sans faute).

Dans ces deux dernières figures, la ligne en pointillés montre le comportement pour le fonctionnement normal et la ligne en continue pour le cas de faute. On visualise une variation importante de la norme de la matrice β ; par contre, il n'existe pas d'une variation considérable de l'évolution de la matrice α. Cela donne de l'importance à l'utilisation d'un ajustement en ligne des paramètres de MFAD, pour obtenir plus d'information pertinente pour le niveau de diagnostic.

On remarque, à nouveau, la performance du schéma de commande prédictive utilisant le MFAD comme identificateur. La performance de la commande dans le cas de faute est dégradée par rapport à celle obtenue lors d'un fonctionnement normal ; on peut donc utiliser l'information extraite des fonctions d'appartenance lors des tâches de supervision afin d'ajuster les paramètres de la commande ; en conséquence, il semble important de développer des approaches dans ce sens.

7.3 Conclusion

Dans ce chapitre, on a présenté une vision générale du problème de supervision et diagnostic et on a remarqué que le MFAD a de caractéristiques intéressantes pouvant être explotées dans ces tâches de niveau supérieur. En particulier, on s'est aperçu que, en présence de perturbations ou de fautes dans le système, les variations trouvées tant dans l'évolution temporelle des fonctions d'appartenance que dans celle des paramètres ajustés en ligne, donnent des informations soit pour la détection/le diagnostic, soit pour l'ajustement des contrôleurs au niveau supervision.

Dans le chapitre suivant, on souligne les aspects les plus importants trouvés dans cette deuxième partie, lors de l'utilisation du MFAD et de l'AR dans des tâches de niveau supérieur dans une structure générale d'automatisation integrée.

Chapitre 8

Conclusions de la partie II

L'obtention de modèles pour l'identification des systèmes pouvant offrir des informations importantes pour la synthèse des tâches de commande et supervision, est un pas crucial dans le domaine de la commande des processus. Dans cette partie, on a présenté, d'un côté, l'utilisation du MFAD pour la prédiction de la sortie à un pas d'un processus de fermentation, dans un schéma de commande prédictive ; d'un autre côté, on a discuté de l'utilisation de l'information contenue dans le MFAD comme modèle de prédiction, dans les tâches de diagnostic et supervision.

1. **Conclusions sur l'application à la commande prédictive**

   Pour ce qui est de la commande prédictive, on a remarqué la performance adéquate du MFAD pour l'identification d'un processus invariant, en utilisant une modification de la définition de la moyenne et de la variance associées aux variables d'entrée du modèle flou. Au vu de la caractéristique invariante du système, on n'a pas besoin de calculer la moyenne et la variance sur une fenêtre de temps, mais de les calculer de façon récursive, en intégrant chaque donnée arrivant au temps courant pour le calcul de la valeur actuelle de la moyenne et de la variance. Cela conduit à proposer des fonctions d'appartenance statiques en absence de perturbations sur le système réel (on peut parler de "régime permanent" des fonctions d'appartenance).

   On a montré, aussi, dans cette partie d'applications, comment utiliser l'apprentissage par renforcement pour traiter un problème d'optimisation : l'amélioration du taux d'apprentissage pour le calcul de la commande prédictive, en formulant un problème de programmation non linéaire avec une fonction objectif dépendant d'un renforcement.

   Bien que les simulations aient montré une performance adéquate du schéma de commande prédictive en utilisant l'apprentissage par renforcement et le modéle flou dynamique, il faut fournir des conditions additionnelles pour garantir la stabilité en boucle fermée du système de commande, par exemple en utilisant des fonctions de Lyapunov contenant une fonction d'erreur entre la sortie du processus commandé et la sortie de référence.

2. **Conclusion sur l'application au diagnostic et la supervision**

   Pour le diagnostic et la supervision, remarquons, à nouveau, l'intérêt d'avoir un modèle avec les propriétés du MFAD qui :

– Préserve le sens physique des variables du modèle.
– Conserve la notion de causalité.
– Intègre la notion du temps.

Les cas présentés montrent qu'il est possible d'utiliser l'information contenue dans l'évolution temporelle des normes associées aux matrices des fonctions d'appartenance des variables d'entrée du MFAD, pour détecter l'existence de perturbations ou fautes sur le système réel. Cette information peut être analysée par un autre modèle (flou, flou-dynamique, non-flou) afin de donner des indicateurs de fautes (diagnostic) ou proposer des changements sur le contrôleur (supervision). De plus, si les paramètres des fonctions d'appartenance son ajustés en ligne (on a utilisé l'AR pour cet objectif), l'évolution des normes associées aux matrices des paramètres peut aussi être utilisée, avec celle des fonctions d'appartenance, pour les fonctions de diagnostic ou supervision.

Remarquons, ici, que nous avons seulement donné des idées sur l'utilisation du MFAD comme modèle de diagnostic ou supervision et nous avons profité des données obtenues d'un autre MFAD d'identification pour avoir plus d'information sur l'évolution du système. Il reste à proposer un MFAD de diagnostic, ou supervision, utilisant cette information.

CONCLUSION GÉNÉRALE

Ce travail de recherche a consisté en la proposition d'un modèle flou avec des fonctions d'appartenance dynamiques, à paramètres ajustables en ligne, par un algorithme basé sur l'Apprentissage par Renforcement (AR). Nous avons été motivée par deux aspects importants sur la modélisation floue :

- L'incorporation de la variable "temps" dans les modèles flous.
- La minimisation de la possibilité d'obtenir des ensembles flous disjoints, problème sensible dans les approches adaptatives de modélisation floue.

L'approche présentée dans ce travail prend en compte la dynamique des variables du système en introduisant, dans les fonctions d'appartenance d'un modèle flou, la valeur moyenne et la variance des variables d'entrée et de sortie du modèle au temps t. De cette manière, les ensembles flous se déplacent sur le domaine de discours des variables, en fonction des valeurs de la moyenne et de la variance échantillonnées. La propriété dynamique du modèle flou proposé est un atout pour l'identification de systèmes variant avec le temps, par exemple.

La modélisation floue temporelle et adaptative

L'obtention de modèles satisfaisants, à partir de données d'observation (problème d'identification), pour la caractérisation d'un système, est vraiment une tâche délicate qui va de la sélection d'une structure du modèle la plus adaptée au problème (systèmes linéaires, non-linéaires) jusqu'à trouver une méthode d'ajustement des paramètres du modèle capable d'extraire l'information contenue dans les données disponibles, afin de capter les propriétés essentielles du système dans son modèle.

Lors de la recherche de structures non-linéaires pour l'identification des systèmes, la théorie de la Logique Floue permet de proposer des modèles linguistiques basés sur une partition linguistique des valeurs de ses variables, valeurs décrites par des ensembles flous caractérisés par une fonction d'appartenance ; l'ajustement d'un modèle flou, via les paramètres de ses fonctions d'appartenance, est une tâche d'identification. Ainsi, la recherche de méthodes pour le développement d'une représentation précise des processus réels est un aspect important de la modélisation floue.

Bien que les modèles flous soient très utilisés, depuis plusieurs années, pour la modélisation des systèmes dynamiques, la structure de ces modèles est plutt "statique" et la variable "temps", dans quelques travaux seulement, a été incorporé comme variable d'entrée du modèle, mais pas comme une variable interne permettant de guider l'évolution dynamique du modèle.

D'un autre côté, on s'est attaché à proposer des structures améliorant l'activation des règles floues lorsqu'on utilise des algorithmes d'ajustement de paramètres. On a donc été conduite à s'intéresser aux algorithmes d'ajustement des paramètres. Dans ce sens, on a pensé le problème d'identification flou, visant à l'ajustement de paramètres du modèle, comme un problème d'AR, avec l'objectif d'arriver à des algorithmes utilisables en ligne.

Dans cette problématique temporelle et adaptative liée à la modélisation floue, nous pouvons préciser les deux objectifs principaux de ce travail : d'une part, l'incorporation de la variable "temps" dans les fonctions d'appartenance du modèle flou, partie fondamentale pour proposer un modèle précis du système et, d'autre part, la proposition d'un algorithme d'apprentissage en ligne pour l'ajustement des paramètres de ces fonctions d'appartenance dynamiques.

Notre contribution

Dans ce travail de recherche, nous avons proposé des approches originales par :

1. l'incorporation de fonctions d'appartenance dynamiques dans un modèle flou adaptatif, en proposant un Modèle Flou Adaptatif Dynamique (MFAD).
2. l'utilisation d'un algorithme d'apprentissage en ligne pour l'ajustement des paramètres du modèle flou basé sur l'AR.

En ce qui concerne le premier point, des fonctions paramétriques, qui déterminent les fonctions d'appartenance dynamiques, sont définies à partir de la moyenne et de la variance échantillonnées des valeurs disponibles, au temps t, des variables du système. Ces fonctions d'appartenance dynamiques sont incorporées dans un modèle flou de type Mamdani dont on peut obtenir une expression analytique, ce qui permet d'utiliser des algorithmes d'apprentissage pour l'ajustement de ces paramètres.

Sur le deuxième point, l'algorithme d'ajustement proposé repose sur l'utilisation de la méthode des Différences Temporelles pour la résolution de problèmes d'AR comme un problème de prédiction. On a proposé une fonction de prédiction non linéaire introduite de manière naturelle afin d'ajuster les paramètres du MFAD. L'agent d'apprentissage est utilisé pour prédire la performance du MFAD dans l'identification floue du système à modéliser. De cette manière, la fonction de prédiction est définie comme une fonction dépendant de l'erreur d'identification. La définition de cette fonction de prédiction conduit, aussi, à la définition d'un signal de renforcement en concordance avec la tâche d'identification à développer.

La performance du MFAD avec l'algorithme d'apprentissage en ligne, basé sur l'AR, pour l'identification des fonctions non-linéaires variants avec le temps a été évaluée par comparaison avec les résultats obtenus en utilisant un modèle flou adaptatif classique : on a remarqué que le MFAD montre une meilleure performance quant au nombre de règles utilisées, à la valeur de l'erreur d'identification, tant dans la phase d'apprentissage que dans celle de validation, et à la performance en présence de perturbations.

Après cette démarche algorithmique, d'aucuns pourraient se poser des questions sur l'utilisation de la connaissance experte dans les modèles obtenues et sur leur lisibilité- interprétabilité. Ces points, sur lesquels nous revenons dans notre prospective, n'ont pas été occultés mais, pour le moment, simplement abordés à 2 niveaux :

– initialisation, par l'expert, du système d'inférence floue à optimiser.

– validation, par l'expert, du modèle obtenue.

La capacité de prédiction du MFAD a été mise en valeur dans un problème de commande prédictive d'un processus de fermentation. Au vu de la caractéristique invariante du système, on a proposé une modification de la définition de la moyenne et de la variance associées aux variables d'entrée du modèle flou. Les résultats obtenus mettent en relief la bonne performance du MFAD pour l'identification de ce processus invariant.

De plus, on a montré, dans cette partie "Applications", comment utiliser l'AR pour traiter un problème d'optimisation en améliorant le taux d'apprentissage, pour le calcul de la commande prédictive, en formulant un problème de programmation non linéaire avec une fonction objectif dépendant d'un renforcement. Les simulations ont montré une performance correcte du schéma de commande prédictive utilisant l'apprentissage par renforcement et le MFAD.

Finalement, on a discuté de l'utilisation de l'information contenue dans le MFAD comme modèle de prédiction, dans les tâches de diagnostic et supervision : il est possible d'utiliser l'information contenue dans l'évolution temporelle des normes associées aux matrices des paramètres des fonctions d'appartenance des variables d'entrée du MFAD, pour détecter l'existence de perturbations ou de fautes sur le système réel.

Perspectives

Les résultats obtenus nous suggère des pistes à explorer pour la continuation de ce travail de recherche :

1. Au sujet de la modélisation floue.

   – La définition de fonctions d'appartenance dynamiques du type triangulaire.

     Dans ce travail, nous nous sommes focalisée sur les fonctions d'appartenance de type gaussien ; cependant, on peut proposer, aussi, une structure dynamique pour des fonctions d'appartenance triangulaires en utilisant la valeur moyenne échantillonnée pour définir la valeur modale des fonctions d'appartenance et la variance pour proposer une expression définissant la base de ces fonctions d'appartenance triangulaires. L'utilisation de ces fonctions d'appartenance est très connue pour la proposition d'un type de contrôleur flou [8].

   – L'introduction de fonctions d'appartenance dynamiques dans les modèles du type TSK.

     Nous avons traité, ici, les modèles du type Mamdani, mais l'incorporation de fonctions d'appartenance dynamiques dans les modèles du type TSK pourrait être validée facilement en utilisant l'algorithme d'ajustement de paramètres en ligne proposé. Rappelons qu'une expression analytique est aussi disponible pour ce type de modèle. Rappelons que ce type de modèle est très utilisé pour la proposition de contrôleurs et d'observateurs flous [78].

- Dans ce travail, nous avons retenu, comme critère d'évaluation des méthodes, le critère d'erreur quadratique, critère facile et couramment utilisé, tout à fait justifié, donc, dans ce travail, mais qui doit être élargi dans un travail complémentaire proche, en intégrant les aspects suivants :
  - robustesse du modèle,
  - généricité du modèle,
  - interprétabilité du modèle,
  - compromis précision-interprétabilité du modèle

2. Au sujet des applications au diagnostic et à la supervision.

   Nous avons envisagé l'utilisation du MFAD comme modèle de supervision et de diagnostic. L'information contenue dans le MDAF d'identification pourrait être utilisée comme variable d'entrée d'un MFAD de diagnostic ou supervision. Il reste à valider la performance de ce modèle dans un problème concret.

Bibliographie

[1] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8 :338–353, 1965.

[2] J.S Roger Jang. Anfis : Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics*, 23 :665–685, 1993.

[3] C.J. Lin. An art-based fuzzy adaptive learning control network. *IEEE Transactions on Systems, Man and Cybernetics*, 5 :477–496, 1997.

[4] R.R. Yager. Implementing fuzzy logic controller using a neural network. *Fuzzy Sets and Systems*, 48 :53–64, 1992.

[5] P.J. Werbos. Neural control and fuzzy logic : connections and designs. *Int. Journal Approx. Reasoning*, 6 :185–219, 1992.

[6] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modelling and control. *IEEE Transactions on Systems, Man and Cybernetic*, 15 :116–132, 1985.

[7] L.X. Wang. *Adaptive Fuzzy Systems and Control*. Prentice Hall, New Jersey, 1994.

[8] R. Yager and D. Filev. *Essentials of Fuzzy Modelling and Control*. John Wiley, New York, 1994.

[9] Y. Shi and M. Mizumoto. Some considerations on conventional neuro-fuzzy learning algorithms by gradient descent method. *Fuzzy Sets and Systems*, 112 :51–63, 2000.

[10] C.C. Lee. Fuzzy logic in control systems : Fuzzy logic controller-part i. *IEEE Transactions on Systems, Man and Cybernetics*, 20 :404–418, 1990.

[11] C.C. Lee. Fuzzy logic in control systems : Fuzzy logic controller-part ii. *IEEE Transactions on Systems, Man and Cybernetics*, 20 :419–435, 1990.

[12] J. Virant and N. Zimic. Attention to time in fuzzy logic. *Fuzzy Sets and Systems*, 82 :39–49, 1996.

[13] L. Ljung. *System Identification. Theory for the User*. Prentice Hall, New York, 1997.

[14] Y. Shi, M. Mizumoto, N. Yubazaki, and M. Otani. A learning algorithm for tuning fuzzy rules based on the gradient descent method. *Proceedings of the 5th. IEEE International Conference on Fuzzy Systems*, 1 :55–61, 1996.

[15] L.X. Wang and J.M. Mendel. Back-propagation fuzzy systems as nonlinear dynamic sytem identifiers. *Proceedings of the 4th. IEEE International Conference on Fuzzy Systems*, pages 1409–1416, 1992.

[16] S. Horikawa, T. Furuhashi, and Y. Uchikawa. On fuzzy modelling using fuzzy neural networks with the back-propagation algorithm. *IEEE Transactions on Neural Networks*, 3(5) :801–806, 1992.

[17] R.S. Sutton. Temporal credit assignment in reinforcement learning. PhD. Thesis. University of Massachusetts, 1984.

[18] R.S. Sutton and A.G. Barto. *Reinforcement Learning. An Introduction.* The MIT Press, Cambridge, 1998.

[19] R. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3 :9–44, 1988.

[20] R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8 :229–256, 1992.

[21] S. Miller and R.J. Williams. Temporal difference learning : A chemical process control application. In A.F. Murray, editor, *Applications of Artificial Neural Networks.* Kluwer, Norwell, 1995.

[22] G. Tesauro. Temporal difference learning and td-gammon. *Communications of the Association for Computing Machinery*, 38(3) :58–68, 1995.

[23] L. Pack Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning : A survey. *Journal of Artificial Intelligence Research*, 4 :237–285, 1996.

[24] S.P. Singh and R.S. Sutton. Reinforcement learning with replacing eligibility traces. *Machine Learning*, 22 :123–158, 1996.

[25] R.E. Schapire and M.K. Warmuth. On the worst-case analysis of temporal difference learning algorithmes. *Machine Learning*, 22 :95–121, 1996.

[26] W.M. van Buijtenen, G. Schram, R. Babuska, and H.B. Verbruggen. Adaptive fuzzy control of satellite attitude by reinforcement learning. *IEEE Transactions on Fuzzy Systems*, 6(2) :185–194, 1998.

[27] W. Zhang and T.G. Dietterich. Solving combinatorial optimization tasks by reinforcement learning : A general methodology applied to resource-constrained scheduling. *Journal of Artificial Intelligence Research*, 1, 2000.

[28] J. Si and Y-T. Wang. On line learning control by association and reinforcement. *IEEE Transactions on Neural Networks*, 12(2) :264–276, 2001.

[29] L. Chen and G. Chen. Fuzzy predictive control of uncertain chaotic systems using time series. *International Journal of Bifurcation and Chaos*, 9(4) :757–767, 1999.

[30] S. Marsilli-Libelli and L. Giunti. Fuzzy predictive control for nitrogen removal in biological wastewater treatment. *Water Science and Technology*, 45(4) :37–44, 2002.

[31] R. Babuska, M. Ayala-Botto, J. S da Costa, and H.B Verbruggen. Neural and fuzzy modelling in nonlinear predictive control. *Computational Engineering in Systems Applications*, pages 9–12, 1996.

[32] H. Rake. Step response and frequency response methods. *Automatica*, 16 :519–526, 1980.

[33] N. Nise. *Control Systems Engineering.* Benjamin Cummings, California, 1992.

[34] L. Ljung and T. Glad. *Modeling of Dynamic Systems.* Prentice Hall, New Jersey, 1994.

[35] J.A. Freeman and D.M. Skapura. *Neural Networks.Algorithms, Application and Programming Techniques.* Addison-Wesley, USA, 1992.

[36] E. Colina-Morles and F. Rivas-Echeverría. Redes neuronales artificiales. In J. Aguilar-Castro and F. Rivas-Echeverría, editors, *Introducción a las Técnicas de Computación Inteligente*, chapter 4, pages 103–173. Meritec, Mérida, 2001.

[37] M. Cerrada-Lozada and V. Rodríguez-Graterol. Lógica difusa. In J. Aguilar-Castro and F. Rivas-Echeverría, editors, *Introducción a las Técnicas de Computación Inteligente*, chapter 3, pages 53–101. Meritec, Mérida, 2001.

[38] D. Dubois and H. Prade. Processing fuzzy temporal knowledge. *IEEE Transactions on Systems, Man and Cybernetics*, 19(4) :729–744, 1989.

[39] D. Qian. Representation and use of imprecise temporal knowledge in dynamic systems. *Fuzzy Sets and Systems*, 50 :59–77, 1992.

[40] L.A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man and Cybernetics*, 3 :28–44, 1973.

[41] R.M. Tong. The construction and evaluation of fuzzy models. In M.M. Gupta, R.K. Ragade, and R.R. Yager, editors, *Advances in Fuzzy Set Theory and Applications*, pages 559–576. North-Holland, Masachussets, 1979.

[42] M.M. Gupta, J.B. Kiszka, and G.J. Trojan. Multivariable structure of fuzzy control systems. *IEEE Transactions on Systems, Man and Cybernetics*, 16 :638–656, 1986.

[43] W. Pedrycz. *Fuzzy Control and Fuzzy Systems*. Wiley, New York, 1989.

[44] M. Sugeno and T. Yasukawa. A fuzzy logic-based approach to qualitative modelling. *IEEE Transactions on Fuzzy Systems*, 1 :7–31, 1993.

[45] M. Ma, Y. Zhang, G. Langholz, and A. Kandel. On direct construction of fuzzy systems. *Fuzzy Sets and Systems*, 112 :165–171, 2000.

[46] Y. Shi and M. Mizumoto. A new approach of neuro-fuzzy learning algorithm for tuning fuzzy rules. *Fuzzy Sets and Systems*, 112 :99–116, 2000.

[47] K.S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1 :4–27, 1990.

[48] J.M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, Harlow, 2002.

[49] E.F. Camacho and C. Bordons. *Model Predictive Control in the Process Industry*. Springer-Verlag, London, 1995.

[50] A.I. Propoi. Use of linear programming methods for synthesizing sample-data automatic systems. *Automation and Remote Control*, 24 :837–844, 1963.

[51] C.E. Garcia, D.M. Prett, and M. Morari. Model predictive control : theory and practice - a survey. *Automatica*, 25 :335–348, 1989.

[52] M.A. Henson. Nonlinear model predictive control : current status and future directions. *Computational Chemical Engineering*, 23 :187–202, 1998.

[53] M. Morari. *Advances in Model-Based Predictive Control*. Oxford University Press, Oxford, 1994.

[54] B. Kouvaritakis, J.A. Rossiter, and A.O.T. Chang. Stable generalized predictive control : an algorithm with guaranteed stability. *Proceedings of the IEE. Part D*, 139(4) :349–362, 1992.

[55] J. Rawlings and K. Muske. The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38 :1512–1516, 1993.

[56] D.Q. Maine, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control : stability and optimality. *Automatica*, 36 :789–814, 2000.

[57] P.J. Campo and M. Morari. Robust model predictive control. *Proceedings of the American Control Conference*, pages 1021–1026, 1987.

[58] J.A Roubos, S. Mollov, R. Babuska, and H.B Verbruggen. Fuzzy model-based predictive control using takagi-sugeno models. *International Journal of Approximate Reasoning*, 22 :3–30, 1999.

[59] M. Ayala Botto, J.J Van der Boom, A. Krijgsman, and J.S da Costa. Predictive control based on neural networks models with i/o feedback linearization. *International Journal of Control*, 72(17) :1538–1554, 1999.

[60] D.W. Clarke, C. Mohtadi, and P.S. Tuffs. Generalized predictive control. part i. the basic algorithm. *Automatica*, 23(2) :137–148, 1987.

[61] C. Onnen, R. Babuska, U. Kaymak, J.M. Sousa, H.B Verbruggen, and R. Isermann. Genetic algorithms for optimization in predictive control. *Chemical Engineering Practice*, 5(10) :1363–1372, 1997.

[62] J. Tang and D. Wang. A new genetic algorithm for nonlinear programming problems. *Proceedings of the 36th IEEE CDC*, pages 4906–4907, 1997.

[63] H.J.L. Van Caan, H.A. Te Braake, C. Hellinga, A.J. Krigsman, H.B Verbruggen, K.Ch. Luyben, and J.J. Heijnen. Design and real time testing of a neural model predictive control for a nonlinear system. *Chemical Engineering Science*, 50(15) :2419–2430, 1995.

[64] M. Cerrada, J. Aguilar, E. Colina, and A. Titli. An approach for dynamical adaptive fuzzy modeling. *Proceedings of the IEEE-FUZZ 2002 International Conference on Fuzzy Systems*, pages 156–161, 2002.

[65] M. Cerrada, J. Aguilar, E. Colina, and A. Titli. Dynamical adaptive fuzzy systems : an application on system identification. *Proceedings of the 15th Triennal IFAC World Congress*, pages 1006–1011, 2002.

[66] M. Cerrada, J. Aguilar, E. Colina, and A. Titli. Time-varying non linear system identification using dynamical adaptive fuzzy systems. *Revista Técnica de Ingeniera de la Universidad del Zulia*, 25(3) :171–180, 2002.

[67] T.J. Williams, P. Bernous, J. Brosvic, D. Chen, G. Doumeingts, and et al. Architectures for integrating manufacturing activities and enterprises. *Computer in industry*, 24 :11–139, 1994.

[68] M. Krstic, I. Kanellakopoulos, and P. Kokotovic. *Nonlinear and Adaptive Control Design*. John Wiley, New York, 1995.

[69] J.O. Moody and P.J. Antsaklis. Supervisory control using computationally efficient linear techniques : A tutorial introduction. *Proceedings of the 5th IEEE Mediterranean Conference on Control and Systems*, MP1, 1997.

[70] S. Gentil. Systèmes d'aide à la supervision. In N. Rakoto-Ravalontsalana and J. Aguilar-Martin, editors, *Supervision de processus à l'aide du système expert G2*, pages 7–20. Hermès, Paris, 1995.

[71] J. Aguilar, M. Cerrada, G. Mousalli, and F. Rivas. Application of the agent reference model for intelligent distributed control system. In N. Mastorakis and L. Pecorelli-Peres, editors, *Advances in System Sciences : Measurements, Circuits and Control*, pages 204–210. World Scientific Engineering Society WSES Press, USA, 2001.

[72] D. Dubois and S. Gentil. Intelligence artificielle et automatique. *Revue d'intelligence Artificielle*, 8(1) :7–27, 1994.

[73] M. Cerrada, J. Aguilar, J. Cardillo, and R. Faneite. Agent-based reference model for fault management. Technical Report, Grant project I-621-98-02-A, CDCHT-University of Los Andes, Venezuela, 2002.

[74] F.J. Uppal, R.J. Patton, and V. Palade. Neuro-fuzzy based fault diagnosis applied to an electro-pneumatic valve. *Proceedings of the European Symposium on Artificial Neural Networks*, pages 505–506, 2002.

[75] J. Jantzen. Fuzzy supervisory control. Technical Report, No.98-H-875, Technical University of Denmark, 1998.

[76] A. Fink, M. Fischer, O. Nelles, and R. Iserman. Supervision of nonlinear adaptive controllers based on fuzzy models. *Control Engineering Practice*, 8(10) :1093–1105, 2000.

[77] A. Pu nal, J. Rodriguez, A. Franco, E.F. Carrasco, and E. Roca. Advanced monitoring and control of anaerobic wastewater treatment plants : diagnosis and supervision by a fuzzy-based expert systems. *Water Science & Technology*, 43(7) :191–198, 2001.

[78] K. Tanaka and H.O. Wang. Fuzzy regulators and fuzzy observers : a linear matrix inequality approach. *Proceedings of the 36th IEEE CDC*, pages 1315–1320, 1997.

Thèse INSA, École Doctorale Systèmes, Spécialité Systèmes Automatiques

Titre : SUR LES MODÈLES FLOUS ADAPTATIFS DYNAMIQUES

Résumé :

La contribution principale de ce travail de recherche est la proposition d'un modèle flou avec des fonctions d'appartenance dynamiques à paramètres ajustables en ligne, par un algorithme basé sur l'Apprentissage par Renforcement (AR). L'approche présentée prend en compte la dynamique des variables du système en introduisant, dans les fonctions d'appartenance d'un modèle flou, la valeur moyenne et la variance des valeurs d'entrée et de sortie du modèle au temps t. De cette manière, les ensembles flous se déplacent sur le domaine de discours des variables, en fonction des valeurs de la moyenne et de la variance échantillonnées ; ainsi, la possibilité d'obtenir des ensembles flous disjoints peut être minimisée. La propriété dynamique du modèle flou proposé est un atout pour résoudre les problèmes de commande de systèmes variant avec le temps, par exemple.

Des exemples d'identification de fonctions non-linéaires variant avec le temps, illustrent la capacité du modèle flou adaptatif dynamique pour l'identification des systèmes. Une application à la commande prédictive a été développée, en utilisant le modèle flou proposé comme modèle de prédiction et l'AR pour résoudre le problème d'optimisation de ce type de schéma de commande. Finalement, l'utilisation de l'information contenue dans les fonctions d'appartenance dynamiques du modèle flou à des niveaux supérieurs de supervision et diagnostic, a été aussi discutée comme perspective intéressante d'application de ce type de modèles.

Mots clés : *Modélisation floue, fonctions d'appartenance dynamiques, apprentissage par renforcement, identification des systèmes, commande prédictive, diagnostic, supervision.*

Title : ON DYNAMICAL ADAPTIVE FUZZY MODELS

Abstract :

This work deals with the proposition of an adaptive fuzzy model with dynamical membership functions. The identification of the parameters of these membership functions is performed by a on-line reinforcement learning-based algorithm. This approach takes into account the system variables dynamic by incorporating the mean value and the variance, at time t, of the input and output variables of the fuzzy model into its membership functions. By this way, the fuzzy sets associated to the fuzzy variables, are relocated on the domain of discourse according to the sampled mean and variance values ; thus, a disjointed partition of the fuzzy sets into de fuzzy model could be avoid. The dynamical property of the proposed fuzzy models is an asset in fuzzy control problems in case of time-varying nonlinear systems, for example.

Classical examples related to the identification of time-varying nonlinear functions show the capabilities of the dynamical fuzzy models. An application to predictive control has been developed using the fuzzy model as one step ahead predictor and the reinforcement learning in the optimization problem of this type of control scheme. Finally, a discussion about the use of the information provided by the dynamical membership functions is presented in order to accomplish diagnosis and supervision tasks at upper levels.

Keys words : *Fuzzy modelling, dynamical membership functions, reinforcement learning, systems identification, predictive control, diagnosis, supervision.*