N° d'ordre: 2074

THESE

présentée à

L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE

Pour obtenir le grade de:

DOCTEUR D'UNIVERSITE

Spécialité Productique: Automatique et Informatique Industrielle

par

DELMOTTE François

Ingénieur EUDIL

ANALYSE MULTI-MODELE

Soutenue le 29 septembre 1997 devant le jury d'examen:

Président	Professeur M. Staroswiecki
Rapporteurs	Professeur A. Richard Professeur D. Meizel Professeur M. Ksouri
Examinateurs	Professeur P. Borne Professeur G. Dauphin-Tanguy Docteur A. M. Desodt-Jolly Professeur D. Dubois Professeur P. Penel

Travaux réalisés sous la direction du Professeur P. Borne au Laboratoire d'Automatique et d'Informatique de Lille, à l'Ecole Centrale de Lille.

CHAPITRE TROISIEME

LE CONTROLEUR FLOU

3.1 Introduction: Le succès du contrôleur flou

Ce chapitre a pour objectif de présenter le contrôleur flou classique.

Les implications floues définies au premier chapitre, et leur utilisation dans le Modus Ponens généralisé, permettent de modéliser des liens d'implication entre des variables de natures différentes. En outre la nature floue de la modélisation permet de prendre en compte une certaine imprécision sur les variables, et par là même, permet de mimer la réflexion humaine.

Or la communauté de l'automatique s'était rendu compte auparavant de la faculté de certains opérateurs humains à commander de manière empirique des systèmes complexes. Sans aucune formule mathématique, ces opérateurs arrivent à commander de manière souple et rapide, robuste et efficace, des systèmes que l'automatique classique avait beaucoup de mal à aborder, à cause de trop fortes non linéarités, ou du trop grand ordre des systèmes.

Lorsque ces experts humains furent interrogés, il apparu que la logique floue permettait de modéliser facilement leur réflexion interne, et donc de mimer l'ensemble du raisonnement qui amenait au calcul d'une commande complexe, et non plus d'une seule liaison entre quelques variables.

La représentation canonique de tels raisonnements fut appelée contrôleur flou. Lorsqu'un homme aborde un problème complexe, il cherche d'abord à en réduire la complexité en le découpant en petits problèmes plus simples. C'était effectivement la structure du contrôleur flou, qui est grossièrement une succession de syllogismes en parallèle avec un mécanisme permettant de les fusionner pour obtenir une seule conclusion.
Tel Modus Ponens correspond à tels environnements particuliers, à telles occurrences des variables en jeu, tel autre Modus Ponens à telles autres circonstances particulières, et l'ensemble des Modus Ponens permet d'avoir une vision globale du système.

La structure du contrôleur flou apparut donc très simple au regard d'autres formes de contrôleurs plus classiques. En outre cette structure était facilement compréhensible par un humain, parce que justement elle cherchait à en mimer les raisonnements, et que les représentations internes, floues, étaient basées sur la notion de symboles ou de termes linguistiques qui encore une fois étaient directement interprétables par un homme.

Ces deux caractéristiques du contrôleur flou expliquent sans doute son immense succès. En 1985, un peu plus de 20 ans après le premier article définissant théoriquement les ensembles flous, un article [Maiers et Sherif, 1985] recensait plus de 400 références dans divers domaines. Actuellement le nombre de travaux ayant trait à la logique floue appliquée à l'automatique est impressionnant et montre que cette théorie a révolutionné l'automatique classique.

Aujourd'hui deux écoles s'affrontent à propos du contrôleur flou. La première souhaite que soit gardée la caractéristique de lisibilité des contrôleurs flous, leur capacité à être interprétables par des humains. La seconde école s'abstrait complètement de cette contrainte, et utilise par exemple des algorithmes d'optimisation qui transforment les ensembles flous. De même il existe des méthodes de modélisation floue qui écartent toute intervention humaine, et donnent de même des contrôleurs flous étrangers à la logique humaine, et difficilement

interprétables par les humains. Il serait prétentieux de vouloir résumer l'ensemble des approches et des travaux ayant trait au contrôleur flou, pourtant il existe un élément commun à tous les contrôleurs flous, c'est leur structure.

L'objectif de ce chapitre est donc d'exposer la structure du contrôleur flou et de rappeler quelques résultats intéressants. La première partie aborde cette structure, et la seconde donne quelques propriétés du contrôleur flou, comme par exemple ses liens avec les PID.

3.2. Structure du contrôleur flou

Le schéma 3.1 montre la structure générale d'un contrôleur flou.

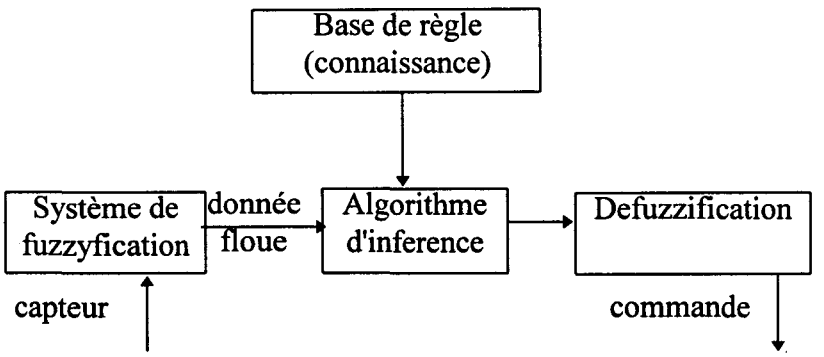

Fig. 3.1. Structure d'un contrôleur flou.

Un contrôleur flou est constitué de 4 blocs [Mandani, 1974; Procyk et Mandani, 1979; Tong, 1980; Lee, 1990 part 1]. Le système de fuzzyfication (bien que d'origine anglaise, ce terme a été gardé par la suite) réalise une interface entre le monde physique et l'algorithme de traitement des données. Par exemple des données fournies par un capteur sont mise en forme et traduites en ensembles flous.
L'algorithme d'inférence utilise les données traduites du monde physique par l'intermédiaire d'une base de règles représentant toute la connaissance sur le système. Il calcule par exemple une commande floue.
Le système de défuzzyfication réalise une opération inverse de celle du système de fuzzification, et transforme des ensembles flous en véritables quantités physiques.

Ces 4 blocs sont détaillés plus précisément ci dessous.

3.2.1 Base de règles

3.2.1.1 Structure générale

Ce bloc représente en fait toute la connaissance disponible sur le système considéré. La formalisation utilisée est celle qui consiste à mimer le raisonnement humain par une série de règles floues liant certaines variables.

Bien que la science actuelle n'ait que très peu de renseignements à propos des processus mentaux qui gouvernent un homme et ses comportements, une modélisation possible est de type procédurale, avec des séries de liens logiques entre les variables.

Cette représentation consiste à assigner à chaque événement possible un comportement, une réponse adaptée, et face à un stimuli particulier, face à une certaine situation, l'ensemble des événements est balayé de manière à trouver la réaction la plus appropriée.

Par exemple lorsqu'un homme souhaite fermer une porte, il fait appel à une connaissance du type R:

(R_1) 'Si la porte est ouverte, alors il faut la fermer'

(R_2) 'Si la porte est fermée, alors ne rien faire'

L'état de la porte étant considéré binaire (soit fermée, soit ouverte, dans ce dernier cas avec n'importe quel angle), la base de connaissance a énuméré tous les cas possibles de configuration et pour chaque cas a associé une commande, l'action fermer étant considérée comme une commande complexe qui permet de prendre en compte l'angle de la porte.

Il en résulte que la base de connaissance comporte deux règles.

Cet exemple paraît totalement artificiel à un humain, parce qu'il est difficile de comprendre la différence entre une porte à moitié ouverte et une porte à moitié fermée. Dans la base de règle la dissymétrie introduite entre les deux états, sous forme d'une commande nulle pour l'un et complexe pour l'autre, n'est pas naturelle.

Mais justement la logique floue va permettre de résoudre ce problème. D'ailleurs en automatique, les variables sont souvent continues, et il serait difficile d'énumérer tous les événements possibles (mathématiquement ce serait difficile).

En généralisant la structure R, une base de données floues est donc une succession de règles floues R_i du type:

(R_i) SI x_1 est X_{1i} et...et x_n est X_{ni} ALORS y est Y_i (3.1)

si un système MISO (multi-input single-output) de n variables entrées X et de 1 sortie Y est modélisé.

Les ensembles X_{ji} sont des ensembles flous qui permettent de caractériser sous un même terme un nombre infini d'événements proches: par exemple un petit angle serait l'ensemble des angles compris entre 0 et 15°, et une température chaude serait l'ensemble des températures entre 25 et 35°C, avec éventuellement des nuances d'appartenance...

Dans la dernière phrase furent utilisés les mots de chaud et de petit. Ce sont des concepts humains, appelés termes linguistiques, qui sont utilisés pour caractériser les ensembles flous. La base de règles doit couvrir l'ensemble des événements possibles, sinon elle serait incomplète. De même les termes linguistiques doivent être compréhensibles, sinon ils perdraient toute valeur pour les hommes. Aussi l'ensemble des termes linguistiques doit respecter certaines contraintes.

Avant cela une dernière remarque sur le nombre de règles m doit être faite. Celui-ci n'a pas encore été évoqué et pourtant il est à l'origine de certains problèmes des contrôleurs flous, nommément leur taille et leur complexité.

En effet soit l'exemple d'un problème à n variables, chaque variable n'étant découpée qu'en 2 termes (par exemple petit et grand), ce classement des variables en deux classes est grossier et pourtant pour que la base de règles couvre tous les cas de figures, il faudrait 2^n règles pour tout énumérer.

Aussi le nombre de règles augmente très vite lorsque la précision souhaitée augmente. Quelques solutions ont été envisagées et sont exposées à la fin du chapitre.

3.2.1.2 Les termes linguistiques

Ces termes linguistiques sont en fait les noms des ensembles flous utilisés dans la base de règles pour les variables X et Y.
Les contraintes qui lient ces ensembles flous sont d'une part une contrainte de lisibilité des ensembles flous par les humains, et d'autre part une contrainte de complétude de la base, qui doit envisager tous les cas de figures possibles.

Les ensembles flous associés aux symboles peuvent être de forme libre, des triangles, des courbes en cloche, mais en général il est difficile à un humain de faire la différence entre une décroissance d'appartenance linéaire et une autre, par exemple gaussienne. L'école qui souhaite que les contrôleurs flous gardent une certaine lisibilité impose ainsi aux termes linguistiques d'avoir une même structure, par exemple uniquement des courbes sigmoïdes.

En outre les hommes ne semblent utiliser qu'un nombre relativement restreint de termes pour caractériser une variable. En général 7±2 termes sont suffisants. Un modèle courant pour caractériser une variable est {grand négatif, petit négatif, nul, petit positif, grand positif} qui comporte 5 éléments.

La contrainte de complétude impose qu'aucune situation particulière ne puisse être oubliée. La figure 3.2 donne un exemple de mauvaise conception due à un manque de complétude:

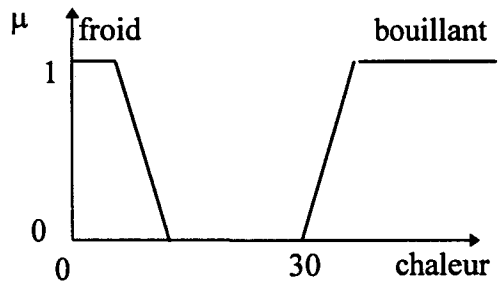

Fig. 3.2. Exemple de mauvaise couverture des termes.

Dans cet exemple la température de 20°C n'est pas prévue dans la base de règles.
Cette contrainte de couverture énonce que le support de l'union de tous les termes linguistiques pour une variable doit être tout l'espace de référence.

Il faut ensuite ajouter à cette contrainte une contrainte de séparabilité ou de lisibilité des ensembles flous (pour ceux qui veulent interpréter leur contrôleur flou). L'exemple de la figure 3.3 est un mauvais exemple de lisibilité.

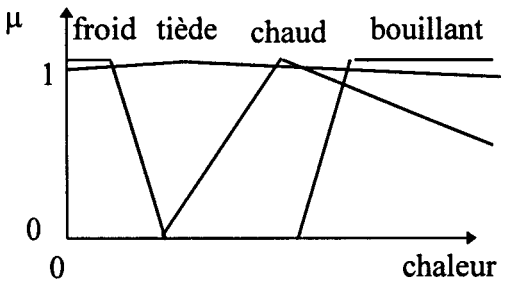

Fig. 3.3. Mauvaise lisibilité.

Dans cet exemple le concept 'tiède' englobe pratiquement tous les autres termes.
La contrainte est

$$\sum_{\text{termes}} \mu_{\text{termes}}(x) = 1 \tag{3.2}$$

la sommation se faisant sur tous les labels caractérisant une variable. En outre il ne faut pas que trop d'ensembles flous contiennent un même élement x. En général, le nombre d'ensembles flous correspondants à $\mu(x) \neq 0$ doit être égal à $2^{\text{Dimension}(X)}$. Si par exemple X est R, au plus 2 ensembles peuvent contenir chaque x.

Lorsque des algorithmes d'optimisation sont utilisés, les symboles sont complétement déformés (au sens de leurs fonctions d'appartenance), et le plus souvent les termes linguistiques ne se réfèrent plus à des notions compréhensibles par les humains.

La figure 3.4 donne un exemple de bonne couverture et de bonne lisibilité.

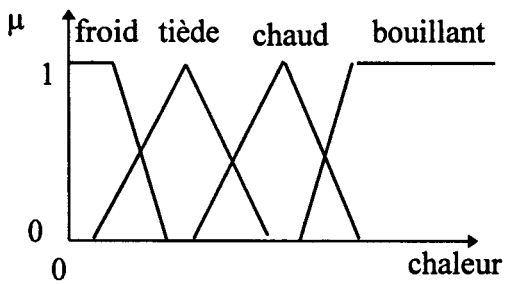

Fig. 3.4. Bonne couverture et bonne lisibilité.

Il existe donc un véritable antagonisme entre d'une part un souci de lisibilité des termes et d'autre part un souci de liberté et d'optimisation. Il existe aujourd'hui des méthodes d'identification floue basées sur des méthodes de classification qui permettent de construire les termes linguistiques et les règles d'inférences. Cependant dans ces cas il n'existe plus d'expert humain, et le plus souvent les règles sont incompréhensibles.

C'est pourquoi des algorithmes de post traitement ont été développés afin de rendre lisibles certains contrôleurs flous, en perdant un peu d'efficacité. Dans [Olivera, 1995] il existe une méthode générale pour concevoir des bases de termes linguistiques lisibles.

Il faut noter que ces contraintes sont valables à la fois pour les variables entrées et pour les variables sorties. Lorsque les variables de sortie sont représentées par de véritables ensembles flous, et non de simples singletons flous comme à la figure 3.7, le contrôleur est appelé

contrôleur symbolique, parce que les sorties sont des symboles ou encore des termes représentés par des ensembles flous. Dans le cas contraire, il est appelé contrôleur numérique, mais à part l'absence ou non de système de défuzzyfication, il n'y a pas vraiment de différences.

Enfin pour conclure ce paragraphe sur les termes linguistiques, il faut présenter les symboles normalisés. Il s'agit de termes linguistiques toujours définis sur un intervalle précis, par exemple [-1,1].

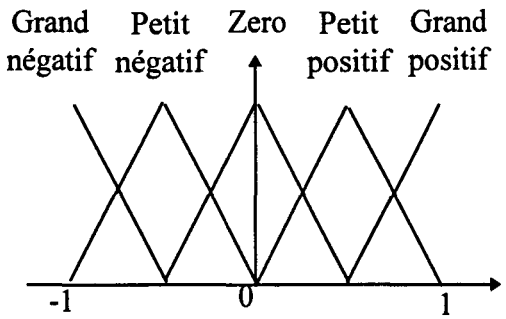

Fig. 3.5. Termes linguistiques normalisés.

Avec cette normalisation, les symboles maniés ont toujours le même sens, quelque soit le contrôleur flou utilisé.

Pour obtenir des valeurs physiques, il faut faire intervenir des gains propres à chaque contrôleur.

Par exemple pour obtenir des volts sur l'échelle -20 volts, +20 volts, il suffit de définir une échelle de conversion des volts dans l'intervalle [-1,1] en introduisant un facteur de proportionnalité. Si la mesure en volts est notée x_V, et la mesure convertie sur l'intervalle [-1,1] est notée par $x_{[-1,1]}$, la relation est $x_V=Gx_{[-1,1]}$, avec G=20 volts.

Dans certains contrôleurs flous, les conclusions sont toujours des termes linguistiques ou des symboles définis sur l'intervalle [-1,1], et c'est en agissant sur le facteur G de proportionnalité de la variable de sortie que les commandes sont alors plus ou moins fortes, ce paramètre pouvant éventuellement être optimisé.

3.2.2 Système de fuzzification

Le système de fuzzification réalise l'interface entre le monde physique et les données utilisées par le contrôleur flou, tout comme un Convertisseur Analogique Numérique lorsque des calculateurs sont utilisés pour la commande. Son objectif est de mettre en forme les données fournies par les capteurs et les transformer en ensembles flous.

A ce stade il est possible d'inclure une modélisation du capteur de façon à tenir compte d'une certaine imprécision.

La figure 3.5 donne des exemples de transformations appliquées aux mesures.

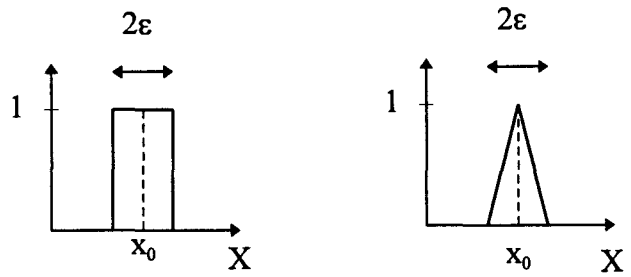

Fig. 3.6. Exemples de transformations utilisées par le système de fuzzification.

Par exemple si une mesure de température est fournie à 5% près, toute mesure numérique t_0 sera transformée en un ensemble flou, qui sera soit un rectangle, soit un triangle.

La modélisation du capteur est nécessaire pour choisir la bonne représentation de l'imprécision. Il faut en outre connaître la précision relative du capteur (par exemple t_0 est fourni à 5%, à 10%, à 15% près), sinon les ensembles flous risquent d'être soit trop précis, soit trop peu précis et inutilisables.

Cette utilisation de l'étape de fuzzification en tant que prise en compte de l'imprécision est cependant toute théorique car dans nombre d'applications réelles, les ensembles flous obtenus sont dégénérés (au sens du degré de flou ou de complexité) en singletons flous, et sont représentatifs de données absolument précises et certaines, comme le montre la figure 3.7:

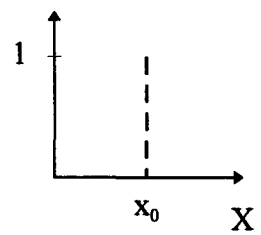

Fig. 3.7. Ensemble flou représentatif d'une mesure x_0.

Ceci se justifie par un souci de rapidité de temps de calcul.

Dans certains cas, la simplification des données est même poussée encore plus loin par l'utilisation des ensembles flous discrets. Le système de fuzzification est alors basé sur les termes linguistiques définis dans la base de règles.

Chaque mesure est alors comparée à chacun des termes utilisés, par des mesures de compatibilité (2.68 à 2.71), et l'ensemble des indices de compatibilité d'une mesure par rapport aux termes définit le vecteur ou l'ensemble flou discret qui représente la mesure.

Par exemple pour la température de l'eau, trois labels ont été définis, froid, tiède et chaud. Une température de 30°C pourrait être convertie dans le vecteur {0, 0.3, 0.7}. 0.3 représente la compatibilité de 30°C par rapport à l'ensemble flou tiède. 30°C est ici une mesure numérique, et donc la compatibilité se résume simplement à la fonction d'appartenance, mais si 30°C avait été au préalable modifié pour prendre en compte l'imprécision du capteur, l'indice de compatibilité aurait rempli le rôle de la valeur d'appartenance.

Le fait d'utiliser des ensembles flous discrets complique un peu l'étape de fuzzification , car il faut calculer une série d'indices de compatibilité, pourtant par la suite les temps de calculs et la place mémoire des données sont fortement réduits. Il faut noter aussi qu'une grande quantité d'information est perdue (car un ensemble continu, par exemple 30°C défini à 10% près, est résumé par un ensemble fini).

3.2.3 Système de défuzzyfication

Ce système réalise une opération inverse de celle du système de fuzzification, un peu à l'image d'un Convertisseur Numérique Analogique.
Le contrôleur flou travaille en effet sur des variables floues, et calcule une sortie en général floue. Il s'agit alors d'avoir pour les actionneurs des valeurs de sortie numériques.

Par exemple la température de l'eau se révèle être froide, et le contrôleur en déduit qu'il faut ajouter de l'eau chaude. Etant donné la sortie floue "chaude", le système de defuzzification va en déduire qu'il faudra ajouter de l'eau dont la température est de 45°C.

En général les sorties calculées par un contrôleur flou sont des ensembles flous complexes, pas du tout précis, et même pas de forme claire, comme des triangles par exemple.

La défuzzification est donc une opération mathématique de $Y^{[0,1]}$ dans Y. Il existe plusieurs types d'opérations réalisant cette fonction, et il n'existe pas de méthode permettant de choisir la meilleure.

Plusieurs méthodes ont été définies, et seulement jugées a posteriori sur les qualités de souplesse et d'efficacité des contrôleurs flous.

Les principales opérations sont données ci dessous.

3.2.3.1 Méthode du maximum

Cette méthode est la plus simple et la plus rapide. Elle consiste à ne considérer que le cœur de l'ensemble flou de sortie Y.
Les ensembles flous de sortie peuvent non seulement être de forme et d'aspect libres, mais en plus ils peuvent être non normalisés.
Il faut donc d'abord définir le cœur de la sortie Y, comme étant l'ensemble classique des valeurs y qui atteignent le maximum de valeur d'appartenance:
$\text{Cœur}(Y_0)=\{y: \mu_{Y_0}(y)=\max_y(\mu_{Y_0}(y))\}$
Alors la défuzzyfication est une simple moyenne:

$$y_0 = \frac{\int_{\text{Coeur}(Y_0)} y\mu(y)dy}{\int_{\text{Coeur}(Y_0)} \mu(y)dy} = \frac{\int_{\text{Coeur}(Y_0)} ydy}{\int_{\text{Coeur}(Y_0)} dy} = 0.5(y_{\min \text{Coeur}} + y_{\max \text{Coeur}}) \qquad (3.3)$$

avec $y_{minCoeur}$ et $y_{maxCoeur}$ les bornes inférieure et supérieure de $\text{Cœur}(Y_0)$, si celui ci est un ensemble connexe dans Y, sinon la dernière égalité ne tient plus.

Parfois pour diminuer encore le temps de calcul, seule la borne inférieure est prise en compte.
La figure 3.8 montre le résultat sur un ensemble flou quelconque.

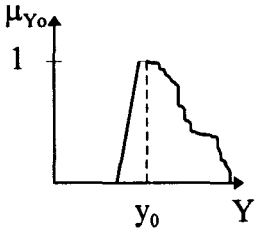

Fig. 3.8. Défuzzyfication basée sur la moyenne du cœur.

Avec cette méthode très rapide, beaucoup d'information est perdue sur la variable de sortie Y. Ainsi la valeur y_0 défuzzyfiée de la figure 3.8 est-elle égale à celle de la valeur y_0 pourtant différente de la figure 3.9.

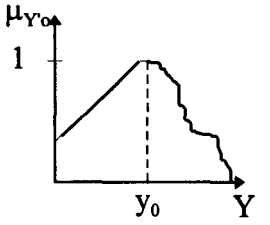

Fig. 3.9. Second ensemble flou défuzzyfié par la méthode du cœur.

Ce problème est résolu en partie par la prochaine méthode.

3.2.3.2 Méthode du centre de gravité

Le nom en anglais est Center of Area (COA).
C'est la principale méthode utilisée, mais elle nécessite un calcul d'intégrale.

La formule est

$$y_0 = \frac{\int_Y y\mu(y)dy}{\int_Y \mu(y)dy} \tag{3.4}$$

Elle prend en compte toute l'information disponible.

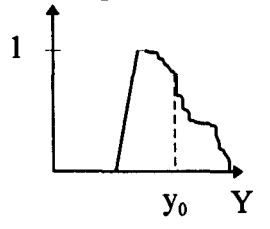

Fig. 3.10. Défuzzyfication par une approche du type centre de gravité.

La méthode suivante est plus précise, mais est très peu utilisée.

3.2.3.3 Méthode du centre de gravité pondéré

Pour comprendre cette opération, il faut prendre un peu d'avance sur le système d'inférence qui permet d'obtenir une sortie Y_0 floue à partir des mesures floues.

En fait les m règles de la base de données calculent chacune une sortie floue Y_{0i}, i=1 à m. Ensuite ces m ensembles flous sont fusionnés, dans le cas des contrôleurs flous un opérateur max est utilisé. Cet opérateur max élimine beaucoup d'information, notamment si 2 règles calculent une même sortie floue Y_{0i}. Avec l'opérateur max, Y_{0i} n'apparaîtra qu'une fois dans le résultat, c'est à dire que cet Y_{0i} n'aura pas plus de poids que les Y_{0j} calculés par d'autres règles.

Le but de l'opérateur de centre de gravité pondéré (WCOA en anglais) est de sauter l'étape de la fusion des m sorties Y_{0i} par un opérateur max, et de les fusionner directement au stade de la défuzzyfication, en combinant les deux étapes.

Le résultat est alors:

$$y_0 = \frac{\int_Y y \sum_i \mu_i(y) dy}{\int_Y \sum_i \mu_i(y) dy} \tag{3.5}$$

3.2.3.4 Combinaison des méthodes de défuzzyfication

Il est possible de combiner les différents opérateurs de défuzzyfication pour obtenir un opérateur plus complexe.

Par exemple avec une combinaison linéaire

$$y_0 = \lambda \frac{\int_{Coeur(Y_0)} y\mu(y)dy}{\int_{Coeur(Y_0)} \mu(y)dy} + (1-\lambda) \frac{\int_Y y\mu(y)dy}{\int_Y \mu(y)dy} \tag{3.6}$$

avec $\lambda \in [0,1]$

Cependant il n'y a plus l'argument de rapidité en faveur de la moyenne du cœur, et le centre de gravité est perturbé par le premier membre.

Le paramètre λ est utilisé comme un paramètre à optimiser, ce qui peut donner une plus grande souplesse au système d'inférence.

3.2.3.5 Compatibilité entre la fuzzification et la défuzzification

Ces deux opérations apparaissent complémentaires, et même inverses l'une de l'autre.

Aussi il serait souhaitable d'imposer aux opérateurs choisis une contrainte sur la composition de ces deux opérateurs [Rondeau et al, 1995].

En effet, pour avoir une certaine consistance des résultats, pour ne pas introduire de dérive artificielle dans les calculs, il faudrait que la défuzzyfication de la fuzzyfication d'un nombre x retourne le même nombre x:

$$\text{Défuzzy(Fuzzy)=Identité} \tag{3.7}$$

Pour une défuzzyfication basée sur le centre de gravité il faut que l'opérateur de fuzzyfication soit un opérateur symétrique dans la définition des ensembles flous, par rapport aux valeurs numériques (par exemple un nombre 35 ne doit pas être transformé en un intervalle [30, 36]).

Dans un souci de lisibilité, certains chercheurs imposent aussi aux termes linguistiques d'avoir un cœur réduit à un singleton.

3.2.4 Systèmes d'inférences

Le système d'inférence est le cœur d'un contrôleur flou. C'est en fait lui qui réalise tous les calculs, les autres blocs étant en fait soit une connaissance statique, soit des interfaces avec le monde extérieur.

Etant donnée une base de connaissance sous forme de règles d'implication, et d'entrées (floues), le système d'inférence calcule une sortie floue pour le système considéré.

3.2.4.1 Structure générale du système d'inférence

La base de règles est une succession de m implications de type 3.1 qui lient des variables floues.

La démarche du système d'inférence reprend celle du Modus Ponens exposé au premier chapitre. Il y aura en fait m syllogismes calculés, la partie finale du système d'inférence se chargeant de réaliser la synthèse des m Modus Ponens.

Les variables des prémisses des règles sont notées $x_i \in X_i$. Chaque règle étudie le comportement approprié à un certain environnement, et cet environnement est représenté par les termes linguistiques qui apparaissent dans les prémisses des règles sous la forme des ensembles flous X_{ij} pour j=1 à n s'il existe n termes linguistiques pour décrire une variable (dans un souci de clarté, les variables sont supposées être toutes décrites par n termes).

Il faut d'abord représenter le lien entre les variables d'entrées X_i et la sortie Y, en définissant la relation entre ces variables. Cette relation est une relation d'implication, qui définit donc un sous ensemble flou dans l'espace $X_1 \times ... \times X_n \times Y$.

Pour définir ce sous ensemble flou, il faut un opérateur d'implication. Le premier chapitre a montré qu'il n'existait pas qu'un seul opérateur possible pour généraliser l'implication booléenne.

Aussi la première étape dans un contrôleur flou est de choisir l'opérateur d'implication floue $\Rightarrow$. Ceci est fait lors de la construction de la base de donnée des règles.

Ensuite à partir des mesures x_i floues, il faut réaliser m Modus Ponens. Pour cela, il faut un opérateur d'extension-projection et un opérateur d'intersection afin de pouvoir d'une part étendre les mesures floues dans l'espace $X_1 \times ... \times X_n \times Y$, comparer ces nouvelles mesures à la relation floue de la règle R_i, et finalement projeter sur Y la solution obtenue.

Pour réaliser les m Modus Ponens, il a donc fallu choisir un opérateur de projection, une union $\oplus$, et un opérateur de comptabilité ou d'intersection, noté $\otimes$.

Enfin il faut fusionner les m différentes valeurs de sorties Y_i calculées par l'application des m Modus Ponens élémentaires. Ceci est réalisé par un nouvel opérateur Θ.

La figure 3.11 montre l'agencement de ces différents opérateurs.

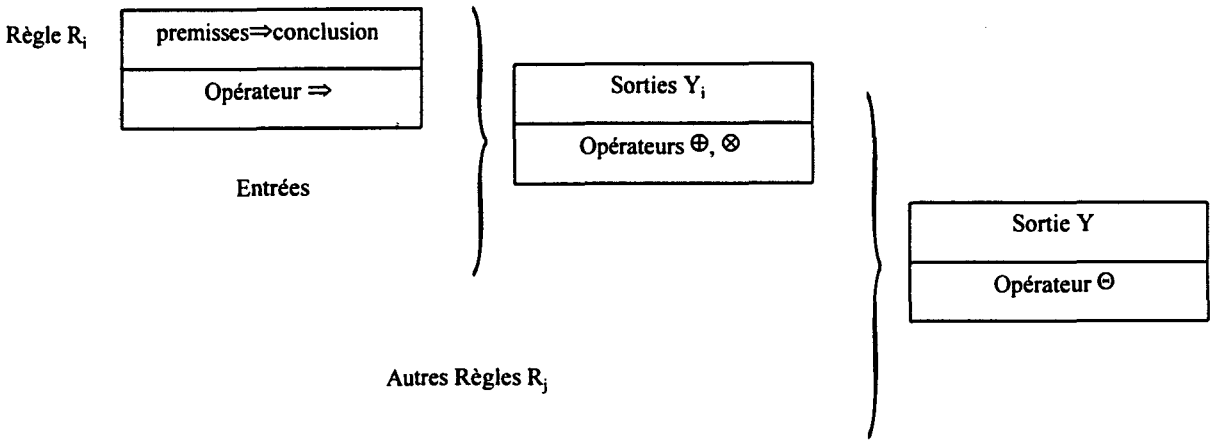

Fig. 3.11. Construction du système d'inférence.

3.2.4.2 Choix des opérateurs.

Dans la construction du système d'inférence sont apparus quatre opérateurs $\Rightarrow,\otimes,\oplus,\Theta$. Le choix de ces opérateurs est surtout un choix de comportement, par exemple un opérateur probabiliste aura un comportement plus continu qu'un opérateur de la famille de Zadeh.

Il existe cependant des contraintes sur le choix de ces opérateurs, dont la plus critique est celle qui lie le choix des opérateurs d'implication $\Rightarrow$ et d'agrégation des règles Θ.

Pour comprendre cette contrainte il faut revenir à l'action de l'implication.

Si l'implication $\Rightarrow$ est une véritable implication floue, alors pour des entrées données:
-Pour la règle R_i qui correspond, en sortie il y aura un ensemble flou Y_i
-Par contre pour les autres règles R_j qui ne correspondent pas, la sortie Y_j sera un ensemble flou équivalent à l'ignorance complète en possibilité, c'est à dire que $Y_j(y)=1\ \forall y$
Donc, pour conclure à la bonne sortie Y_i, l'opérateur d'agrégation Θ doit être une intersection.

Si l'implication $\Rightarrow$ est en fait une simple intersection, alors pour des entrées:
-Pour la règle R_i qui correspond aux bonnes entrées, la sortie sera Y_i (pas de différence)
-Pour les règles R_j, qui ne correspondent pas, les sorties Y_j seront toutes des ensembles nuls $Y_j(y)=0\ \forall y$ (exact contraire au cas d'une véritable implication)
Donc pour conclure à la bonne sortie Y_i, l'opérateur d'agrégation Θ doit être une union.

En dehors de cette contrainte majeure, la plus grande liberté règne pour le choix des quatre opérateurs $\Rightarrow,\otimes,\oplus,\Theta$, et une fois ce choix complètement défini, il constitue un système d'inférence.

Actuellement dans les différents domaines d'utilisation (automatique, intelligence artificielle,...), il existe plus de 72 systèmes d'inférence. Tous ont leurs qualités et aucun n'est universel.

En automatique l'implication $\Rightarrow$ est toujours une intersection, car les systèmes commandés sont toujours passifs, et dépendent donc des entrées. Aussi l'opérateur d'agrégation est-il une union.
Un système d'inférence très simple et très utilisé est le système d'inférence de Mamdani:
$\Rightarrow$ est l'opérateur min ou produit
Θ est donc l'opérateur max
$\oplus$ est l'opérateur max et $\otimes$ est soit l'opérateur min soit l'opérateur produit.

3.3 Résultats divers

Beaucoup d'études ont été réalisées et de résultats obtenus à propos du contrôleur flou défini au paragraphe précèdent. Ce paragraphe présente quelques résultats intéressants et suffisamment généraux pour donner une vision générale des analyses du contrôleur flou.

3.3.1 Fonctions élémentaires floues

Ce sont des structures particulières du contrôleur flou, qui sont très simples et qui peuvent servir soit à approximer des contrôleurs flous efficaces mais illisibles, soit à simplifier la construction des contrôleurs.
En anglais le terme est FBF pour fuzzy basis functions.

Ces contrôleurs sont des cas simplifiés du contrôleur de Mamdani défini au paragraphe 3.2.4.2.

L'implication $\Rightarrow$ est un produit, et n'est donc pas une véritable implication.

$$\mu_{\Rightarrow_i}(x,y) = \mu_{X_{i1}}(x_1)\times\ldots\times\mu_{X_{in}}(x_n)\times\mu_{Y_i}(y) \tag{3.8}$$

avec $x=[x_1...x_n]$.
Le symbole $\Rightarrow_i$ est utilisé pour représenter l'ensemble flou défini par l'implication liant des termes linguistiques de la règle R_i.

L'intersection étant aussi choisie de type produit, le Modus Ponens fournit donc une nouvelle sortie Y'_i (dans le premier chapitre, cet ensemble était noté B', les prémisses étant notées A'), définie par la relation:

$$\mu_{Y'_i=\mathrm{Proj}_Y(A'\cap\Rightarrow_i)}(y) = \max_x(\mu_{A'}(x)\times\mu_{\Rightarrow_i}(x,y)) \tag{3.9}$$

Par rapport au contrôleur de Mamdani classique, une contrainte supplémentaire est ajoutée, celle qui impose que les entrées fournies par les capteurs soient toutes des singletons flous, comme défini à la figure 3.7.

Ainsi les mesures floues sont supposées précises et certaines, étant définies par:

$$\mu_{A'}(x) = \begin{cases} 1 & \text{si } x = x' \\ 0 & \text{autrement} \end{cases}$$

Par conséquent la relation 3.8 peut être simplifiée, et le résultat final de la conclusion modifiée Y'_i est:

$$\begin{aligned}\mu_{Y'_i}(y) &= \mu_{\Rightarrow_i}(x',y) \\ &= \left[\prod_j \mu_{X_{ij}}(x'_j)\right]\mu_{Y_i}(y)\end{aligned} \tag{3.10}$$

avec μ_{Yi} la fonction d'appartenance de la conclusion réelle Y_i, et les μ_{Xij} les n fonctions d'appartenance des différentes prémisses de la règle R_i.

Si x' est noté simplement noté x, le niveau de déclenchement τ_i de la règle R_i (en anglais firing level) est alors défini par:

$$\tau_i(x) = \prod_j \mu_{X_{ij}}(x_j) \tag{3.11}$$

En effet l'ensemble de sortie modifié Y'_i sera égal à l'ensemble de sortie a priori Y_i, mis à part ce facteur de proportionnalité.

Si de plus les ensembles flous de sorties a priori Y_i sont imposés comme étant des singletons flous:

$$\mu_{Y_i}(y) = \begin{cases} 1 & \text{if } y = y_i \\ 0 & \text{sinon} \end{cases} \tag{3.12}$$

alors les étapes d'agrégation des m règles et de la défuzzyfication par centre de gravité peuvent être fusionnées en une simple règle, qui est une somme pondérée des m sorties numériques y_i:

$$y(x) = \frac{\sum_{i=1}^{m} y_i \tau_i(x)}{\sum_{i=1}^{m} \tau_i(x)} \tag{3.13}$$

Plusieurs remarques peuvent être faites à propos de cette relation finale.

3.3.1.1 Degré de confiance des règles

D'abord dans certains contrôleurs flous sont ajoutés des degrés de confiance $\alpha_i \in [0,1]$ dans les règles R_i. Ils ont le même sens que dans la théorie des possibilités, ce sont des coefficients de pondération, qui permettent de renforcer l'influence d'une règle par rapport à une autre. Ainsi plusieurs règles peuvent être définies avec les mêmes prémisses, mais avec des conclusions différentes (leur nombre augmente alors aussi par rapport au nombre maximal de règles correspondant à tous les environnements possibles). Ceci permet d'introduire une certaine contradiction dans les règles.

Par exemple pour régler la température de l'eau, il y aurait trois règles au lieu de 2: si l'eau est chaude, ne rien faire, si l'eau est froide ajouter de l'eau chaude, avec un coefficient de 0.9 et une troisième règle qui énonce que si l'eau est froide, alors ne rien faire, avec un degré de 0.2 (les coefficients α_i n'ont pas de contrainte particulière autre que d'appartenir à [0,1]). La

troisième règle est en contradiction avec la seconde, mais peut s'appliquer si la personne qui va prendre son bain s'entraine à nager en eau glacée...

L'introduction de ces coefficients dépend des circonstances et des méthodes de modélisation. Leur prise en compte est facile, s'ils sont utilisés par rapport aux niveaux de déclenchements τ_i. Des nouveaux niveaux de déclenchements τ'_i sont définis par $\tau'_i=\alpha_i\tau_i$ et la règle 3.12 est utilisée avec τ'_i à la place de τ_i.

3.3.1.2 Les briques élémentaires en tant qu'outil de construction.

La relation 3.12 a la remarquable propriété d'être linéaire en les termes y_i.

Cette propriété est utilisée lors de la construction de contrôleurs flous. En effet, si plusieurs contrôleurs doivent être conçus pour des problèmes très proches, à chaque fois les mêmes variables entrant en jeu, avec seulement quelques coefficients qui changent, il est possible de construire des termes linguistiques, c'est à dire des ensembles flous, uniquement pour les prémisses.

Il est alors possible de définir une partition floue de l'espace d'entrée, de manière à obtenir une partition très claire et très lisible.

Ensuite le véritable problème de modélisation au cas par cas commence avec l'estimation des valeurs y_i. Grâce à la relation 3.12, ces valeurs y_i sont des fonctions linéaires des niveaux de déclenchement, qui ont déjà été définis directement par les termes linguistiques des variables entrées.

Il est alors très facile d'estimer les y_i par des algorithmes d'optimisation linéaire.

Il est possible d'introduire par exemple un critère d'erreur: étant donné une réponse $\acute{y}$ souhaitée, l'erreur $e(k)=\acute{y}(k)-y(k)$ peut être minimisée en fonction des paramètres y_i. Souvent c'est un critère quadratique qui est utilisé.

Les fonctions élémentaires floues apparaissent donc, par leur simplicité, très intéressantes pour la mise au point de contrôleurs flous ou pour la modélisation. De plus en plus la conception des contrôleurs flous repose sur une première étape qui partionne l'espace d'état avec des méthodes de classification et éventuellement des critères de lisibilité, ou simplement en fonction d'un opérateur humain. Ensuite dans une seconde étape les conclusions des règles sont évaluées et optimisées. Cette démarche n'est pas optimale, car bien sûr une optimisation complète adapterait en même temps les premisses et les conclusions, mais les résultats sont corrects, et le gain en temps est immense.

La relation 3.12 peut aussi être utilisée en dehors de toute référence à la commande, selon la nature des variables sorties Y, le système peut être modélisé si les variables Y sont des sorties du système, commandé si les sorties Y sont des variables de commande...
Dans un cadre non linéaire, sans référence aux fonctions élémentaires, ceci est vrai pour tous les contrôleurs flous.

En effet la capacité d'approximation universelle des contrôleurs flous fut demontrée à plusieurs reprises: dans le cas SISO (single input single output), avec les fonctions

élémentaires, voir Zeng [Zeng et Singh, 1994] et Wang [Wang et Mendel, 1992a], dans le cas MISO, voir Zeng [Zeng et Singh, 1994] et dans un cadre plus général englobant la plupart des contrôleurs, voir Castro [Castro, 1995].
C'est pourquoi les fonctions élémentaires font l'objet de nombreux articles. Il est possible de citer Yager [Yager et al, 1994], Wang [Wang, 1993] et Xu [Xu et Lu, 1987]. Dans ces papiers, un modèle flou basé sur les fonctions élémentaires est en effet identifié par divers moyens, puis est utilisé pour la commande.

3.3.2 Perturbations dues à des entrées floues

Lorsque les entrées d'un contrôleur flou sont modélisées par des ensembles flous non réduits à des singletons, comme à la figure 3.7, un problème apparaît qui milite en faveur de l'utilisation d'entrées réduites à des singletons, comme dans toutes les applications floues.

En effet lorsque de véritables ensembles flous sont utilisés pour représenter les mesures, par exemple des triangles pour tenir compte d'une certaine imprécision, les sorties des règles se perturbent entre elles, et s'il arrivait que l'environnement soit exactement celui d'une règle R_i, alors la sortie finale du contrôleur flou ne serait pas la sortie Y_i de la règle R_i.

L'exemple graphique détaille le problème. Le contrôleur utilisé est un contrôleur de Mamdani, avec des intersections et unions basées sur la famille de Zadeh.

Il existe une seule variable d'entrée, et une seule sortie. Chacune des variables est représentée par 2 labels, positif et négatif, comme le montre la figure 3.12.

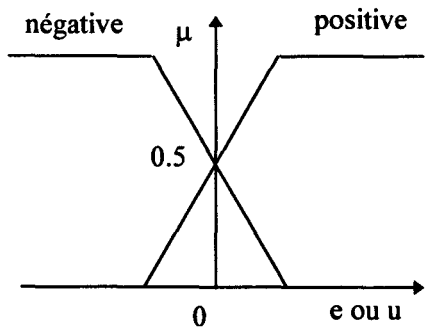

Fig. 3.12. Labels caractérisant l'espace d'entrée e et l'espace de sortie u.

La base de données des règles est réduite à deux règles R_1 et R_2:

(R_1) Si l'erreur e est Positive, alors la commande u est Négative

(R_2) Si l'erreur e est Négative, alors la commande u est Positive

Les dessins qui suivent vont montrer l'évaluation de la sortie lorsque **e est Positive**, c'est à dire un ensemble flou non réduit à un singleton.

D'abord la première règle est envisagée:

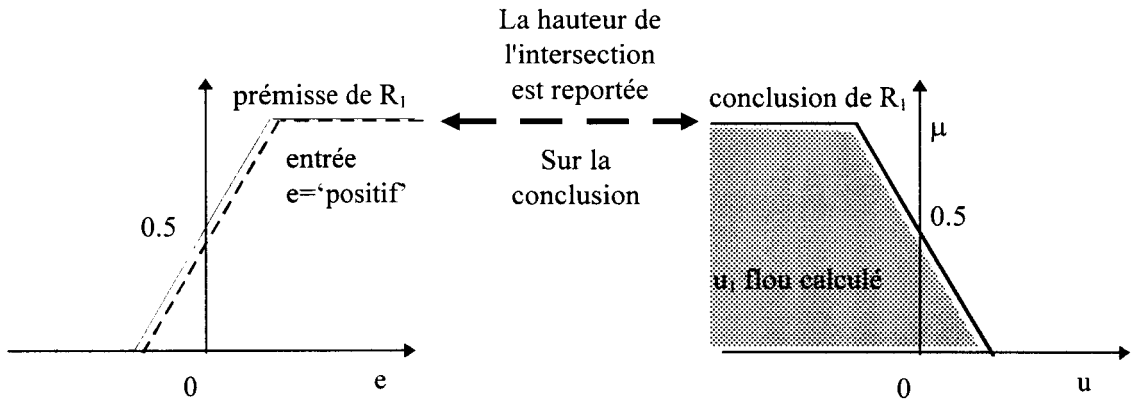

Fig. 3.13. La comparaison de l'entrée e en '----' avec la prémisse en continu qui implique la sortie calculée u_1.

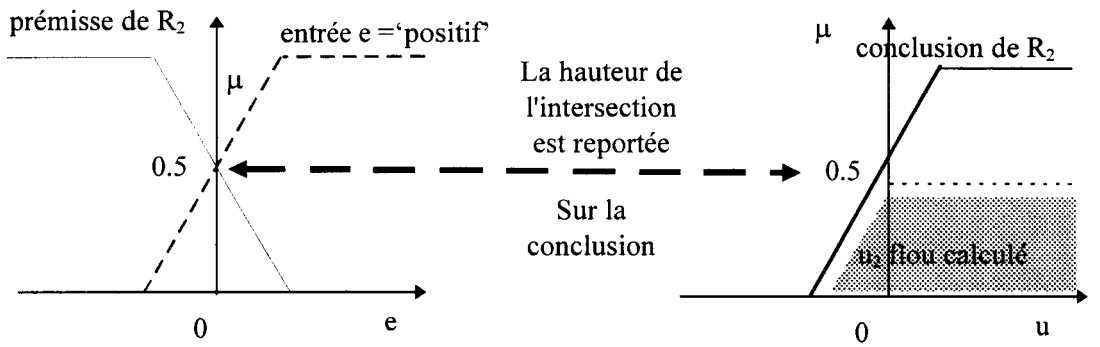

Fig. 3.14. La comparaison de l'entrée e en '----' avec la prémisse en continu qui implique la sortie calculée u_2.

Les deux sorties u_1 et u_2 des règles ayant été calculées, il faut maintenant les agréger.

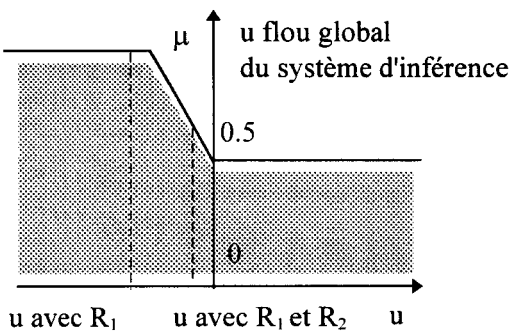

Fig. 3.15. Comparaison des défuzzyfications de u_1 et de la sortie calculée par le système d'inférence (méthode du centre de gravité).

La figure 3.15 montre donc qu'il existe bien une différence entre la sortie u_1 qui était normalement la seule pertinente, puisque les mesures correspondaient parfaitement à la règle R_1, et la sortie u finale qui tient compte de toutes les règles, en l'occurrence R_1 et R_2.

Ceci provient du choix de l'implication floue utilisée dans les controleurs classiques. Même si les mesures correspondent parfaitement à une règle R_i, ce qui serait révélé par un niveau de déclenchement égal à 1, la compatibilité de ces mêmes mesures avec les prémisses des autres

règles, au moins des plus proches, ne sera pas nulle. Leurs niveaux de déclenchement ne seront donc pas nuls, et leurs "conclusions" Y_j seront aussi utilisées par le système d'inférence.

Ceci constitue un inconvénient certain du système d'inférence. Il est possible d'y remédier en utilisant une méthode de défuzzyfication basée sur le cœur du résultat (paragraphe 3.2.3.1), car dans ce cas seule la conclusion Y_i sera prise en compte, puisque son ensemble flou aura une hauteur égale à 1 contrairement aux autres ensembles flous Y_j. Mais dans ce cas pour les cas intermédiaires, la souplesse et le comportement continu de la méthode de défuzzification basée sur le centre de gravité seront perdus.

3.3.3 Liens avec les contrôleurs PID

Le but de ce paragraphe est de montrer que le comportement des contrôleurs flous dégénérés est très semblable à celui des contrôleurs PID.
Bien que les contrôleurs flous sous forme analytique soient totalement non linéaires, comme le montrera le paragraphe suivant, les contrôleurs de Mamdani pour lesquels les symboles ou les termes linguistiques sont conçus lisiblement comme au paragraphe 3.2.1.2 tendent à avoir un comportement linéaire semblable à celui des contrôleurs PID. Dans ce cas plus le nombre de termes augmente, plus les non linéarités diminuent [Foulloy et Galichet, 1992].

3.3.3.1 Cas d'un contrôleur à une variable

La variable d'entrée est l'erreur e, et la sortie du contrôleur est la commande u. Dans ce cas le contrôleur tend vers un contrôleur P.

La figure 3.16 montre la forme des labels pour les deux variables en jeu.

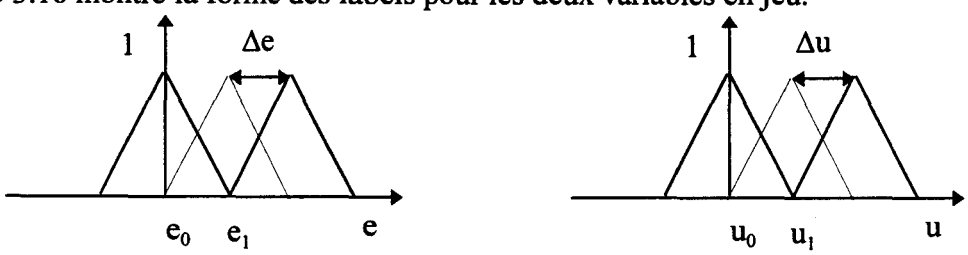

Fig. 3.16. Symboles (e_0, u_1,...) definissant les variables du contrôleur flou (e,u).

Lorsque le nombre de symboles est identique pour la variable e et pour la variable u, le contrôleur flou tend vers un contrôleur du type:

$$u=Ke$$

avec

$$K = \frac{\Delta u}{\Delta e} \tag{3.14}$$

L'égalité est parfaite lorsque la variable entrée vaut exactement $e=e_0+k\Delta e$ ou $e=e_0+(k+0.5)\Delta e$, avec k entier. Si Δe tend vers 0 (c'est à dire si le nombre de termes linguistiques tend vers l'infini), alors l'égalité est vraie pour toute entrée e.

De même, plus la distance entre les symboles pour la variable de sortie u décroît, plus la distance entre la sortie du contrôleur flou et la sortie du contrôleur P équivalent décroît.

3.3.3.2 Cas d'un contrôleur à deux entrées.

Les entrées du contrôleur sont e et δe, la sortie du contrôleur flou est cette fois δu, la variation de commande par rapport à la dernière commande calculée.

Les symboles pour toutes les variables sont des triangles définis comme à la figure 3.16 du paragraphe précèdent sur un contrôleur à une entrée.

Les distances entre les sommets des triangles sont notées respectivement Δe, $\Delta\delta e$ et $\Delta\delta u$.

Dans ce cas le contrôleur flou tend vers un contrôleur PI du type:

$$\delta u = K_p \delta e + K_i e$$

avec

$$K_p = \frac{\Delta\delta u}{\Delta\delta e} \qquad K_i = \frac{\Delta\delta u}{\Delta e} \tag{3.15}$$

L'égalité est même exacte notamment dans le cas où les entrées valent $(e,\delta e)=(e_0+k\Delta e,\delta_0+l\Delta\delta e)$ avec k et l entiers.

3.3.4 Forme analytique exacte.

Ce paragraphe a pour but de montrer que les contrôleurs flous sont parfaitement déterministes, et qu'il est parfaitement possible d'en calculer la forme analytique directe liant les variables entrées et les variables sorties.

Le contrôleur utilisé est un contrôleur de Mamdani à une entrée e et une sortie u avec une inférence basée sur les opérateurs probabilistes. Les symboles, au nombre de n (impair) pour chaque variable, sont définis comme à la figure 3.17. L'intervalle de variation pour l'entrée e est [-1,1], et l'intervalle de variation pour la sortie u est [-k,k]. Comme le nombre de symboles est donné en cas de répartition régulièere , il est possible de le relier avec les distances aux sommets:
$\Delta e=2/(n-1)$ et $\Delta u=2k/(n-1)$.

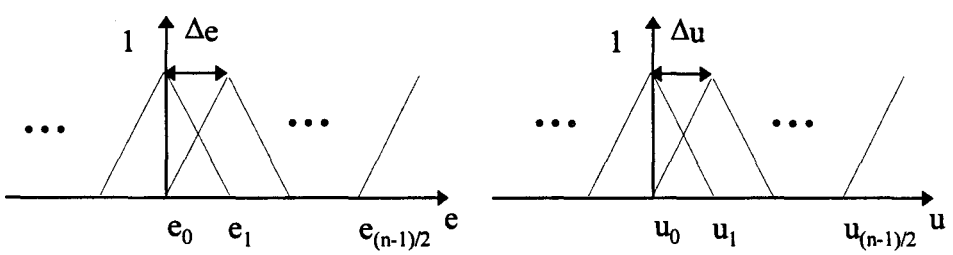

Fig. 3.17. Labels symétriques pour l'exemple de l'expression analytique.

Les règles sont du type:

R_i: Si e est e_i alors u est u_{-i}

En ce qui concerne la variable sortie, les symboles u_i pourront être réduits à des singletons flous (les sommets des triangles) pour envisager le cas où les conclusions sont simplement numériques. Dans ce cas, l'équation est $\mu_{u_i}(u) = \begin{cases} 1 & \text{si } u = i\Delta u \\ 0 & \text{sinon} \end{cases}$

Dans le cas général des triangles, les équations sont pour e_i, pour $i \in \{-(n-1)/2,...,(n-1)/2\}$:

pour $e \in [(i-1)\Delta e, \Delta e]$ l'équation du triangle est $\mu(e)=1+(e-i\Delta e)/\Delta e$

pour $e \in [i\Delta e, (i+1)\Delta e]$ l'équation est $\mu(e)=1-(e-i\Delta e)/\Delta e$

0 ailleurs

et de même pour les u_i (pour les triangles extrêmes il suffit d'enlever le bon terme).

Maintenant, il faut étant donné une mesure e, calculer la conclusion des règles.

L'indice i est d'abord évalué tel que $e \in [i\Delta e, (i+1)\Delta e]$ (ceci est un artifice pour simplifier le calcul de chaque conclusion, dans un contrôleur flou, tout est calculé en parallèle).

Alors il apparaît que seulement deux règles ont un niveau de déclenchement non nul: R_i et R_{i+1}.
La conclusion de chaque règle est donc un ensemble flou défini par:

$$\mu_{R_i}(u) = \mu_{u_{-i}}(u)\left(1-(e-i\Delta e)/\Delta e\right)$$

$$\mu_{R_{i+1}}(u) = \mu_{u_{-i-1}}(u)\left(1+(e-(i+1)\Delta e)/\Delta e\right)$$

Jusqu'à présent la forme des conclusions des règles, le mode d'agrégation des règles et le mode de défuzzyfication n'ont pas été pris en compte.

Le cas où les conclusions u_i sont des singletons flous est d'abord traité.
L'agrégation et la défuzzyfication sont fusionnées pour donner la valeur finale de la commande, puisque le contrôleur est une fonction élémentaire floue.

Dans le cas d'une défuzzyfication par centre de gravité, le résultat est:

$$u = \frac{\left[1-(e-i\Delta e)/\Delta e\right](-i\Delta u)+\left[1+(e-(i+1)\Delta e)/\Delta e\right](-(i+1)\Delta e)}{1-(e-i\Delta e)/\Delta e+1+(e-(i+1)\Delta e)/\Delta e}$$

ou après simplification:

$$u = -e\frac{\Delta u}{\Delta e} \tag{3.16}$$

ce qui est la formulation exacte d'un contrôleur P.

Dans le cas d'une défuzzyfication par le maximum, le résultat est

$$u = \begin{cases} -i\Delta u & \text{si} \quad 1-(e-i\Delta e)/\Delta e > 1+(e-(i+1)\Delta e)/\Delta e \\ -(i+1)\Delta u & \text{sinon} \end{cases} \tag{3.17}$$

qui est une forme non linéaire, qui tend vers un contrôleur P si le nombre de symboles augmente.

Ensuite il faut envisager le cas où les conclusions sont réellement des ensembles flous triangulaires.

Dans le cas d'un système de défuzzyfication basé sur le centre de gravité, il faut fusionner les deux conclusions de R_i et de R_{i+1} par un opérateur max, puis intégrer le résultat pour en déduire la bonne sortie. Le calcul est nettement plus long.
Le résultat est [El Hajjaji, 1994]:

$$u = \frac{1}{3}\frac{-6k^2e^2i\Delta u + 9kei\Delta u^2 + 9kei^2\Delta u^2 - 3k^2e^2\Delta u + k^3e^3 - 4ke\Delta u^2 - 4i^3\Delta u^3 - 6\Delta u^3 i^2 + 2\Delta u^3}{k^2e^2 - ke\Delta u - 2kei\Delta u + i^2\Delta u^2 + 2\Delta u^2} \tag{3.18}$$

Les figures 3.18 montre le résultat obtenu pour les conclusions numériques avec défuzzyfication par maximum.

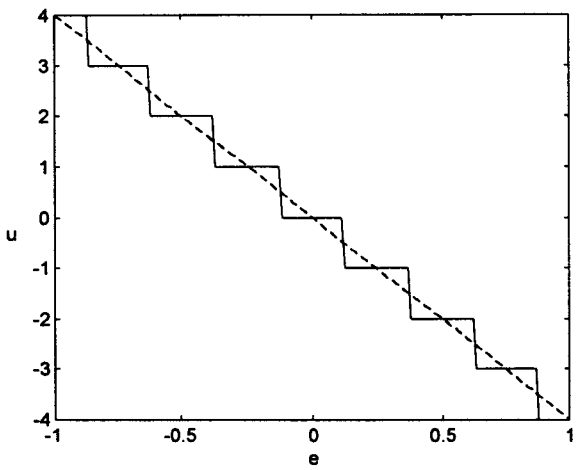

Fig. 3.18. Sortie en fonction de l'entrée pour un type simple de contrôleur flou. En pointillé apparaît le contrôleur P. équivalent.

Il est à remarquer que pour des contrôleurs flous plus complexes, le contrôleur P. équivalent n'a pas forcément un gain égal à $k=\Delta u/\Delta e$.

Lorsque les paramètres des contrôleurs sont modifiés, par exemple la forme des symboles, leur régularité, leurs espacement, en fonction de telle ou telle méthode de défuzzyfication, les non linéarités augment très vite en importance. Les contrôleurs flous peuvent donc être utilisés pour concevoir des contrôleurs linéaires en certaines zones, puis non linéaires en d'autres, en modifiant certaines règles et pas d'autres...
Leur souplesse est très importante [Bucley et Ying, 1989; Ying et al, 1990].

3.3.5 Simplification de contrôleurs flous.

Les contrôleurs flous sont très simples à construire, cependant leur représentation interne dans un calculateur peut nécessiter beaucoup de place.

Le problème vient du nombre de règles nécessaires pour identifier ou commander un système. Par exemple si une représentation utilise 5 variables, chacune étant caractérisée par 5 symboles, il faudra 3125 règles différentes pour couvrir tous les cas, tous les environnements possibles.

Si un algorithme de calcul est utilisé, générer 3125 règles n'est pas plus difficile qu'en générer une centaine, mais pour un opérateur humain cela risque d'être assez fastidieux. De plus, la base de règles résultante risquera de n'être plus lisible. Mais en dehors de ces problèmes de compréhension, des problèmes plus importants de temps de calcul apparaissent. En effet il faudra pouvoir implanter dans le calculateur ces m règles, et si m est très grand, il faudra un calculateur avec un très grand espace mémoire. De même les temps de calculs, bien qu'un certain parallélisme puisse être envisagé, risquent d'être aussi très longs.

Ce problème de complexité des contrôleurs flous, au sens de la taille, est un inconvénient important en pratique. Celui-ci n'arrive dans la réalité que lorsque des algorithmes d'identification sont utilisés, soit pour construire entièrement la base de règles, soit pour enrichir celle fournie par un opérateur humain. En effet ces derniers ne conçoivent jamais de bases de règles gigantesques.

Quelques solutions ont été envisagées, suivant que la base de règles est déjà construite ou non [Lacrosse, 1996].

3.3.5.1 La base de règles existe déjà.

Dans ce cas il faut éliminer les règles inutiles, ou trop peu souvent utilisées.

3.3.5.1.1 Elagage

Cette méthode a pour but d'éliminer les règles qui physiquement ne sont jamais appliquées, par exemple à cause d'une incompatibilité des valeurs de certaines variables, ou encore qui ne sont que très peu appliquées, soit en intensité, soit dû à une rareté des circonstances.
Par exemple si un contrôleur flou était réalisé pour commander l'aménagement du territoire, notamment pour évaluer les zones constructibles, les règles qui s'appliqueraient lorsque les cours d'eau ont un débit tel qu'il ne se produit statistiquement que tous les 10000 ans seraient éliminées.

Cette méthode est appelée pruning en anglais.

La vérification se fait par les niveaux de déclenchements $\tau_i(x)$ de chaque règle.
Il faut indiquer un seuil τ_{min} et toute règle R_i telle que $\max_{x \text{ physiquement possible}} \tau_i(x) < \tau_{min}$ est éliminée.

Le paramètre de niveau de déclenchement minimal τ_{min} est bien sûr un paramètre à évaluer. Des méthodes d'optimisation peuvent être utilisées en fonction d'un critère d'efficacité ou d'erreur du contrôleur.

3.3.5.1.2 Fusion des symboles proches

Cette méthode est utilisée pour supprimer les symboles qui ont des sens très proches.

Pour comparer les symboles, les indices de ressemblance et de similarité définis en analyse de données, et rappelés dans le chapitre des possibilités, sont utilisés.

Il faut dans cette application un indice relativement précis, qui évalue la forme dans son ensemble, et pas seulement un indice qui évalue seulement la contradiction entre les deux symboles.

L'indice de Jacquard peut être utilisé pour comparer le symbole i au symbole j:

$$r_{ij} = \frac{\int_a^b \min(\mu_i, \mu_j)dx}{\int_a^b \max(\mu_i, \mu_j)dx} \tag{3.19}$$

Il faut aussi définir un seuil r_{max} au delà duquel les symboles sont considérés identiques. Ce seuil de ressemblance minimal est important, ainsi si r_{max} est mis à 0, tous les symboles seront unifiés. Il peut être optimisé aussi avec un critère d'efficacité ou d'erreur du contrôleur associé à la base de règles.

Pour la fusion des deux symboles, une approche linéaire est possible, qui pondère dans le nouveau symbole, le nombre de fois où sont utilisés les deux symboles dans la base de règles dans son ensemble. Soit n_i et n_j les nombres de fois où les symboles i et j sont utilisés, il s'ensuit alors que le nouveau symbole k résumant les symboles i et j est défini par:

$$\mu_k = \frac{n_i\mu_i + n_j\mu_j}{n_i + n_j} \tag{3.20}$$

3.3.5.1.3 Approximation de la surface de contrôle

Le précèdent paragraphe a montré que les contrôleurs flous réalisaient exactement des fonctions analytiques, bien que complexes.

Etant donné la surface entrée-sortie que représente le contrôleur flou, il est possible d'utiliser des outils mathématiques pour approximer cette surface au moyen de fonctions analytiques plus simples. Dans ce cas, toute référence au contrôleur flou original est oubliée, et seule une fonction l'approximant est conservée.

Les outils mathématiques correspondants relèvent du champ de l'approximation des fonctions, et ne sont pas développés ici. Il est possible de citer les courbes splines dans le cas 2D en définissant un maillage plus ou moins lâche.

3.3.5.2 Le contrôleur flou n'est pas construit.

Les méthodes développées dans ce paragraphe ont pour but de simplifier directement la base de règles au niveau de la conception, soit en réduisant le nombre de symboles, soit en réduisant la complexité des règles.

3.3.5.2.1 Symboles complexes

Cette approche a pour origine la couche cachée des réseaux de neurones.
Etant donné n variables, certaines d'entre elles sont agrégées pour donner naissance à des variables, plus complexes, mais surtout en plus faible nombre.

En général les méthodes d'agrégation sont linéaires, mais d'autres types de méthodes peuvent être utilisés.

La figure 3.19 montre le but de cette approche.

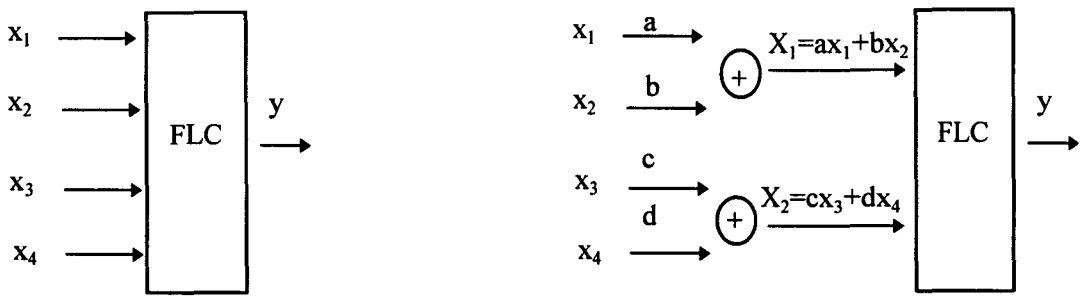

Fig. 3.19. Simplification de contrôleurs flous par des variables complexes.

Le problème de cette approche est qu'il faut pouvoir construire les variables complexes, et leur donner un sens pour le contrôleur simplifié afin de pouvoir en déduire des règles d'inférence.

Il est nécessaire d'avoir une certaine connaissance du système afin de pouvoir apparier certaines variables.

3.3.5.2.2 Principe de hiérarchie.

Cette approche suppose qu'il est possible de trier les variables d'entrée de telle sorte que des variables soient considérées plus importantes que d'autres, ces dernières agissant comme des correcteurs sur les premières sorties.

La figure 3.20 montre le principe de cette approche.

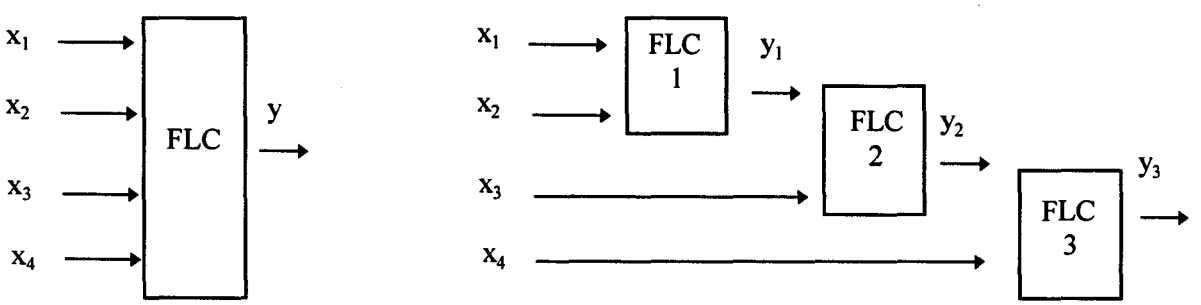

Fig. 3.20. Principe de l'approche par hiérarchie.

Chacun des contrôleurs flous est alors beaucoup plus simple que le contrôleur flou global. Mais le problème de cette approche est qu'il faut pouvoir introduire une notion de prépondérance entre les variables.

3.3.5.2.3 Contrôleur flou à couche

Cette approche reprend un peu la précédente, mais elle ne suppose pas une hiérarchie complète entre les variables. Elle simplifie donc moins les contrôleurs, mais elle peut être beaucoup plus souvent appliquée. En outre son principe apparaît très naturel.

Cette approche suppose que certaines variables sont plus importantes que d'autres, et qu'elles conditionnent le comportement des réponses en utilisant les autres variables.

Il existe donc une couche, appelée couche de supervision, qui fusionne les variables prépondérantes. Cette couche a en sortie des poids qui permettent alors de fusionner des contrôleurs flous fusionnant les variables moins importantes, qui sont tous traités en parallèle.

La couche de supervision sert à établir dans quel environnement global se situe le système, et les contrôleurs flous élémentaires règlent les détails avec les variables mineures. Ces contrôleurs élémentaires, encore appelés locaux, peuvent être en contradiction les uns avec les autres, le rôle de la couche de supervision est justement de savoir quel contrôleur local utiliser, et dans quelle mesure.

La figure 3.21 montre le principe de cette approche lorsqu'il y a simple commutation entre les contrôleurs flous locaux.

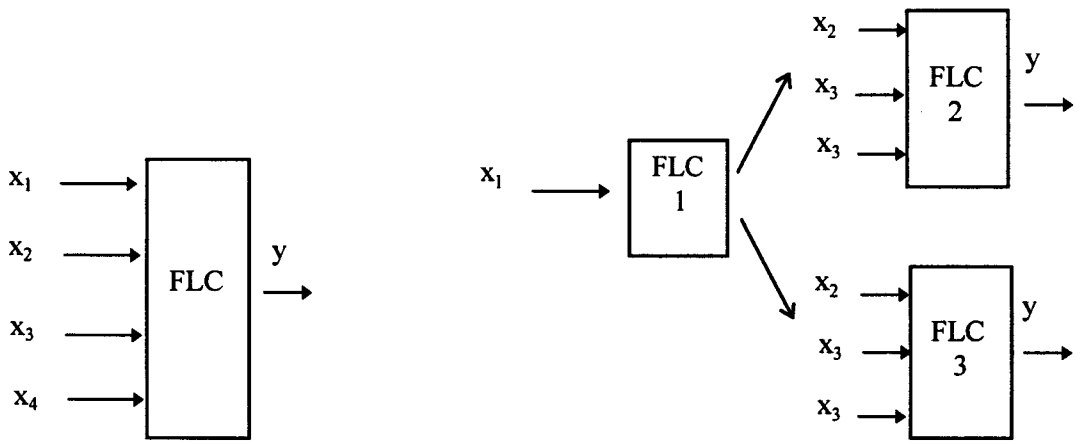

Fig. 3.21. Principe de contrôleur flous à couche (commutation).

Cette dernière méthode de simplification est une des idées qui amènent à la notion de contrôleur multi-modèle développée dans les prochains chapitres.

En effet lorsqu'un système complexe est analysé, il peut être intéressant de le modéliser par une série de modèles locaux, par exemple des linéarisations autour de points de fonctionnement. Dans ce cas il faut un algorithme qui permette, en fonction de l'environnement, de choisir quel modèle est le plus approprié. Cet algorithme est dans ce paragraphe le contrôleur flou superviseur qui choisit quel contrôleur flou local appliquer. Ces contrôleurs élémentaires sont les modèles utilisés pour décrire le système.

3.4 Conclusion: Apport de la logique floue en automatique

Le contrôleur flou, c'est à dire l'application de la théorie des ensembles flous à la modélisation et la commande, a apporté à l'automatique de nouveaux moyens d'identification et de commande totalement différents des approches classiques.

En permettant de décomposer la solution un problème en une série de règles énonçant un comportement adapté à un environnement particulier, représenté par les prémisses des règles, comme le fait un opérateur humain, la logique floue a simplifié considérablement la synthèse de contrôleurs complexes. En modélisation pure, si les conclusions des règles ne sont plus des variables de commande, mais des sorties à estimer, les apports de la logique floue sont aussi

considérables. Cette approche par décomposition différe totalement des approches classiques, qui supposent une synthèse directe d'un contrôleur ou d'un modèle dans sa totalité.

A l'origine les contrôleurs flous étaient extraits de la connaissance humaine, et si certains contrôleurs sont encore construits de cette manière, de plus en plus de méthodes de modélisation et de commande purement floues, sans intervention humaine, sont mises au point pour la construction de contrôleurs flous à partir de données entrées-sorties.

Cependant de plus en plus des systèmes très complexes sont étudiés pour lesquels même la logique floue, sous la forme présentée dans ce chapitre, est insuffisante, à moins de requérir des structures gigantesques sous forme de bases de règles.

Or le contrôleur flou possède en son sein même les germes qui donnent naissance à l'analyse multi-modèle. En quelques mots cette dernière étudie les systèmes très complexes par le moyen d'une série de modèles valides sous certaines conditions, un superviseur se chargeant de choisir quel modèle est le bon pour un environnement donné, et dans quelle mesure. Cette dernière approche est l'aboutissement du contrôleur flou dans son esprit de décomposition, même si certaines analyses multi-modèles ne font plus aucune référence à la logique floue.

Cette approche puissante pour l'identification et la commande des systèmes sera exposée dans les prochains chapitres.

CHAPITRE QUATRIEME

FONDATIONS DES APPROCHES MULTI-MODELES

L'union fait la force
Proverbe commun

4.1 Origine des approches multi-modèles

En automatique il se révèle parfois impossible de représenter un système très complexe par un modèle qui en traiterait tous les aspects particuliers, tous ses comportements. Les origines de ce problème sont multiples: par exemple une trop grande complexité des phénomènes physiques, de fortes non linéarités ou encore des incertitudes sur les mesures, ou les variables mises en jeu. De même un problème similaire apparait lorsqu'un modèle a pu finalement être estimé, mimant fidèlement les réponses du système, mais cette fidélité se payant d'une trop grande complexité, le modèle se révèle finalement inutilisable. Ainsi le calcul d'une commande optimale avec un critère quadratique se révèle vite impossible pour des systèmes relativement simples.
Une solution consiste alors soit à étudier localement le système pour définir un modèle, soit, si un modèle a déjà été défini, à le linéariser autour d'un point de fonctionnement. Le modèle obtenu est alors plus simple que le système, et il est alors facile d'en déduire des commandes efficaces. S'il arrive que le système évolue trop pour être toujours dans un voisinage du premier point de fonctionnement, alors plusieurs modèles simplifiés peuvent être définis au voisinage d'autres points de fonctionnement remarquables. Lors de l'évolution du système, il y a alors une série de commutations entre les modèles. Eventuellement des gains peuvent être introduits afin de lisser les transitions entre les commandes. Un exemple bien connu est celui des systèmes à deux dynamiques, rapide, et lente. Un premier modèle est alors défini pour le régime transitoire, en négligeant les constantes de temps lentes, et un second modèle est défini de même pour le régime permanent, cette fois en négligeant les constantes de temps rapides.

L'utilisation d'une série de représentations partielles d'un système est donc naturelle en automatique classique, et permet de répondre au problème du compromis entre robustesse et performance. En effet si un système est très complexe et qu'il faut le représenter par un seul modèle, simplifié en outre, alors les lois de commandes qui lui seront associées devront être robustes, afin de pallier aux erreurs de modélisations. Cette robustesse se fera au détriment des performances. Si au contraire plusieurs modèles sont définis, il sera alors possible de définir des commandes performantes, et la robustesse de l'ensemble sera acquise en considérant l'ensemble des commandes associées aux modèles localement parfaits, car l'erreur de modélisation finale sera plus faible.

De son coté la logique floue et son utilisation en automatique par l'intermédiaire du contrôleur flou a donné naissance a d'autres approches multi-modèles. En effet chacune des règles d'un contrôleur flou correspond à une vision particulière du système complet, les conclusions correspondant selon l'objectif poursuivi soit à la détermination des variables sorties du système pour des problèmes de modélisation, soit au calcul de commandes pour les problèmes de poursuite. Cependant pour réellement parler d'approche multi-modèle, impliquant une structure plus complexe que celle utilisant des conclusions réduites à des valeurs de variables, la structure du contrôleur flou a du être généralisée. La première généralisation fut présentée au chapitre précèdent, avec la notion de contrôleur à couches. Dans celle-ci un contrôleur superviseur choisit quel modèle est le plus pertinent parmi toute une série de modèles locaux, représentés par autant de contrôleurs flous particuliers. La seconde généralisation est connue sous le nom de modélisation de Takagi-Kang-Sugeno. Dans cette approche les conclusions des règles sont en effet généralisées à de véritables fonctions:

(R_i) Si x_1 est X_{1i} et...et x_n est X_{ni} Alors y est $f_i(x)$ (4.1)

Avec cette approche les fonctions $f_i(x)$ sont la plupart du temps des polynômes de la variable x. $f_i(x)$ représente alors un modèle particulier dont la pertinence ou la validité est estimée par les prémisses de la règle R_i. La fusion des n fonctions $f_i(x)$ par le système d'inférence crée alors une fonction de sortie f(x) fortement non linéaire.

Il apparaît donc que dans plusieurs théories différentes est définie de manière concourante la notion de modélisation et de commande multi-modèles. De manière imagée, "lorsqu'un écheveau compliqué doit être démêlé, il est plus utile de faire appel à plusieurs petites mains agiles qu'a une seule grande et musclée". En automatique la même approche sera utilisée: plusieurs modèles simples seront plus maniables qu'un seul incompréhensible.

Cependant cet aphorisme ne devrait pas cacher le fait que l'application des approches multi-modèles soulève de nouveaux problèmes, par exemple comme l'estimation de la pertinence d'un modèle élémentaire, ou encore la réalisation de la fusion des modèles ou des commandes élémentaires afin d'obtenir un seul modèle ou une seule commande globale. Les principes généraux des approches multi-modèles seront exposés dans ce chapitre, et des améliorations et développements possibles seront présentés aux chapitres suivants.

Dans un souci de perspective synthétique, la structure générale des approches multi-modèles sera exposée dans une première partie. Ensuite chacune de ses composantes fondamentales, c'est à dire l'estimation de la pertinence instantanée des modèles élémentaires et la fusion de ceux-ci seront détaillées plus précisément. Finalement afin de montrer la souplesse et la puissance des approches multi-modèles pour l'automatique, de nombreux exemples concluront ce chapitre.

4.2 Structure générale des approches multi-modèles.

Les figures 4.1 à 4.3 montrent la modélisation d'un système par une approche multi-modèle.

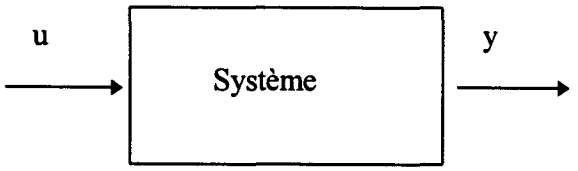

Fig. 4.1. Le système est une boite noire.

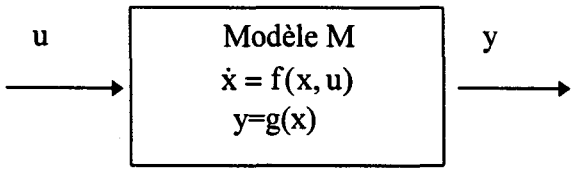

Fig. 4.2. Modélisation du système par un seul modèle.

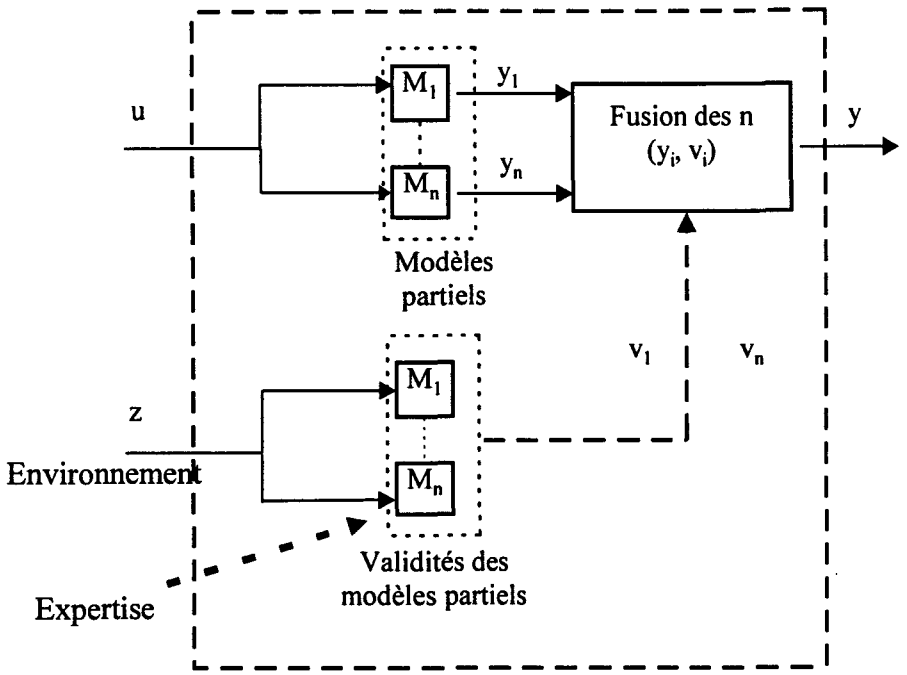

Fig. 4.3. Modélisation du système par une approche multi-modèle à modèle global implicite (les fonctions f et g ne peuvent pas être décrites explicitement).

Tout système est en fait un lien entre des sorties et des entrées. Sur la figure 4.1 les entrées sont notées $u \in U$ et les sorties $y \in Y$. Si le système est considéré comme une boite noire, une modélisation faisant appel à un seul modèle permet alors d'avoir une formulation mathématique du lien entre les entrées et les sorties, sous la forme des fonctions f et g.

Avec un modèle multi-modèle, le schéma est radicalement diffèrent, et sa structure apparaît déjà nettement plus complexe. En effet comme son nom l'indique, un multi-modèle fait appel à plusieurs modèles. C'est pourquoi dans la figure 4.3 il y a plusieurs modèles M_i pour i=1 à n. Chaque modèle est en fait une formulation mathématique simplifiée du système. Simplifiée car l'objectif principal d'une approche multi-modèle est justement de permettre la réduction de la complexité d'un système en l'étudiant sous certaines circonstances particulières.
Aussi chaque modèle n'est pas une représentation fidèle du système, et en général il est même faux, sauf exceptionnellement lorsque le système a le comportement local correspondant.
Ceci explique qu'il soit nécessaire avec les *Contrôleurs Multi-Modèles* (CMM) de pouvoir déclarer fidèle ou non un modèle en fonction de différents paramètres, et de pouvoir quantifier cette fidélité.

En fait dans tous les multi-modèles, à coté de l'ensemble des modèles, ensemble appelé par la suite bibliothèque, est toujours présent un mécanisme évaluant la pertinence de chaque modèle. Dans la figure 4.3 ce mécanisme calcule les indices v_i représentant cette pertinence, encore appelée validité. Cette quantité peut aussi s'interpréter comme un indice de confiance ou de fiabilité du multi-modèle dans le modèle M_i. Les entrées de ce mécanisme sont notées $z \in Z$, Z étant appelé l'environnement. Elles peuvent être de natures très diverses, comme des entrées du système, certaines variables internes des modèles, des paramètres environementaux ou contextuels, comme par exemple un sentiment de préférence propre à l'utilisateur, indépendamment de toute autre référence.

Enfin dans la structure interne d'un multi-modèle apparaît un troisième élément, très important lui aussi. Il s'agit de l'opération de fusion des modèles élémentaires, en fonction de leur validité courante.
Etant donné les sorties des modèles M_i et les validités des modèles correspondant, le mécanisme de fusion calcule une valeur de sortie résumant toute la connaissance incluse dans le multi-modèle, et non plus dans un seul modèle en particulier.
Le titre de la figure est *Multi-Modèle à Modèle Global Implicite* (3MGI). En effet si la fusion est effectuée sur les sorties des modèles, le multi-modèle permettra d'estimer à chaque instant une sortie y en fonction de l'entrée. Cependant il n'y aura aucune indication de la nature du lien entre ces variables.
Il est alors possible d'introduire une seconde classe de multi-modèles fondamentalement différente de la première exposée, appelée classe des *Multi-Modèles à Modèle Global Explicite* (3MGE). Son schéma est donné à la figure 4.4.

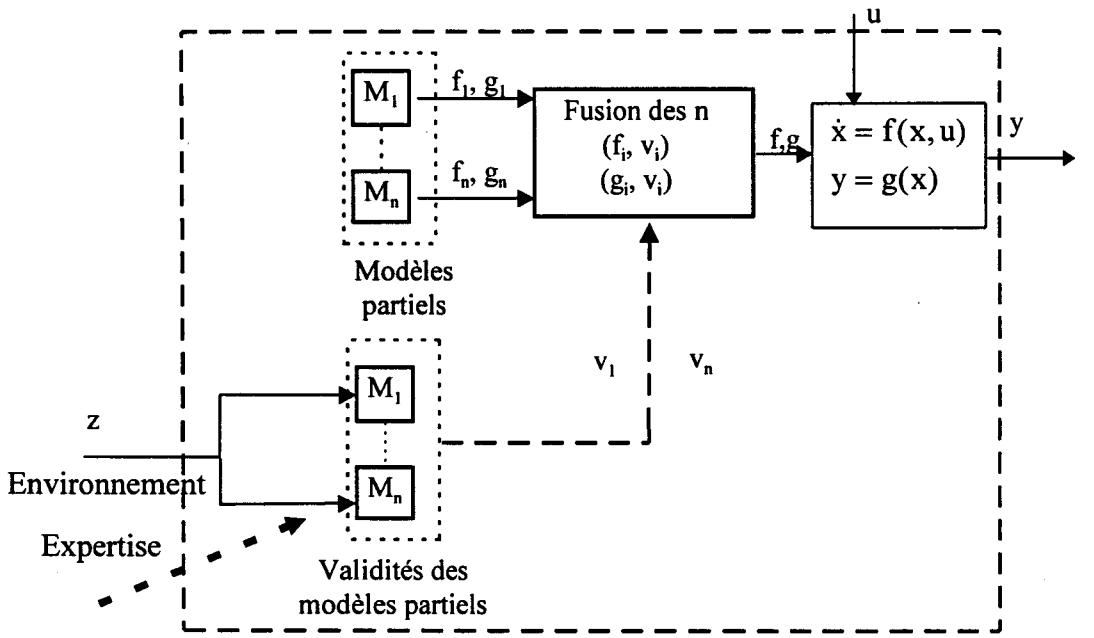

Fig. 4.4. Structure de multi-modèle à modèle global explicite (3MGE).

Avec cette classe de multi-modèles la fusion est effectuée non plus sur les sorties y_i de chaque modèles, mais sur les modèles eux-mêmes, par l'intermédiaire de leur formulation mathématique sous forme de fonctions f_i et g_i.
Dans ce cas, pour définir l'opération de fusion, il est alors nécessaire d'introduire la contrainte très forte que tous les modèles aient la même structure. Ainsi la structure de la fonction f est identique à celle des modèles, ceux-ci variant simplement sur la valeur des coefficients requis. Par exemple Johansen [1995] utilise des modèles ARMAX, chaque modèle M_i ayant des paramètres différents. Le modèle final est donc lui aussi de structure ARMAX.

Si cette classe de multi-modèle apparaît intéressante pour construire effectivement des modèles de systèmes complexes et étudier certaines propriétés, elle offre cependant moins de liberté et de souplesse que la classe des multi-modèles à modèle global implicite.

Jusqu'à ce point de présentation des multi-modèles, seul l'aspect de la modélisation a été envisagé. Néanmoins les multi-modèles peuvent être utilisés aussi en commande, donnant alors naissance aux CMM. La distinction entre 3MGI et 3MGE se prolonge naturellement

avec cet aspect. Les figures 4.5 et 4.6 donnent les structures générales, que ce soit pour un objectif de poursuite ou de régulation.

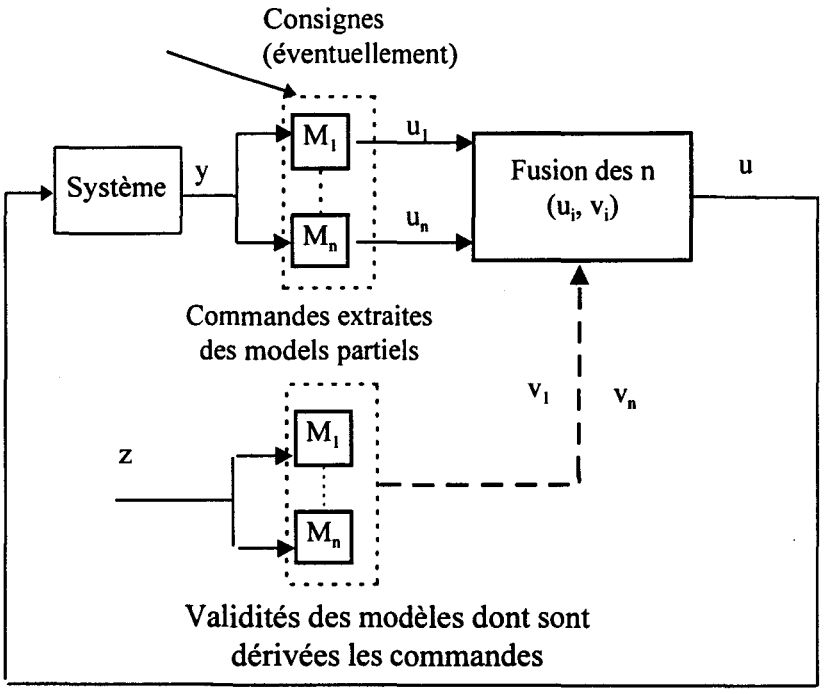

Fig. 4.5. Structure d'un contrôleur multi-modèle à modèle global implicite.

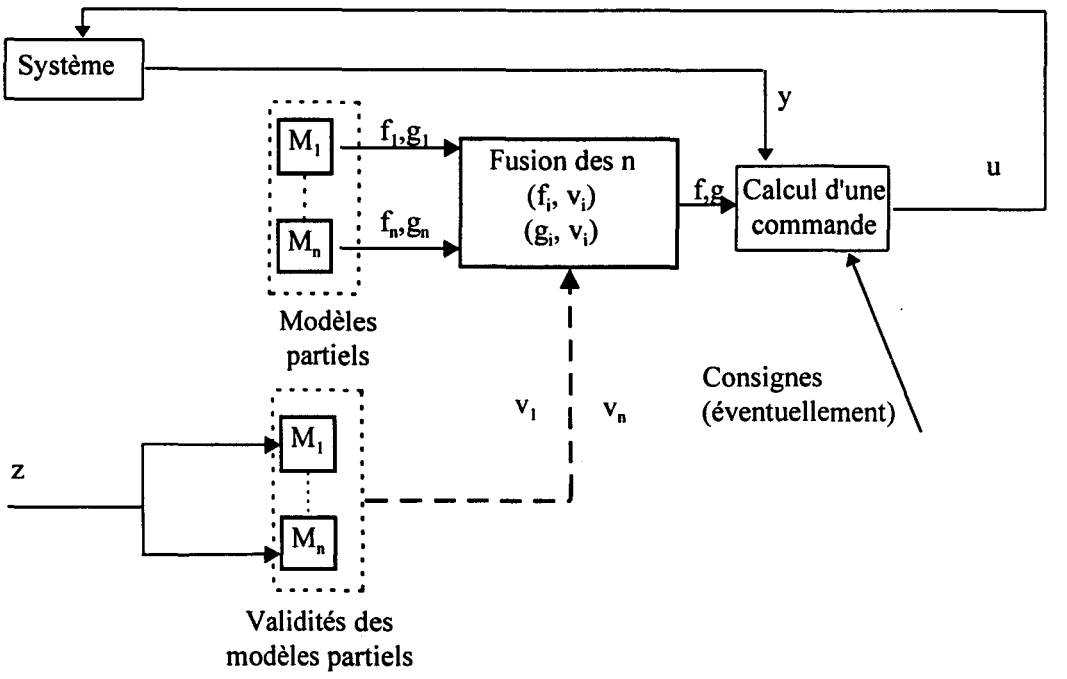

Fig. 4.6. Structure d'un contrôleur multi-modèle à modèle global explicite.

Un question qui vient immédiatement à l'esprit en examinant la structure représentée à la figure 4.5 est que les sorties des modèles sont cette fois des commandes u_i et non plus des sorties du système y_i. En effet dans ce schéma de contrôleur a été posée l'hypothèse qu'à chaque modèle correspondait une et une seule loi de commande possible (mais elles peuvent

être éventuellement distinctes pour les différents modèles). Aussi il est possible d'identifier la validité du modèle et la validité de la loi de commande qui lui est associée, puisque dans ce schéma seul l'aspect loi de commande intervient.
Bien entendu cette restriction n'est pas inhérente à la nature des approches multi-modèles, et il est tout à fait envisageable de considérer plusieurs lois de commande différentes pour chaque modèle considéré. Chaque loi de commande aura ses avantages propres, et donc des degrés de validité distincts de ceux des modèles. La structure d'un CMM à n modèles et m types de lois de commande, est néanmoins plus complexe, et sera exposée au 5ème chapitre.

Avec les C3MGE, il n'existe qu'un seul type de loi de commande, celle effectivement appliquée au modèle global.

En résumé, il ne faut retenir que trois éléments caractéristiques des approches multi-modèles. D'abord il y a toute une série de modèles, chacun représentant un comportement particulier du système étudié. Ensuite il y a le mécanisme évaluant les validités des modèles, il est appelé souvent superviseur dans la littérature. Enfin il y a l'étape de fusion des informations fournies par les modèles, en fonction de leur fiabilité estimée.

Par ailleurs il est possible de classer les différents approches rencontrées. Une première classe n'effectue pas de fusion sur les modèles, mais sur leur sorties, et ne donne par conséquent pas de modèle général de manière claire, la seconde effectue au contraire une fusion sur les modèles eux-mêmes, et donne donc une formulation analytique du système.

Les paragraphes qui suivent vont revenir plus en détail sur les différents constituants d'un multi-modèle.

4.3 Les modèles

Les modèles mis en œuvre dans un CMM représentent des vues partielles du système complet. Vues partielles parce que l'objectif d'un CMM est de réduire la complexité d'un système par une série de modèles simples. Chaque modèle pourra par exemple reproduire le comportement du système au voisinage d'un point de fonctionnement particulier. Il est aussi possible d'utiliser des modèles robustes, très simples et jamais parfaits précisément mais correspondant peu ou prou au système en général (l'estimation de leur validité sera envisagée au paragraphe suivant), de telle sorte que le calculateur puisse réagir à toutes les situations. L'objectif est d'avoir des modèles utilisables et maniables, soit pour étudier localement certaines propriétés, soit pour en déduire des commandes performantes.

Une question fréquemment soulevée à l'encontre des approches multi-modèles est l'obtention de ces n modèles. La réponse est que les mêmes méthodes qu'en automatique classique sont utilisées. Il faut en effet insister sur l'idée que les CMM, *loin de se substituer aux résultats de l'automatique classique, ou de les remplacer, les englobent.* Aussi toutes les méthodes de modélisation et d'identification peuvent être appliquées.

Ainsi tel modèle M_i correspondra à la linéarisation du système dans tel voisinage, tel autre correspondra à la linéarisation au voisinage d'un autre point.

Si aucun modèle n'est disponible alors il faudra utiliser des algorithmes d'identification. Ceux-ci pourront être utilisés avec un sous ensemble de toutes les mesures, afin de définir plusieurs modèles locaux, correspondants chacun à un sous ensemble de données particuliers.

Certains modèles peuvent être fournis par une simple expertise. Par exemple un modèle flou approximatif sera utilisé si tous les modèles locaux sont faux, lorsque le système diverge vers une zone imprévue. A ce modèle approximatif sera associé une commande particulièrement robuste, dont le seul but sera d'assurer la sécurité de fonctionnement de l'ensemble.

La seule limitation à cette liberté, au moins pour les C3MGI, les C3MGE devant conserver la même structure, est qu'il faut conserver en mémoire la méthode de construction de ces modèles, afin de pouvoir par la suite en déduire la validité en cours de fonctionnement. Ainsi si un modèle est le linearisé du système en un point, il faudra informer le superviseur calculant les validités que ce modèle est bien valide pour cet environnement particulier.

A coté des méthodes classiques de modélisation ont été développés des algorithmes d'identification dans le cadre de la logique floue. Bien que peu connu dans la communauté de l'automatique classique, ils sont souvent utilisés dans les articles traitant des multi-modèles, car ces derniers sont encore majoritairement abordés par la logique floue. Aussi il n'est pas inutile de les mentionner rapidement.

4.3.1 Classification floue

Cette méthode de modélisation est une généralisation de la classification classique (en particulier les méthodes par les k plus proches voisins et celles minimisant l'inertie intra-classe par rapport à l'inertie inter-classe).
Cette approche, utilisée dans les contrôleurs flous lorsqu'il n'y a pas d'experts disponibles pour donner un modèle, repose sur un ensemble de mesures entrées-sorties. Ces mesures seront alors groupées en différentes classes. En classification classique, ces classes sont des ensembles normaux, avec la classification floue, ces classes sont des sous ensembles flous caractérisés chacun par une fonction d'appartenance. Les algorithmes sont plus complexes qu'avec les méthodes classiques [Bezdek, 1981].

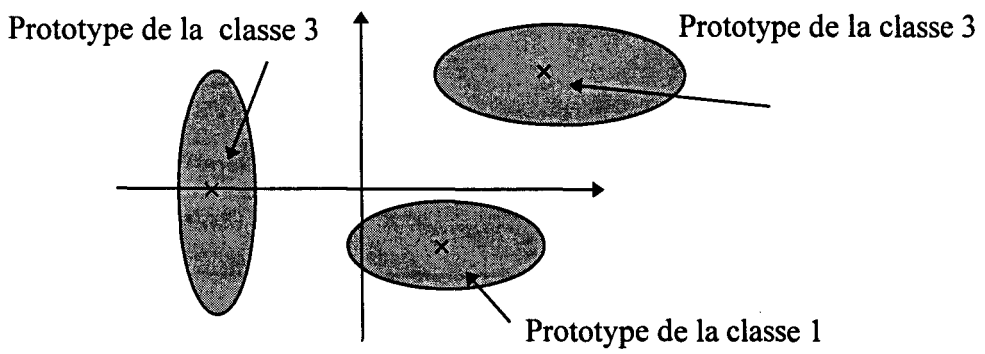

Fig. 4.7. Classification floue (fuzzy clustering) donnant trois modèles.

Une fois que chaque classe a été définie, il s'agit de lui associer un modèle. Ce modèle est appelé prototype, et est en fait l'individu (i.e. la mesure) de la classe correspondante qui est le moins différent par rapport aux autres individus de la même classe (i.e. il faut minimiser un critère).

A partir de ce prototype représentatif d'un lien ou d'un comportement entre les entrées et les sorties, il est possible de définir mathématiquement un modèle.
En outre avec la classification floue, la validité des modèles obtenus est directement fournie par l'algorithme, puisque la validité du modèle est celle de la classe correspondante, qui a été forcèment calculée dans sa construction.

En conclusion cette modélisation donne des systèmes de multi-modèle dont la structure est une base de règles du type de l'équation 4.1, lorsque les entrées et sorties sont appelées en variables x_i et y, c'est à dire des multi-modèles du type de Sugeno-Takagi-Kang. Wang [Wang, 1996] utilise la classification floue avec des ensembles flous discretisés. Yager [Kelman et Yager, 1997] a présenté aussi une autre approche de classification floue. Elle permet, en utilisant l'approche géométrique et des domaines de validités réduits à des singletons, de trouver les paramètres des modèles, linéaires, ainsi que les centres de validités.

Une version simplifiée de cet algorithme a été donnée au chapitre 3 dans le paragraphe des fonctions élémentaires floues 3.3.1. Dans ce cas, les classes sont calculées a priori en réalisant une partition floue des entrées variables selon des critères de lisibilité ou une connaissance experte. Cette classification imposée n'est cependant pas définie dans l'espace des entrées-sorties, mais simplement dans l'espace des entrées, et il s'agit ensuite de calculer un prototype, en introduisant les variables sorties.

4.3.2 Systèmes d'équations de relations floues

Ce problème a été défini indépendamment de l'automatique, il a pour but de résoudre des systèmes d'équations maniant des variables floues. Cependant ce type d'équation étant la plupart du temps basé sur l'implication floue, fonction utilisée aussi dans les contrôleurs flous, cette approche se révèle intéressante pour la modélisation.

Afin d'augmenter la rapidité des calculs, les résultats obtenus sont la plupart du temps définis pour des ensembles flous discrets (voir 3.2.2):

Si les entrées sont caractérisées par n labels $Label_i$ et les sorties par m labels $Labels_j$ (définis a priori), étant donné une mesure entrées-sorties floue (x,y), leurs transformations en ensembles flous discrets (x',y') sont données par $x'=(x'_1,...,x'_n)$ et $y'=(y'_1,...,y'_m)$ avec x'_i=Compatibilité(x_i, $Label_i$) et y'_j= Compatibilité(y_j, $Labels_j$), avec Compatibilité un opérateur de compatibilité (voir 2.71 ou 3.19).

Une équation de relation floue est d'abord définie lorsqu'il y a une et une seule mesure (x,y). L'utilisateur choisit alors un système d'inférence particulier, par exemple une inférence de Mamdani.

Le problème est alors de trouver la règle R représentant la connaissance telle $x' \bullet R = y'$.

Comme x' et y' sont discrets, les inconnues forment un ensemble fini de poids $R=(r_{ji})$, et le choix du système d'inférence donne alors une série d'équations classiques liant les variables numériques classiques x'_i, y'_j et r_{ji}.

En général il existe plusieurs solutions, suivant le choix du système d'inférence.

Si plusieurs mesures (x,y) sont disponibles, un système d'équations de relation floues est alors obtenu.

Etant donné K couples $(x,y)_k$, il ne faut cependant obtenir qu'une seule matrice R. En général les solutions définies une par une pour chaque mesure ne se recoupent pas, et il n'existe en général pas de solution parfaite. Une solution approximative est obtenue par des interpolations sur les K matrices R_k.

Cependant si le système d'inférence choisi est celui de Mamdani, comme dans la plupart des contrôleurs flous, il est possible de définir une solution R exacte.

Le résultat final est donné par

$$r_{ji} = \min_k \tau(y'_{ik}, x'_{jk}) \tag{4.2}$$

avec τ l'opérateur de maximisation défini au premier chapitre.

Etant donné ce modèle flou, il est alors possible de calculer de nouvelle sortie y_{nouv} étant donné une nouvelle entrée x_{nouv}, par une simple application de l'inférence de Mamdani:

$$y_{nouv,j} = \max_i(\min(r_{ji}, x_{nouv,i})) \tag{4.3}$$

Une étude exhaustive de ce problème se trouve dans [Dubois, 1993] et dans [Lamotte, 1993].

4.3.3 Type de modèles fréquents

Ce paragraphe donne les principaux modèles utilisés jusqu'à présent.

Historiquement les premiers modèles avaient généralisé les conclusions des règles des contrôleurs flous à des fonctions des prémices.
Les modèles étaient donc sous la forme

$$y = f_i(x)$$

En général les f_i sont des polynômes de x. Sugeno [Takagi et Sugeno, 1985] utilisèrent une telle approche. En fait le modèle était statique.
Les travaux actuels portent sur des modèles plus complexes. Par exemple il est possible de citer Johansen, Wang, Nakamori et Ishigame, qui utilisent des modeles ARX:

$$y = ARX_i(y,u)$$

De même, Tanaka, Cao et Narendra utilisent des modèles linéaires classiques:

$$\dot{x} = A_i x + B_i u \quad y = C_i x$$

D'autres modèles peuvent être utilisés suivant les circonstances. Ainsi Pedrycz [Pecrycz, 1995] utilise la classification floue pour modéliser des relations, et non des fonctions.

4.4 Estimation de la pertinence des modèles

Le superviseur a pour objectif l'estimation de la validité des modèles. Cette fonction évaluant les indices de qualité $v_i(z)$ des modèles est de première importance.

En effet étant donné les différents modèles, il s'agit, avant d'effectuer la fusion, de savoir quel modèle est correct ou non, et dans quelle mesure il faudra l'utiliser pour le modèle global, en fonction de l'utilisation souhaitée (modèle de commande ou de simulation.

Par ailleurs lorsqu'un système non linéaire est linéarisé en certains points de fonctionnement, il y a une perte d'information. Les validités qui conditionnent les poids relatifs des modèles dans la fusion peuvent alors jouer un certain rôle à ce niveau, car leur ajustement peut permettre de récupérer une certaine information sur le système non linéaire initial.

Dans la plupart des CMM, les indices v_i sont compris entre 0 et 1. Par définition si la validité v_i d'un modèle M_i est égale à 0, celui-ci est considéré comme absolument faux et il doit avoir une influence nulle sur le modèle final. Au contraire si la validité d'un modèle est égale à 1, le modèle final devra, en théorie, être identique au modèle correspondant jugé parfait et idéal (en théorie seulement, car bien souvent cette contrainte n'est pas respectée dans la plupart des CMM).

Bien souvent le calcul de la validité d'un modèle est lié à la façon dont a été obtenu ce modèle. La figure 4.8 résume les principales approches pour un modèle M_i donné, car les approches peuvent être différentes pour l'ensemble des modèles.

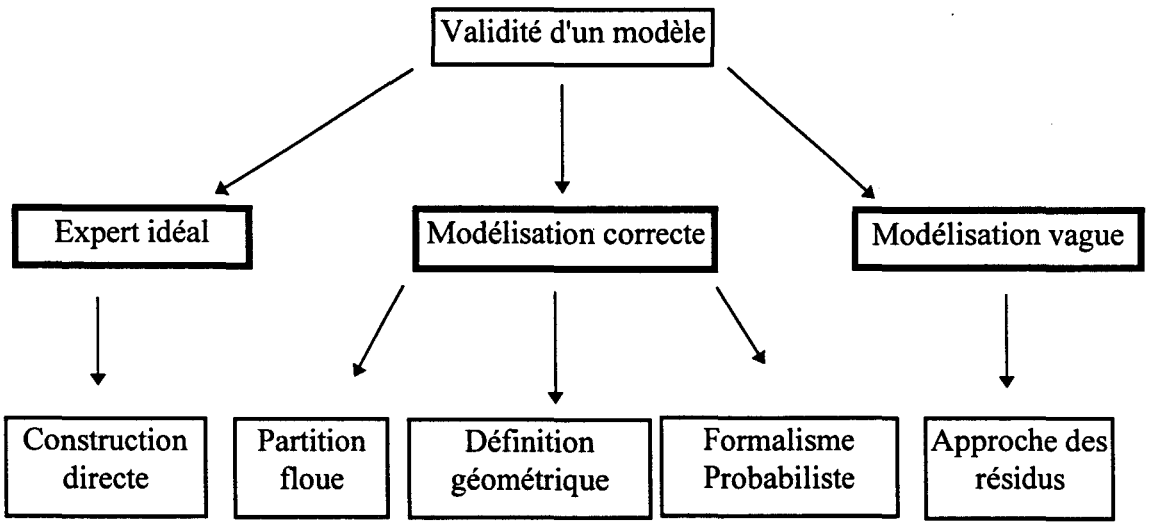

Fig. 4.8. Les différentes approches concourant à l'estimation des validités des modèles.

La première approche est un cas d'école, et est mentionnée uniquement pour faire contraste avec les autres. Elle suppose une telle connaissance que la fonction $v_i(z)$ peut être directement construite. Par exemple un expert idéal pourrait estimer la validité d'un modèle par simple comparaison avec le système. Cette approche permet aussi de forcer ou de transformer la fonction $v_i(z)$ de telle sorte que la validité d'un modèle soit surestimée ou sous-estimée une fois qu'elle a été obtenue par d'autres moyens.

Les trois approches suivantes, qui utilisent respectivement un formalisme flou, géométrique ou encore un formalisme probabiliste, supposent moins d'information a priori sur les liens entre les modèles et le système étudié. Par exemple tel modèle est connu pour être valide a peu près dans telle ou telle situation, tel autre modèle est admis fiable pour d'autres circonstances. Si ces trois approches représentent les mêmes connaissances en vue du calcul de la validité d'un modèle, elles diffèrent sur les moyens d'y parvenir. Jusqu'à présent le terme local a souvent été utilisé pour les modèles. Ceci fait directement appel à un formalisme géométrique et ce type de notion est utilisé dans la troisième approche. La seconde approche modélise cette notion de voisinage sous forme de fonction d'appartenance liée à des partitions floues, et est utilisée avec les CMM de Sugeno-Takagi-Kang.
La dernière approche peut être considérée comme plus riche. Elle ne suppose pas d'information a priori et se base en quelque sorte sur une connaissance a posteriori. Elle permet par exemple d'évaluer la validité de modèles grossiers qui ne sont jamais vraiment exacts.

Les paragraphes qui suivent détaillent les dernières méthodes, ainsi que certaines opérations appliquées aux coefficients de validité une fois qu'ils sont obtenus.

4.4.1 Formalisme flou

Lorsque des méthodes d'identification floues sont utilisées en général l'algorithme donne l'ensemble des modèles sous la forme d'une série d'équations du type 4.1. C'est la base de règles du contrôleur flou généralisé.
Dans ce cas les prémisses de chaque règle permettent de mesurer si telle règle s'applique, et donc si le modèle correspondant est valide ou non.

Si le système d'inférence est simplifié et si les seuils de déclenchements τ_i de chaque règle R_i sont introduits (relation 3.11), alors les indices de validité v_i sont ces seuils de déclenchements τ_i.
Par exemple avec un CMM de Sugeno-Takagi-Kang, le résultat final après fusion généralise la relation 3.13 et donne pour f:

$$f(x) = \frac{\sum_{i=1}^{n} f_i(x)\tau_i(z)}{\sum_{i=1}^{n} \tau_i(z)} \tag{4.4}$$

($Z \subset X$ avec ces CMM).

4.4.2 Approche géométrique

La présentation de cette approche sera plus longue que la précédente, puisque cette dernière bénéficiait des résultats présentés au chapitre 3. Elle est cependant très naturelle en automatique classique.

Par exemple si le système a été linéarisé au voisinage d'un point de fonctionnement, celui-ci sera référencé dans Z et le fait que le modèle correspondant soit valide en ce point sera connu.

A mesure que l'environnement courant z de Z s'éloignera de ce point de fonctionnement, il sera logique de penser que la fiabilité du modèle associé décroîtra.

Par exemple en économie, si un modèle a été défini pour un taux de croissance de 1%, celui ci sera bon si le taux de croissance est de 1,1%, mais il sera très mauvais si le taux de croissance est de 6% par an, ou au moins ce modèle sera moins bon que le modèle conçu pour un taux de croissance de 4% par an.

En général, la mesure du point de fonctionnement nécessite des observateurs (cf. 5.2.2.1).
Pour représenter ces raisonnements est associé au modèle M_i un domaine de validité D_i dans Z. Ce domaine de validité représente une zone dans laquelle il est connu que le modèle a un certain degré de validité. Ce degré de validité certain dans le domaine de validité D_i est représenté par la validité propre $c_i \in [0,1]$. Il est possible d'apparenter ce degré de validité propre aux degrés de confiance assignés aux règles dans un contrôleur flous (3.3.1.1).

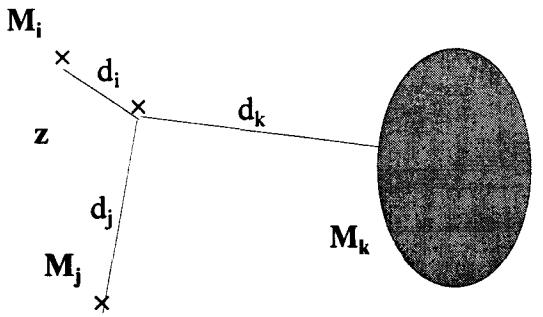

Fig. 4.9. Calcul des validités v_i avec une approche géométrique. M_i et M_j ont des domaines de validité ponctuels, alors que M_k a un domaine de validité D_k plus important.

Le calcul de la validité $v_i(z)$ est alors le suivant [Delmotte et al, 1996a]:

A chaque étape la distance d_i entre z et D_i est calculée. Il faut alors normaliser cette distance. Deux approches sont possibles. La première consiste à introduire la plus grande distance d_m admissible dans l'espace Z. Dans ce cas la distance normalisée est $d'_i = d_i/d_m$.
Le seconde approche consiste à tenir compte de l'ensemble des distances d_j pour obtenir des distances relatives:

$$d'_i = \frac{d_i}{\sum d_i} \tag{4.5}$$

Cette méthode permet de ne pas introduire de paramètre supplémentaire d_m.

Cette distance permet d'avoir un premier indice de validité par l'introduction d'une fonction $g(d'_i)$: $[0,1] \rightarrow [0,1]$, g décroissante avec $g(0)=1$ et $g(1)=0$. L'exemple le plus simple donne:

$$v_i = 1 - d'_i \tag{4.6}$$

Si par exemple la mesure z est équidistante de tous les domaines de validité, alors ils auront tous une validité v_i égale à 1-1/n. Cette valeur tend vers 1 lorsque le nombre de modèles devient grand, indépendamment de la situation précise, et cette situation peut survenir si les modèles sont tous bons ou tous faux. Ceci est en contradiction avec la définition de l'indice de validité, mais cette situation a peu de chance de se rencontrer dans la réalité. De toute façon

ceci ne remet pas en cause l'algorithme en entier, parce qu'à l'étape de fusion, si tous les modèles ont le même indice de validité, que celui-ci soit proche de 0 ou de 1 ne change pas le résultat, et il reste toujours la possibilité de commuter alors sur une normalisation faisant intervenir le paramètre d_m.
Une situation beaucoup plus fréquente est celle où z est proche d'un domaine, et loin des autres. Dans ce cas la validité du bon modèle sera proche de 1 et celles des autres proches de 0.

Si le paramètre de validité propre c_i est diffèrent de 1, alors le résultat final est

$$v_i=c_i(1-d'_i) \tag{4.7}$$

Dans [Dubois, 1995b] des domaines de non validité furent introduits pour représenter les connaissances affirmant que pour tel environnement, le modèle étudié est absolument faux. Avec ces domaines de non validité, 4.7 devient:

$$v_i = c_i(1-d'_i)\frac{\neg d_i}{\neg d_i + d_i} \tag{4.8}$$

avec $\neg d_i$ la distance au domaine de non validité du i^{eme} modèle.
Neanmoins ces formules simples peuvent être mises en défaut lorsque le domaine D_i n'est plus connexe. Dans ce cas le domaine est une union de sous domaines connexes, et, complication suprême, chaque sous domaine peut avoir un degré de fiabilité propre diffèrent.

Ainsi un modèle pourra être considéré parfait pour certaines circonstances, et moyennement pour d'autres.

Il devient alors difficile d'estimer la fiabilité d'un modèle en fonction de ces différentes informations.
En effet l'état courant z peut être proche d'un sous domaine de validité, mais d'une faible fiabilité propre, et aussi loin d'un autre sous domaine, mais cette fois de validité propre élevée. Comme les distances sont normalisées, le problème apparaît complexe.

Il est cependant possible d'obtenir des indices en un temps raisonnable lorsqu'un seul domaine D_i n'est pas connexe: $D_i=\cup(D_{ik})$ pour k=1 à K, si K sous-domaines de validité sont définis, avec K indices de fiabilité propre.
Dans ce cas il existe pour n-1 modèles M_j, n-1 distances d_j pour j=1 à n sauf i, et K distances d_{ik} correspondant aux K sous domaines.

K indices de validité v_{ik} sont alors calculés pour le modèle M_i pour chacun de ses sous domaines:

$$v_{ik} = c_{ik}\left(1-\frac{d_{ik}}{\sum_{j=1,\, j\neq i}^{n} d_j + d_{ik}}\right) \tag{4.9}$$

Son indice de validité final v_i est alors

$$v_i=\max_k v_{ik} \tag{4.10}$$

Le distance d_i est alors définie comme celle correspondant au domaine de validité correspondant à la validité finale maximale:

$$d_i = d_{ik_m} \text{ avec } k_m = \mathrm{Arg}(\max_k v_{ik}) \tag{4.11}$$

et finalement il est possible de calculer les autres indices de validité v_j avec la relation (4.8).

Si plus d'un modèle est associé à des sous domaines, le calcul des validités finales devient trop complexe avec les distances normalisées, aussi vaut il mieux introduire la distance maximale d_m pour obtenir finalement:

$$v_i = \max_k \left[c_{ik} \left(1 - \frac{d_{ik}}{d_m} \right) \right] \tag{4.12}$$

4.4.3 Approche Probabiliste

Ce formalisme est très utilisé en classification et en théorie de la décision.
Par exemple en classification, il faut regrouper un grand nombre d'individus en grandes classes. En théorie de la décision, un exemple possible est donné par la médecine. Un docteur examine un patient, il remarque qu'il a de la fièvre et des boutons rouges sur le corps, et il doit trouver la maladie la plus vraisemblable.

De manière générale, il faut donc trouver la meilleure hypothèse possible, étant donné des observations, et des connaissances statistiques.

L'observation est Z, un vecteur de différent paramètres. Il y a n hypothèses H_i. Les connaissances statistiques sont de deux ordres. D'abord un grand nombre d'expériences a permis d'établir les probabilités a priori $P(H_i)$, indépendamment des observations (par exemple la peste est plus rare que le rhume). Ces données permettent de savoir quel hypothèse est la plus courante. Ensuite il faut pouvoir estimer précisément les densités de probabilités $p(Z|H_i)$, qui permettent de savoir quel type d'observations est lié plus particulièrement à tel type d'hypothèses.

Alors le problème de classification se résoud très simplement. La règle de Bayes permet en effet d'estimer les probabilités a posteriori $P(H_i|Z)$:

$$P(H_i|Z) = \frac{p(Z|H_i)P(H_i)}{p(Z)} \tag{4.13}$$

avec

$$p(Z) = \sum_{i=1}^{n} p(Z|H_i)P(H_i) \tag{4.14}$$

Habituellement Z est un vecteur, donc

$$p(Z|H_i) = \prod_{j=1}^{m} p(z_j|H_i) \tag{4.15}$$

Avec les problèmes de classification, l'étape de fusion n'existe pas en général. Ou alors elle apparaît sous forme discontinue. La meilleure classe ou la meilleure explication H^* est en effet celle qui maximise la probabilité a posteriori:

$$P(H^*|Z) = \max_i P(H_i|Z) \tag{4.16}$$

Cette démarche est tout à fait envisageable avec un contrôleur multi-modèle. Il faut alors définir les probabilités a priori de chaque modèle élémentaire, assimilé à une hypothèse, et les densités de probabilités liant l'environnement aux modèles.
La formule 4.13 permet alors d'estimer les probabilités a posteriori de chaque modèle en fonction des observations.

A l'étape de la fusion, il pourra alors y avoir une fusion discontinue, en choisissant le meilleur modèle possible, ou une fusion linéaire, en assimilant les probabilités a posteriori à des degrés de validité. Un modèle global serait alors obtenu. En automatique les contrôleurs utilisant une telle démarche pour estimer les validités sont rares (il ne faut pas confondre cette approche avec la construction des modèles par une classification utilisant des outils statistiques) .

4.4.4 Approches des résidus

Cette méthode très puissante fonctionne sans autre connaissance que celle des modèles et des réponses du système.
Elle est fondée sur l'estimation par chacun des modèles de certaines variables de sorties du système, qui sont ensuite comparées aux véritables valeurs [Delmotte et al, 1995b; Dubois, 1995]. Plus un système fera de bonnes estimations, plus il aura un indice de validité élevé. Au contraire, si un modèle fait trop d'erreurs, son indice de validité tendra vers 0.
Les comparaisons peuvent éventuellement être effectuées sur d'autres variables que les sorties, et font partie dans un cadre plus général de l'analyse des résidus exploitée en surveillance.

Les données dont les suivantes. A l'étape k, le modèle M_i fait une estimation de la variable y à k+1: $\hat{y}_i(k+1)$. A k+1 le résidu $R_i(k+1)$ est calculé:

$$R_i(k+1) = \mathrm{Abs}\big(y(k+1) - \hat{y}_i(k+1)\big) \tag{4.17}$$

Ces résidus bruts sont ensuite normalisés. Comme avec les distances de l'approche géométrique, deux choix sont possibles:

$$R'_i(k) = \frac{R_i(k)}{\sum_{i=1}^{n} R_i(k)} \tag{4.18}$$

$$R'_i(k) = \frac{R_i(k)}{\mathrm{Abs}(y(k))} \tag{4.19}$$

(si la variable y s'annule, les résidus normalisés sont prolongés par continuité dans 4.19)

Cependant dans le cas présent le principe de résidus relatifs est moins intéressant. En effet cette méthode avait été conçue à l'origine dans un objectif d'aide à l'identification et à la modélisation des systèmes. Ainsi l'étude des fonctions $v_i(z)$ obtenues avec cette approche permet de savoir quand un modèle M_i est valide ou non, et donc de lui attribuer un domaine de validité. Il est aussi possible de détecter des erreurs de conception lorsqu'aucun modèle de la bibliothèque de modèle (voir le chapitre 5) n'est valide pour une situation particulière.

Narendra utilise aussi les résidus pour évaluer la qualité des modèles, mais il prend en compte tous les résidus précédents dans son estimation des défauts de chaque modèle:

$$\text{Défaut}_i(k+1) = \alpha R_i^2(k+1) + \beta \sum_{i=1}^{k+1} e^{-\lambda(k+1-i)} R_i^2(i)$$

Le facteur exponentiel est un facteur d'oubli sur les précédents résidus. Cette relation permet d'adoucir les évolutions des validités, et de filtrer les bruits. Cependant avec cette relation il faut estimer $\alpha \geq 0$, $\beta > 0$ et $\lambda > 0$. Dans les trois séries de d'expériences présentées par Narendra [Narendra, 1997], les triplets (α, β, λ) étaient tous différents, et ceci n'était pas justifié.

L'étude des validités obtenues par ce moyen ainsi que la corrélation de cette méthode avec l'approche géométrique ou floue est très intéressante et fait l'objet de nombreux commentaires dans les exemples qui suivent et au chapitre 5. En tout état de cause, pour définir des domaines de validité qui ne soient pas tributaires de la bibliothèque dans son ensemble, les résidus ne doivent pas être obtenus de manière relative, sinon chaque fois qu'un nouveau modèle sera ajouté ou au contraire enlevé à la bibliothèque, son domaine de validité ne sera plus correctement défini.
Aussi seule la relation 4.14 a été retenue comme étape de normalisation.

Les résidus étant normalisés il devient possible de calculer les validités, comme avec les distances. L'équivalent de 4.6 donne:

$$v_i = 1 - R_i' \tag{4.20}$$

Cependant l'introduction de deux paramètres est intéressante:

$$v_i = \max(v_{min}, 1 - R_i' / err_{max}) \tag{4.21}$$

v_{min} est un simple paramètre pour éviter que dans les calculs numériques il n'y ait une division par zéro lorsque tous les modèles sont mauvais.
err_{max} est un paramètre plus intéressant. Il permet en effet de moduler la vitesse de décroissance de validité d'un modèle en fonction de ses erreurs. En effet un modèle ne faisant que 10% d'erreur sera dans la plupart des cas jugé correct, à moins justement que l'utilisateur ne veuille que les modèles ne soient influents uniquement lorsqu'ils sont très précis. Ce paramètre err_{max} permet de régler ce seuil. La figure 4.10 donne le résultat final de la validité v_i d'un modèle en fonction de son résidu R_i:

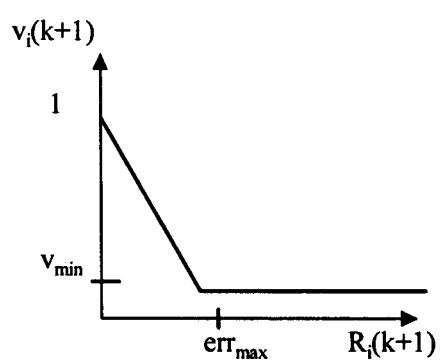

Fig. 4.10. Validité d'un modèle en fonction de son erreur.

(il faut préciser que les validités sont initialisées pour k=0, puisque aucune comparaison n'est encore possible).

Trois remarques peuvent compléter cette présentation.

Tout d'abord en supervision les résidus peuvent porter sur plusieurs variables et sorties. L'approche présentée n'est basée que sur un seul résidu. Un axe de recherche est la généralisation à l'analyse simultanée de plusieurs résidus pour l'estimation de la validité des modèles. Cependant il faudra alors utiliser les résultats de la décision multi-critére, car par exemple un modèle pourra se révéler excellent pour estimer une variable et très mauvais pour les autres, alors qu'un autre sera toujours moyen.
Ensuite il faut insister sur le fait que l'estimation de la validité d'un modèle par cette approche n'est effectuée que sur la trajectoire suivie par le système, comme le montre la figure 4.11.

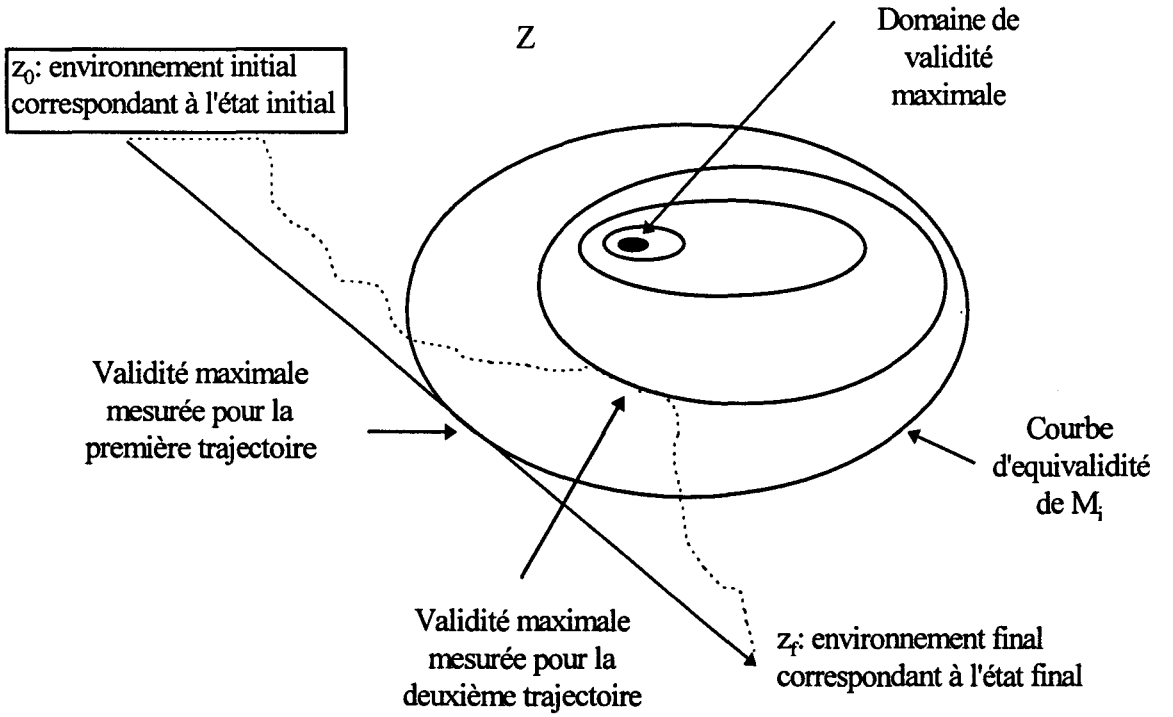

Fig. 4.11. Mesure de validité d'un modèle en fonction de différentes trajectoires (approche des résidus).

Pour des conditions initiales ou un paramètre err_{max} légèrement différents, ou encore pour des perturbations ou une commande différentes, la trajectoire pourra changer légèrement et les validités changer beaucoup si les modèles ont des validités très sensibles.
Il est même possible d'imaginer des exemples où, par le fait d'augmenter la valeur err_{max}, la trajectoire du système se trouve si bouleversée que des modèles qui étaient utilisés à un moment quelconque ne le soient plus du tout, car la trajectoire ne traverse plus de zones où ils sont valides.
Pour résoudre ce problème, et définir des domaines de validités dans un environnement très général Z=X×T×(tous les paramètres), il faudrait alors recourir à une série d'essais hors-ligne qui permettrait de mesurer dans Z les validités des modèles, et de reconstruire précisément son domaine de validité dans un espace Z restreint. Hors ligne il serait possible également d'utiliser des approches de type gradient pour converger en un minimum d'essais vers le ou les sous-domaines de validité.
Enfin de nombreux exemples ont montré que le paramètre err_{max} était d'une grande importance, et que, mal conditionné, il pouvait induire en erreur le CMM. En effet si err_{max} est

assigné a une trop petite valeur, tous les modèles auront une validité égale à v_{min}, et la fusion réalisera une moyenne des modèles. De même, et paradoxalement en fait, si err_{max} est trop grand, tous les modèles auront une validité très proche de 1, et la fusion réalisera alors en pratique une moyenne des modèles. Dans les deux cas la fusion ne sera pas satisfaisante. L'adaptation du paramètre est donc souhaitable, et sera présentée au chapitre 5. Avec cette adaptation l'approche des résidus se révélera très performante.

4.4.5 Opérations sur les validités finales

Une fois obtenu l'ensemble des validités v_i que ce soit par une méthode ou une autre, des traitements supplémentaires peuvent leur être appliqués. Ce paragraphe détaille l'étape de renforcement.

Son objet est de supprimer le phénomène de perturbation des bons modèles par les mauvais, déjà rencontré avec les contrôleurs à logique floue (paragraphe 3.3.2). En fonction du mécanisme de fusion retenu, ce problème peut advenir pour des variables non floues.

L'exemple qui suit le montre. 10 modèles sont considérés.
Le premier est parfait, sa validité $v_1=1$. Il calcule une commande $u_1=1$.
Les autres modèles correspondent a des situations totalement différentes, et sont tous faux. Leur validité $v_i=0.01$ pour $i>1$. Comme les modèles sont faux, leurs commandes dérivées sont toutes aberrantes: $u_i=1000$ pour $i>1$.
Si une règle de fusion linéaire est choisie, le résultat final sera
$u=(1\times1+1000\times9\times0.01)/(1+9\times0.01)\approx83$
Ce résultat final est donc totalement diffèrent de celui préconisé par le modèle M_1, considéré comme parfait par le superviseur.

Afin de remédier à ce problème, une opération supplémentaire est introduite sur les validités juste avant l'opération de fusion des informations. Cette opération de renforcement est définie de la manière suivante:

$$\forall i \in \{1,n\}$$
$$v_i^{renf} = v_i \prod_{\substack{j=1 \\ j\neq i}}^{n} \left(1 - v_j\right) \tag{4.22}$$

A l'origine une autre relation avait été utilisée lors de travaux portant exclusivement sur des calculs de validité par l'approche géométrique:

$$\forall i \in \{1,n\}$$
$$v_i^{renf} = v_i \prod_{\substack{j=1 \\ j\neq i}}^{n} \left(1 - c_j e^{-d_j'^2/\sigma_j^2}\right) \tag{4.23}$$

Avec ces deux relations, toutes les validités sont multipliées par une fonction décroissante de toutes les autres validités. Ainsi si un modèle M_i est parfait, $v_i=1$, alors $v_j^{renf}=0$ pour j=1 à n sauf i. Dans ce cas le modèle parfait n'est plus perturbé par les autres modèles lors de la fusion. Ceci est vrai même si sa validité renforcée est finalement diminuée par les autres validités, puisque ces dernières sont rigoureusement nulles.

Dans 4.23 apparaît le terme σ_j qui permet de moduler l'influence de la décroissance de validité d'un modèle sur les autres modèles. Plus σ_j est grand, plus le modèle M_j sera considéré valide dans un grand voisinage de D_j, et donc plus les autres verront leur validité décroître dans ce voisinage. Normalement ces paramètres doivent être petits par rapports aux distances entre les domaines de validités, sinon les variations de validités deviennent brusques.
Il est aussi possible d'envisager d'autres opérateurs de renforcement que ceux présentés en 4.22 et 4.23, avec d'autres paramètres.
Il faut remarquer qu'une règle spéciale a été introduite dans le cas où plusieurs modèles seraient parfaits pour les mêmes circonstances ($\exists$ z tel que $v_i(z)=v_k(z)=1$). Cette règle est un simple test qui évite que les validités des modèles correspondants ne soient mises aussi à zéro.

Dans la littérature ce problème n'a que rarement été évoqué et résolu. En général un seuil v_{min} est introduit en deçà duquel toutes les validités sont mises à zéro. Ceci permet de supprimer l'influence des modèles à faible validité, mais ne résoud pas complètement le problème (notamment si un modèle est parfait et quelques autres moyens).

Dans ce paragraphe détaillant quelques opérations sur les validités juste avant la fusion, il est possible de citer Narendra [Narendra, 1997], qui introduit pour sa fusion discontinue un temps de relaxation minimal T_{min} entre deux commutations, afin d'en limiter la fréquence.

4.5 Fusion des modèles

Que ce soit avec les 3MGI ou les 3MGE, l'opération de fusion des modèles n'a pas encore été explorée. C'est pourtant, avec le superviseur estimant les validités, la seconde opération fondamentale dans un CMM.

Les données sont les suivantes:
pour les 3MGE: les fonctions $f_i(x)$ et $g_i(x)$
pour les 3MGI: les sorties y_i
enfin dans les deux cas, le vecteur des validités $v=(v_1,...,v_n)$

Le résultat après la fusion est selon les cas, soit des fonctions, soit une sortie y.

Il est possible à ce niveau d'introduire deux grandes classes de multi-modèles: ceux qui effectuent une fusion et ceux qui commutent entre les modèles.
Pour la seconde classe, la fusion est une opération élémentaire:

$$f(x)=f_i(x) \text{ avec } i=\mathrm{Arg}(\max_j v_j) \tag{4.24}$$

et de même si la fusion est effectuée sur les sorties

$$y=y_i \text{ avec } i=\mathrm{Arg}(\max_j v_j) \tag{4.25}$$

Néanmoins avec cette méthode il y a une commutation entre les modèles, et donc des discontinuités sur toutes les variables étudiées. Par contre l'analyse est plus facile. L'école de Yale [Narendra et al, 1995, 1997] a privilégié cette fusion, qui permet des calculs élaborés par la suite. Cependant comme l'interpolation est inexistante, lorsque les contrôleurs dérivés n'utilisent que des modèles fixes, les résultats sont mauvais. Narendra préfère soit des modèles tous adaptatifs, soit des mélanges de modèles fixes et adaptatifs.

Avec la première classe de multi-modèle, il y a une véritable fusion effectuée, c'est à dire que le résultat final dépend de tous les modèles, en fonction bien évidemment de leur validité.
Il faudrait beaucoup d'ambition pour prétendre les recenser toutes. Trois méthodes seront exposées.

4.5.1 Fusion linéaire

Cette fusion est très simple. La relation 4.26 en donne l'expression pour une fusion sur les sorties.

$$y = \frac{\sum_i v_i y_i}{\sum_i v_i} \tag{4.26}$$

Cet opérateur peut être appliqué tel quel si la fusion est effectuée sur des fonctions. Cette fusion peut aussi être effectuée sur des vecteurs. En effet les CMM peuvent affronter des systèmes multivariables, dans ce dernier cas le problème se pose lors de la fusion, mais en réalisant la fusion composante par composante le problème disparaît. S'il est vrai que l'étude des systèmes multivariables est plus complexe que les autres, ce problème de complexité est à traiter au niveau des modèles et des lois de commande, pas au niveau du multi-modèle.

Ainsi avec les modèles évoqués au 4.3.3, le modèle global devient par exemple:

$$\dot{x} = Ax + Bu \quad y = Cx$$

avec

$$A = \frac{\sum_{i=1}^{n} A_i v_i}{\sum_{i=1}^{n} v_i} \qquad B = \frac{\sum_{i=1}^{n} B_i v_i}{\sum_{i=1}^{n} v_i} \qquad C = \frac{\sum_{i=1}^{n} C_i v_i}{\sum_{i=1}^{n} v_i}$$

4.5.2 Fusion rencontrée dans les contrôleurs flous

Les contrôleurs flous sont des CMM dégénérés. Leur étape de fusion des conclusions des règles est utilisée dans les MM pour lesquels les sorties y_i sont floues. Dans ce cas elles sont associées à des fonctions d'appartenance μ_{yi}.
Le chapitre 3 a détaillé le processus. Dans le cas d'une inférence de Mamdani, les indices de validité v_i sont considérés comme les seuils de déclenchements.
Alors les fonctions μ_{yi} sont modifiées en $\mu'_{yi}=\min(v_i, \mu_{yi})$, et finalement un opérateur max est appliqué sur toutes les conclusions, pour définir un nouvel ensemble flou de sortie:

$$\mu(y)=\max_i(\mu'_{yi}(y)) \tag{4.27}$$

4.5.3 Opérateur non linéaire

Les opérateurs de fusion évoqués dans la théorie des possibilités au chapitre 2 peuvent aussi être utilisés en considérant les modèles comme autant de sources d'informations. Cependant il faut dans ce cas utiliser les opérateurs traitant des degrés de fiabilité ou de validité des

données. Le nouvel opérateur de fusion défini au chapitre 2 apparaît alors comme une possibilité intéressante.
Par rapport à la fusion classique des contrôleurs flous, cet opérateur renforce les informations qui sont en accord au détriment des autres. Par exemple si 2 modèles sont en accord pour estimer des sorties voisines par rapport à d'autres toutes différentes, dans le résultat ceci apparaîtra par une plus grande valeur d'appartenance accordée à la valeur commune.

Lorsque les sorties ne sont plus des ensembles flous, mais de simples variables numériques, il est possible de montrer que les deux opérations de fusion floues dégénèrent en la fusion linéaire (4.26), avec éventuellement des coefficients de validité déformés dans le cas de l'opérateur non linéaire.

4.5.4 Etape de défuzzyfication

Si les modèles élémentaires évaluent les sorties sous forme d'ensembles flous, il faut défuzzyfier les variables pour obtenir des résultats classiques. Cette opération peut se réaliser avec les opérateurs définis au chapitre 3.

Il apparaît néanmoins un problème lorsque les modèles estiment les variables sous forme hétérogène, c'est à dire sous forme floue et non floue.

Dans ce cas la fusion ne peut pas être effectuée telle quelle. Selon la proportion de variables floues et non floues, soit il faut, avant la fusion, fuzzyfier les variables non floues en singletons flous, soit défuzzifier les variables floues. Cependant dans le cas d'un grand nombre de transformations, il peut être avantageux soit pour un gain de rapidité, soit pour conserver le plus longtemps possible le maximum d'information, de réaliser une fusion séparée pour les deux groupes de données.

Le résultat est alors une sortie y_{1*} correspondant à la fusion des sorties floues, et une sortie y_{2*} correspondant à la fusion des informations non floues. Les coefficients de validité attachés respectivement à ces deux quantités sont les sommes des validités des informations fusionnées respectivement dans les deux groupes.

Il s'agit alors de défuzzyfier y_{1*} puis de la combiner lineairement à y_{2*}.

4.6 Exemple de contrôleur à validité a priori

Avec cet exemple le système est parfaitement modélisé, et une formulation mathématique est disponible. Cependant elle est supposée trop complexe pour en extraire une commande.

4.6.1 Le système

Le système est représenté par:

$$x(t) + \tau(x)\dot{x}(t) = k(x)u(t) \tag{4.28}$$

avec

$$\begin{aligned} k &= 36x(t)(x(t)-1)+10 \\ \tau &= 15-10x(t) \end{aligned} \tag{4.29}$$

Il apparaît donc que le système est un premier ordre pour lesquels le gain et la constante de temps sont des fonctions de l'état. L'état est en outre observable à chaque instant. Pour les simulations, seul le comportement stable ($\tau>0$) est considéré admissible.

La figure 4.12 montre les évolutions respectives du gain et de la constante de temps.

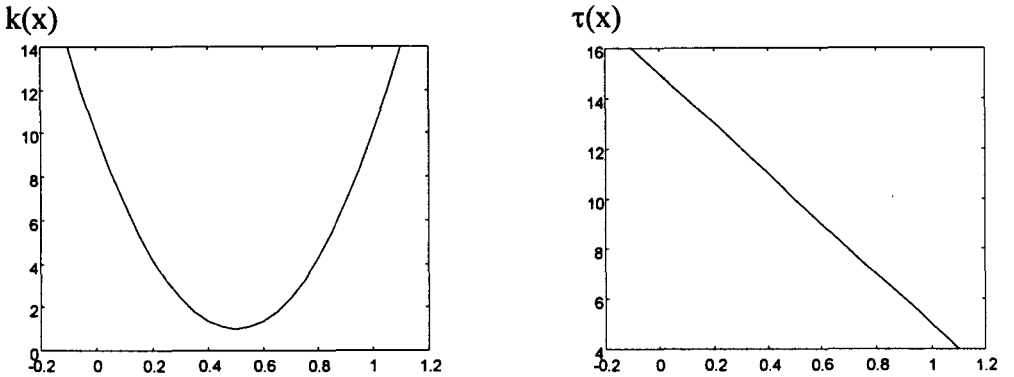

Fig. 4.12. Evolutions des paramètres k et de τ du système du premier exemple.

Le gain est une fonction non linéaire de l'état qui varie d'un facteur 10 pour x entre 0 et 1, donc la non linéarité n'est pas négligeable.

L'objectif est d'amener le système d'un état initial $x_0=0$ à un état final $x_f=1$ en $t_f=5$ secondes tout en minimisant le critère quadratique suivant :

$$J = \int_{t_0}^{t_f} \left((x - x_f)^2 + u^2\right) dt \tag{4.30}$$

Il s'agit donc d'un problème de commande optimale à horizon fixe. Pour les besoins de la démonstration l'utilisateur est supposé ne pas pouvoir calculer la commande optimale du problème posé. En réalité, cette commande est calculable et sa valeur sera comparée avec les différentes commandes réellement appliquées au système.

4.6.2 Les modèles

En examinant les caractéristiques du système il apparaît qu'il existe plusieurs situations remarquables entre x=0 et x=1.
D'abord le gain varie d'un facteur 10. L'utilisateur suppose donc qu'il faudra au moins deux modèles correspondant pour le premier à un gain de 10 et pour le deuxième à un gain de 1. La structure de ces deux modèles étant bien entendu celle d'un premier ordre à paramètres constants.
Cependant la constante de temps ne présente pas une telle symétrie, aussi l'utilisateur hésite entre 2 et 3 modèles, ce dernier cas correspondant au dédoublement du modèle de gain 10 pour les constantes de temps respectivement égale à 15 et 5.

Les comparaisons seront faites avec un CMM comprenant respectivement 2 et 3 modèles. Dans la définition des modèles sera donnée la définition de leur domaine de validité, car leur obtention se fait naturellement par une approche géométrique. Dans ce cas Z s'identifie à X, puisque c'est la variable x qui permettra de choisir quel modèle est bon ou non.

4.6.2.1 Modèles dans le cas d'une configuration à trois modèles

M_1 est décrit par la relation suivante:

$$x = \frac{10}{1+15s}u \tag{4.31}$$

$D_1=\{z=0\}$ et $c_1=1$ car le modèle est parfait dans son domaine
M_2 est décrit par la relation suivante:

$$x = \frac{1}{1+10s}u \tag{4.32}$$

$D_2=\{z=0,5\}$ et $c_2=1$ car le modèle est parfait dans son domaine.

M_3 est décrit par la relation suivante:

$$x = \frac{10}{1+5s}u \tag{4.33}$$

$D_3=\{z=1\}$ et $c_3=1$ car le modèle est parfait dans son domaine.

4.6.2.2 Modèles dans le cas d'une configuration à 2 modèles

L'utilisateur remarque que les modèles M_1 et M_3 sont cependant proches, aussi il décide de les résumer par un seul modèle commun, le second modèle M_2 étant gardé tel quel.
Le nouveau modèle est M'_1:

$$x = \frac{10}{1+10s}u \tag{4.34}$$

Son domaine de validité est $D'_1=\{z=0\}\cup\{z=1\}$. Cependant ce modèle n'est pas identique au système dans son domaine de validité, même s'il en est proche. Aussi sa validité propre c'_1 sera égale à 0,9, valeur prise empiriquement pour les deux sous-domaines de validité.

4.6.3 Les commandes

Les modèles étant linéaires, il est possible d'en déduire les commandes optimales associées, selon les termes du problème défini [Borne et al, 1990].

En boucle ouverte, le résultat est

$$u = c(K_1 e^{\alpha t} + K_2 e^{-\alpha t} + 1) \tag{4.35}$$

avec

$$\begin{aligned}
\alpha &= \frac{1}{\tau}\sqrt{k^2+1} \\
K_1 &= \frac{e^{\alpha t_f} + k^2}{\left(e^{2\alpha t_f} - 1\right)\left(\sqrt{k^2+1} - 1\right)} \\
K_2 &= \frac{e^{\alpha t_f}(1 + k^2 e^{\alpha t_f})}{\left(e^{2\alpha t_f} - 1\right)\left(\sqrt{k^2+1} + 1\right)} \\
c &= \frac{k x_f}{k^2+1}
\end{aligned} \tag{4.36}$$

En boucle fermée, le résultat est:

$$u = -Lx + l \tag{4.37}$$

avec

$$L = \frac{1}{k}\left[\sqrt{k^2+1}\,\frac{\beta^2+1}{\beta^2-1} - 1\right]$$

$$l = x_f\,\frac{2\beta + k^2\left(\beta^2+1\right)}{k\sqrt{k^2+1}\left(\beta^2-1\right)} \tag{4.38}$$

$$\beta = \exp\left[\frac{1}{\tau}\sqrt{k^2+1}\left(t_f - t\right)\right]$$

4.6.4 Calcul des validités a priori

Toutes les informations nécessaires au calcul des validités des modèles ont été fournies, aussi il est possible d'analyser leurs évolutions avant même de faire appel au multi-modèle. La figure 4.13 donne les valeurs des validités pour les 3 modèles de la configuration à 3 modèles.

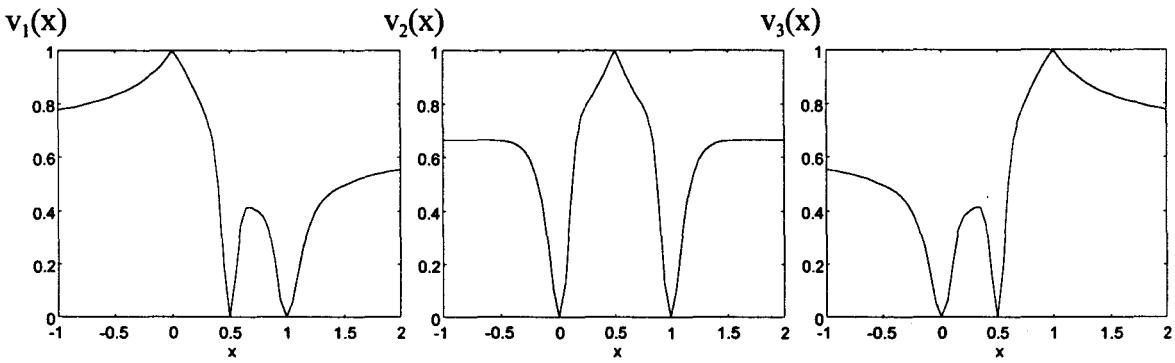

Fig. 4.13 Validités des 3 modèles en fonction de x.

Il est possible dans ce problème d'imaginer ce que donnerait une approche floue pour le calcul des validités. Dans ce cas, il faut définir les fonctions d'appartenance (i.e. les validités) $\mu_i(z)$ des modèles. Il est possible de prendre des fonctions ayant la forme de triangles, ce qui est très courant avec les CMM du type de Sugeno-Takagi-Kang. Le résultat est alors donné par la figure 4.14.

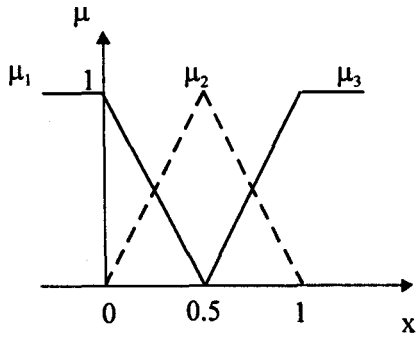

Fig. 4.14. Validités obtenue par une approche floue.

Les validités obtenues par l'approche géométrique et l'approche floue sont donc grosso modo les mêmes. Ceci n'est pas étonnant, ce serait même le contraire qui le serait, car ces deux approches représentent à peu près la même connaissance.
Dans le cas de l'approche géométrique, les courbes sont non linéaires, mais avec la logique floue des fonctions d'appartenance gaussienne auraient pu été choisie aussi.

La seule différence importante apparaît en dehors des zones de fonctionnement (pour $x<0$ et $x>1$). En effet avec la logique floue, en général seuls les règles ou les modèles les plus proches sont appliqués (ceci étant du aux raisons de lisibilité et de couverture). Ainsi pour $x=-100$, seul le premier modèle serait utilisé. Avec les algorithmes géométriques, les validités ne peuvent jamais être nulles si la normalisation est relative, à moins de supposer que le système atteigne les bornes $\pm\infty$. Aussi avec l'approche géométrique, si aucun domaine de non validité n'est défini et si l'environnement courant z n'est pas dans un domaine de validité, tous les modèles sont pris en compte.

4.6.5 Résultat en boucle ouverte avec une configuration à 3 modèles

La figure 4.15 montre les résultats obtenus en appliquant la commande en boucle ouverte dérivée d'un seul modèle pour $x_f = 1$. x(t) et u(t) sont respectivement tracés.

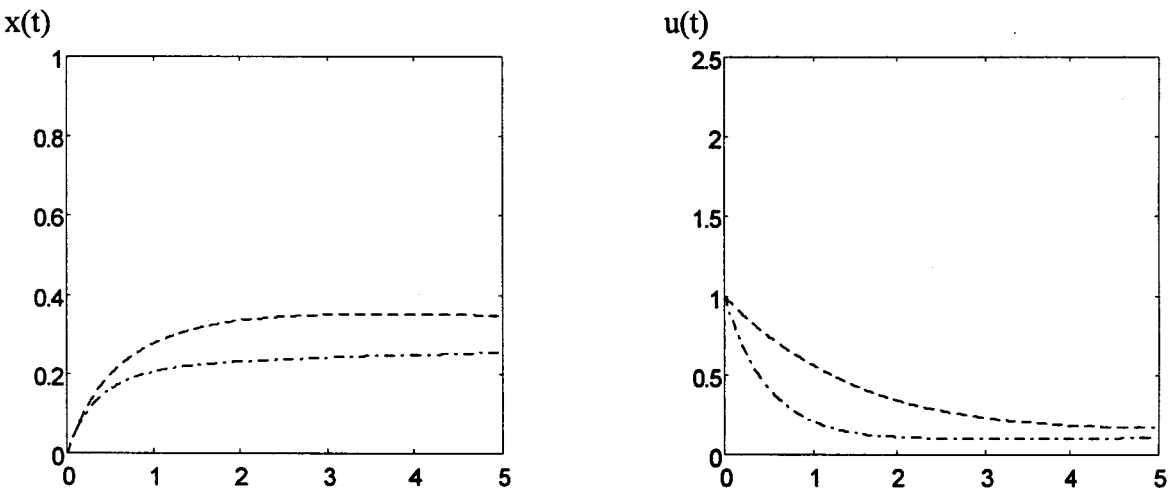

Fig. 4.15. x(t) et u(t) en appliquant au premier système la commande en boucle ouverte dérivée d'un seul modèle. En '----' les résultats pour M_1, et en '-.-.' les résultats pour M_3.

Comme cela était prévisible, en boucle ouverte les résultats sont très mauvais. Etant conçus pour un système à fort gain, les commandes dérivées du premier et du troisième modèle ne sont pas assez puissantes, et l'erreur sur la consigne est très importante.

A cet égard, le deuxième modèle se révèle encore pire. En effet, conçu pour un système à faible gain, la commande est cette fois trop importante. La consigne est très vite dépassée, x(t) devient si important que le système devient instable.

Les approches multi-modèles sur cet exemple en boucle ouverte montrent immédiatement leur puissance. Avec seulement 3 modèles linéaires, le résultat est presque parfait. Pour l'approche floue, les validités sont données par les fonctions d'appartenance, et la fusion choisie est linéaire. Pour l'approche géométrique, une étape de renforcement a été appliquée ($\sigma_i=0.25\ \forall i$), et le nouvel opérateur de fusion a été choisi. Les résultats sont donnés à la figure 4.16.

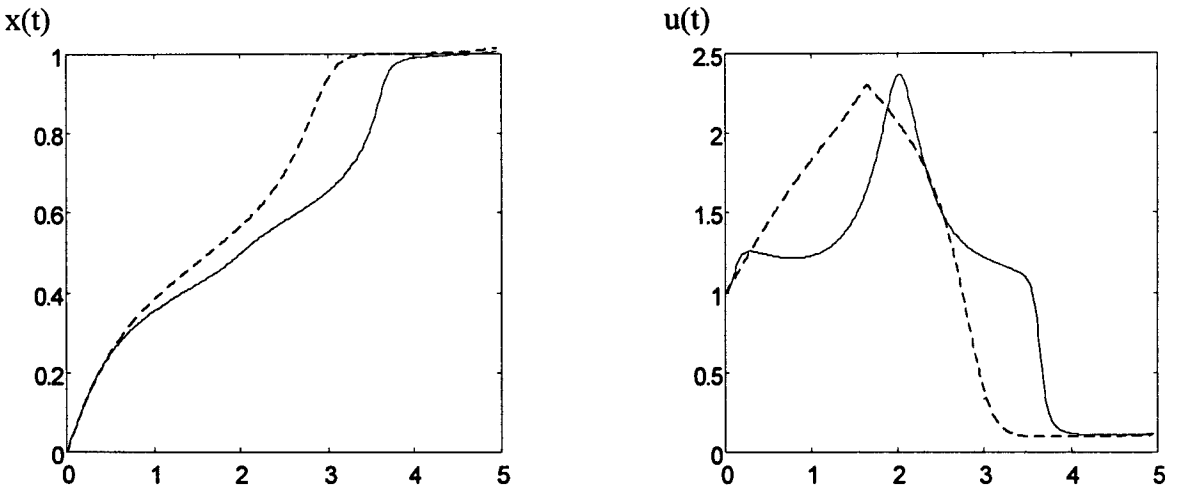

Fig. 4.16. x(t) et u(t) obtenu pour le premier système avec une commande CMM en boucle ouverte. En '----' les résultats concernants le CMM de type Sugeno, en '—' ceux concernants le CMM basé sur l'approche géométrique.

Pour le CMM de type Sugeno (les validités sont données par une approche floue), l'erreur finale est de 16.10^{-3}, pour le CMM basé sur l'approche géométrique, l'erreur finale est de 8.10^{-3}.

Sur les courbes, il est possible de voir l'influence du second modèle sur la commande, notamment pour le CMM avec approche géométrique pour $t \approx 2s$.

4.6.6 Résultats en boucle fermée avec une configuration à 3 modèles

La figure 4.17 donne x(t) et u(t) en appliquant les commandes en boucle fermée obtenues pour un seul modèle a chaque fois.

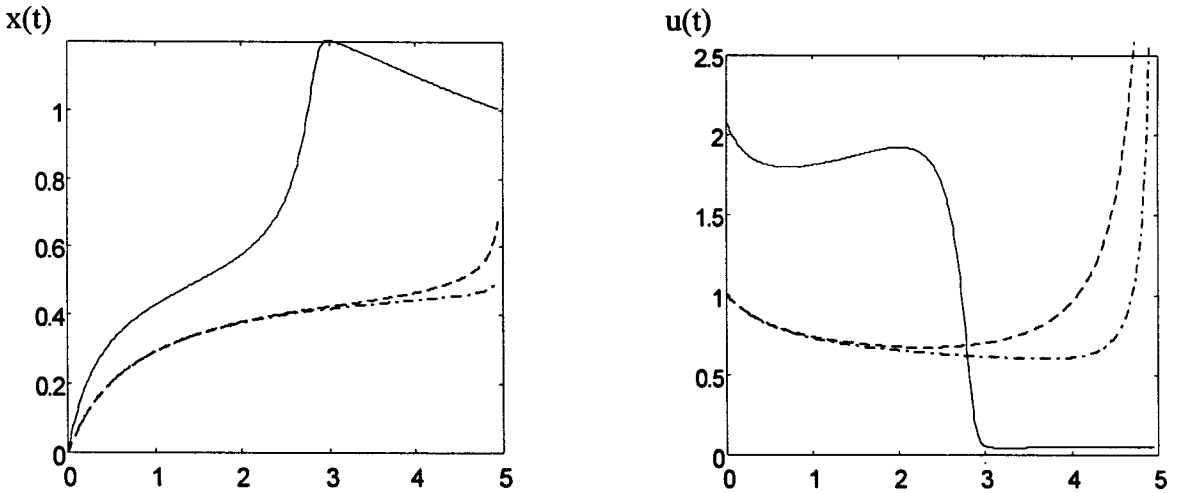

Fig. 4.17. x(t) et u(t) en appliquant la commande en boucle fermée dérivée d'un seul modèle. En '----' les résultats pour M_1, en '-.-.' les résultats pour M_3. En '—' les résultats pour M_2.

Cette fois le critère de comparaison est le coût total défini pour le calcul de la commande optimale, puisque la consigne est atteinte dans tous les cas. Les commandes dérivées des modèles à forts gains ne réagissent pas assez au début, seulement à la fin, mais en injectant

alors une commande très importante pour rattraper la consigne. Le coût associé à ces 2 modèles est très fort: 16,53 pour le premier modèle, et supérieur à 85 pour le troisième modèle (les simulations furent stoppées à t=4,995s car dans les expressions des commandes apparaît au dénominateur un terme qui tend vers 0 lorsque t tend vers t_f, et le logiciel de simulation utilisé gèrait mal ce problème). Cette fois le deuxième modèle se révèle être le meilleur. En effet la commande, calculée pour un faible gain, est trop importante, mais en raison du bouclage, elle chute dès qu'il y a un dépassement. Son coût est alors de 10,15.
La figure 4.18 donne les résultats obtenus avec les deux CMM.

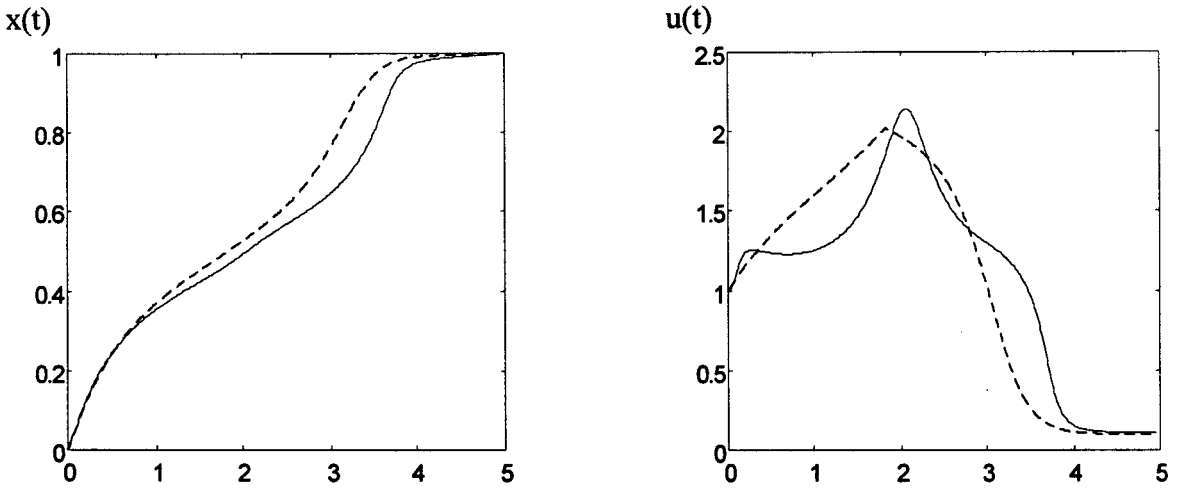

Fig. 4.18. x(t) et u(t) pour les deux CMM en boucle fermée. En '----' les résultats concernants le CMM de type Sugeno, en '—' ceux concernants le CMM basé sur l'approche géométrique.

Le coût est de 9,23 pour le CMM de type Sugeno et de 8,85 pour l'autre CMM.

L'analyse des résultats en boucle ouverte et en boucle fermée montre donc que l'utilisation conjointe de plusieurs modèles simplifiés est très facile et donne une image fidèle d'un système complexe.

4.6.7 Résultats en boucle fermée avec une configuration à 2 modèles

La figure 4.19 donne x(t) et u(t) lorsque 2 modèles sont utilisés avec l'approche géométrique.

Le coût obtenu est de 9,29. Les résultats sont pratiquement les mêmes que ceux obtenu avec une approche de type Sugeno avec 3 modèles.
Ceci s'explique par le fait qu'avec l'approche de type Sugeno, seuls les plus proches modèles sont utilisés, et en l'occurrence dans cet exemple, seules, au plus, deux commandes sont fusionnées à chaque instant, la troisième ayant toujours une validité nulle.

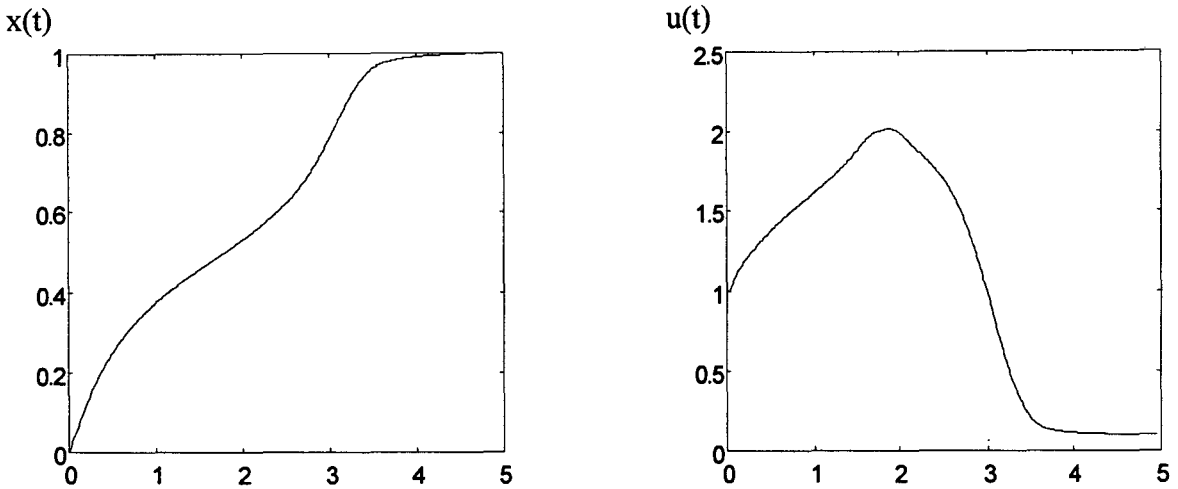

Fig. 4.19. x(t) et u(t) en boucle fermée pour une configuration à deux modèles.

Dans cette configuration, le coût est un peu plus grand qu'avec l'approche de type Sugeno. Ceci s'explique parce que le modèle à grand gain n'a jamais une validité égale à 1, mais au plus égale à 0,9, et le modèle à faible gain se trouve de ce fait avantagé.

A ce propos il paraît difficile dans une approche de type Sugeno d'utiliser des modèles qui ne soient jamais parfaits, c'est à dire dont la fonction d'appartenance n'atteigne jamais 1. Il faudrait pour cela remettre en question la contrainte de lisibilité et de complétude (qui donne finalement la contrainte que la somme des fonctions d'appartenance soit égale à 1 en tout point).

Plutôt que le seul critère d'efficacité, car les résultats sont très proches avec l'approche géométrique et l'approche floue, il faudrait également utiliser un critère de souplesse ou de lisibilité pour choisir.

L'approche géométrique donne la plus grande liberté de conception, il est possible de construire des zones où de multiples domaines de validité, afférents à différents modèles, se recoupent, avec différents degrés de validité propre, sans se préoccuper de la valeur exacte des fonctions d'appartenance.

4.6.8 Commande optimale et CMM avec une infinité de modèles

Les remarques qui suivent concluent l'analyse du premier exemple.
D'abord cet exemple montre les difficultés à calculer une commande optimale pour des systèmes complexes. En effet l'utilisateur imaginaire utilisé, qui comparait les différentes méthodes de commande avec ou sans multi-modèle, ne savait pas que la commande optimale pouvait être calculée.

Elle l'est pourtant, bien que son calcul fasse un saut dans la complexité par rapport aux commandes optimales des modèles linéaires. En effet dans le hamiltonien utilisé pour la résolution du problème est introduit le vecteur adjoint λ. Ce vecteur adjoint qui apparaît dans la forme finale de la commande, est solution d'une équation différentielle dont il faut connaître la valeur initiale λ_0. Pour le problème évoqué la seule manière d'évaluer λ_0 est de recourir à une intégration numérique, et de recourir aux méthodes classiques de recherche de zéro. Une solution a pu être obtenue, et donne x(t) et u(t) sur la figure 4.20.

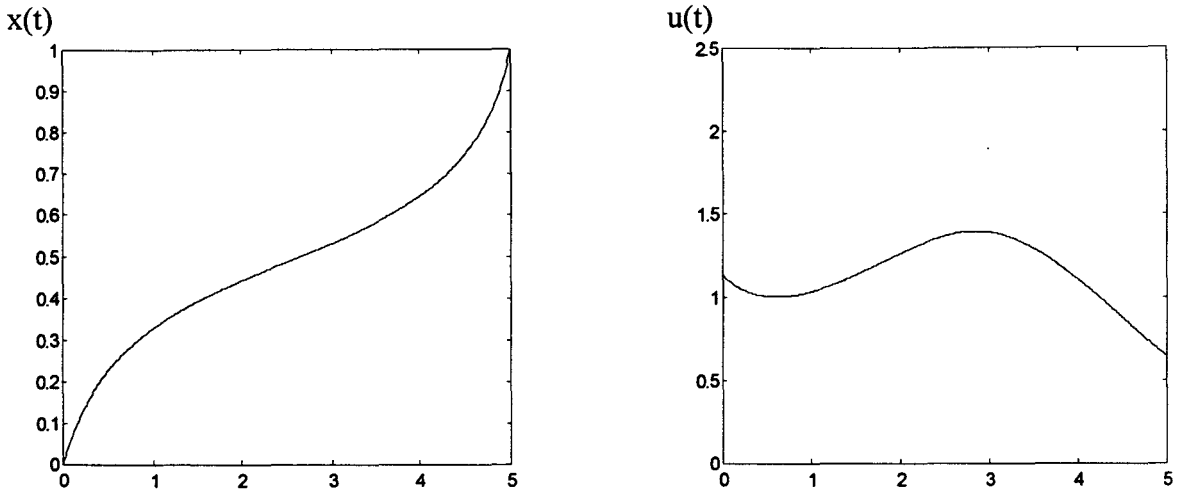

Fig. 4.20. x(t) et u(t) obtenu avec la véritable commande optimale pour le système complet.

Le coût final est de 8,16, ce qui veut dire que les approches par MMC donnaient une solution proche de l'optimale.

Des simulations furent effectuées avec 4 et 5 modèles (dont les domaines de validité étaient régulièrement espacés) pour vérifier si le coût diminuait ou non. Avec 4 et 5 modèles une amélioration de 1% sur le coût total fut constatée. Il serait alors intéressant de savoir si l'efficacité d'un CMM est une fonction croissante du nombre de modèle. Une dernière simulation fut effectuée avec un CMM basé sur un numbre de modèle aussi grand que possible, correspondant chacun au linéaire instantané.

Dans ce cas il n'est pas nécessaire de tenir compte des autres modèles, puisque la validité du modèle courant est par définition égale à 1. L'étape de renforcement permet donc de ne tenir compte à chaque étape que d'un seul modèle, et il n'y a plus d'étape de fusion. L'algorithme est même plus rapide qu'avec 3 modèles. La figure 4.21 donne les résultats.

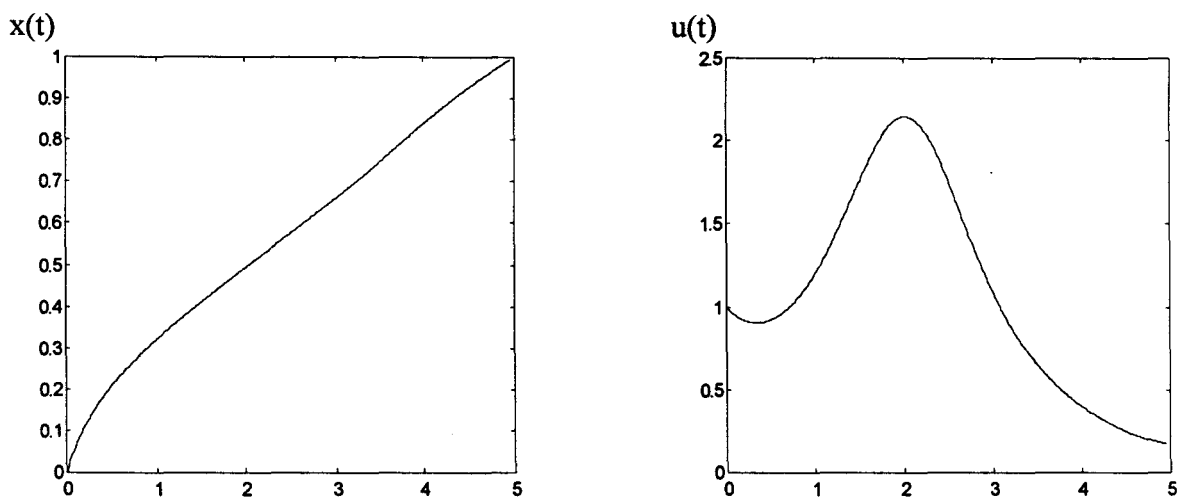

Fig. 4.21. x(t) et u(t) avec un CMM contenant une infinité de modèles.

Le coût final est de 8,93, qui est supérieur au coût obtenu avec 3, 4 et 5 modèles. Dans un CMM il n'est donc pas utile de multiplier à l'envie le nombre de modèles pour obtenir un contrôleur efficace.

La commande obtenue est totalement différente de la commande optimale. Ce résultat confirme la théorie de la commande optimale indiquant que la commande résultant de la succession des commandes optimales dérivées des modèles linéaires d'un système non linéaire n'a aucun lien avec la véritable commande optimale dérivée du système non linéaire en entier.

Cette expérience rappelle les limites des approches utilisées en automatique classique qui consistent à linéariser en différents points un système non linéaire.

Le nombre de modèles mis en œuvre et la complexité du CMM associé est souvent une critique adressée à l'encontre des approches multi-modèles en général. Pourtant cette critique n'est pas justifiée.

Dans toute méthode d'analyse et de commande, plus un système est complexe, plus la structure utilisée est complexe. Mais avec les CMM certaines économies peuvent être faites. Ainsi si un seul paramètre est étudié, il suffira de définir deux modèles aux bornes de variation et l'interpolation permettra de retrouver tous les comportements possibles. Si d'aventure le paramètre a certaines valeurs critiques telles que le système ait un comportement très diffèrent en ces valeurs, quelques modèles particuliers pourront être définis.
En poursuivant ce raisonnement il est possible de penser qu'avec 2 paramètres, il faudra au minimum 4 modèles, avec 3, 8 modèles et ainsi de suite avec une progression géométrique.

Ceci est mal comprendre l'esprit de l'analyse multi-modèle.
En effet les modèles utilisés ne doivent pas traduire toutes les situations possibles, toutes les valeurs possibles des paramètres, mais seulement certaines grandes classes de réponses. Or il se peut très bien que des couples différents de valeurs pour des groupes de variables donnent les mêmes comportements et se recoupent en définitive. Ainsi avec les premier-ordres, qui sont caractérisés par leur gain et leur constante de temps, il est possible de ne s'intéresser en régime dynamique qu'au rapport k/τ. Dans ce cas un modèle avec grand gain et grande constante de temps aura un comportement identique à un modèle de petit gain et de petite constante de temps, avec un rapport identique. Pour le régime statique, il ne faudra tenir compte que du gain.

La figure 4.22 résume cette idée, avec 4 modèles pour 2 constantes de temps τ_1 et τ_2, et 2 gains k_1 et k_2, avec $k_1/\tau_1=k_2/\tau_2$.

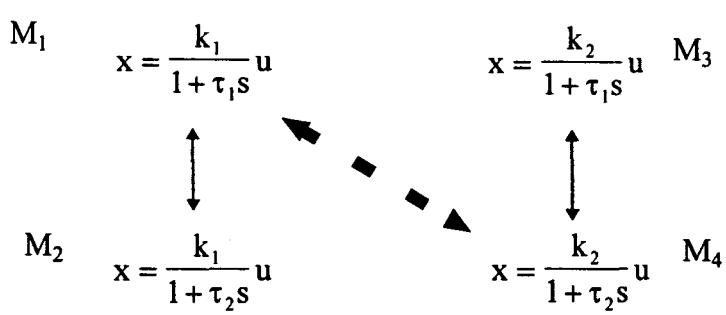

Fig. 4.22. Equivalences entre les modèles. La grosse flèche indique une équivalence en dynamique, les petites en statique.

En dynamique, M_4 est équivalent à M_1, et en statique respectivement M_1 avec M_2 et M_3 avec M_4. Sur ces quatre modèles, il ne pourrait donc en être conservé que 3, par exemple M_1, M_2 et M_3.

Enfin pour des systèmes très complexes, il reste toujours possible de définir quelques modèles robustes, très approximatifs et moyennement valables dans de très grandes zones de fonctionnement, ainsi que quelques modèles très précis valides dans de petites zones, là où le système a des comportements spécifiques.

En liaison avec le nombre de modèles est généralement soulevée la question du volume de calcul exigé par un CMM. Ce volume correspond approximativement à la somme des volumes de calculs exigés par chaque modèle ou algorithme de calcul des commandes inclus dans le contrôleur. Ainsi pour la configuration à 3 modèles, le temps de calcul était 4 fois supérieur à celui de la commande par un seul modèle de même taille. En fait seul le superviseur et le mécanisme de fusion, nécessitent quelques opérations supplémentaires, et encore sont-elles élémentaires.

4.7 Exemple: Contrôleur à validité estimée par les résidus

L'approche par résidus permet la mise en œuvre de modèles dont la validité est absolument inconnue, l'algorithme se chargeant de l'évaluer en temps réel au cours des calculs.

4.7.1 Premier exemple

Le premier système étudié est le système présenté au paragraphe 4.6, donné par les relations 4.24 et 4.28.
Les trois modèles définis par les relations 4.27 à 4.29 sont utilisés sans information particulière sur leur validité. L'approche des résidus est utilisée, les comparaisons étant faites sur la variable dx/dt (et non pas x).

Cette dernière remarque est importante, car avec l'approche des résidus, il faut choisir une variable pour laquelle seront calculés les résidus. Or plusieurs variables sont possibles, surtout avec les systèmes d'ordre élevé, ou les systèmes multi-variables. Il n'est pas certain que le choix d'une variable particulière n'influe pas sur la qualité des résidus obtenus. Il ne semble pas évident de déterminer une variable qui soit de manière générale discriminante pour tous les problèmes. Actuellement des travaux du LAIL [Pierrot et al, 1997] porte sur l'analyse de plusieurs résidus en même temps.

Les paramètres du CMM sont ceux du 4.7 basé sur l'approche géométrique. Les paramètres de l'approche par résidus sont les suivants. Le paramètre v_{min} de validité minimale a été fixé à 0,1 et le paramètre err_{max} modulant la décroissance de validité en fonction de l'erreur à 50%. Les validités sont initialisées à v_{min}.
La figure 4.23 donne les validités v_i obtenues en fonction du temps.

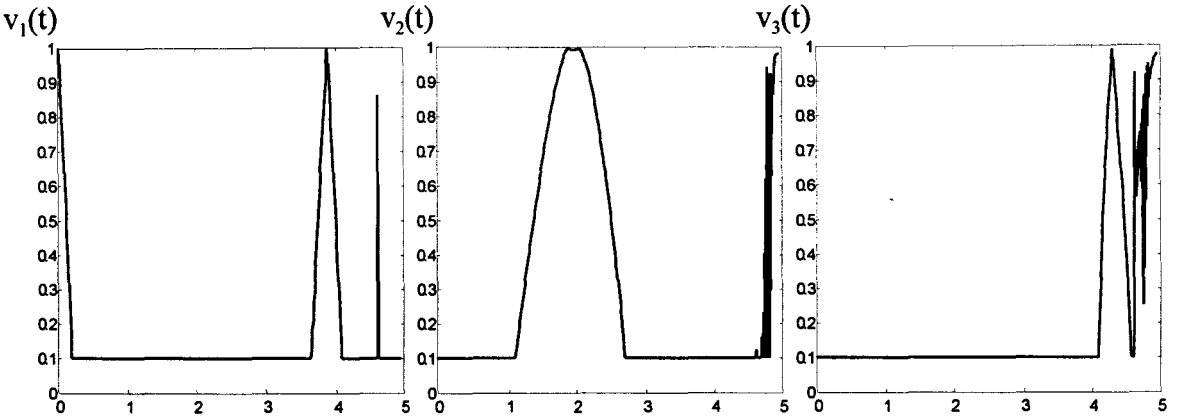

Fig. 4.23. Validités des trois modèles mesurées par l'approche des résidus.

La figure 4.24 donne x(t) et u(t) obtenus en même temps.

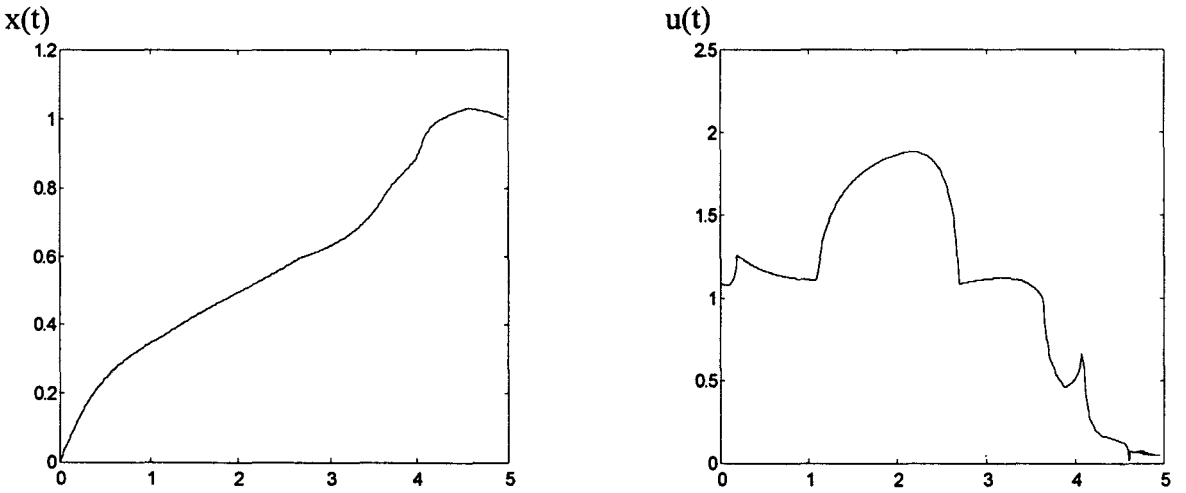

Fig. 4.24. x(t) et u(t) obtenus pour le premier système avec 3 modèles dont les validités sont obtenues par l'approche par résidus.

Cette fois il y a un léger dépassement de x, et la commande est plus perturbée qu'auparavant. Le coût total est de 8,74, mais le plus intéressant réside dans l'analyse des courbes $v_i(t)$.

Sans aucune information sur les domaines de validité, l'algorithme a su les reconnaître immédiatement.
Ainsi après l'initialisation (invisible sur les courbes, vu l'échelle), la validité de M_1 atteint tout de suite 1, pour décroître par la suite très vite jusqu'à v_{min}. La vitesse de décroissance importante montre que le système évolue très vite au démarrage.
Ensuite le deuxième modèle prend le relais, et sa validité dessine une courbe d'allure parabolique au voisinage de $x \approx 0,5$.
Fait non prévisible, le premier modèle voit sa validité augmenter à nouveau pour $x \approx 0,75$. Le premier modèle, pour ces valeurs d'environnement et de commande particulières, doit être alors relativement proche du système.
Puis pour $x \approx 1$, la validité du troisième modèle croit, très vite. Cette vitesse souligne que le système n'est pas du tout linéaire au voisinage de $x=1$. D'ailleurs dans zone de dépassement la validité du troisième modèle chute jusqu'à zéro. Un calcul montre ainsi que pour $x=1,1$, tous les modèles font des erreurs supérieures à 300%.

Finalement lorsque la consigne est pratiquement atteinte et que le système n'est plus excité, les validités apparaissent bruitées, et le troisième et le deuxième modèle voient leur validités converger de nouveau vers 1.
Ceci est très surprenant pour le deuxième modèle. En fait le problème tient dans la calibration du paramètre de décroissance de validité err_{max}, qui est trop élévé (l'approche par résidu n'est pas assez sévère).

L'approche par les résidus a donc permis de reconstruire les domaines de validité des 3 modèles, et les courbes obtenues sont très simples à analyser. Le seul problème apparaît lorsque le système n'est plus excité. Un simple test permettrait de ne pas tenir compte alors des résultats possibles, mais plusieurs autres approches peuvent résoudre ce problème.

La première consiste à adapter le paramètre err_{max} pour permettre une meilleure discrimination entre les modèles. La figure 4.25 donne ainsi les résultats pour err_{max}=10%.

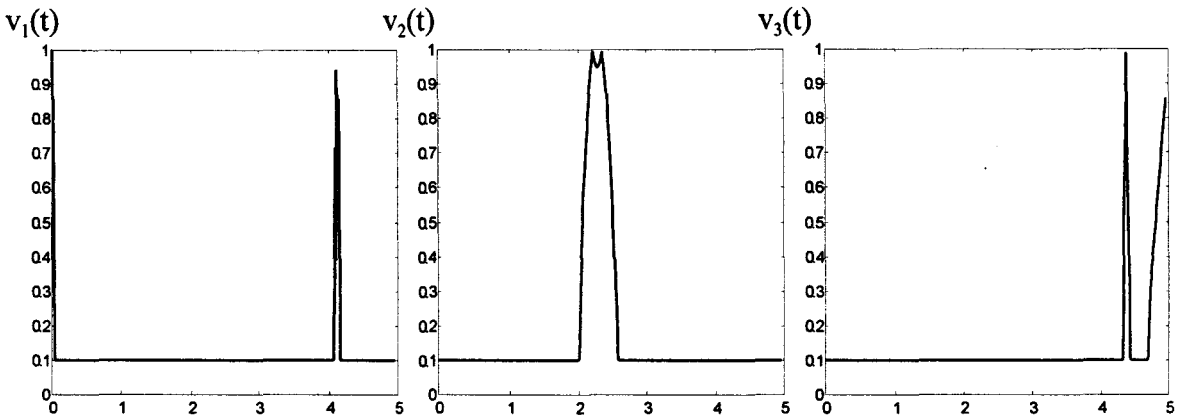

Fig. 4.25. Validités obtenues pour err_{max}=10%.

Il est aussi envisageable d'utiliser des techniques de filtrage. Sur la figure 4.24, les courbes apparaissent complètement lisses, seulement l'assignement de err_{max} à un très faible seuil (si un modèle fait 10% d'erreur, sa validité sera ainsi égale à v_{min}) retentit sur toute l'analyse. Ainsi les courbes apparaissent très resserrées par rapport à la première simulation. De même il existe de larges zones où aucun modèle n'apparaît meilleur que les autres, ce qui implique que dans ces zones la fusion réalise une simple moyenne des commandes u_i.

Lorsque des modèles approximatifs sont utilisés seuls avec un système complexe, il est préférable dans un premier temps de régler err_{max} à des valeurs plus raisonnables.

Il est ainsi intéressant de se construire une base de modèles (par exemple une petite dizaine), d'utiliser un CMM avec des commandes un peu robustes sur un système "biscornu" avec des termes "exotiques", éventuellement des discontinuités, de régler le paramètre err_{max} à 1000% ou plus, et de regarder finalement les résultats obtenus, souvent surprenants.

En étant plus sérieux, il faut admettre que l'approche par résidus est un outil puissant pour calculer les validités des modèles utilisés. L'analyse des résultats peut ainsi indiquer où et quand un modèle est nécessaire parce que les autres sont vraiment trop mauvais. Ces résultats peuvent aussi servir à la construction des domaines de validité pour les approches géométriques ou à la définition des fonctions d'appartenance en flou. Les implications de cette

approche en surveillance sont importantes, car elle permet de détecter des variations du système étudié par comparaison avec les validités calculées a priori, comme le montre l'exemple suivant.

Cependant cette approche, entièrement automatique, n'autorise pas l'intervention de l'utilisateur. Il n'est pas possible de forcer la validité d'un modèle par l'intervention sur son domaine de validité, afin de privilégier certaines réponses.

En outre les validités calculées sont théoriquement celles qui étaient vraies un pas de calcul précédent, puisque cette approche est basée sur des estimations futures.

Enfin cette approche ne permet pas d'estimer la qualité de modèles qualitatifs qui ne peuvent faire d'évaluation précises du système.

4.7.2 Système perturbé

Un CMM conçu à partir d'une bonne base de modèles est difficilement mis en défaut, néanmoins si le système évolue de telle manière que les modèles ne soient plus en adéquation avec le système, les performances peuvent être fortement dégradées. Au contraire l'approche par résidus permet de trouver dans la base quels sont les modèles qui correspondent réellement au système.

L'exemple qui suit va montrer cette propriété.

Le système étudié est celui déjà abordé, cependant il y a une perturbation sur le gain de telle sorte que le système entier est non stationnaire (τ est par hypothèse de travail toujours positif).

$$x(t) + \tau(x)\dot{x}(t) = k_1(x,t)u(t) \tag{4.39}$$

avec

$$\begin{aligned} k(x) &= 36x(t)(x(t)-1)+10 \\ \tau(x) &= 15-10x(t) \\ k_1(x,t) &= k(x)(\sin(t)+1.1) \end{aligned} \tag{4.40}$$

La perturbation a été conçue de telle sorte que le second modèle soit inopérant puisque pour x=0,5, le gain du système est augmenté.

La figure 4.26 montre le résultat obtenu avec un CMM basé sur une approche géométrique, les informations (à moitié fausses) étant celles du paragraphe 4.6.

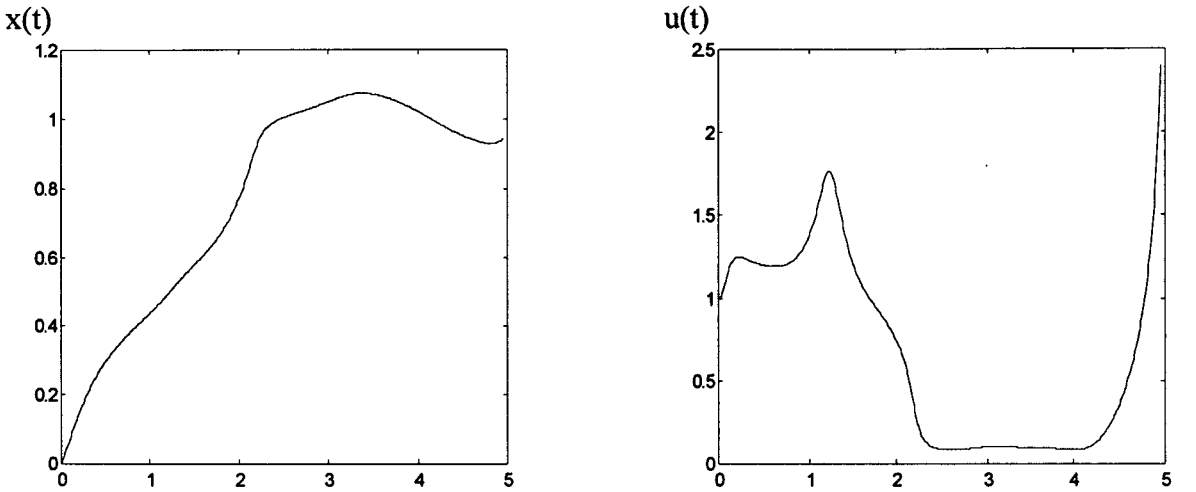

Fig. 4.26. x(t) et u(t) obtenus avec un CMM conçu pour le premier système, appliqué en fait au système perturbé.

Les figures 4.27 et 4.28 donnent respectivement les validités puis x(t) et u(t) pour le CMM dont les validités sont obtenues avec l'approche par résidus.

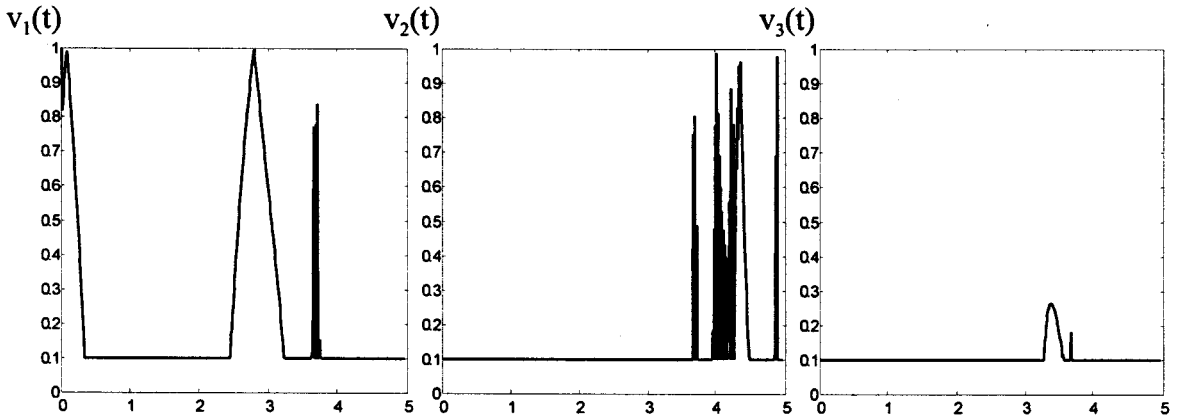

Fig. 4.27. $v_i(t)$ pour le système non stationnaire avec l'approche par résidus.

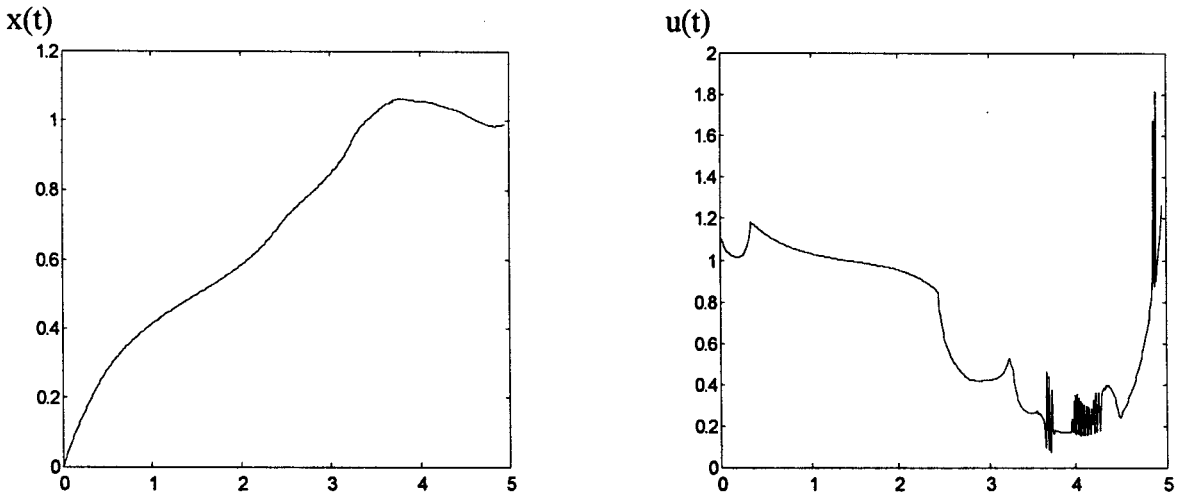

Fig. 4.28. x(t) et u(t) pour le CMM basé sur les résidus.

Dans les deux cas il y a un dépassement, et la commande s'effondre. Cependant puisque le gain s'effondre à son tour, les commandes augmentent fortement juste à la fin. Le coût avec le

CMM conçu pour le système non perturbé atteint 6,2, celui pour le CMM utilisant les résidus, 4,5.

L'analyse des validités obtenues peut maintenant être effectuée. Seul le premier modèle apparaît correspondre au système avec la commande calculée. La courbe $v_1(t)$ définit deux sous domaines de validité, pour x=0 et pour x≈0,7.

Par contre le deuxième et le troisième modèles ne semblent pas avoir une grande importance. En outre il existe de larges zones dans lesquelles aucun modèle n'est valide par rapport aux autres. Pour éviter cette situation, une seconde expérience est alors réalisée avec un paramètre err_{max} réglé à 200%. Les figures 4.29 et 4.30 donnent respectivement les validités puis x(t) et u(t) pour le CMM dont les validités sont obtenues par l'approche des résidus.

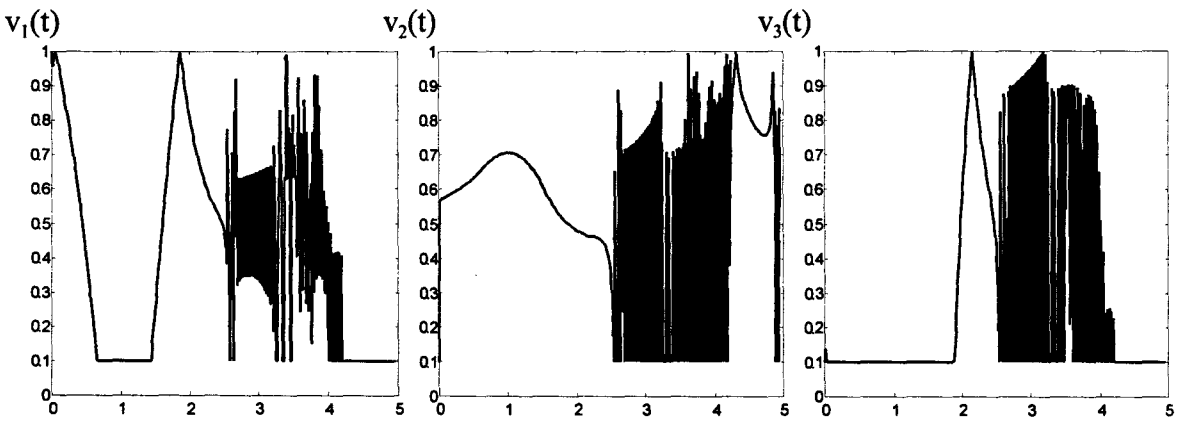

Fig. 4.29. $v_i(t)$ pour le système non stationnaire avec l'approche des résidus, lorsque err_{max}=200%.

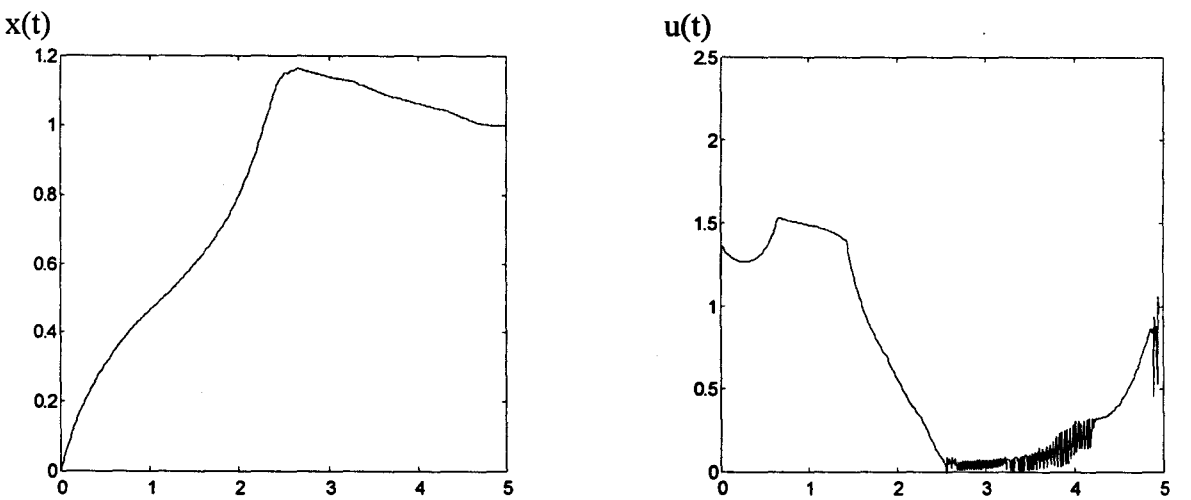

Fig. 4.30. x(t) et u(t) pour le CMM basé sur les résidus, lorsque err_{max}=200%.

Les signaux sont cette fois très perturbés lorsque le système n'est pratiquement plus excité. Pour une analyse fine il faudrait recourir à un filtrage car le paramètre err_{max} a justement été augmenté pour avoir plus d'information.

La zone d'influence du premier modèle ne fait pas apparaître de nouveaux sous domaines.

Vers 1 seconde, M_2 a un pic de validité à 0,7. Le troisième modèle a un pic de validité pour $t \approx 2s$, puis finalement le deuxième modèle pour $t \approx 4s$.

Il est possible de donner une première idée de la construction des domaines de validité à partir de ces informations. Par exemple pour $t \approx 4s$, le deuxième modèle a un haut degré de validité qui correspond sûrement à un domaine de validité maximale pour $t \approx 4s$, $x \approx 1$. Donc le domaine de validité D_2 de M_2 devrait inclure au moins $\{z=1\}$, si Z est défini égal à X. Cependant pour $t \approx 2{,}5s$, $x \approx 1$ aussi, or le deuxième modèle n'a pas de valeur de validité élevée en ce point comme le montre la courbe $v_2(t)$. Ces informations sont paradoxales, et le conflit ne peut être résolu qu'en supposant que Z contienne X et d'autres paramétres. Si Z est défini comme $X \times [0s,5s]$, alors il est possible de définir sans aucune contradiction un sous domaine de validité de M_2 par $D_2=\{z=1\} \times [4s,5s]$, et le système apparaît non stationnaire.

Beaucoup d'arguments militent en faveur de l'approche par résidus pour le calcul des validités. Très souple elle permet notamment une reconfiguration, et si le système évolue, cette méthode permettra d'utiliser immédiatement le meilleur modèle disponible.
Cependant cette méthode n'est pas une panacée. En effet deux arguments défendent les deux autres approches, floue et géométrique.
D'une part l'approche des résidus nécessite un plus gros volume de calcul. En effet si une approche multi-modèle est utilisée dans un but de commande, avec un C3MGI utilisant les approches floue ou géométrique, seules les équations traitant de la commande seront implantées, les équations relevant des modèles étant implicites dans celles de la commande. Le calcul des validités se fera très vite par les fonctions d'appartenance ou les distances. Au contraire avec l'approche par résidus, il faudra à la fois les équations de commande et les équations des modèles pour calculer les validités.
D'autre part l'approche des résidus est entièrement automatique et n'autorise aucune liberté au concepteur.

4.8 Conclusion: utilisation des approches multi-modèles

Les approches multi-modèles permettent l'étude et la commande des systèmes complexes pour lesquels l'utilisation d'un seul modèle est soit trop restrictive soit trop complexe. Elles autorisent alors la représentation de ces systèmes par plusieurs modèles, et leur commande par plusieurs lois de commande différentes. En cela, tous les résultats de l'automatique classique peuvent être utilisés.

Ce chapitre, mis à part les exemples, contient peu d'équations, et les rares présentes qui caractérisent les CMM sont très simples. Si la structure initiale d'un CMM est plus complexe que lorsqu'un seul modèle est utilisé, c'est uniquement pour permettre la fusion des informations diverses. Une fois définie cette structure initiale, l'utilisation d'un multi-modèle autorise une importante réduction de complexité dans les modèles, et au total les CMM sont d'une grande simplicité.

Ce chapitre n'a pas abordé la stabilité des contrôleurs multi-modèles. Si les premiers travaux tentèrent de définir une stabilité floue [Kiska et al, 1985], les plus récents utilisent les résultats de la stabilité des systèmes non linéaires [Tanaka, 1995], ou se ramenant à l'étude des systèmes linéaires au moyen d'hypothèses fortes [Cao et al, 1996]. L'étude de tels contrôleurs est complexe, et dépend toujours de cas particuliers [Marin et al, 1995a, 1995b].

D'un autre point de vue, les approches multi-modèles pourraient avoir des justifications théoriques venant de la théorie de l'algèbre différentielle. En effet un des objectifs de cette théorie est la décomposition en éléments simples différentiels des systèmes différentiels complexes.

Les approches multi-modèles apparaissent aussi très modulaires, il est ainsi possible de choisir quel type d'approche utiliser pour estimer les validités des modèles.
De manière très générale les multi-modèles se résument simplement à une nouvelle façon de penser en utilisant plusieurs sources d'information pour analyser un problème complexe. Cette manière de penser dépasse les cadres de l'automatique, et peut avoir des applications dans divers domaines, notamment en analyse de la décision [Delmotte et Dubois, 1996]. Il est aussi possible de transformer le multi-modèle en multi-critère [Dubois et al, 1995a].
Les derniers chapitres de cette thèse présentent quelques avancées et améliorations de l'approche multi-modèle.

CHAPITRE CINQUIEME

TECHNIQUES AVANCEES AVEC LE CONTROLEUR MULTI-MODELE

5.1 Introduction: Et maintenant, que faire?

Le chapitre précèdent a introduit l'idée de multi-modèle, qui consiste simplement à utiliser plusieurs outils en même temps au lieu d'un seul. Cette idée nouvelle a été lancée à partir de la brique élémentaire pour un automaticien, c'est à dire le modèle du système étudié. Dans certains cas il peut en effet être plus avantageux de recourir à plusieurs modèles qu'à un seul.

Le chapitre précèdent a défini les structures fondamentales utilisées par un contrôleur multi-modèle: d'abord les modèles élémentaires, qui doivent être complémentaires, sinon il n'y aurait aucun avantage à en utiliser plusieurs. Ensuite un mécanisme estimant la qualité de chaque modèle a été présenté. Son rôle est de pouvoir nuancer l'utilisation de tel ou tel modèle en fonction de sa pertinence, puisque, par définition, les modèles étant complémentaires, à chaque instant un modèle est meilleur que les autres. Cette pertinence a été représentée par des indices appelés validités. Enfin a été défini le mécanisme de fusion qui assure le calcul d'une loi de commande à partir des informations élémentaires. Plusieurs schémas d'architecture ont été donnés en fonction du but poursuivi, modélisation pure ou commande. Cette présentation a souligné les liens entre ces approches et les contrôleurs flous. Quelques simulations ont montré l'intérêt de ces approches, et leur simplicité d'usage.

Bien que ces architectures offrent déjà une grande souplesse et puissent aider à résoudre certains problèmes, elles peuvent être encore améliorées pour apporter plus de possibilités à l'automaticien. Ce chapitre détaille quelques raffinements qui furent apportés aux structures initiales. Ces raffinements vont montrer que les approches multi-modèles sont très modulaires, et facilement adaptables aux différents problèmes rencontrés.
La première amélioration concerne la définition d'une bibliothèque de lois de commandes. Il est en effet naturel, étant donné plusieurs modèles, de penser à plusieurs lois de commandes complémentaires, chacune avec ses avantages et ses inconvénients. Cette amélioration fera l'objet de la première partie.
La seconde partie concernera la présentation de quelques algorithmes d'adaptation de paramètres utilisés par les approches multi-modèles, en l'occurrence ceux utilisés pour estimer la validité des modèles.
La troisième partie présentera quelques perspectives de travail.

5.2 Bibliothèque de commandes

Parfois, il peut arriver que, étant donné un modèle représentatif d'un système, plusieurs lois de commandes soient intéressantes. Souvent un choix difficile s'offre: telle loi de commande serait particulièrement robuste, ce qui serait nécessaire au vu de la modélisation simplifiée disponible, telle autre serait par contre beaucoup plus efficace en régime nominal. En général un compromis est défini, qui ne satisfait parfaitement aucun des critères retenus en premier lieu.
Il serait très intéressant de définir un système qui permette d'utiliser plusieurs lois de commande diffèrentes en même temps, en choisissant la commande idoine en fonction de divers paramètres. Ainsi, l'utilisation de chaque loi de commande serait optimisée, et la commande finale serait adaptée à toutes les situations.
La notion de bibliothèque de modèle ayant été introduite, il est naturel de définir une bibliothèque de lois de commandes. Cette partie présente une structure de contrôleur multi-modèle et multi-commande [Delmotte et al, 1996b].

5.2.1 Structure d'un contrôleur à plusieurs modèles et commandes

Pour la description, n modèles M_i sont utilisés, avec la structure donnée ci dessous:

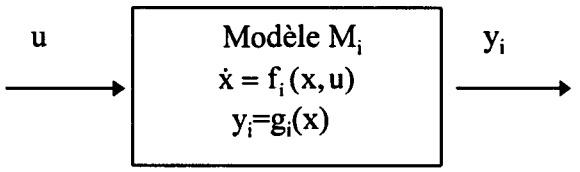

Fig. 5.1. Modèle élémentaire du contrôleur

De même le contrôleur dispose de m lois de commandes U_j différentes. Pour alléger l'écriture, ces lois de commandes sont supposées être représentées par des fonctions de l'état:

$$U_j: h_j: X \rightarrow U$$

Pour un schéma en poursuite, il faut définir la consigne. D'autres paramètres peuvent bien évidemment être introduits, et notamment les fonctions f_i et g_i.
Comme avec les modèles, est associé à chaque loi de commande U_j un indice de validité ou de pertinence, noté $t_j \in [0,1]$.
Plus t_j sera proche de 1, plus la loi de commande associée sera dite performante, ou valide.

La figure 5.2 donne un schéma de contrôleur utilisant ces n modèles et ces m commandes. Pour simplifier, l'estimation des validités des différents éléments n'est pas indiquée, seuls apparaissent les indices finaux.

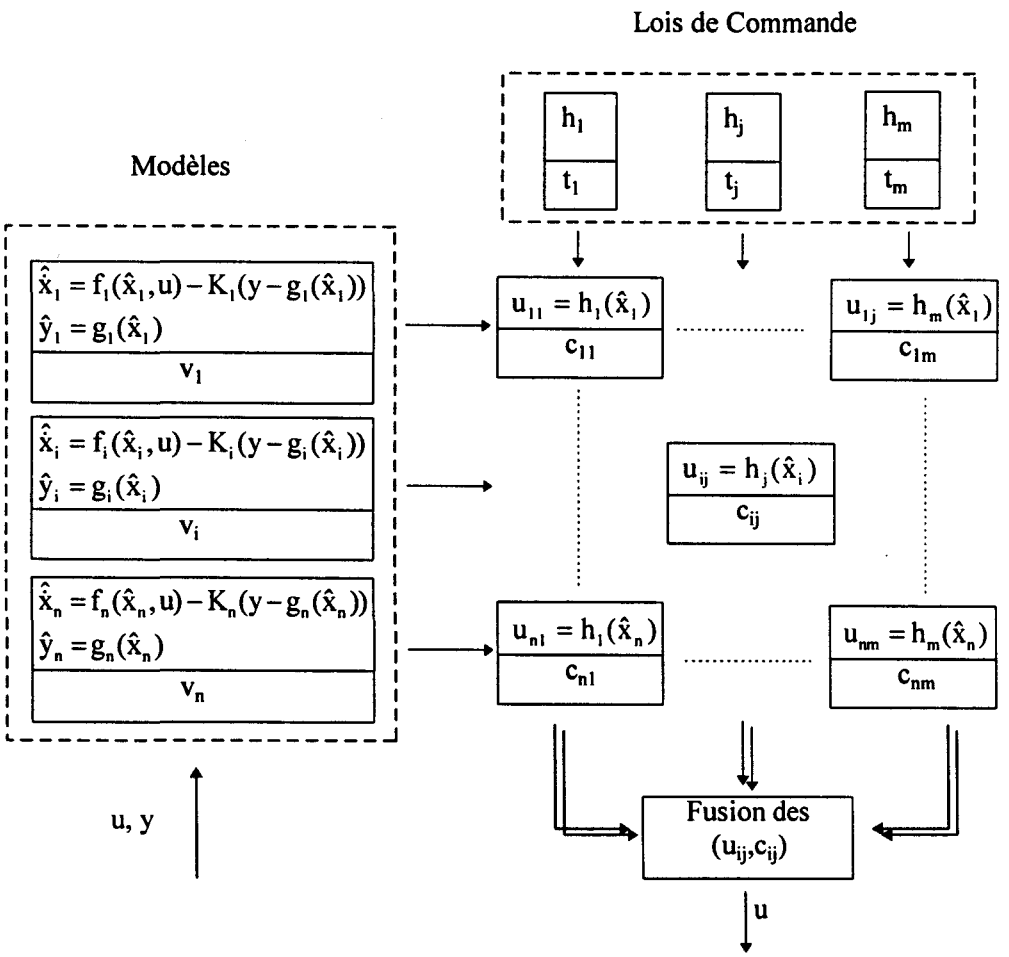

Fig. 5.2. Structure d'un contrôleur à plusieurs modèles et plusieurs commandes.

Dans ce schéma, la seconde case de chaque élément en est sa validité: v_i pour les modèles, t_j pour les lois de commande, et c_{ij} pour les lois de commandes effectives. Au maximum il y aura donc n×m cas possibles, mais il n'est pas obligatoire de calculer tous les types de commande pour tous les modèles donnés. Ainsi une commande très robuste pourra être calculée simplement pour un modèle très simplifié, à n'utiliser qu'en cas de divergence.

Pour comparaison, les exemples utilisés dans le chapitre 4 faisaient référence à la même loi de commande et au même état partagé par tous les modèles. Les variables de la loi de commande étaient simplement les paramètres des modèles.

Cette structure appelle plusieurs commentaires.

5.2.1.1 Observateurs

Pour la première fois sont mentionnés les observateurs. En effet la plupart des lois de commande nécessitent la connaissance de l'état du système. Celui ci sera fourni par les modèles eux-mêmes, mais il est nécessaire d'introduire des observateurs associés à ces modèles.
Dans la structure donnée à la figure 5.2, ceux ci apparaissent par l'entremise des gains K_i sur l'erreur de sortie.
Ces observateurs ne sont pas l'objet de cette étude, pourtant leur utilisation amène une remarque sur la validité de l'estimation de état. En effet la qualité de cette estimation repose à la fois sur la qualité du modèle, et sur l'adéquation du gain de l'observateur par rapport aux autres paramètres.
Une étude ultérieure de l'utilisation de tels observateurs impliquerait donc une définition de la qualité de l'estimation de l'état étant donné les validités des modèles et des observateurs associés.

5.2.1.2 Validité des lois de commande

L'utilisation de la bibliothèque de lois de commande est très similaire à celle des modèles. En ce qui concerne ces lois, seules apparaissent aujourd'hui intéressantes les méthodes traduisant une connaissance a priori:
- l'approche floue
- l'approche probabiliste
- l'approche géométrique

L'estimation en ligne de la validité des lois de commande soulève en effet le problème du critère choisi. Pour les modèles, il était naturel de faire appel à la qualité de leur modélisation du système, par le calcul des résidus.
Pour les commandes il est beaucoup plus difficile de définir un tel critère, permettant d'évaluer en temps réel la qualité d'une commande. Il faut en effet donner un sens à cette qualité, et pour l'instant aucun critère n'est vraiment universel.
Pour certain cas particuliers il pourrait être envisagé de définir des critères pour pallier à certains inconvénients. Ainsi l'exemple traité dans cette partie pour illustrer l'approche multi-commande utilise les modes glissants. Ceux ci introduisent des discontinuités dans la commande, et amènent à un phénomène de broutement de la commande. Avec de tels problèmes, un critère permettant d'évaluer la qualité des lois de commandes pourrait être basé sur la dérivée de la commande. Mais même ce dernier critère est discutable, car certaines lois de commande peuvent solliciter fortement les actionneurs en début de commande, et peu par la suite...

Il pourrait être plus intéressant de parler de critère de préférence: en cas de poursuite, si l'erreur restait toujours importante sur une longue période (l'utilisateur devrait alors créer un indice mesurant cette erreur au cours du temps), l'algorithme estimant les validités des lois de commande pourrait privilégier les lois minimisant l'erreur au détriment de celles qui minimisent la consommation.
Une telle approche a été développée dans Dubois [Dubois et al, 1995], mais pour remplacer un critère quadratique, compliqué à calculer, par deux critères associés à deux commandes optimales.

5.2.1.3 Validité d'une commande élémentaire

Une commande élémentaire u_{ij} est celle qui résulte de l'application d'une loi de commande U_j au modèle M_i. Etant données la validité v_i du modèle M_i et celle t_j de la loi U_j, quelle peut être la validité de u_{ij}?

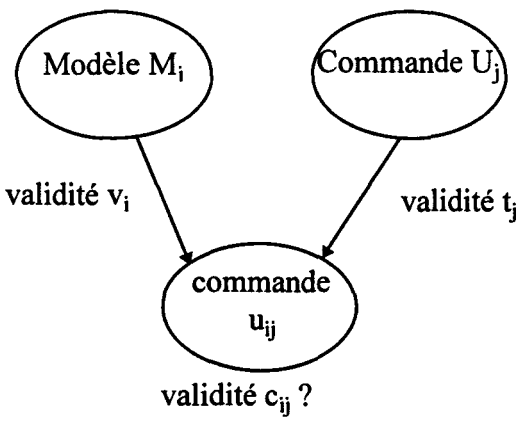

Fig. 5.3. Quelle est la validité d'une commande obtenue par un type de commande U_j appliqué à un modèle M_i particulier?

Le premier exemple est celui d'un modèle parfait (v_j=1), associé à une très mauvaise loi de commande. Par exemple en telle zone de fonctionnement, l'utilisateur préfère minimiser les erreurs, et cette loi minimise en fait la consommation...Dans ce cas il ne faudra pas appliquer la commande élémentaire associée. Un autre cas est celui d'un modèle totalement faux utilisé avec une commande parfaite. A moins que la commande ne dépende pas du tout du modèle, il ne faudra surtout pas utiliser la commande élémentaire.

Il apparaît donc que la validité d'une commande élémentaire u_{ij} est une t-norme de la validité du modèle M_i et de la loi U_j. Afin d'avoir un comportement continu, le choix s'est porté vers l'opérateur produit:

$$c_{ij} = v_i t_j \tag{5.1}$$

5.2.2 Exemple de contrôleur à plusieurs commandes

L'application qui suit utilise la commande à modes glissants. Le paragraphe 5.2.2.1 détaille quelques notions concernant ce type de commande. Grossièrement elle introduit des commutations sur le vecteur commande au voisinage d'une surface de l'espace d'état qui représente la dynamique de retour choisie. Cette loi est robuste, mais du fait des discontinuités sur la commande, un phénomène de broutement apparaît: les variables du système subissent

des oscillations à hautes fréquences. Ce phénomène de broutement est le principal inconvénient de cette loi de commande.
Ce broutement fait l'objet de l'étude qui suit. Cette dernière n'a pas pour objectif de révolutionner la commande à modes glissants, mais simplement de montrer la facilité de mise en œuvre de la multi-commande.

5.2.2.1 Notions de la commande à modes glissants

Soit un modèle linéaire monovariable représenté dans l'espace d'état par l'équation 5.2:

$$\frac{dx}{dt} = Ax + Bu \tag{5.2}$$

L'utilisateur souhaite implanter une commande telle que le système bouclé ait une dynamique de retour imposée à l'avance. Cette dynamique de retour est caractérisée dans l'espace d'état par une surface de glissement S(x), caractérisée par le vecteur c:

$$S(x) = c^t x = 0 \tag{5.3}$$

En effet lorsque $S(x) = 0$ et $\dot{S}(x) = 0$, le système se comporte comme s'il était soumis à une entrée appelée commande équivalente u_e:

$$u_e = -(c^t B)^{-1} c^t Ax \tag{5.4}$$

Lorsque ces deux conditions sont réunies, un mouvement de glissement apparaît. Mouvement, parce que le système tend toujours vers la consigne, objet de la commande, et glissement, parce que ce mouvement se fait avec S(x)=0.

Lorsque $S(x) \neq 0$, l'objet de la commande est de forcer le système à rejoindre cette surface. Utkin [Utkin, 1977, 1992], a montré que la commande devait être telle que

$$\forall x, S\dot{S} < 0 \tag{5.5}$$

pour qu'un mouvement de glissement apparaisse.
Une solution à ce problème consiste alors à prendre une commande de la forme

$$u = (c^t B)^{-1}\left(-c^t Ax + u_d\right) \tag{5.6}$$

avec

$$u_d = -k\,\mathrm{sign}(S) \tag{5.7}$$

u_d est appelée la partie discontinue de la commande. La commande est donc la somme d'un terme continu, la commande équivalente, et d'un terme discontinu, fonction de S(x).

Le paramètre k (positif) est appelé le gain de la commande, et module l'importance de la valeur discontinue.
Plus le gain est important, plus la commande est robuste et peut absorber des erreurs de modélisation [Buhler, 1994; Perruquetti, 1992]. Cependant plus ce gain est important, plus des oscillations sur les variables apparaissent, augmentant le phénomène de broutement.

5.2.2.2 Les données

L'évolution du système est caractérisée par les équations:

$$(S) \qquad \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = (5+4\sin(t))x_1x_2 - x_1 + u \\ y = x_1 \end{cases} \tag{5.8}$$

et le seul modèle accessible est:

$$(M) \qquad \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \alpha x_1x_2 - x_1 + u \\ y = x_1 \end{cases} \tag{5.9}$$

En fait la partie non stationnaire est considérée comme une perturbation dont seule la plage de variation est connue: $\alpha \in [1,9]$.
L'objectif de la commande est de conduire le système de x_0=0.5 à x_f=1 en 10 secondes. La dynamique de retour choisie est donnée par l'équation suivante:

$$\frac{y_d}{u} = \frac{2}{(s+1)(s+2)} \tag{5.10}$$

Cette dynamique correspond à la surface de glissement S:

$$S = \frac{2}{3}\left(x_1 - \Gamma(t)\right) + x_2 \tag{5.11}$$

avec $\Gamma(t)$ la consigne à poursuivre, ici constante et égale à 1.

5.2.2.3 Le contrôleur à plusieurs commandes

Une solution classique au problème de broutement consiste à adapter les gains utilisés en fonction de la distance de l'état à la surface de glissement. En effet les commutations n'apparaissent qu'à son voisinage. C'est pourquoi il est tout à fait licite d'utiliser un grand gain lorsque l'erreur est importante, afin de rejoindre le plus vite possible la surface de glissement, et au contraire il faut utiliser un petit gain au voisinage de celle-ci.
Afin d'implanter cette fonction par un contrôleur il faut d'abord définir des modèles à paramètres constants. Un modèle pourrait suffire, mais il serait parfois trop imprécis. Avec une interpolation linéaire, l'utilisation de deux modèles semble un choix correct.

Les deux modèles seront notés M_α, avec une valeur α choisie. La validité de ces deux modèles n'étant pas connue a priori, elle sera estimée par l'approche des résidus (définis sur y, avec une décroissance de validité err_{max} égale à 50%).

Maintenant il faut définir les lois de commande. Pour retrouver le résultat classique il suffit de considèrer 2 lois de commande, chacune avec un gain diffèrent. Avec 2 lois, la fusion donne une loi similaire à une commande par mode glissant avec adaptation du gain.

La première loi U_1 correspond à un petit gain, et la seconde à un grand gain. L'expertise nous permet de définir leurs domaines de validité dans l'espace d'état:

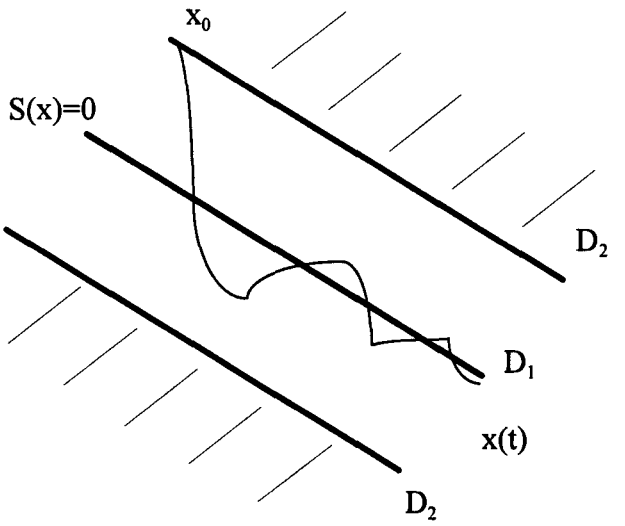

Fig. 5.4. Domaines de validité des deux lois de commande à modes glissants.

La variable utilisée est la distance de l'état à la surface de glissement S(x)=0. Cette distance qui peut varier beaucoup, est normalisée afin d'en déduire les validités t_j. Cette normalisation fait appel à la plus grande distance mesurée, habituellement celle de l'état initial x_0.

$$D_1=\{x\in X : d(x,S)=0\}$$

$$D_2=\{x\in X : d(x,S)\geq d_m\} \tag{5.12}$$

La conjonction de plusieurs modèles et plusieurs commandes donne finalement les commandes élémentaires par application de 5.6:

$$u_{ij}=(x_1-\tfrac{2}{3}x_2)-\alpha_i x_1 x_2-k_j \mathrm{Sign}(S) \tag{5.13}$$

5.2.2.4 Résultat classique

La première simulation correspond à une commande simple n'utilisant qu'un seul modèle, et une seule commande. Le modèle a été pris pour α=5 qui est la valeur médiane.
La commande correspond à un gain k=0.7, valeur intermédiaire entre 0.4 et 1. 0.4 est une borne approximative nécessaire pour qu'apparaisse un mouvement de glissement, et 1 est une valeur faisant converger relativement vite le système (ces deux valeurs sont prises comme gains des deux commandes du multi-modèle).
Les résultats donnent l'erreur relative définie par:

$$e_r=(y_d-y)/y_d \tag{5.14}$$

ainsi que la valeur de S, et la commande u.
Celle ci est fournie par l'équation suivante:

$$u=(x_1-\tfrac{2}{3}x_2)-5x_1x_2-0.7\mathrm{Sign}(S) \tag{5.15}$$

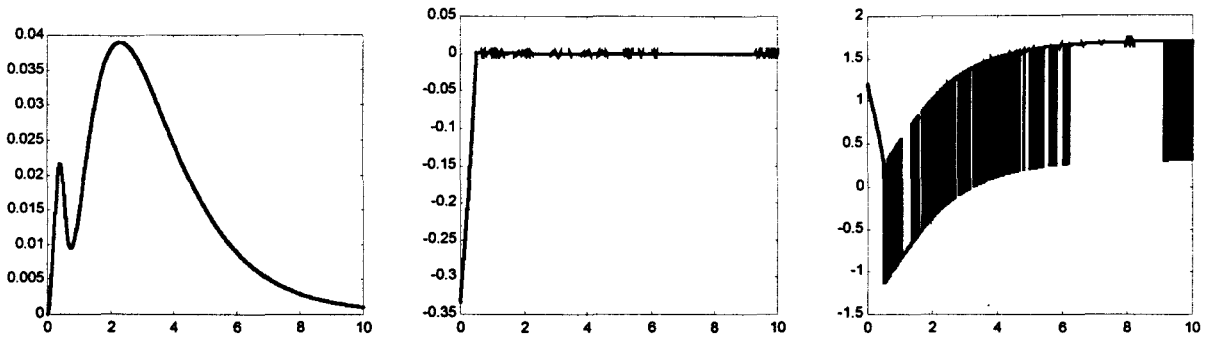

Fig. 5.5. $e_r(t)$, S(t) et u(t) pour la commande à mode glissant classique (cf. 5.15)

L'erreur relative est toujours faible, inférieure à 4%, et tend vers 0, le mouvement de glissement apparaît bien après 0.3 secondes environ, par contre il existe un broutement très important sur la commande.

5.2.2.5. Résultats avec deux modèles bien espacés.

L'utilisateur, par prudence, a choisi a priori deux modèles pour deux valeurs de α symétriques sur sa plage de fonctionnement.
Les deux modèles correspondent respectivement à M_2 pour α=2 et M_8 pour α=8.

La figure 5.6 donne respectivement $e_r(t)$, S(t) et u(t), et la figure 5.7 donne les validités (avant renforcement) des deux modèles.

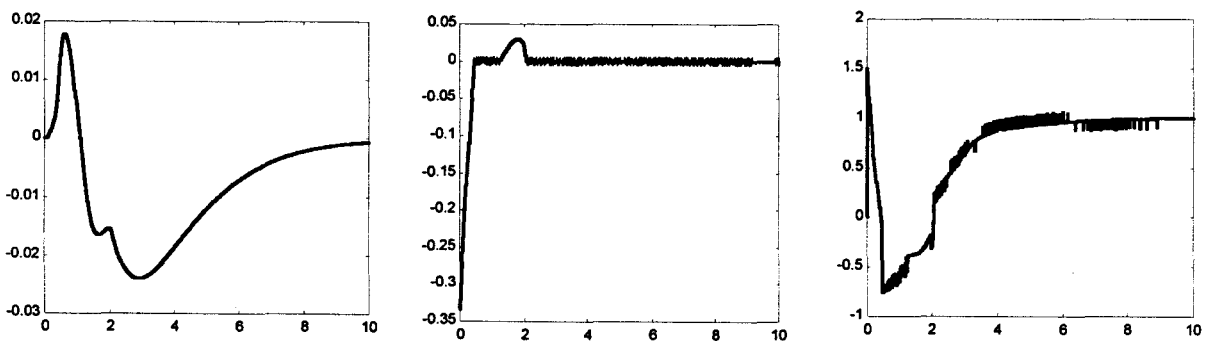

Fig. 5.6. $e_r(t)$, S(t) et u(t) pour le contrôleur à deux modèles et deux commandes.

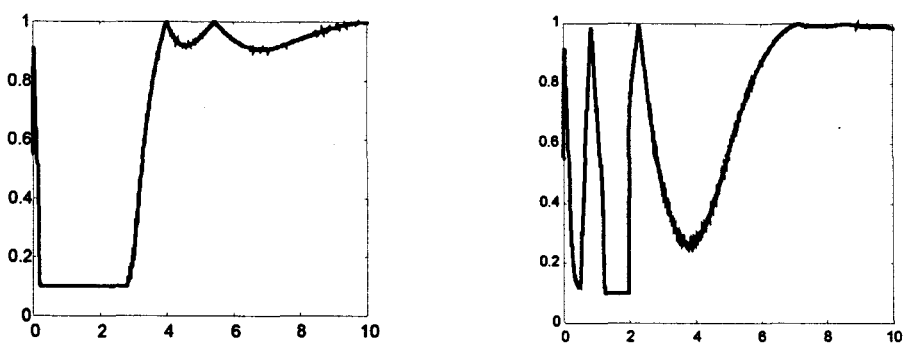

Fig. 5.7. Validité de M_2 et Validité de M_8.

L'erreur est cette fois un peu plus faible, puisqu'elle n'atteint pas 3%. Le broutement a été fortement réduit. Cependant, si le mouvement de glissement est bien présent, pour t≈2

secondes, S(x) est différente de 0 de manière visible et le mouvement de glissement disparaît. Une explication est donnée par l'analyse des courbes de validité des modèles.

En effet, après une phase d'initialisation, les deux validités atteignent un seuil minimal, puis le modèle M_8 prend le relais. Les deux pics correspondent à une valeur de 5+4sin(t) égale à 8. Pour t≈2 secondes, le modèle M_8 fait cependant beaucoup d'erreur, et sa validité atteint de nouveau le seuil minimal.
Le contrôleur multi-modèle réalise donc une moyenne entre les deux modèles, donnant un modèle implicite associé à α=0.7. Contrairement à la première simulation, les valeurs particulières de x_1 , x_2 et des autres paramètres sont telles que la condition $S\dot{S} < 0$ n'est plus respectée. Une solution classique est alors d'augmenter le gain de la commande, cependant cela augmente aussi le broutement.

Le contrôleur à plusieurs modèles peut apporter deux solutions différentes.
La première consiste à adapter la valeur de err_{max}. En effet la décroissance de validité en fonction des résidus est mal conditionnée, puisque pour t≈2 secondes, le modèle M_8 correspondant à α=8 est considéré comme faisant autant d'erreur que le modèle M_2 correspondant à α=2. Si le paramètre err_{max} était adapté au cours du temps, le modèle M_8 apparaîtrait bien meilleur que le modèle M_2, et la différence entre le modèle implicite et le système serait finalement beaucoup plus faible.

La seconde solution consiste à changer les modèles. En effet l'analyse complète des courbes de validités montre que le modèle 2 est en fait inutile: pour t supérieur à 3 secondes, même si le modèle M_2 a un peu d'avance sur le modèle M_8, les deux validités tendent vers 1. Ceci s'explique par le fait que la consigne est atteinte, et que le régime permanent est pratiquement atteint. Comme en régime permanent x_2=0, et que la seule incertitude porte justement sur x_2, les deux modèles sont équivalents. La seconde simulation sera donc effectuée en choisissant d'autres modèles.

5.2.2.6. Résultats avec deux modèles mieux choisis.

Puisque le modèle M_2 est inutile, il sera remplacé par un modèle correspondant à un α moyen, égal à 5. Le modèle M_8 sera donc remplacé par le modèle M_9 pour optimiser l'interpolation entre les modèles.
Les figures 5.8 et 5.9 donnent respectivement $e_r(t)$, S(t) et u(t), puis les validités.

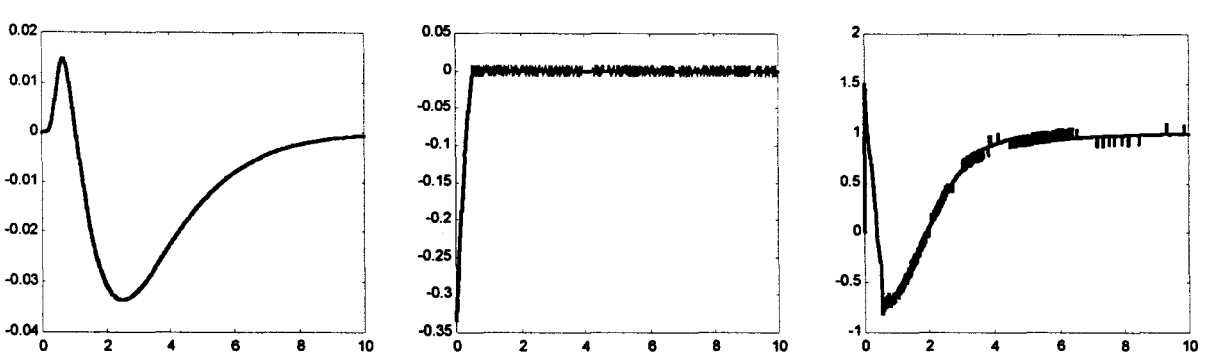

Fig. 5.8. $e_r(t)$, S(t) et u(t) pour le contrôleur utilisant M_5 et M_9.

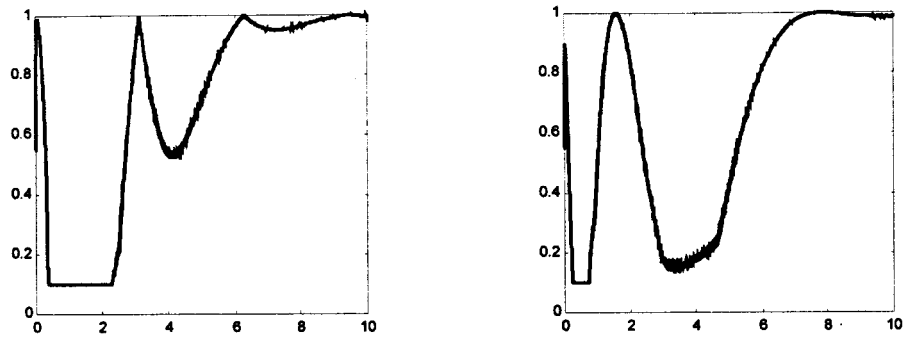

Fig. 5.9. Validité de M_5 et Validité de M_9.

Cette fois ci le mouvement de glissement est continu. Cependant l'erreur relative est un peu plus importante pour $t \approx 2$ secondes, puisque le modèle M_9 remplaçant le modèle M_8 ne joue un rôle important qu'un peu plus tard, ce qui entraîne une approximation du système par le modèle implicite un peu moins bonne.

5.3 Adaptation de l'estimation des validités

Toute approche multi-modèle est basée sur une évaluation correcte de la validité des éléments à fusionner. Si cette évaluation n'est pas correcte, la fusion fournira des résultats faux. Cette partie donne quelques outils pour affiner l'estimation des validités, afin de rendre le contrôleur multi-modèle plus autonome et indépendant des erreurs de l'utilisateur.
Le premier paragraphe donne un affinement de l'estimation par les résidus, avec une adaptation en ligne, le second paragraphe en propose un autre, concernant l'étape de renforcement.
Encore une fois ces deux paragraphes vont souligner la grande modularité des approches multi-modèles, et d'autres techniques adaptatives peuvent évidemment être utilisées.

5.3.1 Ajustement des validités fournies par les résidus

Ce paragraphe propose une amélioration de l'estimation des validités fournies par l'approche des résidus.
En effet, comme le montre le paragraphe suivant, l'approche classique nécessitait une intervention a priori de l'utilisateur pour évaluer un paramètre très important. Plusieurs stratégies d'évaluation automatique de ce paramètre seront alors proposées, et leur intérêt sera montré sur un exemple.

5.3.1.1 Retour sur l'approche classique des résidus

Cette approche repose sur l'idée très simple que la qualité d'un modèle est liée à sa qualité d'estimation de certaines variables du système.
Ainsi si le modèle ne fait pas d'erreur d'estimation, sa validité est définie égale à 1, le maximum. Si le modèle fait des erreurs, sa validité est diminuée.
Une fonction décroissante de l'erreur en module représente donc facilement cette idée.
La figure 5.10 rappelle la structure de la fonction utilisée:

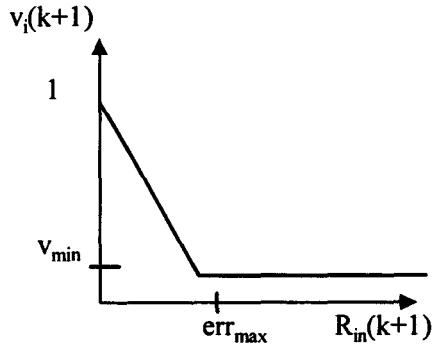

Fig. 5.10. Validité du modèle M_i en fonction de l'erreur d'estimation R_{in} normalisée.

La fonction la plus simple possible est en fait une fonction linéaire, et c'est celle qui avait été utilisée.
Pour définir cette fonction, deux limites doivent être introduites. Le première est naturelle, et n'appelle pas de commentaires: si l'erreur est nulle, la qualité du modèle est considérée comme maximale.
Par contre la deuxième limite est plus problématique. Quand est-il possible d'affirmer qu'un modèle est complètement faux, et qu'il doit être par consequent complètement rejeté au niveau de la fusion?
Un modèle qui fait des erreurs de 10% doit-il être rejeté? Ou bien une erreur de 50% peut-elle être encore acceptable, avec une validité moyenne, et dans cette acception seuls les modèles dépassant 100% d'erreur auront-ils une validité minimale?

Avec l'approche classique, l'utilisateur doit donc fournir une estimation du paramètre réglant cette décroissance de validité en fonction de l'erreur. Ce paramètre est noté err_{max}. Dans le chapitre 4, les simulations avaient donné de bons résultats, parce que ce paramètre err_{max} était bien adapté pour les problèmes considérés.

En général cependant, l'utilisateur peut avoir beaucoup de mal à estimer correctement ce paramètre, puisque justement l'approche des résidus est utilisée lorsqu'il n'existe pas de moyen simple d'estimer la validité des modèles.

Or une estimation, même grossière, de ce paramètre, est indispensable au bon fonctionnement de la fusion. En effet si la bibliothèque ne comprend que de bons modèles, qui ne font que de faibles erreurs, le multi-modèle conçu dans un souci de performances de servira à rien si le paramètre err_{max} est très important. Dans ce cas toutes les validités seront proches de 1, et même l'étape de renforcement ne pourra corriger complètement cette mauvaise estimation initiale. Dès lors, à l'étape de fusion, le modèle implicite sera une moyenne des modèles initiaux, avec de petites fluctuations.

De même, dans le cas contraire, si le paramètre err_{max} est trop petit, l'approche des résidus notera très sévèrement les modèles, et si la bibliothèque ne comprend que des modèles moyens, encore une fois tous les modèles seront notés de la même manière, cette fois avec une validité minimale égale à v_{min}.

C'est pourquoi il serait intéressant de définir un mécanisme permettant d'estimer en ligne la valeur de ce paramètre, sans intervention de l'utilisateur, de telle sorte qu'à chaque cycle, l'approche des résidus permette d'introduire un certain échelonnement entre les validités. La

suite de cette partie propose un tel mécanisme qui sera ensuite testé sur un exemple [Delmotte et al, 1997b].

5.3.1.2 Stratégies d'adaptation

La méthode proposée se veut le plus possible indépendante de l'utilisateur. Le résultat est en fait purement mécanique, la méthode ne dépendant que des résidus générés à chaque étape. En fait il est possible de faire en sorte, en adaptant le paramètre err_{max}, que le meilleur modèle atteigne une valeur élevée de validité, ou qu'au contraire, le plus mauvais modèle ne descende pas en dessous d'un certain seuil de validité.
De telles contraintes amènent à ce qui sera appelé par la suite des stratégies d'adaptation de err_{max}. Deux seulement seront détaillées, les autres étant générées suivant le même principe (par exemple faire en sorte que le résidu moyen corresponde à une validité égale à 0.5).

La première stratégie, notée S_1, impose que le meilleur modèle atteigne un seuil de validité v_{meill} donné, entre 0.5 et 1 par exemple.
Il faut donc que err_{max} respecte l'équation:

$$v_{meill} = 1 - \min(R_{in}(k+1)) / err_{max}(k+1) \tag{5.16}$$

La valeur du paramètre recherché est donc fournie par:

$$err_{max}(k+1) = \min(R_{in}(k+1)) / (1 - v_{meill}) \tag{5.17}$$

Le calcul des validités pose alors problème uniquement lorsque tous les modèles s'avèrent parfaits ($\min(R_{in})=0$), mais ce cas rare est traité en assignant err_{max} à une valeur différente de 0, ce qui n'a pas d'influence sur la fusion.

La seconde stratégie développée, S_2, est la réciproque de S_1. Elle impose que le plus mauvais modèle atteigne au moins un seuil minimal v_{pire}.

$$err_{max}(k+1) = \max(R_{in}(k+1)) / (1 - v_{pire}) \tag{5.18}$$

Les deux méthodes d'estimation du paramètre de décroissance de validité err_{max} imposent donc l'introduction de deux nouveaux paramètres, v_{meill} ou v_{pire}. Cependant ces deux paramètres n'ont pas d'influence sur le résultat, ce sont juste des facteurs d'échelonnement. Dans les simulations qui suivent, v_{meill} a été pris égal à 0.9 et v_{pire} à 0.1.

L'objectif principal du mécanisme d'adaptation a donc été parfaitement atteint: le paramètre err_{max} est ajusté automatiquement à chaque étape de telle sorte que les modèles, si leurs résidus sont différents, aient des validités différentes.

En fait les deux stratégies d'adaptation peuvent même servir d'étape de pré-renforcement. En effet, si le meilleur modèle fait une erreur normalisée de 10^{-4}, et si l'erreur sur le deuxième meilleur modèle passe immédiatement à 10^{-2}, du fait de l'approche linéaire retenue, si la validité du premier est égale à $v_{meill}=0.9$, celles du deuxième et des suivants seront égales à 0 ou v_{pire}, c'est à dire une valeur faible.

Une dernière remarque concernant ce mécanisme peut être faite. En effet, en ligne, l'estimation des validités est effectuée à paramètre err_{max} variable. Il pourrait être intéressant, a posteriori, d'étudier, à paramètre err_{max} constant, les valeurs de ces validités, afin d'en déduire

dans quelle zone il faudrait introduire de nouveaux modèles, et dans quelle zone un modèle pourrait éventuellement être enlevé de la bibliothèque.
La formule qui suit réalise la conversion entre les validités:

$$v_i^{err_{max2}} = \max\left(0{,}1 - \frac{err_{max1}(1 - v_i^{err_{max1}})}{err_{max2}}\right) \tag{5.19}$$

5.3.1.3 Exemple avec le second système perturbé

Le système étudié sera le premier système déjà défini, mais avec une perturbation non stationnaire différente. En effet au chapitre 4 les simulations ont été réalisées avec un paramètre err_{max}, qui, bien que constant, était particulièrement bien adapté. Le système qui suit, en étant relativement proche du premier système perturbé, montre, qu'avec la même estimation "raisonnable" de err_{max}, les résultats ne sont pas les meilleurs possibles.

Le système est représenté par la classique équation:

$$x(t) + \tau(x)\dot{x}(t) = k_1(x,t)u(t) \tag{5.20}$$

avec

$$\begin{aligned} k(x) &= 36x(t)(x(t)-1)+10 \\ \tau(x) &= 15-10x(t) \\ k_1(x,t) &= k(x)(\sin(t)+1.1) \end{aligned} \tag{5.21}$$

La figure 5.11 montre les évolutions de ces différents paramètres

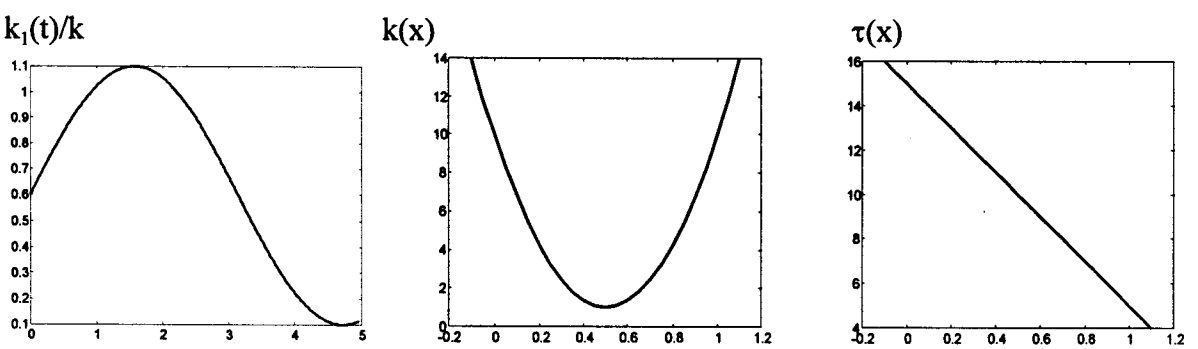

Fig. 5.11. Evolutions des paramètres du second système perturbé.

L'utilisateur n'a en fait obtenu que la relation suivante:

$$x + \tau\dot{x} = ku$$

qui est celle du premier système, mais non perturbé.
Comme avec les simulations du chapitre 4, l'utilisateur va donc définir un multi-modèle avec 3 modèles:

M_1	M_2	M_3
$D_1=\{x=0\}$	$D_2=\{x=0.5\}$	$D_3=\{x=1\}$
$x = \frac{10}{1+15s}u$	$x = \frac{1}{1+10s}u$	$x = \frac{10}{1+5s}u$

La figure 5.12 donne les résultats avec une estimation des validités utilisant l'approche géométrique:

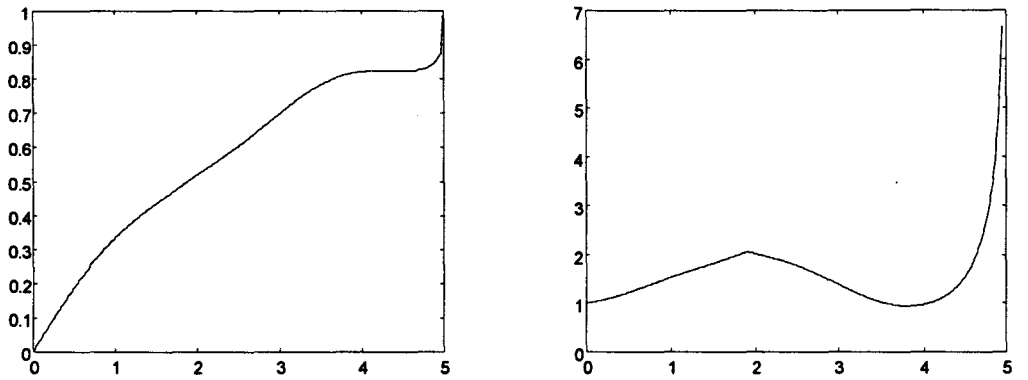

Fig. 5.12. x(t) et u(t) pour le second système perturbé. Approche géométrique.

En fait, comme avec le premier système perturbé, les informations a priori sont fausses. Cette fois les modèles 1 et 3 sont utilisés alors qu'il ne le devraient pas, puisque le gain réel est beaucoup plus faible. Le coût dépasse 45 puisque la commande dépasse 800.

La figure 5.13 donne les validités obtenues avec l'approche des résidus, avec un paramètre err_{max} constant égal à 50%. La figure 5.14 donne x(t) et u(t) respectivement.

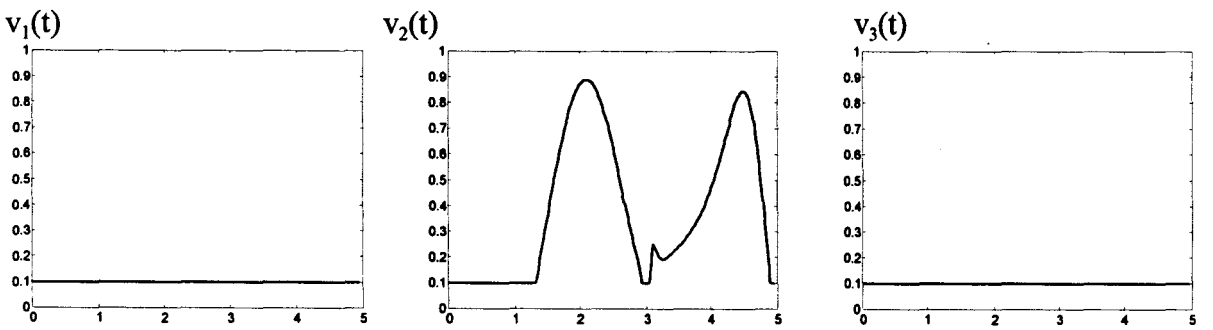

Fig. 5.13. Validités des trois modèles, avec l'approche des résidus classiques.

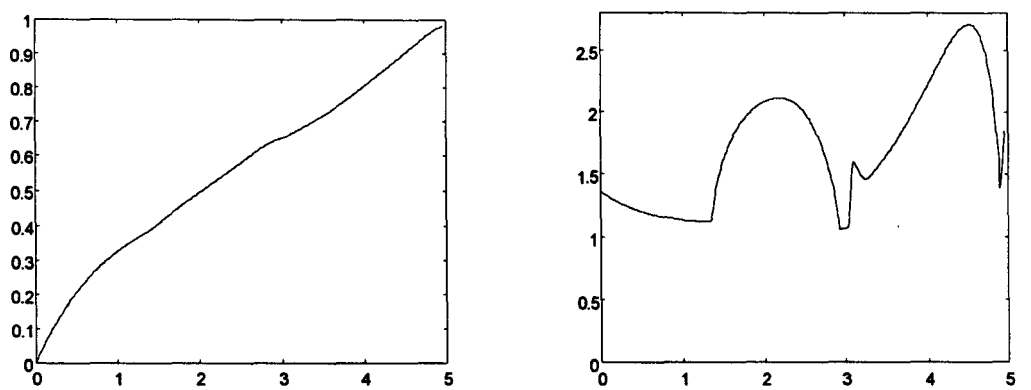

Fig. 5.14. x(t) et u(t) pour le second système perturbé. Approche des résidus classiques.

Jusqu'à t = 1 seconde, l'estimation des validités ne permet pas de faire une différence entre les modèles, puisqu'ils sont tous considérés comme très mauvais. Ensuite le deuxième modèle prend le relais, puis voit sa validité retomber à un niveau égal à celle des autres. Il connaît un

pic de validité à la fin, à nouveau avant de retomber. En fait avec cette expérience, les modèles 1 et 3 ne servent qu'à perturber le rôle du modèle 2.
Le coût est égal à 18, car l'approche des résidus classique a permis de détecter la fausseté des modèles 1 et 3, et la commande est finalement plus raisonnable qu'avec l'approche géométrique. Maintenant, les figures 5.15 et 5.16 donnent les validités estimées grâce à l'approche des résidus avec adaptation, respectivement avec S_1 et S_2.

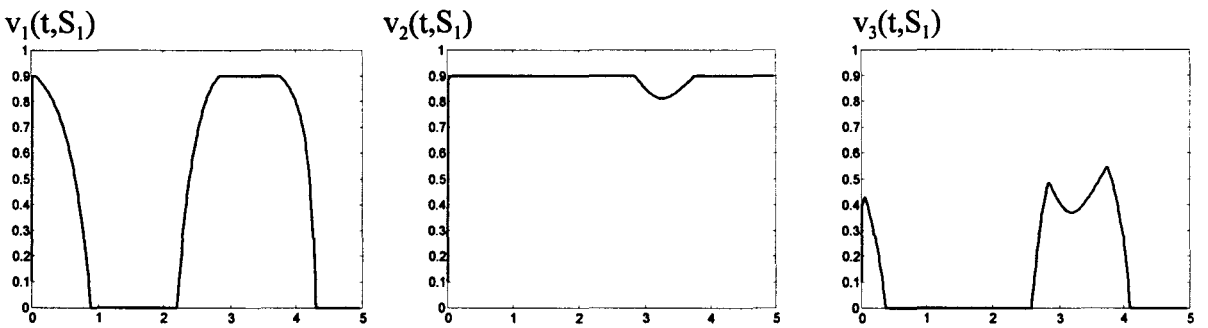

Fig. 5.15. Validités avec les résidus adaptés (S_1).

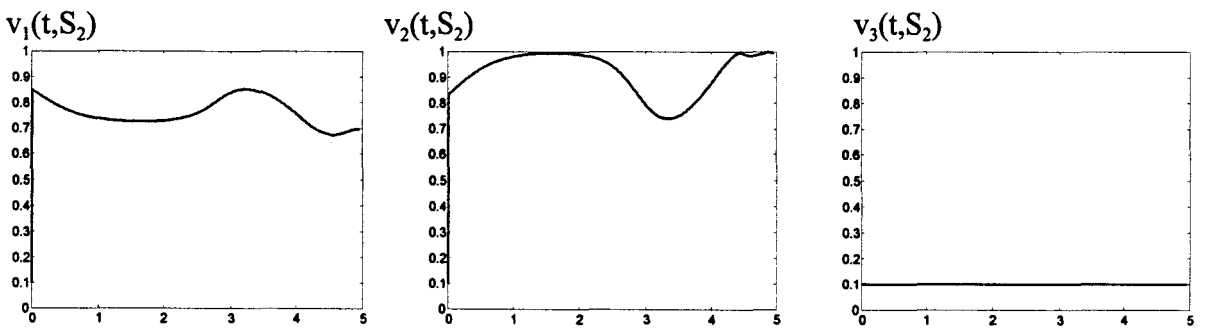

Fig. 5.16. Validités avec les résidus adaptés (S_2).

Avec la première stratégie le meilleur modèle a toujours une validité égale à 0.9, tandis qu'avec le seconde stratégie, le moins bon modèle à toujours une validité égale à 0.1.
Dans les deux cas, c'est le deuxième modèle qui apparaît le plus souvent comme le meilleur modèle. Seul le modèle 1 le dépasse en qualité pour $t \approx 3.5$ secondes. Le troisième modèle est toujours considéré comme le plus mauvais modèle.

La figure 5.17 donne x(t) et u(t) pour les deux stratégies d'adaptation de err_{max}.

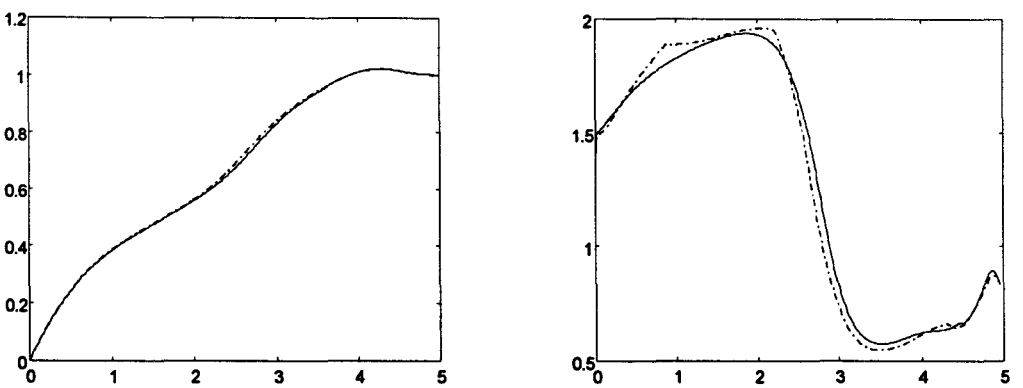

Fig. 5.17. x(t) et u(t) avec une approche des résidus avec adaptation de err_{max}.
'-.-.' correspond à S_1, et '—' à S_2.

Les deux résultats sont très comparables, et le coût ne dépasse pas 10.88. Ceci s'explique pour deux raisons: d'abord au début, le modèle M_2 est bien perçu comme le meilleur modèle, et la commande est beaucoup plus importante qu'avec l'approche des résidus classique, et ensuite, pour $t \approx 3.5$ secondes, le premier modèle est utilisé seul, et empêche la commande d'atteindre un niveau trop important, car x atteint déjà la consigne.

Ces résultats ont été fournis à paramètre de décroissance de validité err_{max} variable. La figure 5.18 donne les évolutions de ce paramètre.

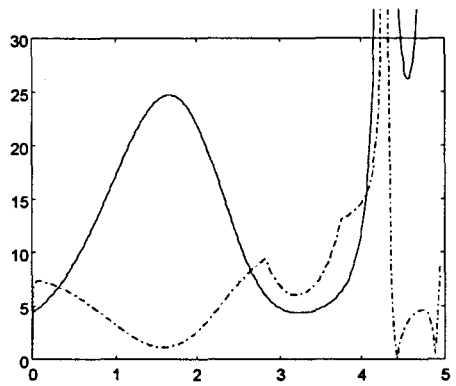

Fig. 5.18. Evolutions du paramètre $err_{max}(t)$ avec les deux stratégies d'adaptation.
'-.-.' correspond à S_1, et '—' à S_2

Le paramètre err_{max} varie entre 800% et 1800% pour S_1, basée sur le meilleur modèle, et entre 500% et 3500% pour le seconde stratégie, basée sur le pire modèle.
Si pour $t \approx 0$, err_{max} est plus faible pour S_2 que pour S_1, ce qui peut paraître paradoxal, c'est pour une raison simple: le modèle 3, le moins bon, fait relativement peu d'erreur au démarrage, alors que les modèles 1 et 2 en font à peine moins, ce qui explique qu'avec l'adaptation le mécanisme est très peu sévère avec la stratégie S_2, et qu'il doit l'être plus avec la stratégie S_1.

Lorsque err_{max} tend vers 0 avec la stratégie S_1, c'est simplement parce que les résidus du meilleur modèle tendent vers 0.

Il est possible d'examiner les validités avec une décroissance de validité constante. La figure 5.19 donne par exemple $v_1(t)$ et $v_2(t)$ avec un paramètre err_{max} égal à 500%.

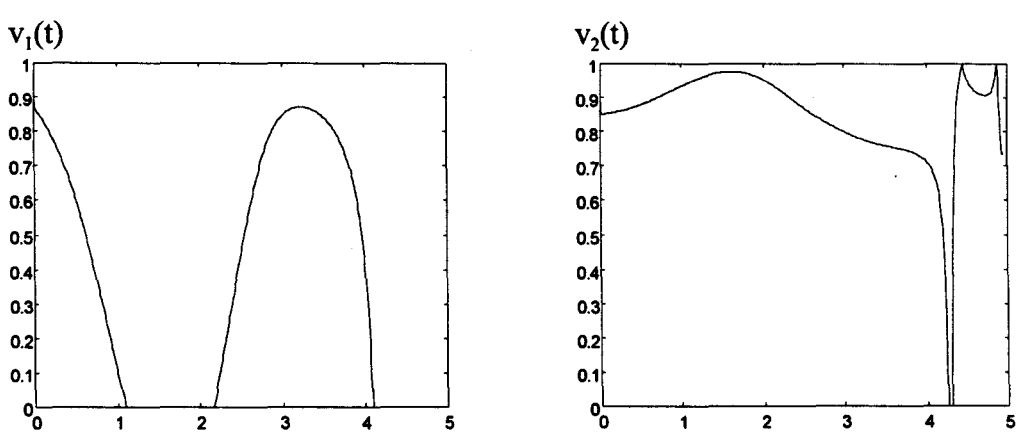

Fig. 5.19. Validités obtenues a posteriori avec un paramètre err_{max} égal à 500%, à partir des validités fournies par la stratégie d'adaptation S_2.

500% est une valeur très peu sévère, et pourtant les deux modèles M_1 et M_2 n'ont pas une validité proche de 1, sauf pour le modèle M_2, pour $t \approx 1.5$ seconde. Les validités des deux modèles tendent même vers 0 pour $t \approx 4.2$ secondes.

Comme le modèle M_2 apparaît en moyenne bien meilleur que les deux autres modèles, ces résultats montrent qu'il ne serait pas inutile de remplacer le modèle M_3 par un modèle de gain inférieur au modèle M_2, ou en tout cas, de rapport k/τ plus faible.

5.3.2 Optimisation des validités avec l'étape de renforcement

Ce paragraphe aborde l'optimisation des validités afin de diminuer l'erreur de modélisation entre le système réel et le modèle global implicite, sans changer la bibliothèque de modèles.
L'objectif de cette optimisation est de remplacer un modèle réaliste du système initial par une série de petits modèles. Sans introduire de nouveaux modèles, plusieurs paramètres peuvent être modifiés: les domaines de validité en sont un premier exemple. Cependant leurs déplacements dans l'espace Z de l'environnement ainsi que leurs déformations apparaissent comme des problèmes très compliqués. Afin d'obtenir un certain degré de liberté l'optimisation se fera au niveau des paramètres de l'étape de renforcement.
Un exemple d'application en modélisation pure sera finalement donné avec une estimation initiale des validités avec l'approche géométrique.

5.3.2.1 Formulation du problème

Une fusion linéaire est utilisée. Etant donné les coefficients de validité v_i, le résultat est donc:

$$M = \frac{v_i}{\sum v_j} M_i \tag{5.22}$$

Les M_i peuvent être par exemple des équations différentielles. Dans cet exemple, tous les modèles ont la même structure, seuls leur paramètres diffèrent (sinon 5.22 n'aurait aucun sens, les contraintes des CMM à modèle global explicite sont donc retrouvées).

Les validités avec l'étape de renforcement sont données par:

$$v_i = (1 - d_i) \prod_{j=1,\, j \neq i}^{m} \left(1 - e^{-(d_j/\sigma_{ij})^2}\right) \tag{5.23}$$

expression dans laquelle les d_i sont les distances normalisées.
Avec cette formule les σ_{ij} permettent de modifier les validités entre les modèles M_i et M_j.
Plus ils sont grands, plus le modèle M_i est perturbé par le modèle M_j, et sa validité tend vite vers 0 lorsque le modèle M_j a une validité supérieure (représentée par une distance normalisée d_j très petite). Au contraire plus σ_{ij} est petit, plus les transitions sont douces.

L'objectif est de minimiser l'erreur entre le modèle réel connu et le modèle global explicite:

$$J = \frac{1}{2}\left(f - f_{MM}\right)^2 \tag{5.24}$$

avec f_{MM} l'équation fournie par 5.22 [Dubois et al, 1996c].

Les calculs qui suivent n'utilisent que deux modèles pour alléger l'écriture, mais ils peuvent être généralisés. Ces deux modèles à validité non nulle sont notés M_i et M_j. Aussi la fonction J peut elle être réécrite sous la forme suivante:

$$J = \frac{1}{2}\left(f - \frac{v_i f_i + v_j f_j}{v_i + v_j}\right)^2 \tag{5.25}$$

Cette notation représente le carré de la somme des erreurs depuis l'instant initial.
Les dérivées partielles de J en fonction des paramètres à adapter σ_{ij} sont données par:

$$\frac{\partial J}{\partial \sigma_{ij}} = \frac{\partial J}{\partial v_{v_i}} \frac{\partial v_{v_i}}{\partial \sigma_{ij}} \tag{5.26}$$

Avec 5.22 et 5.23, les termes sont:

$$\begin{aligned} \frac{\partial J}{\partial v_{v_i}} &= -\left(f - f_{MM}\right) v_j \frac{f_i - f_j}{\left(v_i + v_j\right)^2} \\ \frac{\partial v_{v_i}}{\partial \sigma_{ij}} &= -2\left(1 - d_i\right) e^{-\left(\frac{d_j}{\sigma_{ij}}\right)^2} \frac{d_j^{\,2}}{\sigma_{ij}^{\,3}} \end{aligned} \tag{5.27}$$

Il est donc possible de converger vers de meilleurs valeures pour les σ_{ij} avec un algorithme de type gradient, avec μ un facteur d'apprentissage:

$$\sigma_{ij}(k+1) = \sigma_{ij}(k) - \mu \frac{\partial J}{\partial \sigma_{ij}} \tag{5.28}$$

5.3.2.2 Modélisation d'une série temporelle

Soit un signal discret modélisé par l'équation suivante:

$$\begin{aligned} y(k) &= \left(0.8 - 0.5e^{-y^2(k-1)}\right) y(k-1) \\ &- \left(0.3 + 0.9e^{-y^2(k-1)}\right) y(k-2) + 0.1 \sin\left(\pi y(k-1)\right) \end{aligned} \tag{5.29}$$

Cette série temporelle tend vers un cycle limite, donné à la figure 5.20:

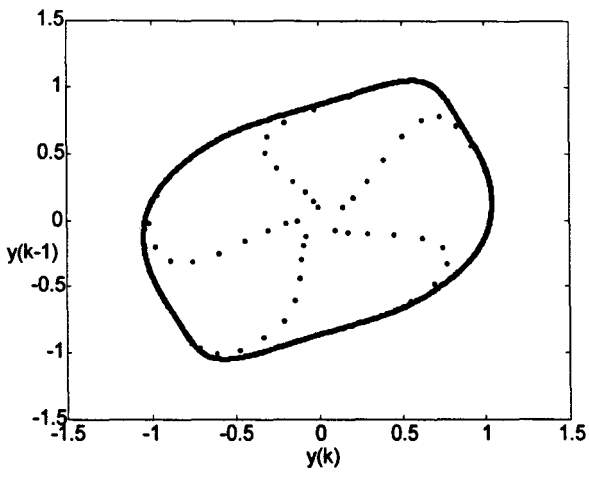

Fig. 5.20. Cycle limite de la série temporelle à modéliser. y(0)=0.1, y(1)=0.

La figure 5.21 donne de même la surface y(k) en fonction de y(k-1) et y(k-2):

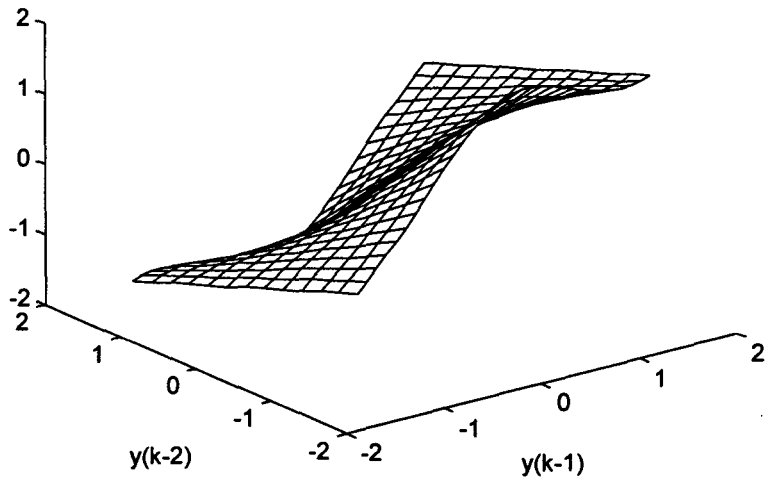

Fig. 5.21. y(k) en fonction de y(k-1) et de y(k-2).

L'objectif est de déterminer un modèle simplifié de 5.29, en supprimant le terme sin(y(k-1)). Pour cela trois modèles seront considérés suivant la valeur de ce terme.

Les trois modèles sont:

M_1:
$$\begin{aligned} y(k) &= \left(0.8 - 0.5e^{-y^2(k-1)}\right)y(k-1) \\ &\quad - \left(0.3 + 0.9e^{-y^2(k-1)}\right)y(k-2) \end{aligned} \tag{5.30}$$

M_2:
$$\begin{aligned} y(k) &= \left(0.8 - 0.5e^{-y^2(k-1)}\right)y(k-1) \\ &\quad - \left(0.3 + 0.9e^{-y^2(k-1)}\right)y(k-2) + 0.1 \end{aligned} \tag{5.31}$$

M_3:
$$\begin{aligned} y(k) &= \left(0.8 - 0.5e^{-y^2(k-1)}\right)y(k-1) \\ &\quad - \left(0.3 + 0.9e^{-y^2(k-1)}\right)y(k-2) - 0.1 \end{aligned} \tag{5.32}$$

Ces modèles sont donc encore non linéaires.
Leurs domaines de validité sont:

$$\begin{aligned} D_1 &= \{\pi y(k-1)[\pi] = 0\} \\ D_2 &= \{\pi y(k-1)[2\pi] = \pi/2\} \\ D_3 &= \{\pi y(k-1)[2\pi] = -\pi/2\} \end{aligned} \tag{5.33}$$

Leurs domaines de non validité sont:

$$\begin{aligned} \neg D_1 &= \varnothing \\ \neg D_2 &= \{\pi y(k-1)[2\pi] \in [-\pi, 0[\} \\ \neg D_3 &= \{\pi y(k-1)[2\pi] \in [\pi, 0]\} \end{aligned} \tag{5.34}$$

Avant de donner les résultats de la modélisation multi-modèle, il est intéressant de donner les cycles limites ainsi que les surfaces y(k) pour ces trois modèles.

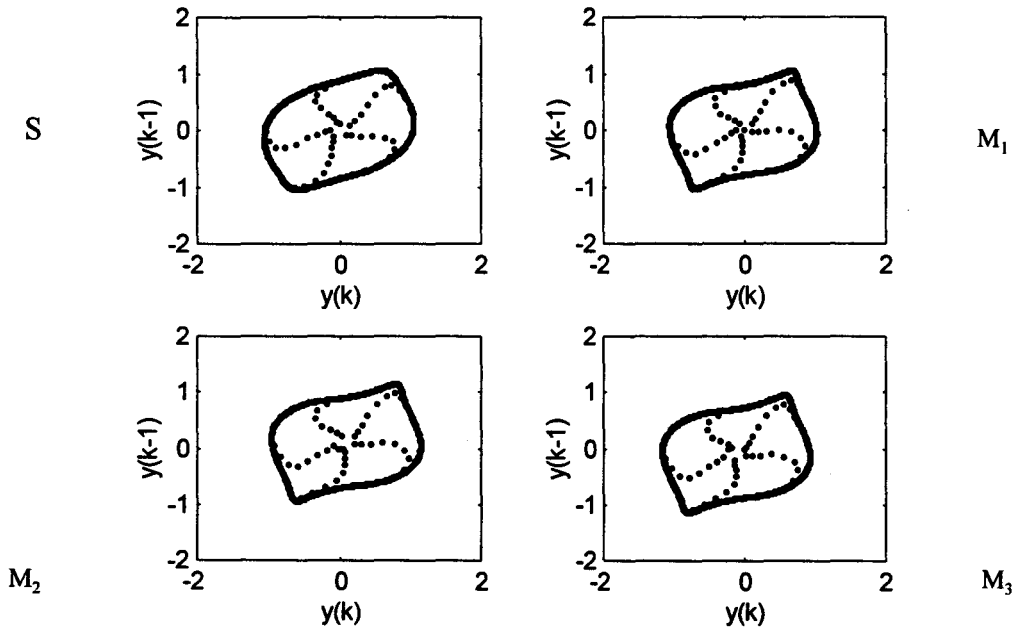

Fig. 5.22. Cycles limites des trois modèles comparés à celui de la série.

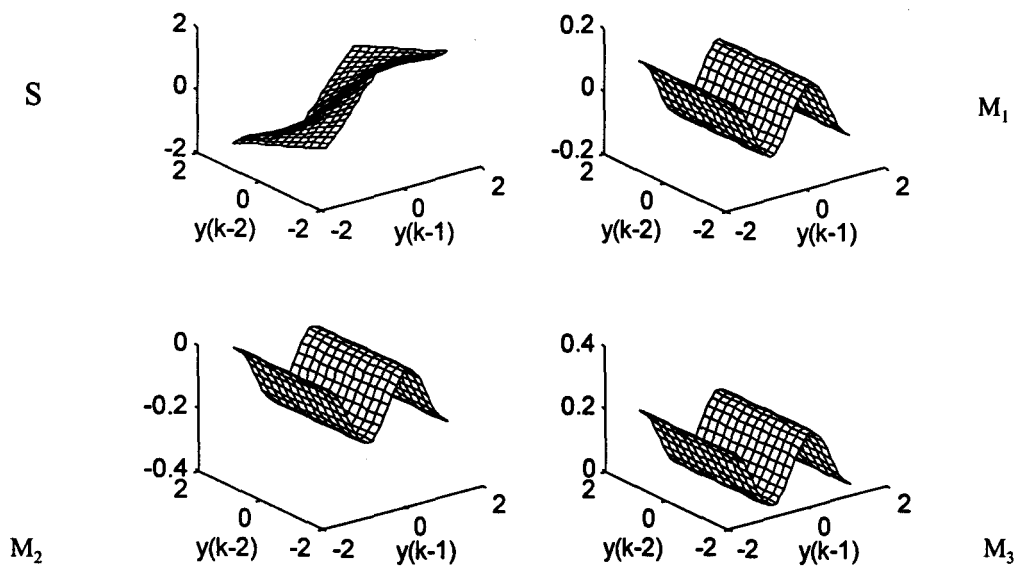

Fig. 5.23. Surfaces y(k) pour les trois modèles, comparés à celle de la série.

Il apparaît donc que le terme variable engendré par le sinus joue un rôle important, puisque les surfaces y(k) en fonction de y(k-1) et y(k-2) sont très différentes entre les modèles et la série originale.

La première simulation est effectuée sans adaptation des paramètres σ_{ij}. La figure 5.24 donne le cycle obtenu, et la figure 5.24 l'erreur entre les deux surfaces.

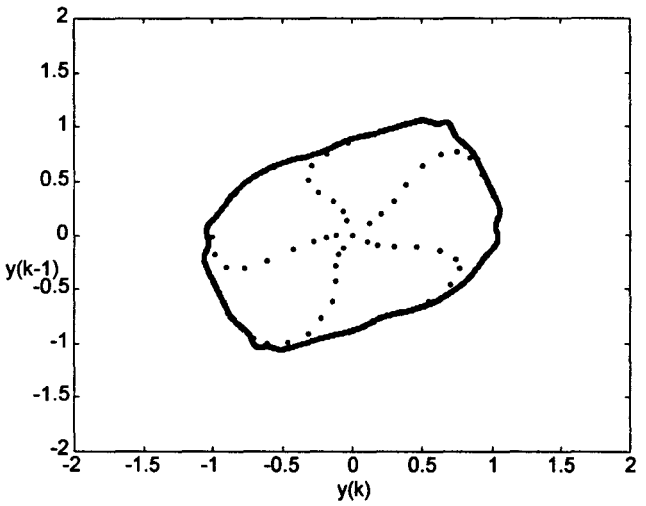

Fig. 5.24. Cycle limite avec le multi-modèle non adapté.

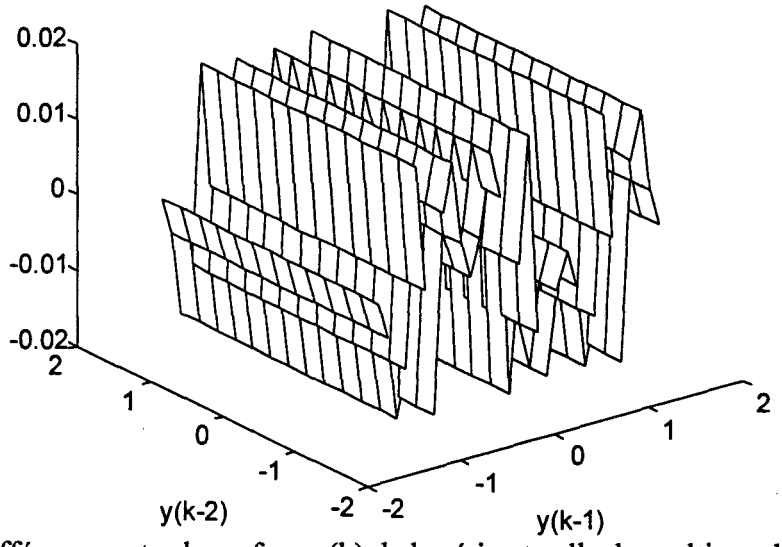

Fig. 5.25. Différence entre la surface y(k) de la série et celle du multi-modèle non adapté (σ_{ij}=0.5).

Par rapport aux modèles pris indépendamment, la modélisation multi-modèle est beaucoup plus proche de celle de la série, puisque les erreurs sont plutôt faibles.

Cette qualité va être améliorée avec une adaptation des coefficients σ_{ij} choisis au hasard. Les figures 5.26 et 5.27 montrent respectivement le cycle limite et la surface d'erreur pour le modèle obtenu avec adaptation.

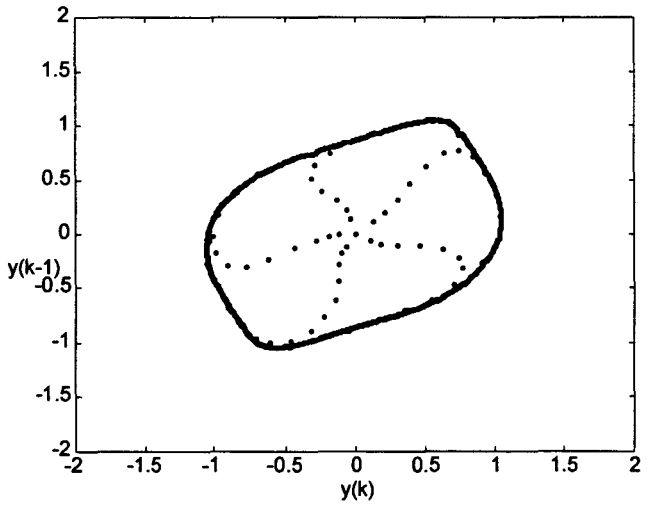

Fig. 5.26. Cycle limite obtenu avec l'adaptation des paramètres du renforcement.

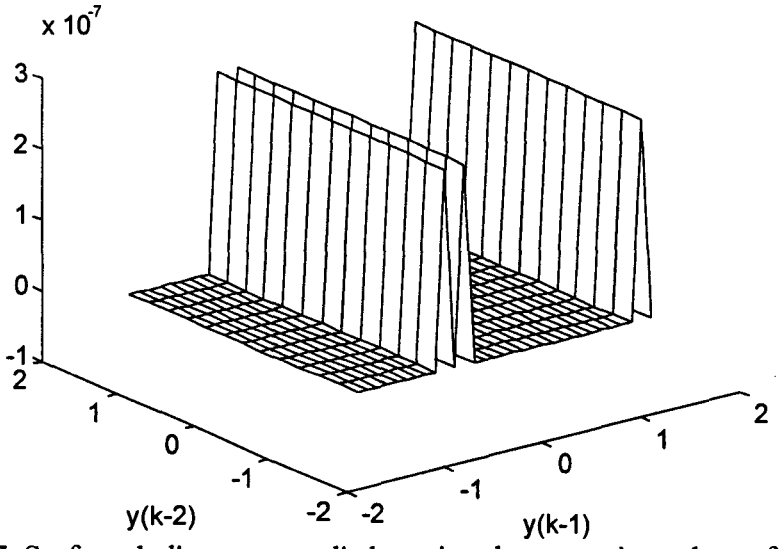

Fig. 5.27. Surface de l'erreur avec l'adaptation des paramètres du renforcement.

L'erreur est considérablement diminuée. La figure 5.28 donne les évolutions des paramètres σ_{ij}.

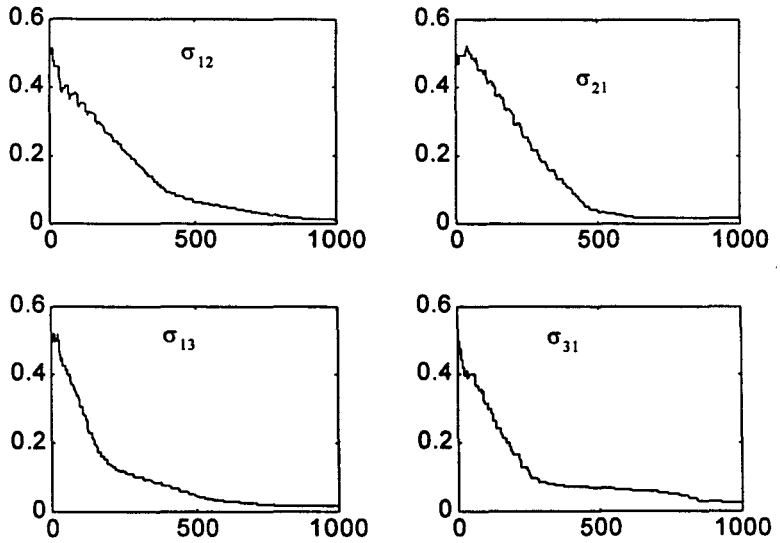

Fig. 5.28. Evolutions des paramètres σ_{ij}: σ_{12}, σ_{21}, σ_{13} et σ_{14}.

Il faut remarquer que les coefficients deviennent tous petits, ce qui augmente les transitions douces entre les modèles. Par contre les coefficients ne sont pas symétriques lorsque le multi-modèle passe du modèle M_i au modèle M_j et inversement. Par exemple, la transition est plus douce lorsque le multi-modèle passe du modèle M_1 au modèle M_3, et plus brusque du modèle M_3 au modèle M_1. Aucune explication n'a été trouvée à ce jour.

Une critique qui peut être faite à l'encontre de cette adaptation est qu'elle utilise l'algorithme du gradient, qui ne permet pas d'affirmer que les optimums globaux aient été trouvés. En effet la fonction 5.24 est peut être multi-modale en fonction des σ_{ij}. Pour en obtenir les meilleurs valeurs, il faudrait recourir à un algorithme adapté à ce type de problème, comme par exemple les algorithmes génétiques, mais l'optimisation ne pourrait se faire qu'a posteriori, en essayant des valeurs différentes pour les paramètres σ_{ij}.

5.3.4 Complexité des représentations multi-modèles

Yager et Filev ont introduit en 1993 [Yager et Filev, 1993] la notion de complexité des approches multi-modèles. Pour ces derniers, la complexité de ces dernières se réfèrent à la complexité de l'étape de la fusion. Cette complexité est donc liée aussi à l'estimation des validités. En effet si à chaque instant les validités sont telles qu'un seul modèle est considéré parfait, et tous les autres absolument mauvais, l'étape de fusion dégénère en une commutation.

Pour cela les degrés de validités sont interprétés comme des distributions de probabilités des modèles dans l'espace de l'environnement. Elles sont en effet données par:

$$v_i(z) = d_i(z) \Big/ \sum_{j=1}^{m} d_j(z) \tag{5.35}$$

et

$$d_i(z) = \left\| z - \beta_i \right\|^{-\alpha} \tag{5.36}$$

(β_i étant les domaines de validités réduits à des singletons). Le modèle global est donné par:

$$f = \sum_{i=1}^{m} v_i f_i \tag{5.37}$$

Cette approche est donc une structure à modèle global explicite, avec une fusion linéaire, des validités obtenues par l'approche géometrique et une normalisation relative.

Ils peuvent alors calculer l'entropie associée à l'ensemble des validités.

$$H(z) = -\sum_{i=1}^{m} v_i(z) \ln(v_i(z)) \tag{5.38}$$

Cette entropie est alors appelée complexité de la représentation multi-modèle.

Les auteurs montrèrent que la complexité d'un multi-modèle est minimale lorsqu'un seul modèle est utilisé, ce qui n'est pas surprenant, mais de même lorsque l'étape de fusion est une commutation entre les meilleurs modèles (comme avec l'école de Yale). La complexité est maximale lorsque tous les modèles ont une validité identique à 1/m. Ils utilisent la notion de complexité pour adapter le paramètre α qui intervient dans la définition de la distance utilisée par leur approche multi-modèle, afin d'en diminuer la complexité. Ils utilisent pour cela une

méthode du gradient, pour minimiser l'erreur de modélisation sur les données entrées-sorties et la complexité de la représentation obtenue..

Ils montrent que α doit tendre vers 0, ce qui revient à tendre vers une fusion de type commutation (mais pas complètement). En effet avec une approche de type commutation, comme avec l'école de Yale, l'interpolation n'existe plus, et les erreurs de modèlisation augmentent énormément.

5.4 Perspectives

Ce paragraphe donne deux perspectives de recherche qui n'ont pas encore été détaillées dans les paragraphes précédents. La première concerne la gestion des très grandes bibliothèques, et la seconde l'utilisation des multi-modèles en surveillance.

5.4.1 Très grande bibliothèque

Il est envisageable, pour de petits problèmes, de concevoir une bibliothèque avec un certain nombre de modèles, par exemple une dizaine, ou un peu plus.

Au fur et à mesure que le temps passera, pour améliorer les performances, ou pour travailler sur des systèmes de plus en plus complexes, les bibliothèques augmenteront en taille.

Pour un environnement donné, il est impensable que tous les modèles et toutes les lois de commandes soient de validités égales. Il y aura beaucoup de modèles très valides, et beaucoup qui ne seront pas valides du tout.

Ce serait alors une grande perte de temps de calculer toutes les lois de commandes élémentaires pour tous les modèles et toutes les lois de commandes.

Une solution serait alors de trier les modèles pour n'utiliser que ceux apparaissant comme les meilleurs au cours des précédents pas de calcul.
Si le système est par exemple représenté par 50 bons modèles, et qu'il y en ait 950 d'autres moyens ou mauvais, beaucoup de temps serait gagné si le multi-modèle ne faisait ses calculs que sur les 50 meilleurs modèles.

Pour de tels problèmes, au schéma classique de multi-modèle sera ajouté un mécanisme, appelé superviseur, qui vérifiera que les validités de la bibliothèque courante, petite, restent correctes. Le rôle du superviseur serait alors de mettre à jour la bibliothèque courante en piochant dans la mémoire morte les modèles complémentaires.

Ainsi il serait tout à fait possible de mettre au point des bibliothèques de modèles de très grande taille pour couvrir tous les cas possibles.
La figure 5.29 donne un exemple de modélisation avec un très grand nombre de modèles.

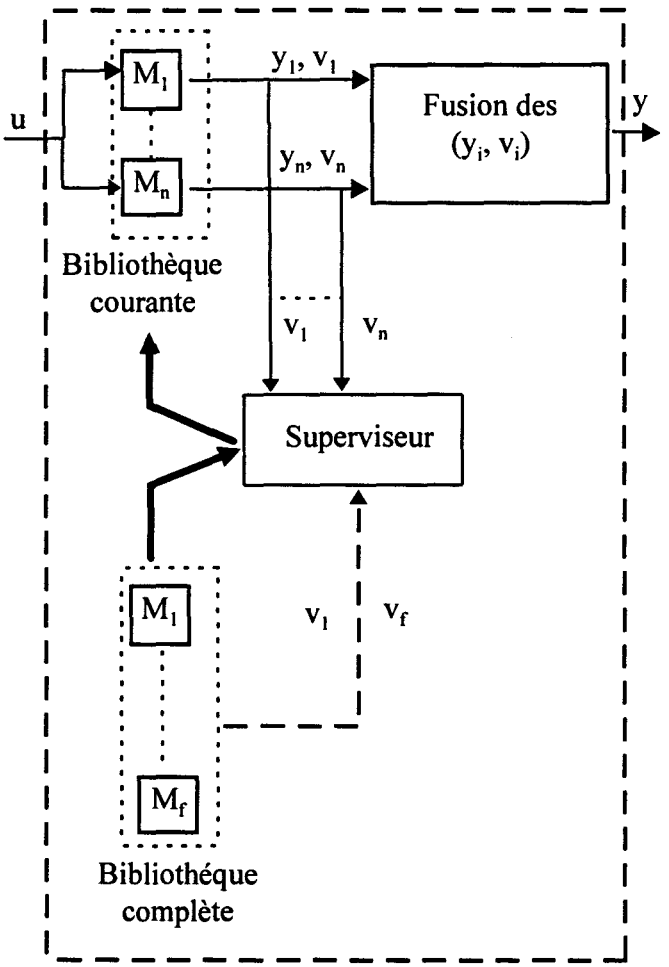

Fig. 5.29. Exemple de modélisation avec une très grande bibliothèque.

A chaque instant la modélisation n'est effectuée que sur un nombre restreint de modèles. Leurs validités sont contrôlées à chaque instant. Si ces validités diminuent trop, ce qui sera vérifié par un superviseur, alors ce dernier cherchera dans l'ensemble de la bibliothèque de meilleurs modèles. Les flèches de validités en pointillés, pour v_1 à v_f, indiquent que ce balayage ne sera pas fait à chaque instant.

Bien que ce schéma soit très simple, les mécanismes gouvernants la recherche de bons modèles pour rafraîchir la bibliothèque courante sont complexes, tout au moins s'ils sont souhaités optimisés, pour réduire les temps de calcul.

Il est à remarquer que l'arrivée des contrôleurs multi-modèles avec une très grande bibliothèque est imminente. En effet dans un récent article [Narendra et al, 1997], un contrôleur utilisant jusqu'à 512 modèles fut présenté (dans le cas de modèles fixes, et "seulement" 125 dans le cas de modèles tous adaptatifs).

5.4.2 Application en surveillance

Au chapitre 4 furent présentés les principales méthodes d'estimation des validités des modèles. Il y avait deux grandes familles. D'une part il y avait celle qui incluait les méthodes nécessitant une connaissance a priori, et qui regroupait les approches floues, probabilistes, et géométriques. D'autre part il y avait celle qui était représentée en fait seulement par l'approche

des résidus. Cette approche des résidus apparait très intéressante, car elle peut être appliquée sans connaissance a priori. Par contre, étant une approche purement réactive, elle ne permet pas des optimisations sur les validités, pour privilégier tel ou tel modèle dans telle ou telle perspective. Elle pourrait servir dans un premier temps pour fournir une information a priori sur les validités, de telle sorte qu'il soit possible d'utiliser une des premières méthodes.

Une autre utilisation de l'approche des résidus est de contrôler la vraisemblance des validités estimées a priori. En effet les systèmes sont toujours susceptibles d'évoluer de manière imprévue, et ce n'est jamais au le bénéfice de l'utilisateur.

L'objet de la surveillance est ainsi de vérifier si les modèles retenus correspondent bien au système, si les capteurs tombent en panne ou si les modèles sont défectueux dans telle plage de fonctionnement, ou encore si les paramètres du système évoluent réellement.

La figure 5.30 propose en schéma permettant de vérifier la validité des informations fournies a priori pour construire les validités.

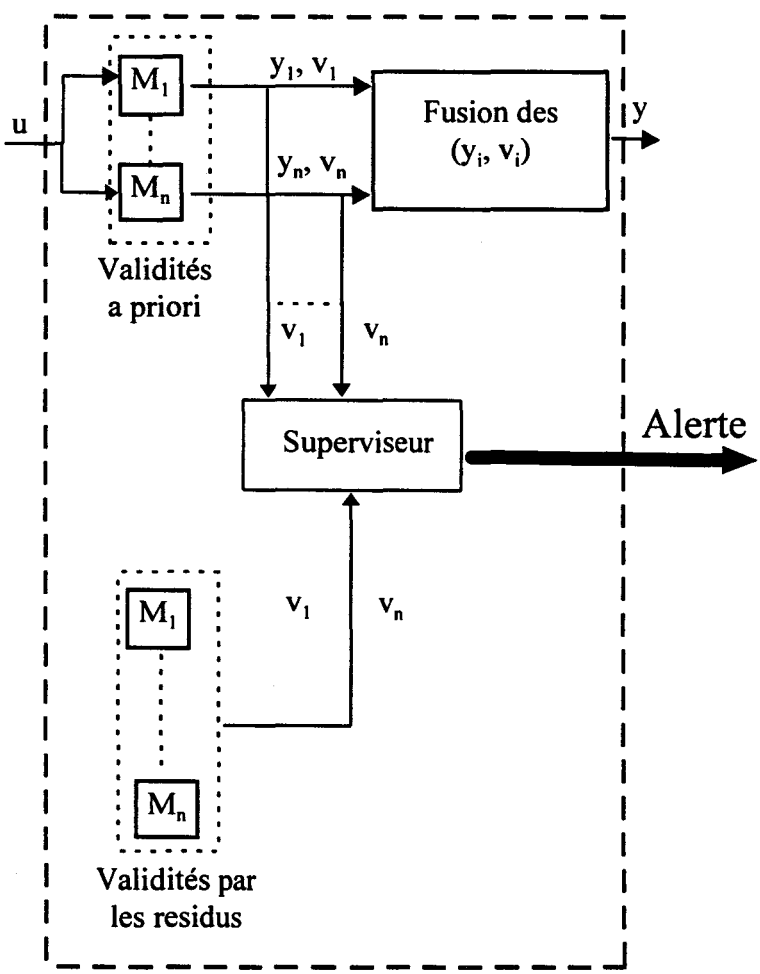

Fig. 5.30. Comparaisons des validités estimées a priori avec les validités calculées par l'approche des résidus.

Dans ce schéma, l'expression "validités estimées a priori" veut simplement dire que les validités sont estimées à chaque instant grâce à la mesure de l'environnement, mais que les équations qui fournissent ces validités ont été établies préalablement.

Pour l'instant la seule sortie du superviseur qui contrôle les données est un signal d'alerte. Il faudrait d'abord corréler les évolutions des validités par les approches a priori avec les évolutions des approches par les résidus, par exemple pour éviter les variations dues simplement à la normalisation relative avec l'approche géométrique, variations qui ne sont pas dues aux modèles.

Une étape ultérieure permettrait au superviseur d'utiliser les validités obtenues par les résidus au cas où les différences seraient trop importantes.

Il existe de fortes similitudes entre les outils de la théorie de la surveillance, dont l'objet est entre autres de vérifier que les modèles utilisés sont corrects on non, et ceux des approches multi-modèles qui se doivent de quantifier la validité des modèles utilisés. Ainsi en surveillance, un problème est d'isoler, c'est à dire de reconnaître, les capteurs ou les paramètres physiques qui provoquent des perturbations. Même si les finalités des deux théories diffèrent, il semble qu'elles pourraient bénéficier de leurs avancées réciproques

5.5 Conclusion

Ce chapitre a permis d'approfondir la présentation des contrôleurs multi-modèles dont les bases avaient été données au précèdent chapitre.
La première partie a présenté une généralisation intéressante de l'idée de multi-modèle à l'idée de multi-commandes, qui est le prolongement de la commande multi-critère. Cependant le formalisme du contrôleur multi-modèle avec multi-commande permet de représenter clairement, puis, par la suite, d'étudier et d'optimiser les commandes associées aux modèles, ainsi que leur liens, et leur complémentarités. Il est par exemple possible d'associer une commande robuste à un modèle global, une commande optimisée avec un modèle précis, en un point de fonctionnement important, ou même d'associer à un même modèle deux lois de commandes, qui joueront un rôle dans telle ou telle zone suivant les choix de l'utilisateur.

La deuxième partie présente deux mécanismes qui tendent à donner plus d'autonomie au multi-modèle, et à corriger les erreurs ou à compléter les informations fournies par le concepteur. Par exemple l'adaptation de la vitesse de décroissance de validité dans l'approche des résidus permet de corriger les estimations de validités de telle sorte que les meilleurs (ou moins mauvais selon les cas) modèles soient toujours pris en compte par rapport aux mauvais.

La troisième partie présente quelques axes de recherche actuels, notamment l'optimisation de l'utilisation d'un très grand nombre de modèles et de commandes.

Les outils présentés permettent dès maintenant l'analyse de systèmes complexes réels. Cependant les mécanismes présentés jusqu'ici forment un ensemble relativement statique, en cela que, mis à part l'approche des résidus, toutes les connaissances utilisées pour la conception doivent être fournies à l'avance. Même l'approche des résidus ne pourra rien faire, à moins de disposer d'une commande "génialement robuste", si le meilleur des modèles restant devient, le système ayant évolué de façon imprévue, très mauvais. L'objet du chapitre 6 apparaît donc naturel: identifier en ligne de nouveaux modèles, ce qui permettra d'enrichir la base des modèles.

CHAPITRE SIXIEME

BIBLIOTHEQUE DYNAMIQUE DE MODELES

La carte n'est pas le territoire

A. Korzybski, Science and Sanity

6.1 Pourquoi une bibliothèque de modèles?

Dans l'automatique coexistent deux grandes manières de contrôler les systèmes.
Avec la première approche, un bon modèle du système étudié est disponible, et il est supposé en outre que le système n'évoluera pas. Aussi il est possible de faire des calculs afin d'optimiser la commande dans telle ou telle perspective. Si le système n'a pas été bien modélisé, il est possible de recourir à des commandes robustes, mais ces dernières sont en général abandonnées au profit des méthodes optimales lorsqu'un souci de rentabilité apparait, si, bien entendu, une amélioration de la modélisation est possible.
Au contraire, avec la seconde approche, très peu d'informations sur le système sont nécessaires, et de fait les commandes sont contraintes de s'adapter au système par des algorithmes de réglage en ligne. Le terme de commande adaptative est alors employé pour souligner que les algorithmes de commandes vont littéralement s'adapter pour suivre les évolutions du système. En général ces méthodes sont utilisées lorsque le système est susceptible d'évoluer. Un de leurs avantages est qu'elles peuvent, en général, être utilisées telles quelles avec un minimum de recherche sur des systèmes variés. Par contre elles sont moins performantes que les méthodes conçues avec un plus grand volume d'information.

Les approches multi-modèles peuvent être perçues comme étant à la charnière de ces deux grandes approches. En effet, d'une part la définition d'un contrôleur multi-modèle requiert beaucoup d'informations, par exemple pour la construction des modèles, d'autre part si plusieurs modèles sont utilisés, c'est souvent parce que le système original est complexe, et qu'il est alors perçu comme évoluant parmi toute une gamme de comportements possibles, chacun représenté par un modèle particulier. Le fait que le système évolue entre différents états est donc une propriété attendue avec l'utilisation d'un contrôleur multi-modèle.

Cependant un contrôleur multi-modèle ne peut pas anticiper les évolutions *imprévues* du système. Un contrôleur multi-modèle n'est que la traduction d'une connaissance a priori, qui ne présage en rien des évolutions futures, ou mal calculées, du système. S'il est vrai que l'approche des résidus développée pour l'estimation des validités permet une certaine adaptation par le choix du moins mauvais modèle dans la bibliothèque, la commande finale ne permettra pas d'atteindre les objectifs souhaités si le moins mauvais des modèles n'est pas représentatif du système.

Aussi s'est imposée l'idée qu'il fallait pouvoir traiter de telles déviations. L'architecture très souple du contrôleur multi-modèle permet d'introduire facilement des algorithmes d'adaptation. Ce chapitre a pour objet la définition d'un contrôleur multi-modèle qui pourrait s'adapter aux évolutions quelconques du système.
En automatique classique, deux types de méthodes existent. Il y a d'une part la commande adaptative indirecte, qui suppose qu'un modèle soit d'abord identifié en ligne, et sur lequel s'applique ensuite une commande définie a priori; et d'autre part la commande adaptative directe, qui ne présuppose pas une identification de modèle, mais intervient directement au niveau des paramètres de la commande.

Les deux approches peuvent naturellement être utilisées avec un contrôleur multi-modèle, que ce soit au niveau de la librairie de modèles ou de la librairie de commandes. Pour l'instant seule la première approche a été étudiée.

Le terme de bibliothèque dynamique a été utilisé dans le titre de ce chapitre. Cette expression appelle un commentaire. Dans les méthodes de commande adaptative indirecte, les paramètres varient sans cesse, mis à part les éventuels états stables. Ces méthodes ne considèrent qu'un seul modèle à la fois, qui évolue. C'est pourquoi, au niveau des modèles, ces méthodes sont sans mémoire, même si elles incorporent toutes les données antérieures. Si un système oscille entre deux états (par état il faut entendre ici la structure du système, ainsi que les valeurs des paramètres), une méthode adaptative passera son temps à osciller entre les deux, mais avec une perte de performance due à la continuelle adaptation des paramètres lors des transitions.

Or un contrôleur multi-modèle utilise des modèles bien distincts les uns des autres. C'est en quelque sorte sa mémoire statique. Il pourrait être tout à fait possible d'adjoindre à la bibliothèque un modèle identifié en ligne, dont les paramètres évoluent continûment, comme avec l'école de Yale. Cependant il serait beaucoup plus intéressant de relever les modèles identifiés performants qui pourraient être ajoutés à une mémoire dynamique, qui contiendrait des emplacements mémoires suffisants pour stocker quelques modèles imprévus qui seraient souvent utilisés. Ainsi pour le système qui évolue entre les deux états, après un cycle complet, deux modèles seraient ajoutés à la bibliothèque, et le contrôleur réagirait beaucoup plus vite qu'avec une simple commande adaptative qui demanderait à chaque passe un temps d'adaptation.

Les pages qui suivent vont détailler les mécanismes présentés dans cette introduction. Cette partie aurait très bien pu être incluse dans le chapitre précèdent qui traitait des améliorations de l'architecture élémentaire d'un contrôleur multi-modèle, cependant, par le volume des recherches mises en jeu, et par l'ampleur des perspectives offertes la présentation de cette thématique justifie un chapitre complet.

L'identification de nouveaux modèles avait été prévue dès l'origine de cette étude. Cependant l'identification est vite apparue comme un champ très complexe. Il serait illusoire et prétentieux de présenter une architecture de contrôleur avec identification qui se dirait universelle et autonome. Les résultats présentés ci après ne sont qu'une tentative pour résoudre certains problèmes simples dus aux variations accidentelles d'un système.

Le système utilisé à titre d'exemple depuis le chapitre 4, le premier ordre dont les paramètres sont des fonctions de l'état, s'est révélé particulièrement dur à l'épreuve. Les méthodes classiques d'identification ne semblent pas pouvoir l'aborder. Aussi dans une première partie sera présentée une méthode peu fréquente d'identification, la seconde partie donnant les résultats de simulations.

6.2 Structure d'un contrôleur avec un seul modèle identifié

Ce paragraphe va présenter l'architecture d'un contrôleur qui complète sa bibliothèque statique par un modèle dont les paramètres varient.

Les figures 6.1 et 6.2 donnent deux schémas de contrôleur utilisant un seul modèle identifié.

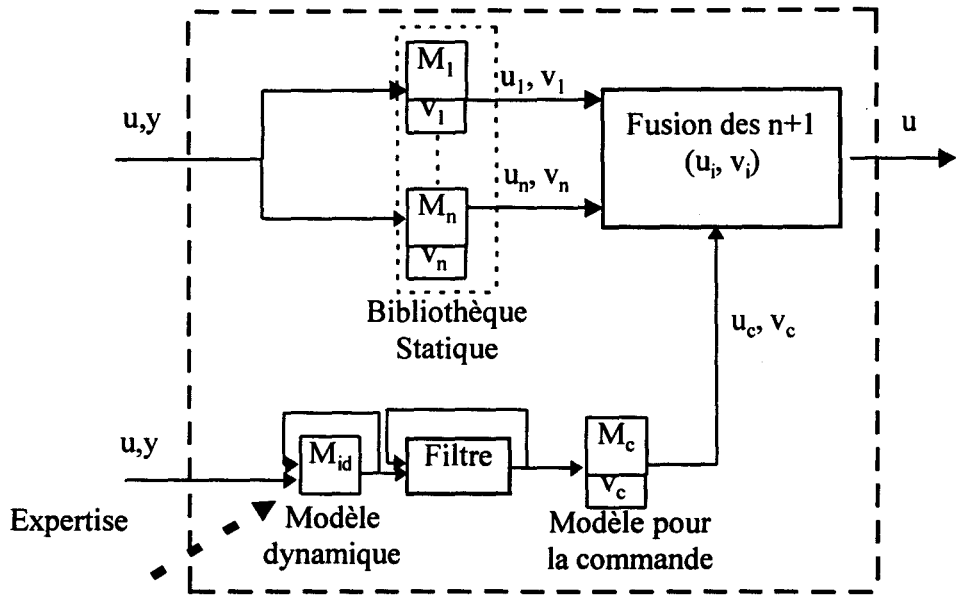

Fig. 6.1. Schéma de contrôleur avec un modèle identifié et un filtre sur ce dernier.

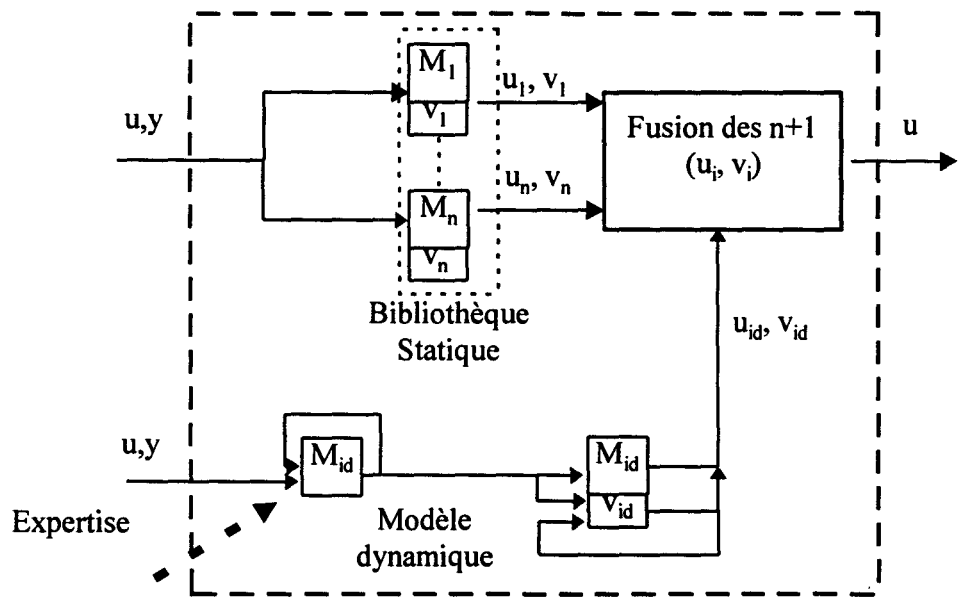

Fig. 6.2. Schéma de contrôleur avec un modèle identifié et superviseur sur la validité.

Le premier apport de ces architectures par rapport aux approches multi-modèles classiques est le modèle identifié M_{id}. Il a fallu pour cela introduire des algorithmes d'identification. Leurs données sont les entrées u(k) et les sorties y(k). M_{id} est donc un modèle discret. Parmi les différents algorithmes possibles, il fallait choisir un algorithme adaptatif avec une formulation récurrente. Puisque l'objectif principal des approches multi-modèles est d'utiliser des modèles simples, la structure du modèle M_{id} est choisie linéaire en les entrées et en les paramètres.

Les algorithmes utilisés sont donc du type "moindres carrés". Il est possible de trouver plus de détails dans un grand nombre d'ouvrage [Najim, 1988; Borne et al, 1992; Landau, 1988; Walter et Pronzato, 1994, pour de nombreux détails; Sudarno, 1996 pour la méthode avec test sur l'excitation permanente]. L'annexe B rappelle les algorithmes de base.

La différence entre les deux architectures provient de l'utilisation du modèle identifié. En effet, numériquement les paramètres du modèle identifié peuvent changer brusquement lors de la phase d'identification, lorsque le système passe d'un état à un autre, et par conséquent lorsque le modèle identifié est utilisé tel quel, la commande risque de diverger à un moment ou à un autre.

Aussi avant d'utiliser le modèle identifié, il faut d'abord s'assurer qu'il ait une fiabilité minimale. C'est l'objet de la différence entre les deux architectures.
Avec la première, un filtre fournit pour la commande un modèle dynamique (au sens des paramètres) dont les paramètres évoluent plus lentement, et plus doucement aussi. Ce modèle peut alors être utilisé tel quel, sa validité étant estimée par exemple par l'approche des résidus.

Si cette approche est la seule possible avec les méthodes de commande adaptative classique, cela n'est plus le cas avec un contrôleur multi-modèle. En effet comme plusieurs autres modèles sont disponibles, la commande finale peut dépendre temporairement uniquement de ces derniers, et ne prendre en compte le modèle identifié que lorsque celui -ci s'est stabilisé. Cette fonction est réalisée par un superviseur au niveau de la validité du modèle identifié. Si celui -ci n'est pas invariant localement, sa validité est mise à 0. Cette approche est celle representée à la figure 2.

L'annexe B détaille par exemple l'approche utilisant les filtres. Cependant avec les deux approches, celle du filtre, et celle du superviseur, de nombreux paramètres apparaissent, et leur réglage est pour l'instant de nature heuristique. Aussi les résultats présentés par la suite utilisent directement le modèle identifié M_{id}.

6.3 Identification de systèmes non linéaires

Ce paragraphe va aborder le traitement de systèmes complexes qui ne sont pas linéaires. La méthode classique des moindres carrés ne fonctionne pas toujours pour de tels systèmes. En effet les systèmes non linéaires peuvent être interprétés comme des systèmes linéaires à paramètres variables. Les méthodes d'identification simples présentées en annexe sont suffisantes pour les systèmes relativement simples, car elles permettent une certaine adaptation des paramètres des modèles. Cependant le système du chapitre 4 a "résisté" à toutes les simulations avec les trois méthodes d'identification, en adaptant par exemple le coefficient d'oubli ou la taille de la fenêtre.

Aussi il a fallu recourir à des méthodes d'identification plus élaborées. Après avoir donné un aperçu des principales approches [Grenier, 1984], le choix final de la méthode d'identification sera finalement présenté.

Il faut insister sur la fait que l'architecture de contrôleur multi-modèle avec une étape d'identification n'est pas tributaire du choix d'une méthode particulière. La méthode

finalement retenue ne l'a été que parce que les autres approches ne fonctionnaient pas. Mais d'autres approches, plus performantes, peuvent évidemment être utilisées.

6.3.1 Modèles pour l'identification

Les données sont les suivantes. Il s'agit d'évaluer les paramètres du modèle représenté par l'équation:

$$\hat{y}_{n+k} = -a_1 y_{n+k-1} - \ldots - a_n y_k + b_1 u_{n+k-1} + \ldots + b_n u_k \tag{6.1}$$

avec n l'ordre du modèle, et $k \geq 1$.
y_{k+i} et u_{k+j} sont les informations disponibles à l'instant n+k.
L'objectif est de trouver les valeurs des coefficients supposés constants a_i et b_j.

Le problème peut se mettre sous la forme matricielle:

$$y_{n+k} = H_{n+k}\Theta + e_{n+k} \tag{6.2}$$

avec

$$H_{n+k} = [-y_{n+k-1} \ldots -y_k \ldots u_{n+k-1} \ldots u_k] \tag{6.3}$$

$$\Theta = [a_1 \ldots a_n \; b_1 \ldots b_n]^T$$

e_{n+k} représentant l'erreur entre le modèle et le système réel

6.3.2 Méthodes adaptatives

Ces méthodes sont le plus souvent utilisées, et représentent la majorité des travaux.
Leur principe est simple: une méthode générale est décrite par un algorithme rendu récursif, puis un oubli est introduit dans l'algorithme pour permettre à la méthode de suivre les évolutions du système.
Un archétype de ces méthodes adaptatives est l'algorithme des moindres carrés avec facteur d'oubli. Les moindres carrés avec fenêtre glissante en sont un autre exemple classique.
Ces méthodes sont très souples, et requièrent peu d'information, à part l'ordre des systèmes. Par contre elles nécessitent l'estimation du ou des paramètres qui permettent l'adaptation, par exemple le facteur d'oubli dans les moindres carrés à facteur d'oubli.
Ce paramètre d'oubli conditionne directement l'apprentissage des dynamiques d'évolution des paramètres des systèmes. Si l'oubli est faible, les méthodes adaptatives ne pourront suivre que des non stationnarités évoluant lentement, s'il est important, les dynamiques pourront évoluer plus vite, mais les modèles seront très sensibles, et les algorithmes seront moins stables.

6.3.3 Méthodes de segmentation

Ces méthodes supposent que le système évolue par sauts entre des états stationnaires. Elles se décomposent donc en deux étapes. Une première phase se charge de détecter si le système est stationnaire ou non, et la seconde phase lance une méthode d'identification sur le système stationnaire.
Pour une modélisation précise, la phase de détection de régime stationnaire est obligatoire, mais elle ne l'est pas pour une modélisation approximative, les instants de transitions peuvent alors être calculés de manière mécanique de sorte à approcher les évolutions du système par une série de modèles constants proches et régulièrement espacés.

Dans ce cas l'algorithme d'identification est réinitialisé à chaque transition a priori, et l'identification elle même se fait uniquement à partir des nouvelles mesures.

6.3.4 Modèles à coefficients aléatoires

Ces méthodes, et celles exposées au paragraphe suivant, sont les seules à supposer que les coefficients des modèles peuvent évoluer et qu'il faille représenter ces évolutions de manière complète.
Dans le cas présent les coefficients du système doivent être modélisés par une équation, et le problème devient complexe, car il faut à la fois identifier les paramètres du modèle, et son état. 6.8 donne un exemple de modélisation simple du problème

$$\begin{cases} y(k+1) = \Theta(k+1)H(k) + \text{bruit} \\ \Theta(k+1) = A\Theta(k) + \text{bruit}' \end{cases} \tag{6.8}$$

Dans cet exemple il faut estimer en même temps Θ et A, ce qui est un problème non linéaire. Le filtre de Kalman étendu pourrait être utilisé. Ces méthodes non pas été testées, mais apparaissent complexes.

6.3.5 Modèles évolutifs

Ces méthodes proposent de projeter les paramètres variables sur une base de fonctions définie a priori. Si la base est bien choisie, la projection donne une série de paramètres à identifier, mais ils ont la remarquable propriété d'être maintenant constants. Le nombre de nouveaux paramètres est bien sûr beaucoup plus important, mais le fait qu'ils soient constants compense largement l'inconvénient du nombre.

La base de fonctions est supposée connue a priori, et les coefficients Θ_i (Θ_i représentant indifféremment a_i ou b_j) s'expriment alors avec:

$$\Theta_i(k) = \sum_{j=1}^{m} \Theta_{ij}(k) f_j(k) \tag{6.9}$$

avec f_j une des fonctions de la base qui en comprend m.

6.9 injectée dans 6.1 donne finalement:

$$\begin{aligned} \hat{y}_{n+k} = & -a_{11}y_{n+k-1}f_1(k) - \ldots - a_{1m}y_{n+k-1}f_m(k) - \ldots - a_{n1}y_k f_1(k) - \ldots - a_{nm}y_k f_m(k) \\ & + b_{01}u_{n+k}f_1(k) + \ldots + b_{0m}u_{n+k}f_m(k) + \ldots + b_{n1}u_k f_1(k) + \ldots + b_{nm}u_k f_m(k) \end{aligned} \tag{6.10}$$

qui est un modèle stationnaire d'ordre n×m sur le signal étendu Y défini par:

$$Y(n+k) = [y_{n+k-1}f_1(k) \ldots y_{n+k-1}f_m(k) \ldots\ldots y_k f_1(k) \ldots y_k f_m(k)] \tag{6.11}$$

Un modèle non stationnaire d'ordre n est donc représenté par un modèle stationnaire, mais d'ordre n×m. En plus de l'augmentation de l'ordre, ces méthodes imposent de devoir choisir une base de fonctions.

Il est finalement possible de recourir à une méthode d'identification classique sur le signal étendu, avec un nombre de paramètres égal à n×m. L'annexe B donne plus de détails sur cette approche peu commune en modélisation.

6.3.6 Choix de la méthode d'identification

Les méthodes adaptatives ne fonctionnent pas bien avec le système complexe choisi comme exemple. Aucun coefficient ne converge. Lors des simulations en régulation, avec le modèle M_1, les paramètres du modèle identifié convergent grossièrement lors de la phase en plateau, lorsque le modèle M_1 à fort gain injecte une commande faible alors que le système a un faible gain, mais cette convergence approximative s'effondre lorsque le modèle M_1 corrige la commande. Dans ce cas les coefficients évoluent à nouveau.
Ces méthodes adaptatives semblent donc ne converger que lorsque les paramètres n'évoluent pas, ou alors très lentement.

Une méthode par segmentation a été testée, en réinitialisant l'algorithme toutes les secondes, pour donner par conséquent 5 modèles. Aucun n'était correct.

Les modèles évolutifs, bien que plus complexes que les précédents, furent testés parce que les résultats présentés dans Najim pour la modélisation des signaux de la parole étaient remarquables. Alors que le filtre de Kalman étendu ne donnait que du bruit, les évolutions obtenues avec les modèles évolutifs étaient parfaitement lisses.

Les résultats obtenus avec le système complexe furent suffisamment intéressants pour que les modèles évolutifs soient finalement retenus.

6.4 Vers la Multi-Identification

Cette partie conclue les résultats présentés dans ce mémoire. Les résultats imparfaits amèneront en conclusion de ce chapitre à la notion de multi-identification.

Le système test est représenté par (par hypothése τ est toujours positif):

$$x(t) + \tau(x)\dot{x}(t) = k(x)u(t) \tag{6.29}$$

avec

$$\begin{aligned} k &= 36x(t)(x(t)-1)+10 \\ \tau &= 15-10x(t) \end{aligned} \tag{6.30}$$

Pour les simulations, les conditions seront identiques au 6.3. Un seul modèle, le modèle M_1 sera utilisé pour la commande, mais bien évidemment, il ne correspond au système que rarement. L'utilisation de M_1 seule va donc provoquer des erreurs.

M_1:
$$x = \frac{10}{1+15s}u \tag{6.31}$$

Les méthodes d'identification présentées à celles du paragraphe 6.3 concernaient des systèmes non stationnaires. Aussi les modèles évolutifs étaient -ils définis sur une base de fonction du temps. Dans le cas présent, les fonctions de la base auront pour argument x.

Théoriquement le système 6.29 ne devrait pas pouvoir être identifié avec la base des polynômes. En effet si les coefficients k et τ s'expriment sous forme de polynômes de x avec le modèle continu, avec le modèle discret ils apparaissent comme des fractions rationnelles de x. Mais par chance, le dénominateur, en fait τ, est un polynôme de degré 1, et x varie entre 0

et 1. Or le rayon de convergence du développement en série de $(1-x)^{-1}$ est justement]-1,1[. L'identification est donc possible, ce qui permet de souligner encore les qualités d'approximation de la base des puissances.

La première simulation concerne le cas de l'identification sans bouclage de la commande du modèle M_{id} (la commande u est en fait seulement u_1, commande dérivée de M_1).

6.4.3.1 Identification pure du système complexe.

Les figures suivantes donnent le résultat de l'identification, sans boucler le modèle M_{id}. L'algorithme avec test sur l'excitation permanente a été utilisé (λ=0.9), avec la base des puissances, de taille 5.

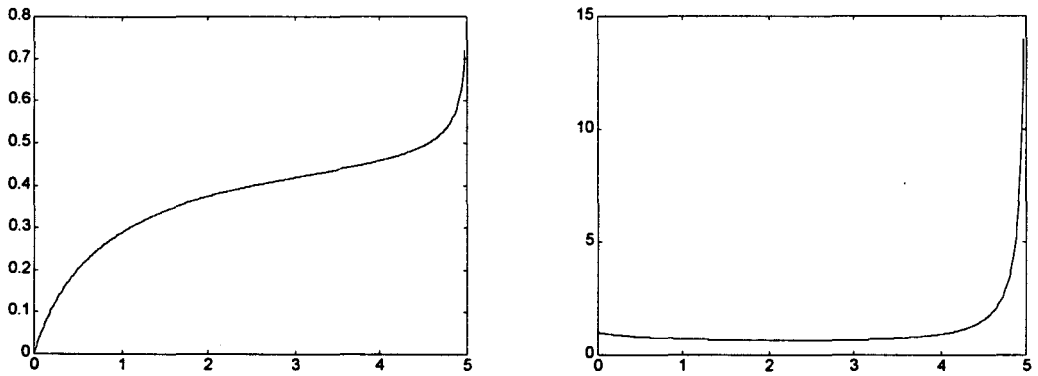

Fig. 6.3. x(t) et u(t) pour le système complexe, avec seulement M_1.

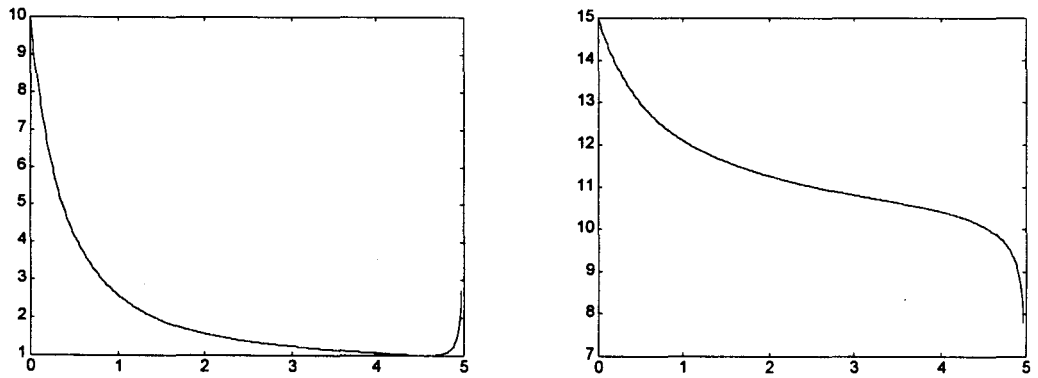

Fig. 6.4. Gain(t) et Constante de temps τ(t) du système.

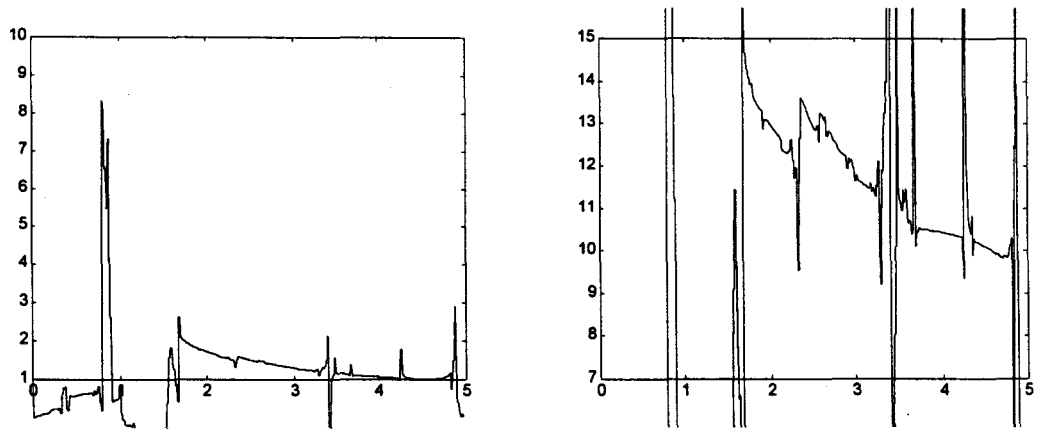

Fig. 6.5. Gain(t) et Constante de temps $\tau(t)$ identifiés.

Comme seul le modèle M_1 intervient pour le calcul de la commande, celle ci s'avère très mauvaise. Les deux paramètres du modèle sont relativement bien identifiés, mais ils sont forts bruités. Pour t<2 secondes, les deux paramètres évoluent trop vite pour que l'algorithme d'identification puisse converger.

Afin de faire quelques comparaisons, les valeurs des paramètres du système avec le modèle discret sont aussi données:

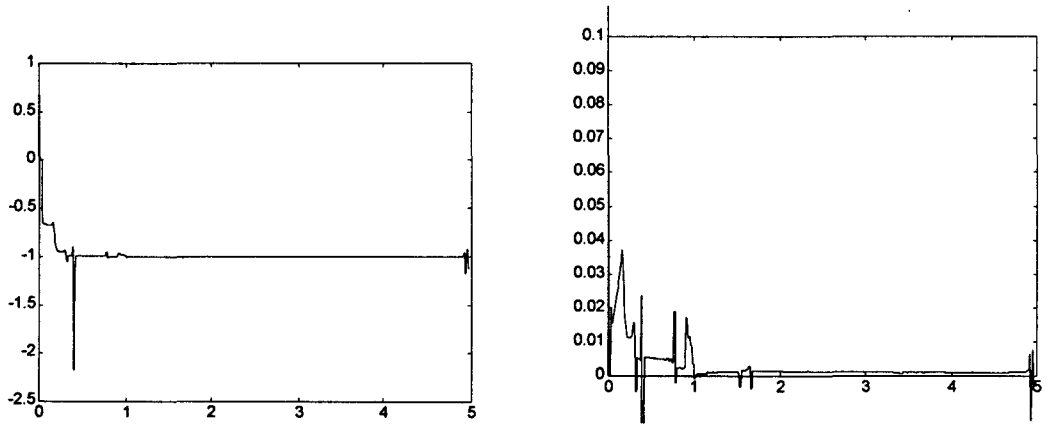

Fig. 6.6. a(t) et b(t) du modèle sous forme discrète.

6.4.3.2 Bouclage de l'identification du système complexe.

Les paramètres sont identiques, mais cette fois le modèle M_{id} intervient au niveau de la commande du système.

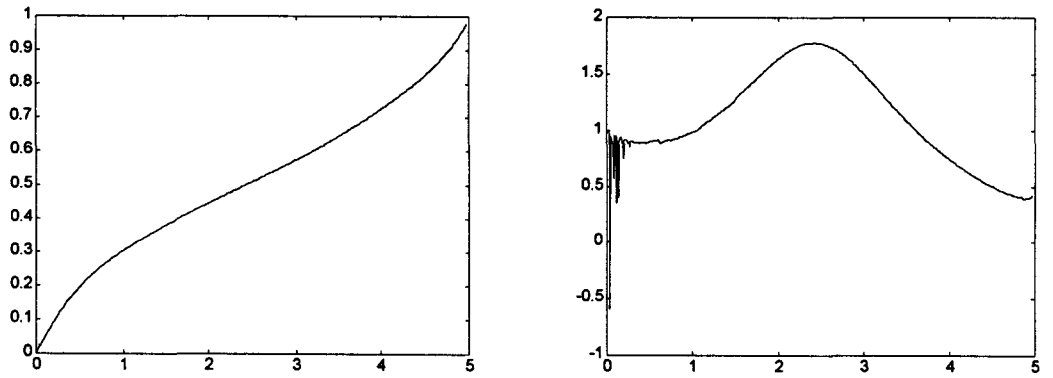

Fig. 6.7. x(t) et u(t) pour le système complexe, avec M_1 et M_{id} (base de polynômes).

La consigne est bien atteinte, et le coût, donné par

$$J = \int_{t_0}^{t_f} \left((x - x_f)^2 + u^2 \right) dt \tag{6.32}$$

atteint 8.4.

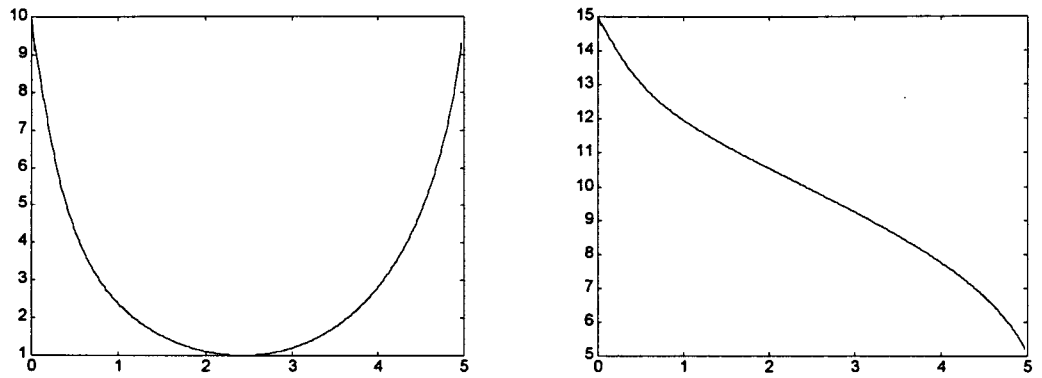

Fig. 6.8. Gain(t) et Constante de temps τ(t) du système.

Il apparaît que les coefficients évoluent de manière moins non linéaire qu'auparavant. L'identification en est beaucoup améliorée:

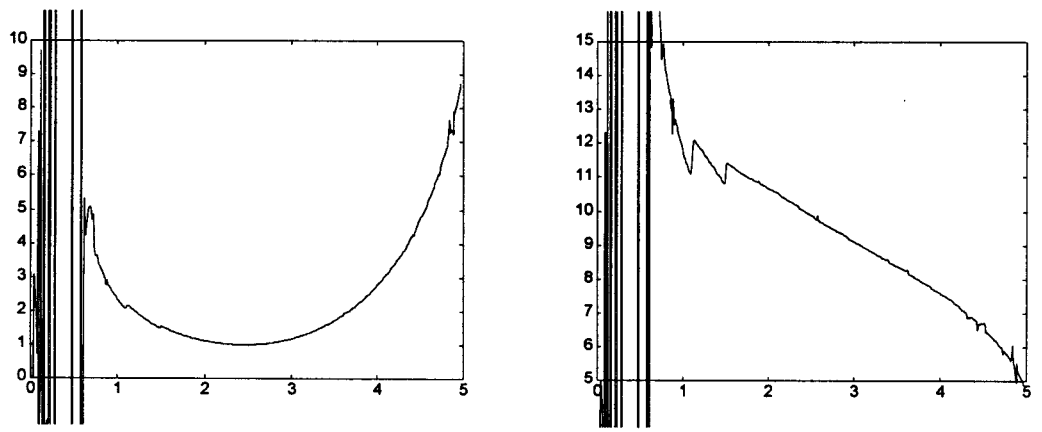

Fig. 6.9. Gain(t) et Constante de temps τ(t) identifiés (base de polynômes).

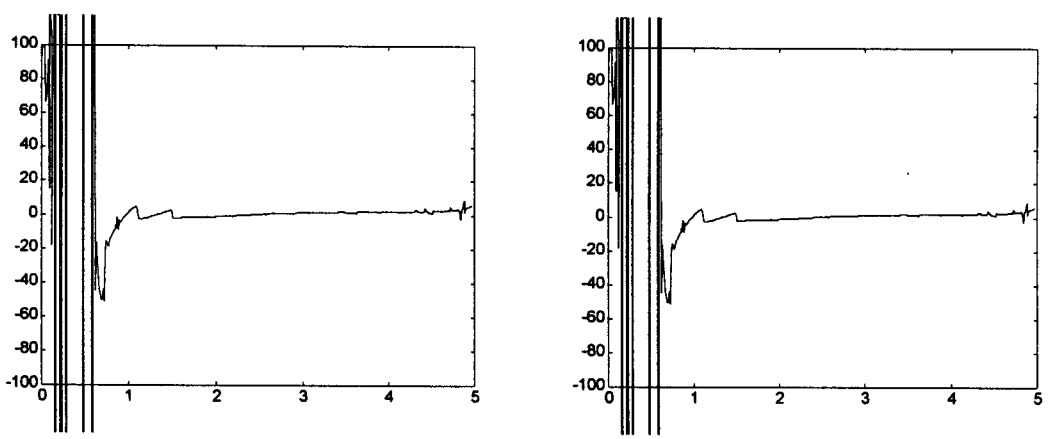

Fig. 6.10. Erreur relative sur le gain et sur τ.

La figure 6.10 montre les erreurs commises sur l'estimation de k(t) et τ(t). Elles sont toutes les deux très faibles pour t>0,5 seconde.

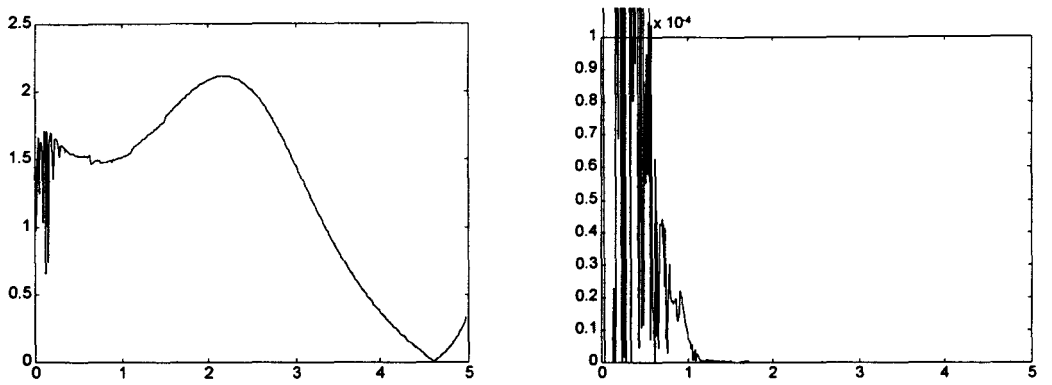

Fig. 6.11. Résidu relatif pour M_1 et M_{id}.

La figure 6.11 montre les résidus normalisés qui vont servir à calculer les validités (sous forme non normalisée ils apparaissent dans l'algorithme d'identification). Ils sont très proches de 0. Le modèle M_1, qui, seul, produit une très mauvaise commande, ne fait ainsi des erreurs que de l'ordre du pour-cent. Pour le modèle M_{id}, ces erreurs sont extrêmement faibles.

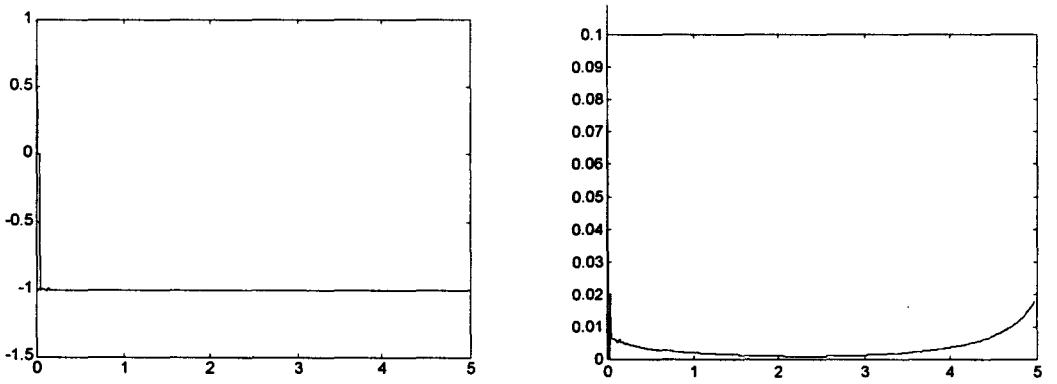

Fig. 6.12. a(t) et b(t) du modèle sous forme discrète.

La figure 6.12 montre les coefficients a(k) et b(k) du modèle sous forme discrète. Par rapport à l'identification non bouclée du paragraphe précèdent, il faut noter que l'amélioration concerne en fait le paramètre b(k), puisque le coefficient a(k) est constant dans les deux cas.

Il est possible de tester l'algorithme d'identification avec la base de Fourier (5 fonctions, mêmes autres paramètres). L'identification est bien réalisée aussi, même si elle est un peu plus bruitée (le facteur T faisant la conversion temps indice a été pris égal à 1/100).

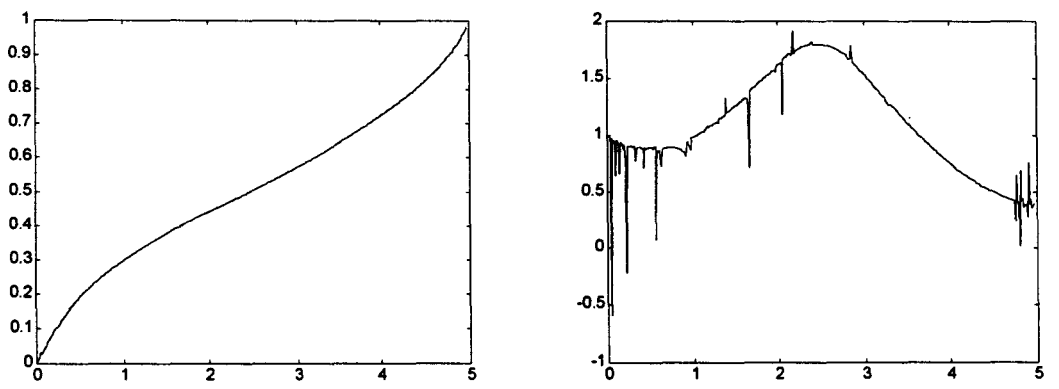

Fig. 6.13. x(t) et u(t) pour le système complexe, avec M_1 et M_{id} (base de Fourier).

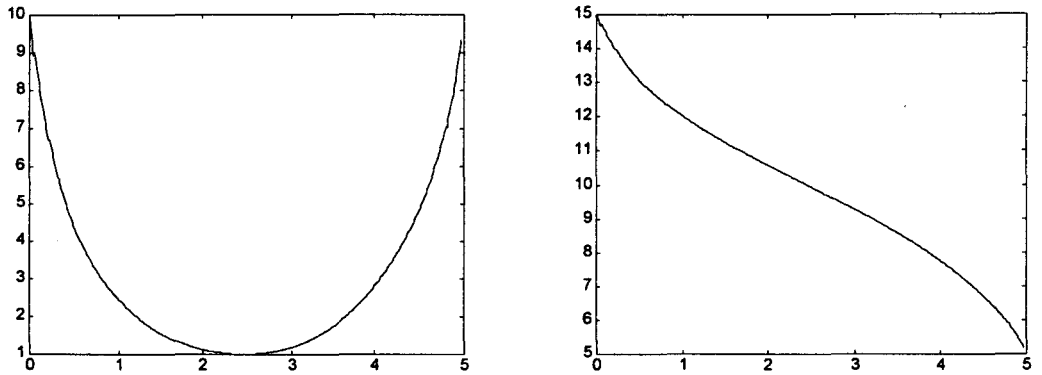

Fig. 6.14. Gain(t) et Constante de temps $\tau(t)$ du système.

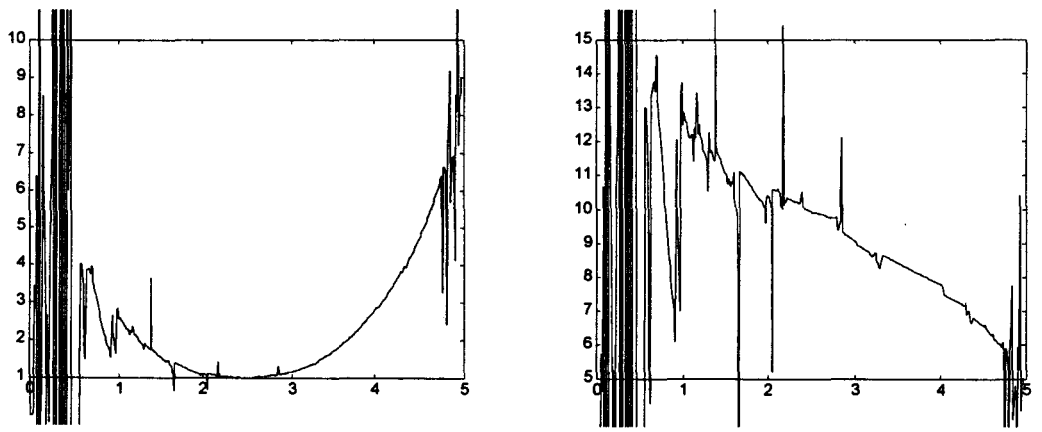

Fig. 6.15. Gain(t) et Constante de temps $\tau(t)$ identifiés (base de Fourier).

Les bons résultats obtenus montrent que finalement, si le choix de la base est important, il n'est pas critique dans tous les cas.

6.4.3.3 Bouclage de l'identification, cas de la poursuite.

Le présent paragraphe a uniquement pour objectif d'illustrer l'identification du système complexe dans un cas de poursuite.

L'algorithme utilisé est celui avec facteur d'oubli et test sur l'excitation permanente (λ=0.99). La consigne est $x_f(t)=0.5+0.5\sin(0.25t)$ (elle est superposée à x(t) sur les courbes).

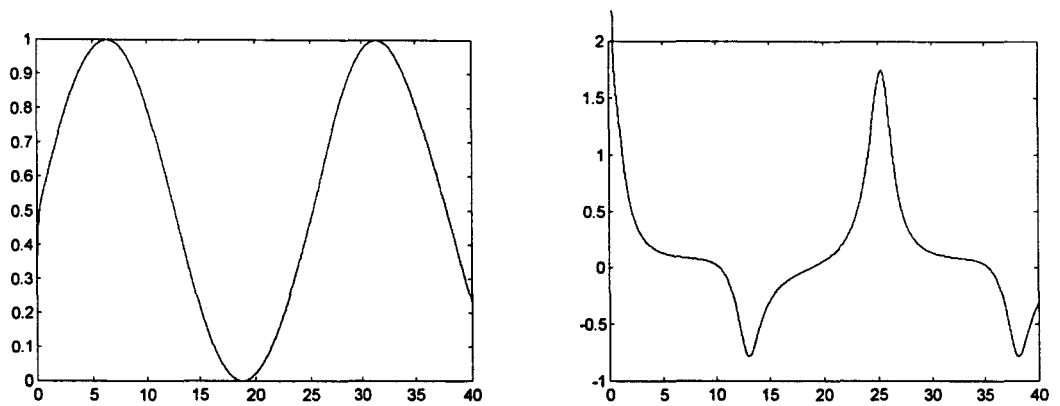

Fig. 6.16. x(t) et u(t) pour le système complexe, avec M_l et M_{id} (base de Fourier).

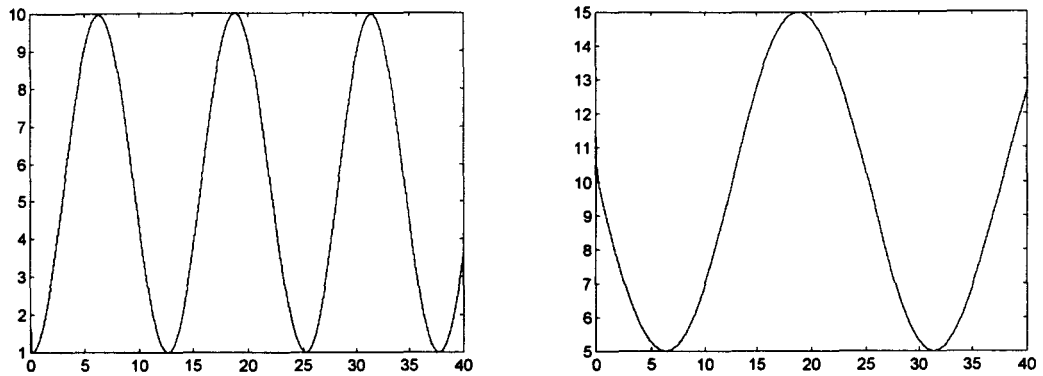

Fig. 6.17. Gain(t) et Constante de temps τ(t) du système.

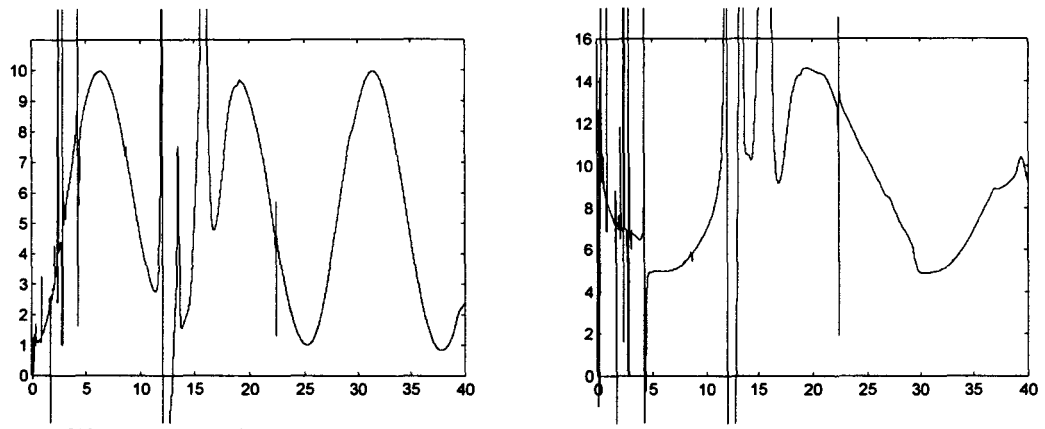

Fig. 6.18. Gain(t) et Constante de temps τ(t) identifiés (base de Fourier).

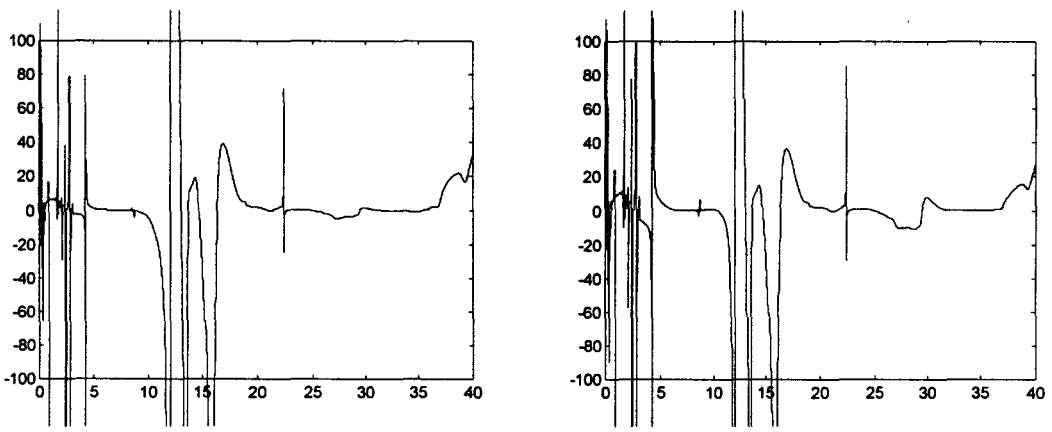

Fig. 6.19. Erreur relative sur le gain et sur τ.

La base de Fourier utilise 7 fonctions, et surtout le facteur τ liant temps et indice est passé à 1/25, ce qui veut dire que les fonctions de la base évoluent 4 fois plus vite par indice qu'avant.

Dans cet exemple il y avait de sévères problèmes numériques, puisque pour λ inférieur à 0.99 les matrices divergeaient. Il faudrait peut être utiliser la forme de Joseph pour la mise à jour des matrices K et F dans l'algorithme d'identification [Borne et al, 1992]

6.4.4 La multi-identification

Les méthodes évolutives gèrent très mal les discontinuités dans les modèles. En fait le développement en série entière d'une fonction au voisinage d'une asymptote est très imprudent. Il faudrait alors plutôt recourir à des fonctions du type t^{-j}, mais cela est difficile à savoir à l'avance.

Par contre les méthodes adaptatives classiques gèrent très bien les transitions, si elles ne sont pas trop rapides.

Il pourrait donc être intéressant de compléter un algorithme de type évolutif par un autre, plus frustre, mais pouvant gérer les discontinuités.

Dans le même ordre d'idée, la commande du système complexe 6.61 s'est faite par l'identification sur une base de fonctions de x, et non pas du temps. Le problème du choix de la base de fonctions n'est pas à négliger.

Jusqu'à présent les méthodes utilisées étaient basées sur des critères quadratiques. Il existe d'autres types d'algorithmes.

Il serait donc tout à fait intéressant de pouvoir jouer sur les complémentarités de ces méthodes d'identification pour définir un système d'identification qui fonctionnerait avec le moins d'hypothèses possibles.

L'idée de multi-identification est alors introduite. A l'image du multi-modèle, qui crée un modèle global à partir de plusieurs petits modèles, une méthode globale d'identification serait définie à partir de méthodes élémentaires. La méthode globale apparaît pour l'instant implicite, et la validité des méthodes serait évaluée à partir de leurs modèles dérivés.

Pour l'étape de fusion, il faudrait déterminer s'il est plus intéressant de fusionner les modèles identifiés, pour fournir un modèle identifié global explicite, qui serait alors utilisé avec les lois de commandes, comme les modèles statiques, ou alors de fusionner l'ensemble de toutes les commandes de tous les modèles, en une seule passe.

Cette orientation vers la multi-identification se plonge donc dans la même dialectique que celle du multi-modèle, notamment pour la question du coût de calcul. Elle pourrait être très intéressante pour résoudre de complexes problèmes d'identification.

6.5 Conclusion

Comme cela a été souligné en introduction, les approches multi-modèles sont à mi chemin des approches adaptatives et des approches classiques. Proche des méthodes classiques, car le concepteur, ayant à sa disposition des modèles, peut calculer des commandes élaborées. Proche des méthodes adaptatives, car l'idée même de plusieurs modèles est liée à l'idée d'évolution des systèmes rencontrés.

Afin de pallier aux évolutions inconnues, il fallait cependant recourir à des méthodes d'identification, comme avec les approches adaptatives. Alors, a été défini un multi-modèle

ayant une librairie statique, et un ou des modèles dynamiques (au sens de modèles à paramètres variables) suivant les évolutions du système.

Pour des problèmes complexes, ce contrôleur multi-modèle peut donc prétendre concurrencer les deux approches, classiques et adaptatives, car il emprunte des outils à chacune d'entres elles.

Les exemples montrés furent très simples, ils consistaient à compléter une librairie défaillante réduite à un modèle par un deuxième modèle adaptatif. Le premier modèle pouvait parfaitement être supprimé, au prix d'une légère perte de performance lors de l'identification initiale. Ces exemples étaient donc très proches de la simple commande adaptative indirecte.

Cependant cette approche ouvre de prometteuses perspectives.

En ce qui concerne l'identification proprement dite, ce chapitre n'a rien apporté de bien original. Des méthodes d'identification classiques furent simplement mises en œuvre. Afin de pouvoir travailler avec le système utilisé dans les simulations des précédents chapitres, il a fallu recourir à des méthodes plus élaborées. Ces méthodes un peu moins courantes sont celles qui utilisent des bases de fonctions.

L'idée de multi-identification fut donc finalement introduite en vue de pouvoir jouer sur les inconvénients et les avantages respectifs des différentes méthodes d'identification, afin de pouvoir en définir une qui ait moins de contraintes que chacune des méthodes prises séparément.

CONCLUSION GENERALE

La logique floue a profondément changé la représentation des connaissances. D'une part elle a permis de traduire mathématiquement des concepts imprécis, incertains ou encore peu fiables, qui tous, jusqu'à lors, n'étaient pas, ou mal, gérés par les sciences; d'autre part elle a permis la décomposition de concepts complexes en plusieurs entités simples.

Ce dernier point a conduit aux contrôleurs flous puis aux représentation et aux contrôleurs multi-modèles. En résumé ces derniers permettent de commander des systèmes très complexes par le mélange de plusieurs modèles et de plusieurs lois de commandes.

La structure d'un contrôleur (ou d'un modèle pour la modélisation) multi-modèle est bien évidemment plus complexe que celle d'une approche avec un seul modèle. Il faut en effet estimer la pertinence des éléments utilisés. Il faut ensuite fusionner toutes ces informations. Cependant cette complexité initiale de structure permet une grande souplesse alliée à une réduction de la complexité des modèles mis en oeuvre, qui montrent que les approches multi-modèles sont très riches. Les premiers chapitres se sont attachés à présenter les racines des ces approches, puis leur architecture interne, et plus précisément les algorithmes évaluant la validité des éléments, et aussi l'algorithme fusionnant ces informations. Pour ce dernier point des outils dérivés de la théorie des possibilités peuvent être utilisés.

Dans cette théorie, en dehors de toute référence aux approches multi-modèles, divers opérateurs ont été proposés pour essayer de gérer mieux les informations et leurs fiabilités respectives.

Pour en revenir à l'analyse multi-modèles, les derniers travaux présentés ont été focalisés sur l'identification de nouveaux modèles pour compléter une bibliothèque défaillante. Même si ces travaux sont loin d'être terminés, il n'est cependant pas dans la philosophie des contrôleurs multi-modèles adaptatifs de vouloir concurrencer les méthodes purement adaptatives, qui ne requièrent que peu d'informations sur les systèmes. Au contraire, les descriptions multi-modèles représentent bien plus souvent la prise en compte des connaissances déjà acquises. Ils ne diffèrent dans leur esprit des méthodes classiques de l'automatique que sur la transcription et la construction des connaissances et de leur outils dérivés. La bibliothèque dynamique qui gère des nouveaux modèles (et pourquoi pas de nouvelles commandes) est conçue uniquement pour répondre à une défaillance des modèles élémentaires. En cela les contrôleurs multi-modèles pourraient servir de pont entre les approches classiques de l'automatique et les approches adaptatives.

Les simulations présentées ont montré sur des exemples simples combien il était facile d'utiliser une approche multi-modèles, et combien ces méthodes étaient souples d'emploi. Dans la littérature sont souvent présentées des méthodes de modélisation et de commande des systèmes complexes. En général plusieurs types d'algorithmes sont utilisés à différentes étapes, et la méthode proposée forme un tout pour lequel il semble difficile de faire la part des choses, par exemple les différents liens entre les algorithmes, leurs avantages et leurs inconvénients respectifs. En quelque sorte la méthode est livrée "clés en main". L'architecture proposée dans les pages précédentes relève d'une démarche opposée. La structure a été bien analysée en étapes fondamentales, et si certains algorithmes de commande et d'identification furent utilisés, ils peuvent être échangés avec d'autres méthodes. L'architecture proposée est donc beaucoup plus ouverte que les méthodes de modélisation et de commande complexes déjà existantes.

Pour la suite, avec des problèmes plus complexes, il faudrait poursuivre la voie tracée. Il faudra pouvoir jouer sur la complémentarité entres les modèles précis mais sensibles et les modèles robustes. Il faudra pouvoir jouer sur les avantages et les inconvénients des commandes robustes et des commandes optimisées. C'est pourquoi il est si difficile d'utiliser pleinement les approches multi-modèles. En effet il ne suffit pas de connaître une seule loi de commande, il faut toutes les connaître. Il faut connaître tous les problèmes liés aux paramètres des modèles, le choix de leur ordre, de leur structure. Il faut connaître les liens entre les commandes et les modèles particuliers dont elles sont déduites.

A l'image d'un architecte qui aurait à sa disposition plusieurs types de briques, plusieurs types de poutres, et plusieurs types de mortier, et qui doit encore beaucoup travailler pour construire de belles maisons, il faudra donc encore beaucoup de mathématiques pour obtenir des bons contrôleurs multi-modèles.

BIBLIOGRAPHIE

Bezdek J. C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, New-York Plenum, 1981.

Bezdek J. C., *Analysis of Fuzzy Information*, Eds Mathematics and Logic, CAC Press Floride, 1986.

Bloch I., *Information Combination Operators for Data Fusion: A comparative Review with Classification*, IEEE T. on Systems, Man, and Cybernetcis, Part A, vol 26, no 1, 52-62, 1995.

Borne P., G. Dauphin Tanguy, J. P. Richard, F. Rotella, I. Zambettakis, *Commande et Optimisation des Processus*, Technip, 1990.

Borne P., G. Dauphin Tanguy, J. P. Richard, F. Rotella, I. Zambettakis, *Modélisation et Identification des Processus*, vol 2, Technip, 1992.

Bouchon-Meunier, *La Logique Floue*, Que sais-je, PUF, 1993.

Buckley J. J., and H. Ying, *Fuzzy Controller Theory: Limit Theorems for Linear Fuzzy Control Rules*, Automatica, vol 25, no 3, 469-472, 1989.

Buhler H., *Réglage par mode de glissement*, Presses Polytechniques Romandes, 1994.

Cao S.G., N. W. Rees and G. Feng, *Stability Analysis of Fuzzy Control Systems,* IEEE T. on Systems, Man and Cybernetics, Part B, vol 26, no 1, 201-204, 1996.

Castro J. L., *Fuzzy Logic Controllers Are Universal Approximators*, IEEE T. on Systems, Man and Cybernetics, vol 25, no 4, 629-635, 1995.

Chen S. J., C. L. Hwang, in collaboration with F. P. Hwang, *Fuzzy Multiple Attribute Decision Making*, Springer Verlag, 1992.

CSN (référence commune): CNRS, Pole Automatisation Intégrée, Projet Commande Symbolique et Neuromimétique.

Delgado M. and S. Moral, *On the Concept of Possibility-Probability Consistency*, Fuzzy Sets and Systems, vol 21, 311-318, 1987.

Delmotte F., L. Dubois, A. M. Desodt and P. Borne, *Using Trust in Uncertainty theories*, Information and Systems Engineering, vol 1, no 3-4, 303-314, 1995a.

Delmotte F., L. Dubois, P. Borne, *Adaptive Multi-Models Using Trust*, Proc. of IEEE SMC, vol 5, 4155-4160, 1995b.

Delmotte F., L. Dubois, P. Borne, *A General Scheme For Multi-Models Controller Using Trust*, Mathematics and Computer in Simulation, vol 41, no 1-2, 173-186, 1996a.

Delmotte F., S. Hajri, P. Borne, *Multi-Models and Sliding Mode Control*, World Automation Congress, vol 4, 144-151, 1996b.

Delmotte F., L. Dubois, *Multi-Models and Belief Functions for Decision Support Systems*, Proc. of IEEE. SMC, Beijing, Chine, 181-186, 1996c.

Delmotte F., L. Dubois, P. Borne, *Context Dependant Trust in Data Fusion within the Possibility Theory*, Proc. of IEEE SMC, Beijing, Chine, 538-543, 1996d.

Delmotte F., A.M. Desodt, P. Borne, *Feedback Adaptive Fusion Rule in Possiblity Theory*, Conf. CESA 96, Lille, France, Symp. on Robotics and Cybernetics, 410-415, 1996e.

Delmotte F., L. Dubois, P. Borne, *Optimization of reliability coefficients for a new family of fusion rule*, IFAC SICICA 97, Annecy, France, à paraître en 1997a

Delmotte F., *Multi-Models and Self Reconfiguration*, IFAC CACSD 97, Gand, Belgique, 243-248, Avril 1997b

Delmotte F., L. Dubois, P. Borne, *Dynamical Library of Models for Multi-Models Controllers*, IMACS 97, Berlin, Allemagne, à paraître en 1997c

Delmotte F., P. Borne, *Modeling of Reliability with Possibility Theory*, IEEE T. on Systems, Man, and Cybernetics, à paraître.

Deveughele S. and B. Dubuisson, *The Influence of a Conclict Index in the Frame of the Adaptive Combination*, Proc. of CESA 1996, Lille, France, 98-103, 1996.

Dubois D., H. Prade, *On Several Representations of an Uncertain Body of Evidence*, Fuzzy Information and Decision Processes, 167-181,1982.

Dubois D., H. Prade, *Unfair Coins and Necessity measures: Towards a Possibilitic Interpretation of Histograms*, Fuzzy Sets and Systems, vol 10, 15-20, 1983.

Dubois D., H. Prade, *Evidence Measures Based on Fuzzy Information*, Automatica, vol 21, no 5, 547-562, 1985.

Dubois D., H. Prade, *Representation and Combination of Uncertainty with Belief Functions and Possibility Measures*, Comput. Intell., vol 4 ,244-264, 1988.

Dubois D., H. Prade, *Théorie des Possibilités*, Masson, 1988

Dubois D. and H. Prade, *Combination of Fuzzy Information in the Framework of Possibility Theory*, in Data Fusion in Robotics and Machine Intelligence, M. A. Abidi and R. C. Gonzales, Academic Press, 1992.

Dubois D., and H. Prade, *Adaptive Combination Rules for Possibility Distributions*, Proc. of EUFIT 1994, 48-52, 1994.

Dubois D., and H. Prade, *La Fusion d'Informations Imprecises, Traitement du signal*, vol 11, no 6, 447-458, 1994.

Dubois G., *Résolution d'un Système d'Equations de Relation Floue, Contribution à l'Identification de Systèmes Complexes*, Thèse de l'université de Nancy 1, 1992.

Dubois L., F. Delmotte, P. Borne, *A Quasi Optimization Method of Control for Ill-defined System Using a Multi-Model Approach*, Proc. of MMAR, Poland, 121-125, 1995a.

Dubois L., *Utilisation de la logique Floue dans la Commande des Systèmes Complexes*, Thèse de Lille 1, 1995b.

Dubois L., F. Delmotte, T. Fukuda, *On a Multi-Strategy Approach in Evolutionary Computation,* IEEE Int. Conf. In Evolutionary Computation, Nagoya, Japon, 576-580, 1996a

Dubois L., F. Delmotte, T. Fukuda, P. Borne, *Multi-Models Systems are Universal Approximators*, Conf. CESA 96, Lille, France, Symp. on Modeling, Analysis and Simulation, 879-884, 1996b.

Dubois L., T. Fukuda, F. Delmotte, P. Borne, *Towards a Self-Organising Multi-Models System*, 12th Fuzzy System Symposium, 961-964, 1996c.

Dubois L., F. Delmotte, T. Fukuda, *Global multi-model approach for the control of a robot in an uncertain environment*, Int. Conf. on Robotics and Automation, Albuquerque, Etats Unis, à paraître en 1997

Foulloy L., S. Galichet, *Controleurs Flous, Représentation, équivalences et études comparatives*, CSN 92-2, 1992.

Goldberg, D. E., *Genetics Algorithms in search, optimization, and machine learning*. Addison Wesley Publishing Compagny, 1989.

Grabish M., H. T. Nguyen, E. A. Walker, *Fundamentals of Uncertainty Calculi With Applications to Fuzzy Uncertainty*, Kluwer Academic Publishers, 1995.

Grabish M., *On Equivalence Classes of Fuzzy Connectives-The Case of Fuzzy Integrals*, IEEE T. on Fuzzy Systems, vol 3, no 1, 96-109, 1995.

Grafenstedt, *Optimization of control parameters for genetic algorithms*, IEEE T. on Systems, Man and Cybernetics, 16, no 1, 122-128, 1986.

Grenier Y., *Modélisation de Signaux Non Stationnaires*, Thèse d'Etat en Sciences Physiques, Orsay, 1984

El Hajjaji, *Représentation Analytique du Controleur Flou*, CSN 94-3, 1994.

Ishigame A., T. Furukawa, S. Kawamoto, and T. Taniguchi, *Sliding Mode Controller Design Based on Fuzzy Inference for Nonlinear Systems*, IEEE T. on Industrial Electronics, vol 40, no 1, 64-70

Johansen T. A., *Fuzzy Model Based Control: Stability, Robustness, and Performance Issues*, IEEE T. on Fuzzy Systems, vol 2, no 3, 221-234, 1995.

Kelman A., R.R. Yager, *Compatibilité et Agrégation Partielle: Méthode de Fusion Modulaire*, Rencontres Francophones sur la Logique Floue et ses Applications, LFA'95, 73-79, 1995.

Kiska J. B., M. M. Gupta, and P. N. Nikiforuk, *Energistic Stability of Fuzzy Dynamic Systems*, IEEE T. on Systems, Man, and Cybernetics, vol 15, no 6, 783-792, 1985.

Klir G. J., *Where do we Stand on Measures of Uncertainty, Ambiguity, Fuzzyness, and the Like*, Fuzzy Sets and Systems, vol 24, 141-160, 1987.

Klir G. J., *Measures of Uncertainty in the Dempster-Shafer Theory on Evidence*, in Advances in the Dempster-Shafer Theory of Evidence, by R. R. Yager, J. Kacprzyk, M. Fedrizzi, John Wiley and Sons, 34-49, 1994.

Lacrosse V., *Simplification des Controleurs Flous*, CSN 1996

Lamotte M., *Equations de Relation Floue*, CSN 91-3, 1991.

Landau I. D., *Identification et Commande des Systèmes*, Traité des nouvelles technologies, HERMES, 1988.

Lee C. C., *Fuzzy Logic in Control Systems: Fuzzy Logic Controller, Part 1-2*, IEEE T. on Systems, Man and Cybernetcis, vol 20, no 2, 404-435, 1990.

Lerman, *La classification automatique*, Gauthiers Villars Paris, 1970.

Maiers J. and Y. S. Sherif, *Applications of Fuzzy Set Theory*, IEEE T. on Systems, Man and Cybernetics, , vol 15, no 1, 175-189, 1985.

Mandani E. H., *Application of Fuzzy Algorithms for simple dynamic plant*, Proc. of IEE, vol 121, no 12, 1585-1588, 1974.

Marin J. P., *Outils pour l'Analyse de la Stabilité des Systèmes Bouclés par Controleurs flous*, CSN 95-1, 1995.

Marin J. P., A. Titli, *Stabilité et Performances des Systèmes Flous, 2 Approches Basées sur la Méthode Directe de Lyapunov*, CSN 95-4, 1995.

McMurty G. J., *Adaptive Optimization Procedures*, in Adaptive, Learning and Pattern Recognition Systems, Eds J. M. Mendel, K. S. Fu, Academic Press, New York, 243-286, 1970.

Perruquetti W., *La Commande par Mode Glissant*, Rapport Interne du LAIL, Ecole Centrale de Lille, 1992.

Pierrot H., F. Delmotte, C. Guernez, P. Borne, *Multi-models based on residuals* IFAC CIS 97, Belfort, France, à paraître en 1997

Najim M., *Modélisation et Identification en Traitement du Signal*, Masson, 1988.

Nakamori Y., K. Suzuki, and T. Yamanaka, *A New Design of a Fuzzy Model predictive Control System For Non linear Processes*, Fuzzy Engineering toward Human Friendly Systems, IFES1991, 788-798, 1991.

Narendra K. S., J Balakrishnan and M. K. Ciliz, *Adaptation and Learning Using Multiple Models, Switching, and Tuning*, IEEE Control Systems, 37-51, June 1995.

Narendra K. S., J Balakrishnan, *Adaptive Control Using Multiple Models*, IEEE T. on Automatic Control, vol 42, no 2, 1997.

Olivera J. V., *A Design Methodology for Fuzzy System Interfaces*, IEEE T. on Fuzzy Systems, vol 3, no 4, 404-414, 1995.

Pearson A. E., *Identification of Linear Time Varying Differential Systems Amid Modal Disturbances*, Proc of 5th IFAC Congress on Identification and System Parameter Estimation, Washington D.C., 1051-1056, 1982.

Pedrycz W., *Fuzzy Multimodels*, BUSEFALL, vol 64, 9-17, 1995.

Procyk T. J. and E. H. Mamdani, *A Self-Organizing Process Controller*, Automatica, vol 15, 15-30, 1979.

Rondeau I, R. Ruelas, E. Levrat, M. Lamotte, *Définition d'une Méthode de Défuzzification Respectant la Fuzzification*, CSN 95-3, 1995

Sandri S. S., D. Dubois, and H. W. Kalfsbeek, *Elicitation, Assessment, and Pooling of Expert Jugements Using Possibility Theory*, IEEE T. on Fuzzy Systems, vol 3, no 3, 313-335, 1995.

Schweizer B., and A. Sklar, *Associative Functions and Abstract Semigroups*, 69-81, 1967.

Silvert W., *Symmetric Summation: A Class of Operations on Fuzzy Sets*, IEEE T. on Systems, Man, and Cybernetics, vol 9, no 10, 657-659, 1979.

Smets P., *Imperfect Information: Imprecision-Uncertainty*, rapport interne de IIRIDIA, Bruxelles, Belgique, no TR/IRIDI/93-3, 1993.

Sudarno W., *Amélioration des méthodes d'identification de types moindres carres, appliquées à la commande adaptative et à la reconnaissance des formes* Thèse en Automatique de l'Université Paul Sabatier, Toulouse, 1996.

Takagi T. and M. Sugeno, *Fuzzy Identification of Systems and Its Applications to Modeling and Control*, IEEE T. on Systems, Man, and Cybernetics, vol 15, no 1, 126-132, 1985.

Tanaka K., *Stability and Stabilizability of Fuzzy-Neural-Linear Control Systems*, IEEE T. on Fuzzy Systems, vol 3, no 4, 438-447, 1995

Tong R. M., *Some Properties of Fuzzy Feedback Systems*, IEEE T. on Systems, Man, and Cybernetics, vol 10, no 6, 327-330, 1980.

Utkin V.I., *Variable Structure Systems with Sliding Modes*, IEEE T. on Automatic Control, vol 22, no 2, 212-222, 1977.

Utkin V.I, *Sliding Modes in Control and Optimization*, Springer-Verlag, 1992.

Walter E et L. Pronzato, *Identification de modèles paramétriques, A partir de Données Expérimentales*, MASC, Masson, 1994.

Wang L. X., and J. M. Mendel, *Fuzzy Basis Functions, Universal Approximation, and Orthogonal Least Squares Learning*, IEEE T. on Neural Networks, vol 3, no 5, 807-814, 1992a.

Wang L. X., and J. M. Mendel, *Generating Fuzzy Rules by Learning from Examples*, IEEE T. on Systems, Man and Cybernetics, vol 22, no 6, 1992b.

Wang L. X., *Stable Adaptive Fuzzy Controller of Nonlinear Systems*, IEEE T. on Fuzzy Systems, vol 1, no 2, 146-155, 1993.

Wang L., and R. Langari, *Complex Systems Modeling via Fuzzy Logic*, IEEE T. on Systems, Man, and Cybernetics, Part B, vol 26, no 1, 100-105, 1996.

Weber S., *A General Concept of Fuzzy Connectives, Negations and Implications Based on t-norms and t-conorms*, Fuzzy Sets and Systems, vol 11, 115-134, 1983.

Xu C. W. and Y. Z. Lu, *Fuzzy Model Identification and Self-Learning for Dynamic Systems*, IEEE T. on Systems, Man and Cybernetics, vol 17, no 4, 683-689, 1987.

Yager R. R., and D. P. Filev, *Including Probabilistic Uncertainty in Fuzzy Logic Controller Modeling using Dempster-Shafer Theory*, BUSEFAL, 54-791992.

Yager R. R., D. P. Filev, *Unified Structure and Parameter Identification of Fuzzy Models*, IEEE T. on Systems, Man and Cybernetics, vol 23, no 4, 1198-1205, 1993.

Yager R.R., A. Kelman, *Fusion of Fuzzy Information with Considerations for Compatibility, Partial Aggregation and Reinforcement*, Int. J. of Approximate Reasoning, vol 15, 93-122, 1996.

Yager R. R., D. P. Filev, and T. Sadeghi, *Analysis of Flexible Structured Fuzzy Logic Controllers*, IEEE T. on Systems, Man and Cybernetics, vol 24, no 7, 1035-1043, 1994.

Ying H., W. Sileri, and J. J. Buckley, *Fuzzy Control Theory: A Nonlinear Case*, Automatica, vol 26, no3, 513-520, 1990.

Zadeh L. A., *Fuzzy Sets*, Informatic Control, vol 8, 338-353, 1965.

Zadeh L. A., *Fuzzy Sets as a Basis for a Theory of Possibility*, Fuzzy Sets and Systems, vol 1, 3-28, 1978.

Zeng X. J., M. G. Singh, *Approximation Theory of Fuzzy Systems-SISO Case*, IEEE T. on Fuzzy Systems, vol 2, no 2, 162-176, 1995.

Zeng X. J., M. G. Singh, *Approximation Theory of Fuzzy Systems-MIMO Case*, IEEE T. on Fuzzy Systems, vol 3, no 2, 219-235, 1995.

ANNEXE A

LES ALGORITHMES GENETIQUES

Les algorithmes génétiques sont des techniques d'optimisation pour les fonctions multi-modales. Ces algorithmes essaient de mimer les mécanismes mis en jeu par la théorie de l'évolution neo-darwinienne [Goldberg, 1989].
Beaucoup de recherches sont actuellement faites sur ces nouvelles techniques d'optimisation, et seules les bases de ces algorithmes seront données.

En fait l'idée principale des algorithmes génétiques est de faire une recherche sur un grand nombre de solutions en même temps, et, grâce à des mécanismes appropriés, ces algorithmes améliorent de manière itérative ces solutions.

La première étape est de recenser l'ensemble des paramètres intervenant dans la recherche pour maximiser la fonction f (pour des fonctions à minimiser, il faut définir une autre fonction à maximiser).

Ensuite il faut trouver une représentation de tous ces paramètres de manière à les coder dans un individu. Ceci est l'étape du codage. Classiquement les algorithmes génétiques utilisent un codage binaire. C'est pourquoi ils ont beaucoup de mal à travailler avec des problèmes qui impliquent des variables continues.
Dans ce cas la solution la plus simple consiste à travailler comme en informatique, en allouant un nombre donné de bits pour représenter chaque nombre.

Par exemple si 8 bits sont utilisés pour représenter un nombre, il y aura 256 valeurs possibles.

Si plusieurs paramètres sont nécessaires, une solution S,encore appelée individu, sera simplement la concaténation de tous les paramètres. Ils peuvent être tous de natures différentes (variables continues codées sur x bits, variables binaires...).

Commence alors l'algorithme d'optimisation, qui va prendre en compte n solutions S_i différentes.
Ces n solutions forment une population.
A la première étape, il faut initialiser la recherche. En général les n solutions sont initialisées au hasard, sauf si plus d'information est disponible.

A chaque étape k seront évaluées les solutions S_i. Alors pour créer la population de l'étape k+1, des mécanismes de reproduction seront utilisés. L'idée principale est que les meilleures solutions sont plus proches de la solution optimale, et qu'il faut donc favoriser leur reproduction.
Il faut donc former des couples de parents. La sélection est aléatoire. Le mécanisme le plus simple est d'assigner à chaque solution S_i une probabilité proportionnelle à son évaluation $f(S_i)$.
Un individu S_i peut donc être choisi plusieurs fois.

Chaque couple va donner naissance à deux enfants, ou deux nouvelles solutions. Cette création utilise le mécanisme du cross-over. Le cross-over mélange les informations de chaque parent, comme indiqué figure A.1

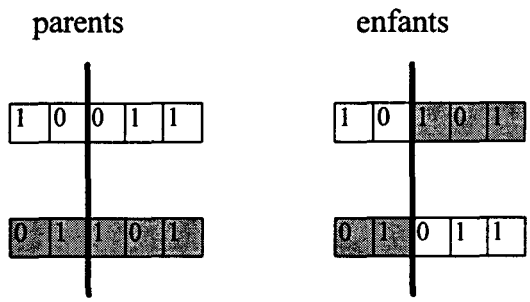

Fig. A.1. Mécanisme de cross-over utilisé pour la reproduction de parents de 5 bits.

Pour chaque couple, le cross-over n'est pas systématique. Il dépend d'une probabilité p_c, en général proche de 0.7. Si un cross-over doit être réalisé, l'endroit de la cassure est choisi aléatoirement.

La mutation est une étape utile pour empêcher que la population ne converge vers un maximum local. Elle dépend d'une probabilité, très faible, (de l'ordre de 1/1000 pour chaque bit), et si une mutation doit être effectuée sur un enfant, un bit est choisi aléatoirement puis inversé.

Ces deux étapes, avec l'étape de sélection, constituent le cœur de tout algorithme génétique. Ils sont donc très simples à mettre en œuvre, et une dizaine de lignes de code permet de réaliser un algorithme d'optimisation multi-modal.
L'algorithme est stoppé de manière empirique, par exemple après x itérations.

Deux mécanismes intéressants peuvent être ajoutés.
Le premier est simple, il s'agit de l'élitisme. En effet entre deux étapes, les parents procréent, puis meurent, étant remplacés par leurs enfants. Il peut donc arriver que la meilleure solution disparaisse. L'élitisme est un mécanisme qui assure que le meilleur parent (ou les x meilleurs parents, s'ils sont meilleurs que leurs enfants), sont automatiquement rajoutés à la jeune population des enfants (donc forcèment des enfants, les moins bons, disparaissent). Ce mécanisme est très simple à mettre en œuvre, mais un peu long car il faut comparer toutes les solutions.

Le rééchelonnement est le deuxième mécanisme intéressant. Lors de la convergence, lorsque une bonne partie de la population converge vers le maximum global, tous les individus ont à peu près la même évaluation $f(S_i)$, sauf les plus mauvais. Aussi le mécanisme de sélection, qui donne une probabilité de reproduction proportionnelle à $f(S_i)$, n'arrive plus à privilégier les meilleurs éléments. Au début cela était possible, car s'il y avait un élément excellent, et tous les autres mauvais, il était distingué très clairement.

Donc lors de la phase de convergence, la population risque de stagner.

Le rééchelonnement est un mécanisme qui amplifie les écarts entre les évaluations des individus, de telle sorte que les meilleurs soient toujours privilégiés. Beaucoup de mécanismes ont été développés. Le plus simple consiste à retrancher à toutes les évaluations celle du plus mauvais élément.
Par exemple, soit une population de quatre individus, $f(S_1)=0.9$, $f(S_2)=f(S_3)=0.8$, $f(S_4)=0.5$.

Sans rééchelonnement, les probabilités de sélection sont $P(S_1)=0.9/3=0.3$, $P(S_2)=P(S_3)=0.27$, et $P(S_4)=0.17$. Finalement l'individu S_1est peu différencié des autres.

Avec un échelonnement simple, le résultat est par exemple $P(S_1)=0.4/1=0.4$, $P(S_2)=P(S_3)=0.3$, et $P(S_4)=0$, et l'écart a été amplifié.

Les algorithmes génétiques sont des méthodes relativement robustes d'optimisation multi-modale [Grafensdtett, 1986]. Les recherches actuelles portent notamment sur des mécanismes nouveaux de reproductions, des changement de précision des codages pour les nombres réels, une adaptation de la taille de la population...

Le succès actuel de ces méthodes masque les anciennes approches de recherche d'optimum.
En effet, en 1970 déjà, McMurty avait donné les principes des algorithmes d'optimisation, et il avait définit la structure de toute méthode de recherche multi-modale. Cette structure se decomposait toujours en quatre étapes, avec une base stochastique:
1) une recherche aléatoire des solutions possibles (grand nombre de solutions en un pas)
2) une identification des principaux modes
3) des algorithmes de convergence rapide pour les fonctions restreintes aux domaines des modes principaux, fonctions qui sont alors uni-modale
4) des algorithmes extrêmement rapides sur le mode qui semble donner le maximum global

Les algorithmes génétiques semblent mélanger les phases 1 et 3, avec une initialisation aléatoire et une phase d'amélioration des solutions avec des mécanismes aléatoires. Ils ne semblent donc pas très performants. Le succès actuel de ces approches pourrait être du aux progrès de l'informatique, et au fait qu'ils sont très faciles d'emploi.
Bien meilleurs sont les algorithmes hybrides, qui mélangent algorithmes génétiques et algorithmes rapides lors de la convergence.

ANNEXE B

IDENTIFICATION DE MODELES DISCRETS

Cette annexe présente les outils utilisés pour manier le modèle identifié. D'abord sont donnés les algorithmes d'identification, puis les relations de passage entre modèle continu et modèle discret, et enfin les équations des filtres utilisés pour lisser le modèle identifié.

B.1 Algorithmes d'identification

Les données sont les suivantes. Il s'agit d'évaluer les paramètres du modèle représenté par l'équation:

$$\hat{y}_{n+k} = -a_1 y_{n+k-1} - \ldots - a_n y_k + b_1 u_{n+k-1} + \ldots + b_n u_k \tag{B.1}$$

avec n l'ordre du modèle, et $k \geq 1$.
y_{k+i} et u_{k+j} sont les informations disponibles à l'instant n+k.
L'objectif est de trouver les valeurs des coefficients supposés constants a_i et b_j.

Le problème peut se mettre sous la forme matricielle:

$$y_{n+k} = H_{n+k}\Theta + e_{n+k} \tag{B.2}$$

avec

$$H_{n+k} = [-y_{n+k-1} \ldots -y_k \ \ldots u_{n+k-1} \ldots u_k] \tag{B.3}$$

$$\Theta = [a_1 \ldots a_n \ b_1 \ldots b_n]^T$$

e_{n+k} représentant l'erreur entre le modèle et le système réel

B.1.2 Algorithme des moindres carrés à facteur d'oubli

Cet algorithme est un exemple classique d'algorithme d'identification adaptatif. L'ensemble des mesures est pris en compte, cependant celles-ci sont pondérées par un terme décroissant de manière exponentielle en fonction du passé. Ceci permet de suivre dans une certaine mesure les évolutions du système. Afin de garder un gain à peu près constant, il s'agit de la version avec trace constante qui est présentée.

Les équations sont les suivantes:
1) nouvelle mesure à l'instant n+k+1
2) erreur de prédiction

$$e_{n+k+1} = y_{n+k+1} - H_{n+k+1}\hat{\Theta}_{n+k} \tag{B.4}$$

3) mise à jour de l'estimation des paramètres $\hat{\Theta}$:

3.1) gain:

$$K_{n+k+1} = \frac{F_{n+k} H^T{}_{n+k+1}}{1 + H_{n+k+1} F_{n+k} H^T{}_{n+k+1}} \tag{B.5}$$

3.2) adaptation des paramètres:

$$\Theta_{n+k+1} = \Theta_{n+k} + K_{n+k+1} e_{n+k+1} \tag{B.6}$$

4) Adaptation de la matrice de pondération F

4.1) calcul de λ_1:

$$\lambda_{1,n+k} = \frac{1}{\mathrm{trace}(F_0)}\left[\mathrm{trace}(F_{n+k}) - \frac{\mathrm{trace}(F_{n+k}H^T{}_{n+k+1}H_{n+k+1}F_{n+k})}{\alpha + H_{n+k+1}F_{n+k}H^T{}_{n+k+1}}\right] \tag{B.7}$$

5.2) Adaptation de la matrice de pondération

$$F_{n+k+1} = \frac{1}{\lambda_{1,n+k+1}}\left[F_{n+k} - \frac{F_{n+k}H^T{}_{n+k+1}H_{n+k+1}F_{n+k}}{\alpha + H_{n+k+1}F_{n+k}H^T{}_{n+k+1}}\right] \tag{B.8}$$

5) retour à 1 pour le pas suivant

La matrice F est initialisée avec une matrice diagonale de grande valeur:
$F_0 = \frac{1}{\delta}I$
avec δ<<1 et θ est initialisé par une matrice nulle.
α est une constante permettant de privilégier plus ou moins les nouvelles mesures par rapport aux anciennes dans la mise à jour de la matrice F et est compris entre 0.5 (exclus) et 1.

B.1.3 Algorithmes des moindres carrés à fenêtre glissante

L'objectif de cet algorithme est de ne prendre en compte que les mesures qui ne sont pas trop anciennes, à l'exclusion des autres. Pour cela est défini un indice en deçà duquel les mesures ne sont plus intégrées dans les calculs. Le terme de fenêtre glissante est employé parce qu'à chaque étape une nouvelle mesure est ajoutée, et la plus ancienne est retirée.

Dans la suite la largeur de la fenêtre est notée L.

Les équations sont les suivantes:

1) mesures à l'instant n+k+1
nouvelle mesure: $H_{n+1+k} = [-y_{n+k-1}\ldots-y_k \ldots u_{n+k}\ldots u_k]$
ancienne mesure: $H_{n+1+k-L} = [-y_{n+k-1-L}\ldots-y_{k-L} \ldots u_{n+k-L}\ldots u_{k-L}]$
2) erreurs de prédiction

$$e_{n+k+1} = y_{n+k+1} - H_{n+k+1}\hat{\Theta}_k \tag{B.9}$$

$$e_{n+k+1-L} = y_{n+k+1-L} - H_{n+k+1-L}\hat{\Theta}_k \tag{B.10}$$

3) calcul des gains et matrices de pondérations

$$K_{n+k+1-L,n+k+1} = \frac{F_{n+k}H^T{}_{n+k+1}}{1 + H_{n+k+1}F_{n+k}H^T{}_{n+k+1}} \tag{B.11}$$

$$F_{n+k+1-L,n+k+1} = F_{n+k} - K_{n+k+1-L,n+k+1}H_{n+k+1}F_{n+k} \tag{B.12}$$

$$K_{n+k+1} = \frac{F_{n+k+1-L,n+k+1}H^T{}_{n+k+1-L}}{1 - H_{n+k+1-L}F_{n+k+1-L,n+k+1}H^T{}_{n+k+1-L}} \tag{B.13}$$

$$F_{n+k+1} = F_{n+k+1-L,n+k+1} + K_{n+k+1}H_{n+k+1-L}F_{n+k+1-L,n+k+1} \tag{B.14}$$

$$K_A = K_{n+k+1-L,n+k+1} + K_{n+k+1}H_{n+k+1-L}K_{n+k+1-L,n+k+1} \tag{B.15}$$

4) adaptation des paramètres:

$$\Theta_{n+k+1} = \Theta_{n+k} + K_A e_{n+k+1} - K_{n+k+1} e_{n+k+1-L} \tag{B.16}$$

5) retour à 1 pour le pas suivant

F et Θ sont initialisés de la même manière qu'avec l'algorithme à trace constante.

B.2.2.3 Algorithmes avec excitation permanente

Ces types d'algorithmes sont plus récents et ont été développés pour l'identification en ligne des systèmes bouclés. Dans ce cas il est probable que le système ne soit plus suffisamment excité et que l'algorithme d'identification ne puisse plus converger.

Les équations dans le cas des moindres carrés avec facteur d'oubli simple sont les suivantes. Un test est effectué à chaque étape pour savoir si l'excitation est nouvelle ou non. En fait la nouvelle mesure ne doit pas appartenir à l'espace vectoriel engendré par les précedentes. Ce test est réalisé par l'introduction d'une matrice A_{n+k}, qui est mise à jour à chaque étape.

1) nouvelle mesure à l'instant n+k+1
2) erreur de prédiction

$$e_{n+k+1} = y_{n+k+1} - H_{n+k+1}\hat{\Theta}_{n+k} \tag{B.17}$$

3) gain et pondération. Deux séries de mise à jour suivant le test sur l'excitation permanente:

3.1) SI $A_{n+k}H_{n+k+1} = 0$
(il y a excitation permanente)

$$K_{n+k+1} = \frac{B_{n+k}H^T{}_{n+k+1}}{\lambda + H_{n+k+1}B_{n+k}H^T{}_{n+k+1}} \tag{B.18}$$

$$A_{n+k+1} = \frac{A_{n+k}}{\lambda} \tag{B.19}$$

$$B_{n+k+1} = \frac{1}{\lambda}\left(B_{n+k} - \frac{B_{n+k}H^T{}_{n+k+1}H^T{}_{n+k+1}B_{n+k}}{\lambda + H_{n+k+1}B_{n+k}H^T{}_{n+k+1}}\right) \tag{B.20}$$

3.2) SI $A_{n+k}H_{n+k+1} \neq 0$
(il n'y a pas d'excitation permanente)

$$K_{n+k+1} = \frac{A_{n+k}H^T{}_{n+k+1}}{H_{n+k+1}A_{n+k}H^T{}_{n+k+1}} \tag{B.21}$$

$$\sigma = H_{n+k+1}A_{n+k}H^T{}_{n+k+1} \tag{B.22}$$

$$A_{n+k+1} = \frac{1}{\lambda}\left(A_{n+k} - \frac{A_{n+k}H^T{}_{n+k+1}H^T{}_{n+k+1}A_{n+k}}{\sigma}\right) \tag{B.23}$$

$$B_{n+k+1} = \frac{1}{\lambda}\left(B_{n+k-} \frac{A_{n+k}H^T{}_{n+k+1}H_{n+k+1}B_{n+k} + B_{n+k}H^T{}_{n+k+1}H^T{}_{n+k+1}A_{n+k}}{\sigma}\right) + \frac{A_{n+k}H^T{}_{n+k+1}H^T{}_{n+k+1}A_{n+k}(\lambda + H_{n+k+1}B_{n+k}H^T{}_{n+k+1})}{\lambda\sigma^2} \tag{B.24}$$

4) Mise à jour des paramètres

$$\Theta_{n+k+1} = \Theta_{n+k} + K_{n+k+1}e_{n+k+1} \tag{B.25}$$

5) retour à 1 pour le pas suivant

La matrice A est initialisée par une matrice Identité de taille nn, la matrice B par une matrice nulle de taille nn.
λ est un facteur d'oubli. Plus il est proche de 0.5, plus les anciennes mesures sont oubliées vite, mais plus la convergence est instable.

B.2. Conversion modèle continu modèle discret

L'architecture du contrôleur à multi-modèle avec un modèle adaptatif utilise à la fois des modèles continus et des modèles discrets. Il est notamment nécessaire d'utiliser des relations de passage pour comparer ces différents modèles lors de l'estimation des validités.

Le système continu est décrit par:

$$\begin{aligned} \dot{x} &= A_C x + B_C u \\ y &= Cx + Du \end{aligned} \tag{B.26}$$

Le système discret est décrit par:

$$\begin{aligned} x_{k+1} &= A_D x_k + B_D u_k \\ y_k &= Cx_k + Du_k \end{aligned} \tag{B.27}$$

Dans le sens Continu-Discret, les relations sont:

$$A_D = e^{A_C T} \tag{B.28}$$

$$B_D = A_C^{-1}(e^{A_C T} - I)B_C \tag{B.29}$$

avec T la période d'échantillonnage, et si A_C est supposée inversible. U(t) est en outre constant entre deux intervalles d'échantillonnage.

Dans le sens Discret-Continu, les relations sont:

$$A_C = \frac{1}{T}\ln(A_D) \tag{B.30}$$

$$B_C = (e^{A_C T} - I)^{-1} A_C B_D \tag{B.31}$$

Pour les simulations réalisées, les validités des modèles étaient toutes basées sur l'erreur de prédiction, même pour les modèles continus. Pour ces derniers, leurs équivalents discrets furent utilisés à cette étape. Les commandes ont été calculées à partir des modèles continus.

B.3 Filtres du modèle identifié

L'objectif de l'étape de filtrage est d'adoucir les évolutions des coefficients intervenant dans le modèle identifié, afin que la commande qui s'en déduit ne fasse pas diverger la commande globale.

Les filtres suivants incorporent tous des paramètres à régler. Toutefois il est apparu qu'un filtre léger laissait passer certaines perturbations, ce qui était utile pour enrichir le signal de commande. Une solution possible serait de modifier la structure de la commande, en ajoutant une partie indépendante, par exemple une séquence binaire pseudo aléatoire, ce qui permettrait d'enrichir le signal, mais cela serait contradictoire avec des objectifs de performances. Cette technique pourrait être testée, mais il apparaît de toute façon que seul un compromis pourra être obtenu.

Les filtres présentés sont de structure élementaire, et des filtres plus complexes conçus dans les régles de l'art pourraient bien evidemment être utilisés.

B.3.1 Structure des filtres

Un premier ordre a été retenu pour tous les filtres.
L'équation B.32 représente les coefficients du modèle identifié pour la commande à partir de ceux du modèle identifié non filtré.

$$\Theta_c(k+1) = \alpha_{k+1}\Theta_{id}(k+1) + (1-\alpha_{k+1})\Theta_c(k) \qquad \text{(B.32)}$$

α est un paramètre qui privilégie les nouvelles mesures par rapport aux anciennes. C'est sur son évaluation que diffèrent les filtres.

B.3.2 Evaluation du gain α

L'objectif est de lisser les évolutions du modèle identifié pour la commande. Ceci est relativement indépendant de la qualité du modèle. En effet de nombreuses simulations ont montré que, si l'erreur de prédiction définie par B.4, B.9 ou B.17 était faible pour une convergence des paramètres, elle pouvait l'être aussi même si le modèle identifié était faux, si le système était en régime permanent par exemple.

Aussi une nouvelle variable a été créée uniquement pour mesurer le fait que le modèle identifié était invariant localement ou non, indépendamment de sa qualité.
Cette variable est en fait basée sur la vitesse d'évolution de Θ_{id}. Plus Θ_{id} évolue vite, moins le modèle sera considéré stable.

Pour cela est d'abord défini un modèle filtré a priori M_{if}, correspondant aux L' derniers modèles M_{id}:

$$\Theta_{if}(k+1) = \frac{1}{L'}\sum_{i=1}^{L'}\Theta_{id}(k+1-i) \qquad \text{(B.33)}$$

Ensuite la vitesse d'évolution est calculée avec la norme du max:

$$vit = \max\left(\left|\frac{a_{0id} - a_{0if}}{a_{0id}}\right|, \ldots, \left|\frac{a_{nid} - a_{nif}}{a_{nid}}\right|, \left|\frac{b_{1id} - b_{1if}}{b_{1id}}\right|, \ldots, \left|\frac{b_{nid} - b_{nif}}{b_{nid}}\right|\right) \quad (B.34)$$

La largeur L' de la fenêtre va avoir une influence directe sur le filtre, puisque la vitesse du modèle instantané M_{id} sera plus ou moins importante par rapport au modèle filtré à priori M_{if}.

B.3.3 Filtre proche de la surveillance

Avec ce filtre, α_{k+1} vaut soit 0 si le modèle identifié varie beaucoup, soit 1 s'il ne varie pas beaucoup. Pour savoir si le modèle évolue vite ou non, l'utilisateur doit donner un seuil v_m au delà duquel le nouveau modèle identifié n'est plus pris en compte, et en deçà duquel le modèle identifié devient le nouveau modèle filtré.

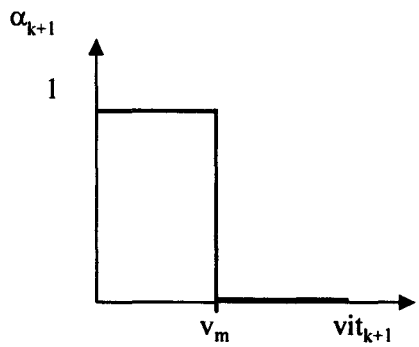

Fig. B.1. Evolutions du gain α en fonction de la vitesse d'évolution de Θ. Filtre proche de la surveillance.

B.3.4 Filtre linéaire

Avec ce filtre, α évolue de manière continu et linéaire entre 0 et 1 suivant la vitesse d'évolution (vit). Cependant comme les évolutions de Θ_{id} sont non linéaires, il peut arriver qu'une forte perturbation bouleverse Θ_c. Par exemple si a_{0c}=10, et que a_{0id}=90, avec α=0.1, le nouveau paramètre a_{0c} vaudra 19.9, c'est à dire qu'il aura doublé.
Aussi deux seuils seront ajoutés, respectivement v_{min} et v_{max} pour mettre plus vite à 1 ou à 0 le gain α.

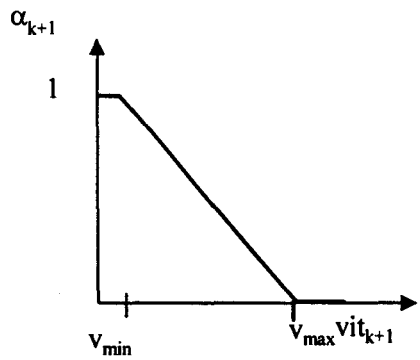

Fig. B.2. Evolutions du gain α en fonction de la vitesse d'évolution de Θ. Filtre relativement linéaire.

B.4 Modélisation avec les bases de fonctions

Les méthodes utilisant une base de fonctions sont peu fréquentes dans la littérature, comparativement aux méthodes adaptatives. Ce paragraphe a donc pour objectif de les présenter, cela grâce à la modélisation de signaux non stationnaires discrets. Plusieurs résultats de simulations seront donnés pour en montrer quelques caractéristiques.

B.4.1 Bases de fonctions

La détermination d'un modèle évolutif requiert une base de fonctions définie a priori. Ce paragraphe présente donc les principales bases de fonctions envisagées.

La première concerne la base des puissances du temps:

$$f_j(k) = \frac{T^j k^j}{j!} \tag{B.35}$$

T est un paramètre introduit pour faire le lien entre temps et indice. La forme orthogonalisée des polynomes (polynômes de Legendre) peut être utilisée pour améliorer la stabilité numérique.
La seconde base est celle des fonctions de Fourrier:

$$\begin{aligned} f_1(k) &= 1 \\ f_{2j}(k) &= \cos(jkT) \\ f_{2j+1}(k) &= \sin(jkT) \end{aligned} \tag{B.36}$$

Les bases de fonctions peuvent être théoriquement différentes pour les coefficients a_i et b_j. Grenier utilise des fonctions dites sphéroïdales aplaties, car elles permettent d'écrire exactement n'importe quel signal étant donné sa bande passante B et une durée T d'analyse.

B.4.2 Signal dans la base

Ce paragraphe donne quelques résultats lorsque le signal se décrit parfaitement dans la base de fonctions.
Tous les modèles sont du premier ordre:

$$y(k+1) = -ay(k) + bu(k) \tag{B.37}$$

B.4.2.1 Première simulation

Cette simulation montre que peu importe la complexité des évolutions des paramètres, s'ils peuvent être définis rigoureusement dans la base.
Les paramètres sont donnés par:

$$\begin{aligned} a(k) &= 1 + 0.5\sin(2kT) - 0.6\sin(3kT) + 1\cos(5kT) \\ b(k) &= 1 + 0.5\sin(2kT) - 0.3\cos(3kT) - 0.6\sin(5kT) + 2\sin(7kT) \end{aligned} \tag{B.38}$$

Dans le cas présent, T=1/50, mais s'il était pris égal à 1 l'identification serait encore effectuée, seulement toutes les courbes des résultats seraient illisibles.

L'identification a été effectuée avec la base de Fourrier, avec 15 fonctions (pour le sin(7k)). L'algorithme utilisé est celui des moindres carrés avec facteur d'oubli et trace constante. α=0.8.

Les figures B.3 à B.5 donnent les résultats (l'abscisse est indexée par l'indice k).

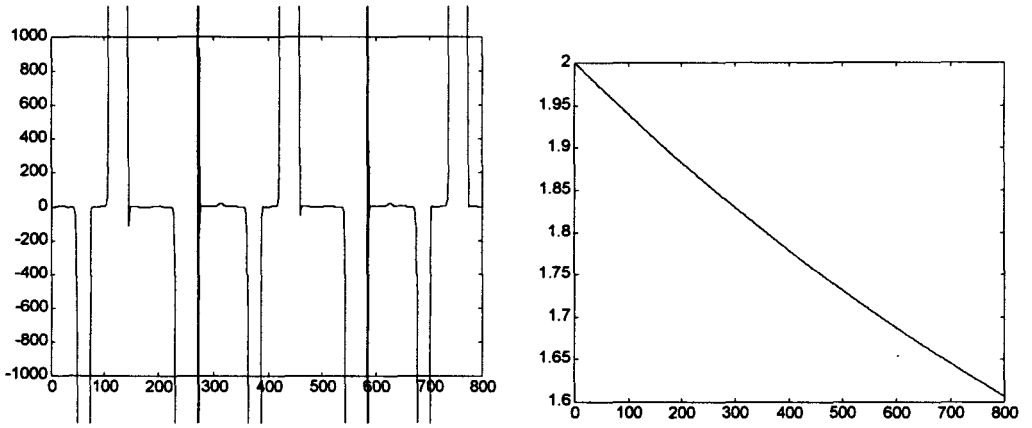

Fig. B.3. y(k) et u(k) pour le modèle correspondants à B.38.

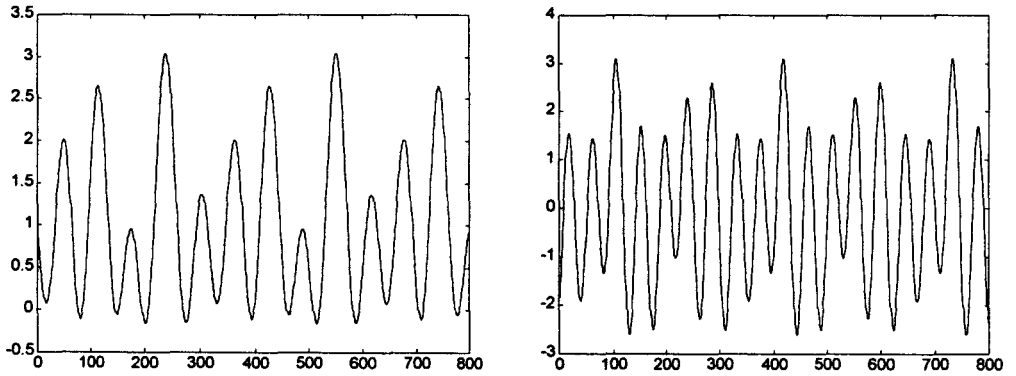

Fig. B.4. a(k) et b(k) réels pour le modèle correspondants à B.38.

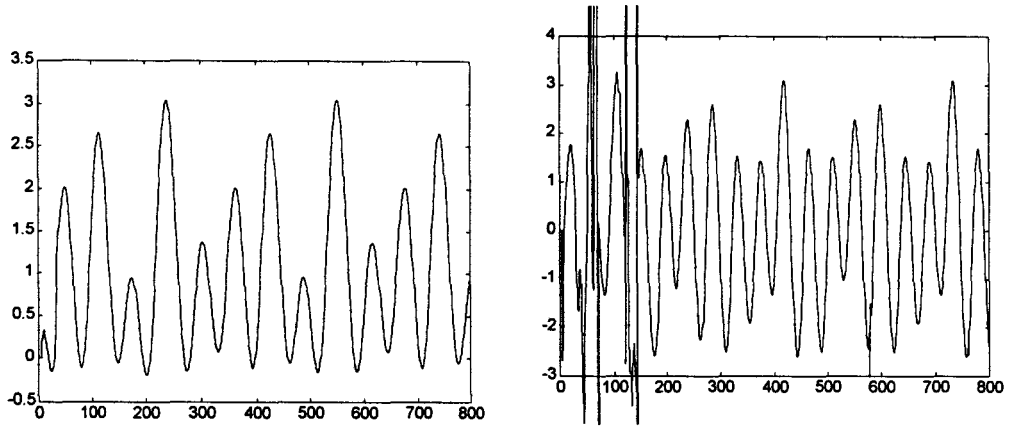

Fig. B.5. a(k) et b(k) estimés pour le modèle correspondant à B.38.

Le système est donc parfaitement modélisé. Les erreurs relatives entre les paramètres et leurs estimations sont très faibles (autour de 10^{-10}), ce qui s'apparente à un bruit numérique. Les écarts visibles sont plus dus à la méthode d'identification qu'au modèle évolutif.

B.4.2.2 Seconde simulation, base petite

L'objet de cette simulation et de la suivante est de montrer le problème du dimensionnent de la base de fonctions.

Les paramètres du modèle sont:

$$a(k) = 0.5 + 0.5\sin(kT)$$
$$b(k) = 0.5 + 0.5\cos(kT) \qquad \text{(B.39)}$$

avec T=1/10.

Les figures B.6 à B.8 donnent respectivement le signal et son entrée, et les paramètres et leurs estimations, avec une base de Fourrier de trois fonctions. L'algorithme en lui même n'a pas changé, il s'agit toujours de celui avec trace constante (α=0.8).

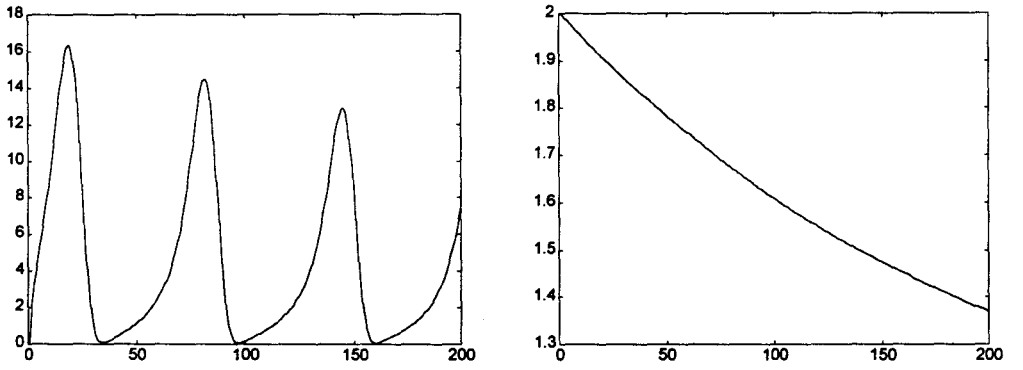

Fig. B.6. y(k) et u(k) pour le modèle correspondants à B.39.

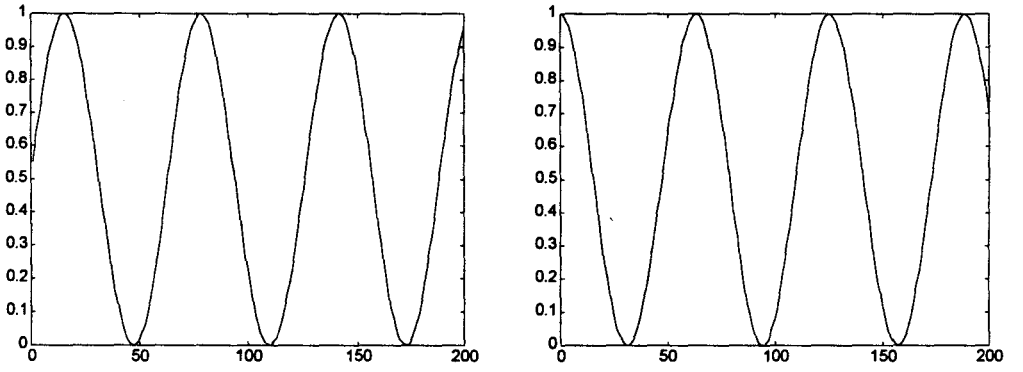

Fig. B.7. a(k) et b(k) pour le modèle correspondants à B.40.

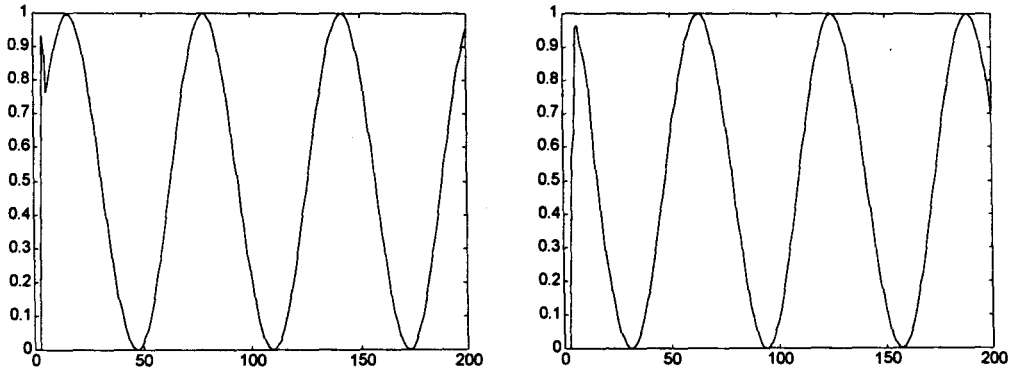

Fig. B.8. a(k) et b(k) estimés pour le modèle correspondants à B.40, petite base.

Il peut être intéressant de voir à l'intérieur même du modèle évolutif, les paramètres Θ_{ij} (pour un premier ordre, il y a seulement le coefficient a et le coefficient b). Les figures B.9 à B.11 donnent les valeurs des paramètres a_i et b_j qui sont les coefficients de projection des paramètres a(k) et b(k) dans la base de fonctions, et qui sont reliés par (c'est l'équation 6.9, à la base des modèles évolutifs):

$$a(k) = \sum_{j=1}^{m} a_j(k) f_j(k)$$

$$b(k) = \sum_{j=1}^{m} b_j(k) f_j(k)$$

(pour un premier ordre).
Ainsi le coefficient a_1, avec la base de Fourrier, représente la composante continue du paramètre a(k), et le coefficient a_2 le coefficient du terme cos(kT) intervenant dans la décomposition.

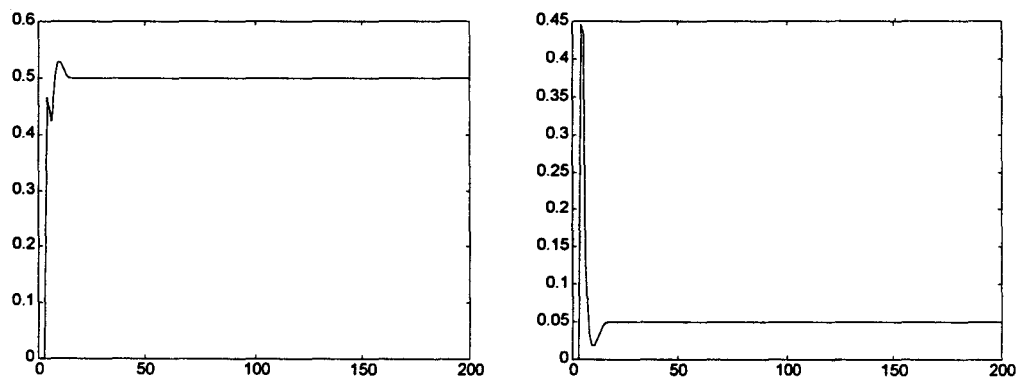

Fig. B.9. a_1 et a_2 pour décomposer a(k) (modèle donné par B.40).

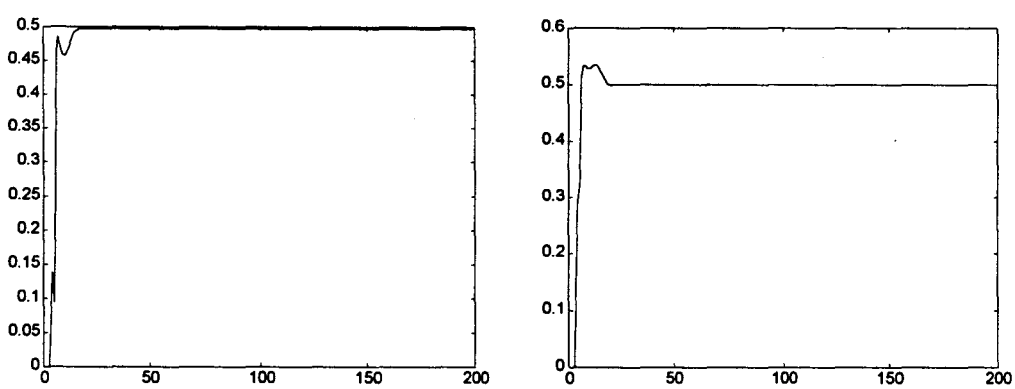

Fig. B.10. a_3 et b_1 pour reconstruire a(k) et b(k) (modèle donné par B.40).

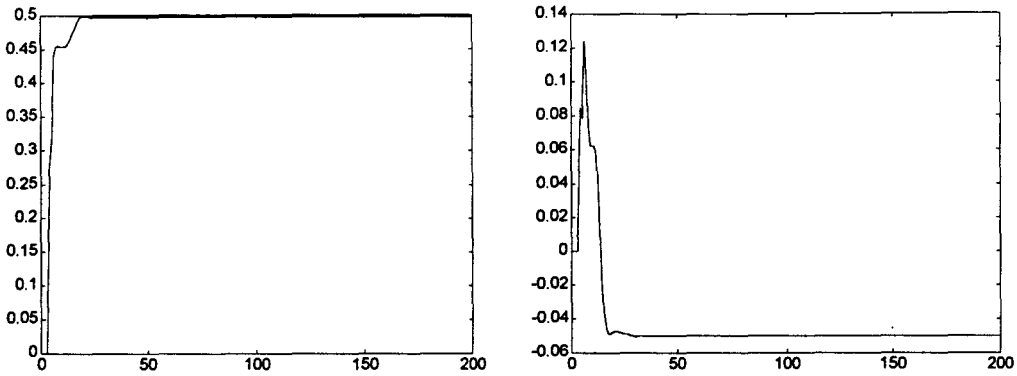

Fig. B.11. b_2 et b_3 pour reconstruire b(k) (modèle donné par B.40).

L'algorithme d'identification converge donc en 20 échantillons à peu près. Les coefficients sont retrouvés, cependant il y a un biais important sur les coefficients nuls (il est de 1 pour mille pour les coefficients non nuls). Comme va le montrer la simulation suivante, ce biais va provoquer de gros problèmes si la base est trop grande.

B.4.2.3 Seconde simulation, base trop grande

Cette expérience reprend les conditions précédentes, mais la base de Fourrier comprend cette fois 15 fonctions. La figure B.12 donne a(k) et b(k) estimés, et les figures B.13 à B.15 donnent les premiers coefficients a_j intervenant dans a(k).

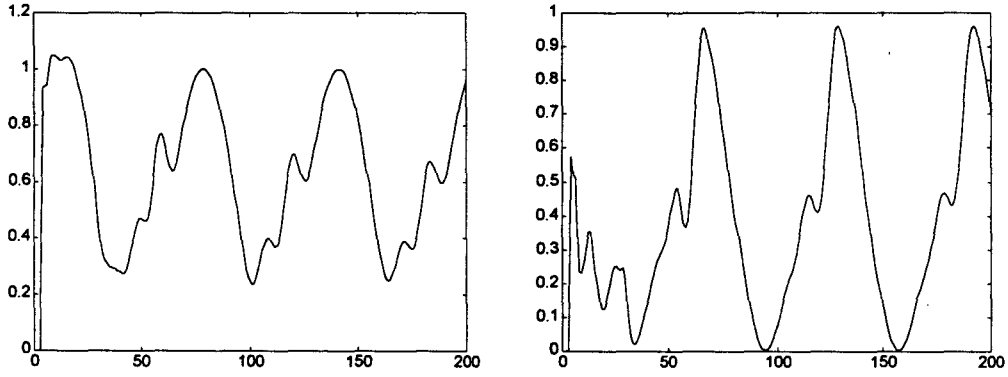

Fig. B.12. a(k) et b(k) estimés (modèle donné par B.40, grande base).

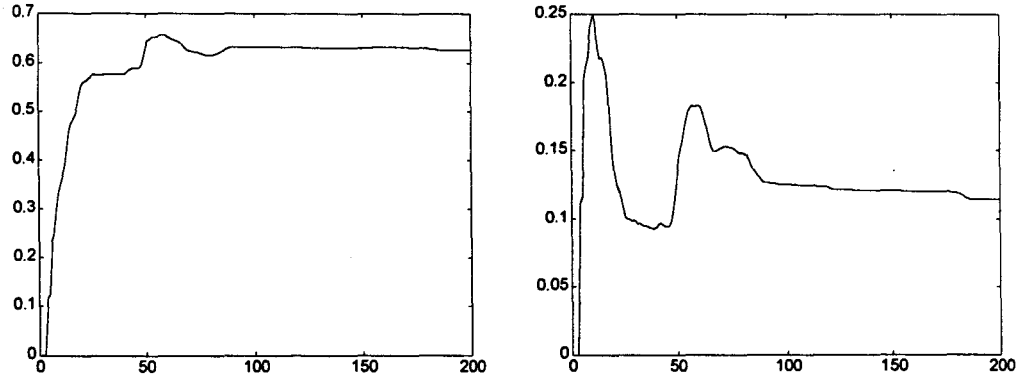

Fig. B.13. a_1 et a_2 pour décomposer a(k) (modèle donné par B.40, grande base).

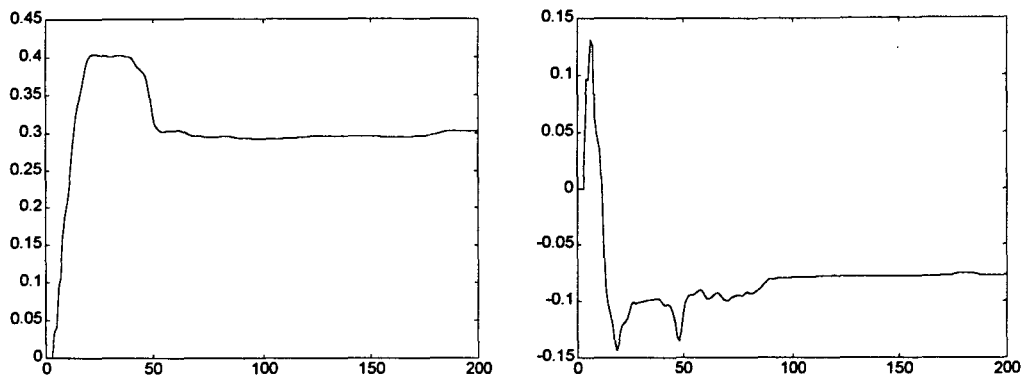

Fig. B.14. a_3 et a_4 pour reconstruire a(k) (modèle donné par B.40, grande base).

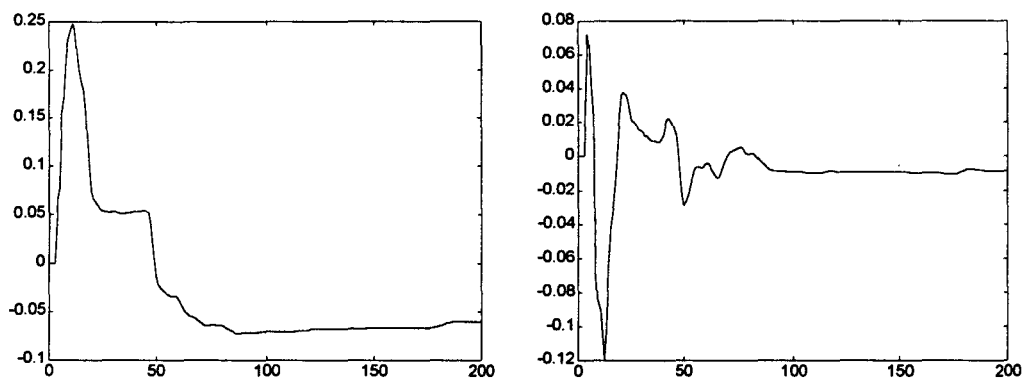

Fig. B.15. a_5 et a_6 pour reconstruire a(k) (modèle donné par B.40, grande base).

Le modèle ne correspond donc plus vraiment au système, même si l'allure générale d'évolution des paramètres est obtenue. La réponse vient de l'analyse des coefficients a_i donnés (les courbes sont similaires pour a_7 à a_{15} et b_1 à b_{15}). Les coefficients non nuls sont en fait perturbés par les coefficients non nuls et biaisés.

Cette simulation montre donc que la taille de la base de fonctions soulève un problème, non seulement en coût, mais aussi en surparametrisant les modèles. De manière générale il faudrait pouvoir résoudre ce problème, ou au moins le diminuer. Pour cela il faudrait recourir à des méthodes d'identification non biaisées, en blanchissant le résidu (comme les moindres carrés généralisés), ou en décorrélant le résidu et les mesures (variables instrumentales).

Une simulation fut effectuée avec les moindres carrés étendus, avec le modèle suivant:

$$y_{k+1} = -a(k)y_k+b(k)u_k+c(k)\varepsilon_k \qquad \text{(B.40)}$$

avec ε_k un bruit approché par l'erreur de prédiction a posteriori.
Ce modèle induisait une augmentation de 50% du nombre de coefficients, mais l'identification n'en fut que faiblement améliorée.

B.4.3 Signal en dehors de la base

L'ensemble des fonctions continues étant de dimension infinie, il n'est pas possible de concevoir une base qui permette de décrire exactement tous les signaux. Il y aurait de toute façon le problème du temps de calcul.

Il est donc nécessaire donc recourir à des bases de dimension réduites et donc incomplètes.
L'identification avec un modèle évolutif est alors nettement plus complexe que précédemment. En effet, les modèles et le système ne pouvant jamais être identiques, il y aura toujours des erreurs.

L'objectif des modèles évolutifs est de rendre suffisamment petites ces erreurs d'approximation pour que l'algorithme d'identification puisse converger localement.

De manière générale, sans savoir si le signal est dans la base, il faudra donc définir une base de fonctions suffisamment complète, et un algorithme adaptatif qui puisse permettre l'adaptation des paramètres lorsque les erreurs d'approximation deviennent trop importantes.

L'étude d'un tel algorithme (base+algorithme) apparaît délicate. De très nombreuses simulations furent effectuées sur des signaux discrets.

Le principal problème concerne la base de fonction et sa taille. En ligne, avec l'algorithme des moindres carrés, il est difficile de savoir quelle est la meilleure combinaison.
Grenier cite un auteur qui, à partir d'un algorithme utilisant le maximum de vraisemblance, permet d'obtenir la base adéquate, sa taille, et l'ordre du système. Il fait notamment appel aux critéres d'Akaike:
le FPE (pour Final Prediction Error), et le AIC (pour Akaike Information Criterion).

$$FPE = E_p\left(\frac{N+p-1}{N-p-1}\right) \tag{B.41}$$

$$AIC = -2\mathrm{Log}(\text{Maximum de vraisemblance}) + 2p \tag{B.42}$$

avec N le nombre d'échantillons, p l'ordre choisi et E_p la somme totale des erreurs de prédiction.

Cependant l'algorithme du maximum de vraisemblance, bien que très puissant, n'est pas un algorithme récursif (des algorithmes récursifs existent, mais ils ont été définis au prix de simplifications). Il semble difficile de pouvoir donner des critéres performants avec une identification en ligne. Les pages suivantes donnent quelques résultats.

B.4.3.1 Signal un peu hors de la base

Cet exemple aborde un système du premier ordre pour lequel les paramètres sont légèrement en dehors de la base.
Les coefficients sont:

$$a(k) = 1 + 0.3\sin T(1.5k) - 0.5\cos T(3k)$$

$$b(k) = 1 - 0.5\sin T(4.5k) + \cos Tk \tag{B.43}$$

En fait ces deux paramètres font intervenir des coefficients qui, bien qu'ils soient dans la même plage de fréquence que la base, ne se projettent pas exactement dans celle-ci. Les fréquences utilisées, i.e. 1.5 et 4.5, sont celles qui soulèvent le plus de problème, car leur distance au coefficients de la base sont les plus importantes. Par exemple pour sinT(4.1k), les erreurs sont plus faibles.

L'algorithme d'identification est à facteur d'oubli avec trace constante. $\alpha = 0.8$. $T = 1/100$. Les figures B.16 à B.18 donnent les évolutions du système, des paramètres et de leur estimés.

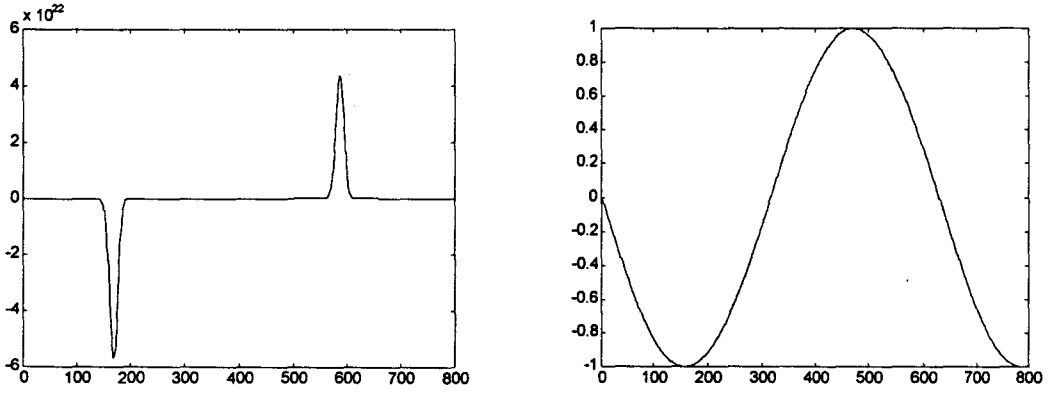

Fig. B.16. y(k) et u(k) pour les coefficients donnés par B.43.

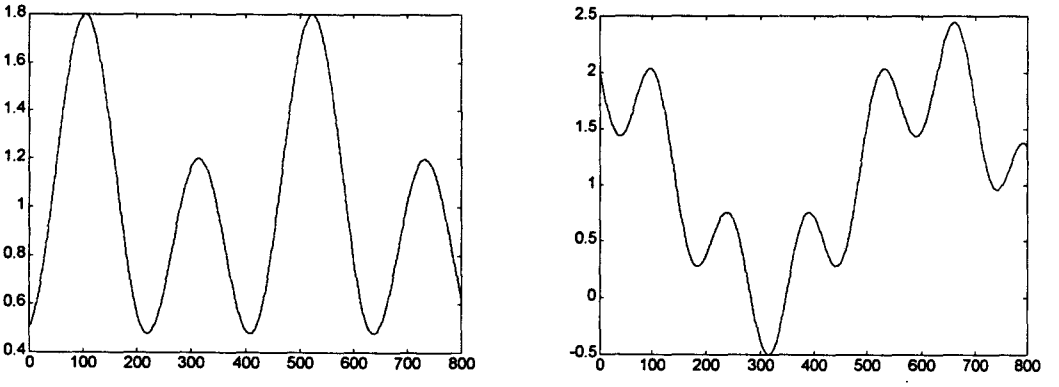

Fig. B.17. a(k) et b(k) donnés par B.43.

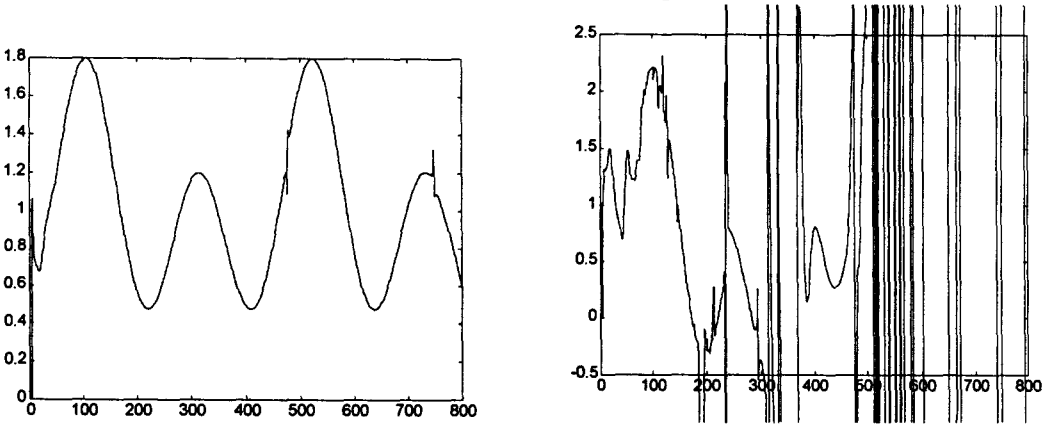

Fig. B.18. a(k) et b(k) estimés.

Le signal a(k) est bien mieux reconstruit que le signal b(k). Une explication à ce phénomène pourrait être que la partie du signal a(k) qui n'est pas dans la base (i.e. sinT1.5k) est à basse

fréquence par rapport au reste du signal (cosT3k), alors que pour b(k) c'est l'inverse. Une autre possibilité fait plutôt appel à la puissance des signaux y(k) et u(k) qui excitent a(k) et b(k) respectivement. En effet y(k) évolue dans une très grande gamme d'amplitude, contrairement à u(k), qui est compris entre -1 et 1.

Ainsi, pour k≈300 et k≈600, l'identification de b(k) est très mauvaise, alors que u(k) est voisin de 0. Pour k>600, l'algorithme n'arrive plus à rattraper les erreurs commises, et ne converge plus du tout.

Pour confirmer cette hypothèse, deux autres simulations ont été effectuées.
La première consistait à augmenter l'amplitude du signal u(k). Les figures B.19 à B.20 donnent les résultats.

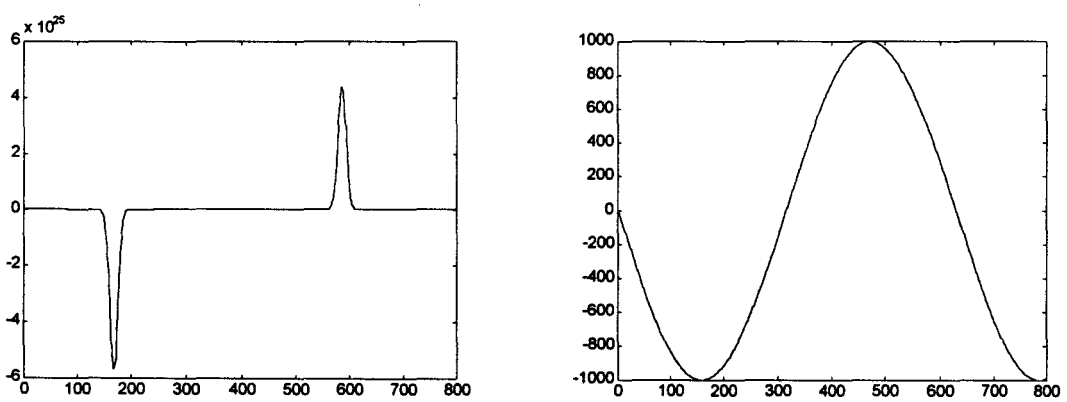

Fig. B.19. y(k) et u(k) pour les coefficients donnés par B.43.

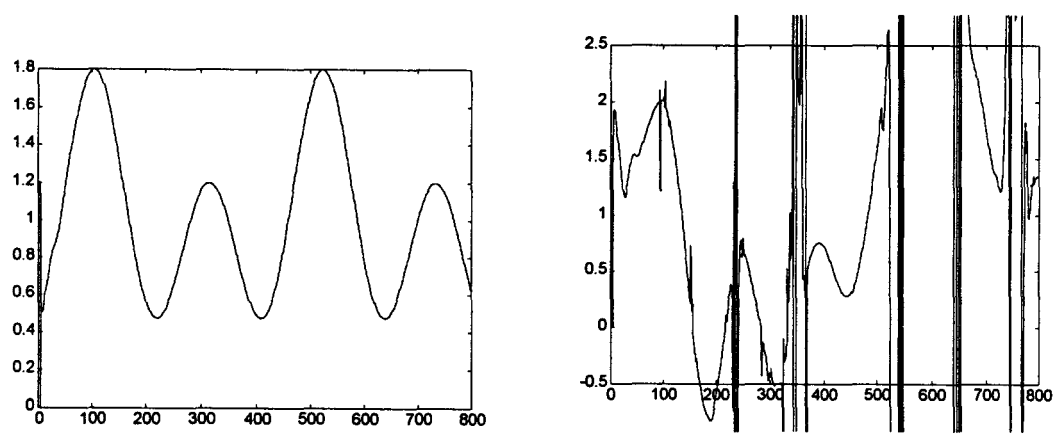

Fig. B.20. a(k) et b(k) estimés.

La figure B.21 donne les erreurs relatives pour les paramètres estimés, par exemple $100\left(a(k)-\hat{a}(k)\right)/a(k)$.

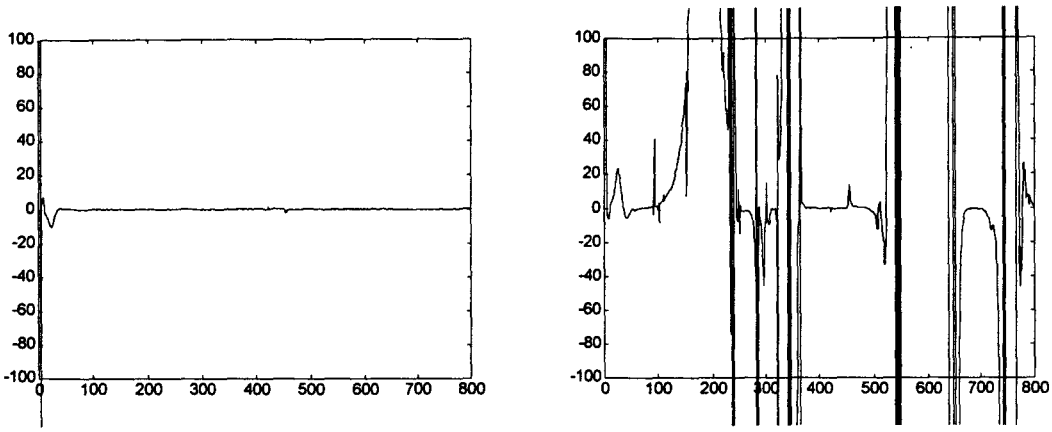

Fig. B.21. erreurs relatives pour a(k) et b(k) en %.

La figure B.22 montre le résidu normalisé défini par $100(y_{n+k}-H_{n+k}\Theta)/y_{n+k}$.

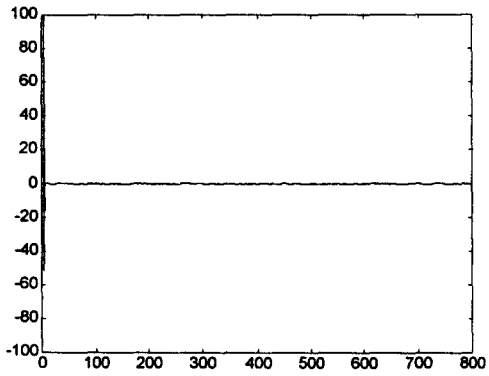

Fig. B.22. e(k) normalisé.

Avec une commande amplifiée, le terme b(k) est un peu mieux identifié, cependant il est encore fortement bruité. Le signal a(k) est quant à lui parfaitement reconstruit. L'erreur relative sur a(k) est voisine de 0. Pour b(k) elle varie entre des zones de convergences locales, parce que l'identification n'a pas convergé. Bien que b(k) ne soit pas parfaitement identifié, le résidu e(k) normalisé est toujours très faible. Les simulations ont souvent montré que si e(k) tendait vers 0 lorsque l'identification avait abouti, e(k) pouvait aussi bien être voisin de zéro même si l'identification n'est pas complète. Ceci peut s'expliquer par l'influence d'autres paramètres, notamment les signaux u(k) et y(k). C'est pourquoi il serait très difficile de construire un critère de validité des modèles identifiés avec ce seul argument.

Pour revenir au problème d'identification de b(k), un dernier résultat est présenté, en augmentant la complexité du signal a(k) par contraste. Celui ci est maintenant donné par:

$$a(k) = 1 + 0.3\sin T(3.5k) - 0.5\cos T(3k) \qquad \text{(B.44)}$$

Les figures B.23 à B.27 donnent quelques résultats.

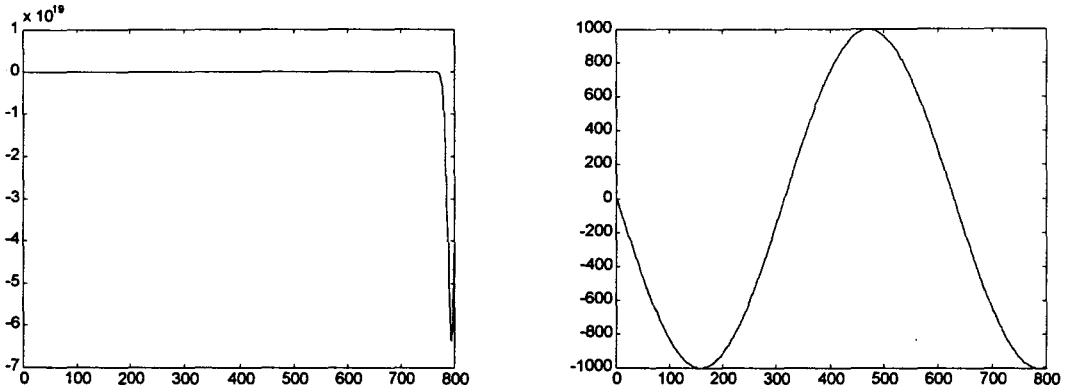

Fig. B.23. y(k) et u(k) pour le nouveau coefficient a(k) donné par B.44.

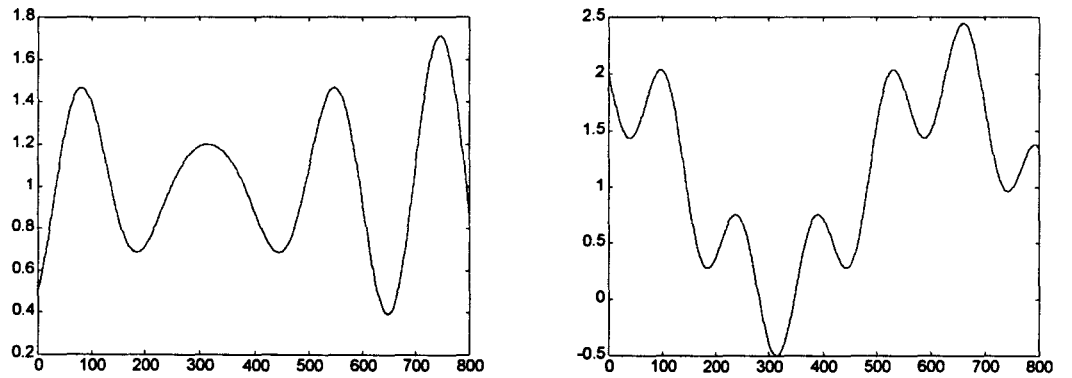

Fig. B.24. a(k) et b(k) (a(k) donné par B.44).

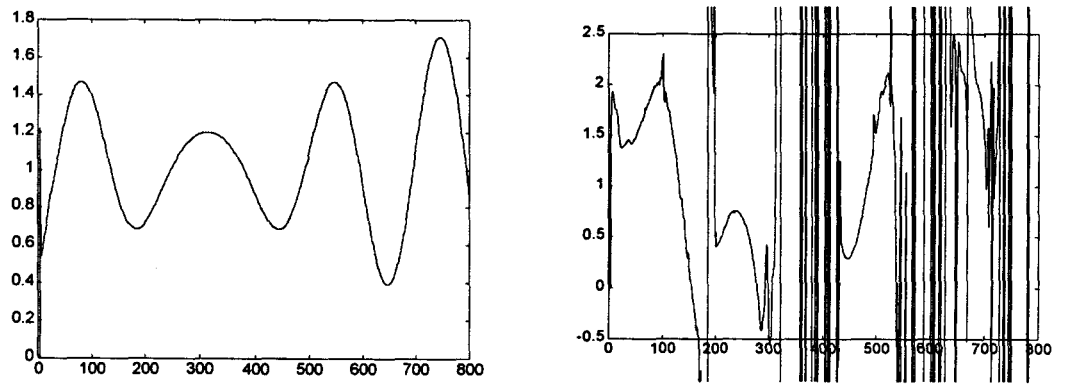

Fig. B.25. a(k) et b(k) estimés.

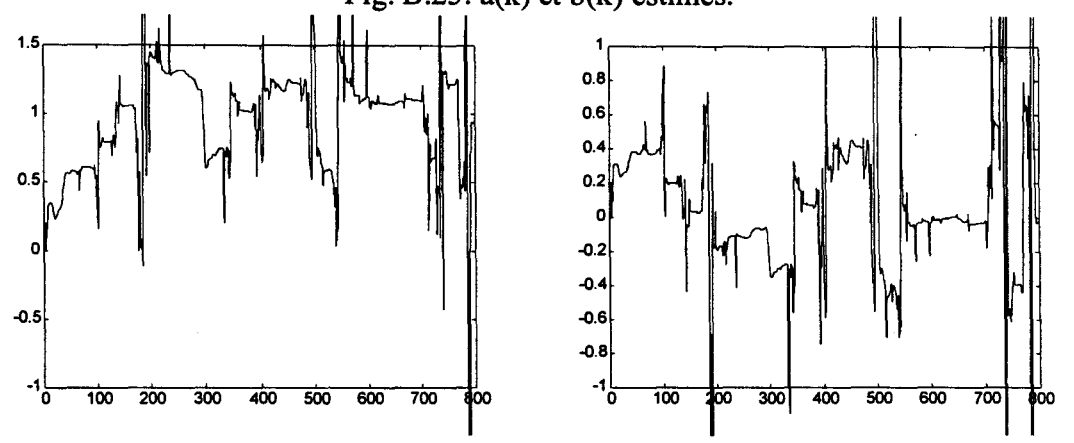

Fig. B.26. a_1 et a_2 utilisés pour décomposer a(k).

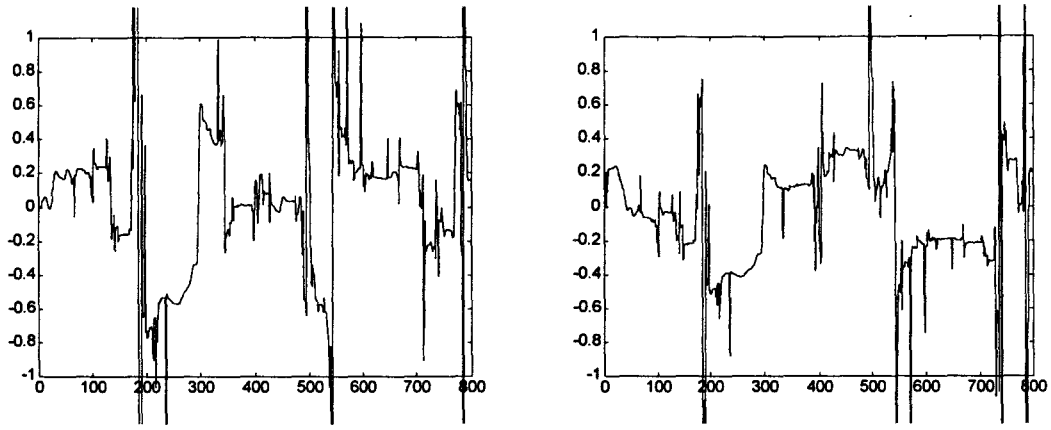

Fig. B.27. a_3 et a_4 utilisés pour reconstruire a(k).

Bien que plus complexe, le paramètre a(k) a parfaitement été reconstruit. Par contre l'identification de b(k) est fortement dégradée.
Le signal a(k) estimé est si bien reconstruit que cela apparaît étonnant, et il est intéressant d'examiner plus précisément chacun des coefficients a_j. Les figures B.26 et B.27 donnent les 4 premiers termes. Les autres sont similaires.
Ce qui est immédiatement visible est que ces coefficients élémentaires sont tous fortement bruités, au contraire du a(k) reconstruit. La sommation de ces termes par l'intermédiaire de la base de fonctions apparaît donc comme jouant le rôle d'un filtre. La seconde remarque est qu'aucun de ces coefficients n'est constant. Chacun de ces coefficients évolue plus ou moins vite. Cela est logique car sin (4.5t) n'est pas une combinaison linéaire constante des fonctions de la base de Fourrier. Par contre l'algorithme d'identification a réussi à trouver une combinaison avec des coefficients qui évoluent lentement.

Plusieurs autres simulations furent effectuées pour essayer améliorer l'identification de b(k). En simplifiant sa complexité, les résultats sont à peine meilleurs. Même si b(k) est parfaitement dans la base, les résultats ne sont pas aussi bons qu'avec un a(k) en dehors de la base.
Ceci est sûrement du à des problèmes numériques, tels que des mauvais conditionnements de matrices. Pour résoudre ces problèmes, il faudrait ajouter d'autres modules à l'algorithme principal d'identification, pour améliorer les traitement numériques ultérieurs.

B.4.3.2 Signal moyennement hors de la base

Cet exemple aborde le système du premier ordre avec des paramètres qui sont cette fois hors de la base. En effet avec les précédentes simulations, ceux -ci étaient en en partie masqués par des composantes dans la base:

$$a(k) = 0.5 + 0.5\sin T(2.5k + 4)$$

$$b(k) = 0.5 + 0.5\cos T(3.5k - 4) \qquad \text{(B.45)}$$

Avec une base de Fourrier de 10 fonctions, le problème ne réside pas dans les déphasages, mais dans les fréquences différentes, qui ne sont pas entières.
L'algorithme d'identification est celui à facteur d'oubli avec trace constante. α=0.8. T=1/100.
Les figures B.28 à B.31 donnent les évolutions du système, des paramètres et de leur estimés.

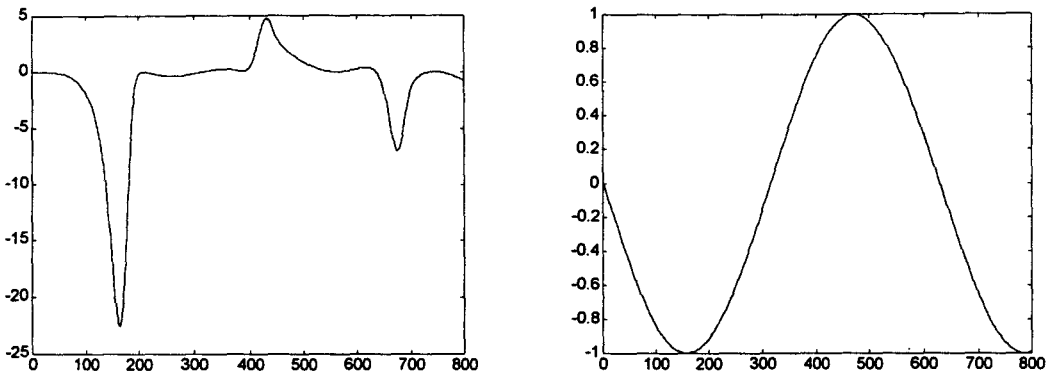

Fig. B.28. y(k) et u(k) avec le système défini par B.45.

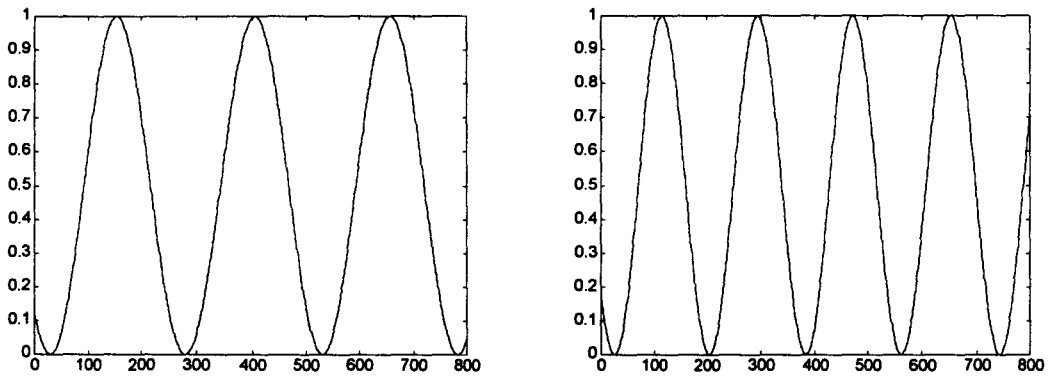

Fig. B.29. a(k) et b(k) défini par B.45.

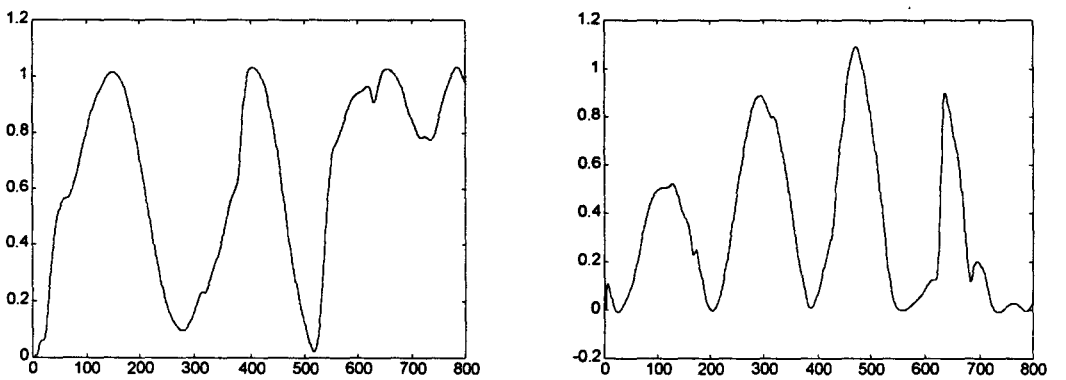

Fig. B.30. a(k) et b(k) estimés avec le système défini par B.45.

Comme les paramètres a(k) et b(k) ne sont pas dans la base, qui pourtant contient des fonctions très proches, il est impossible d'obtenir les bonnes valeurs. Cependant les estimés fournissent des évolutions qui ressemblent aux véritables, et notamment les modes principaux et leur fréquences sont bien obtenus. Il y a seulement un problème pour k>700, qui doit être du à une phase d'adaptation couplée avec un y(k) nul.

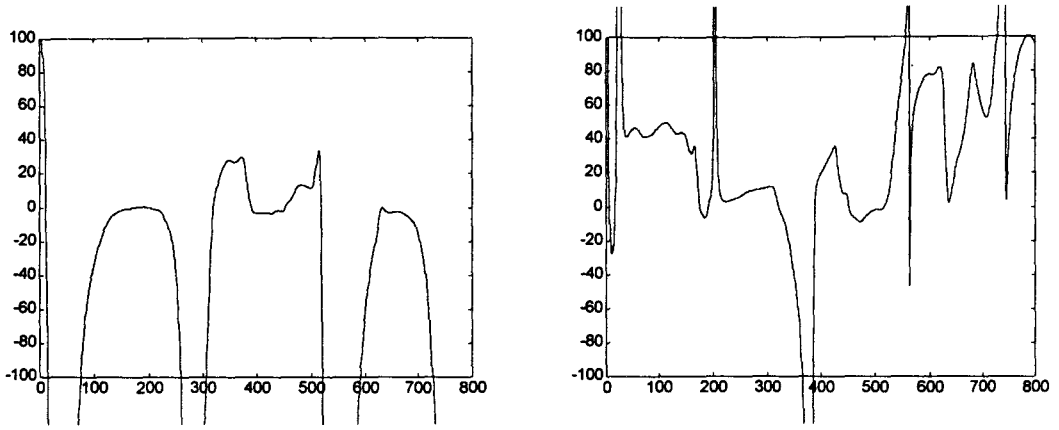

Fig. B.31. erreurs relatives pour a(k) et b(k) en %.

Il est possible de remarquer les zones de convergences pour lesquelles la base de fonctions donne une bonne approximation. Les divergences s'obtiennent lorsque les véritables signaux tendent vers 0.

La figure B.32 montre le résidu normalisé défini par $100(y_{n+k}-H_{n+k}\Theta)/y_{n+k}$.

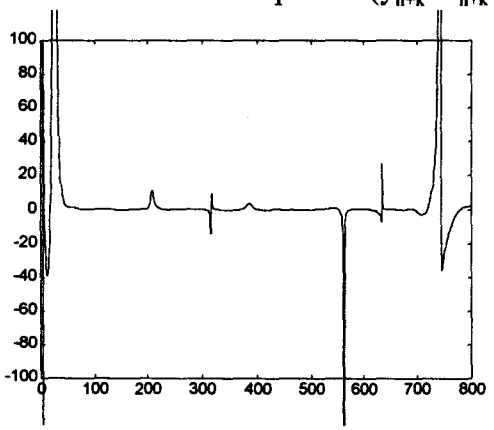

Fig. B.32. e(k) normalisé.

Comme cela a déjà été constaté, l'erreur de prédiction a priori est presque toujours très faible. Pour comparer différents modèles et différentes bases, ou différents ordres, cet indice est donc très mauvais.

B.4.3.3 Signal totalement hors de la base

Cette fois les paramètres a(k) et b(k) de l'exemple choisi ne seront pas du tout exprimables dans la base. En fait ce seront des fonctions périodiques décrites dans la base des polynômes.

Cet exemple va donc montrer qu'il est possible de projeter n'importe quelle fonction sur n'importe quelle base. Les résultats peuvent varier évidemment en qualité, mais si la base est bien choisie, les résultats ne devraient pas être trop mauvais.
En fait cette base de polynômes permet de retrouver la décomposition en série entière des fonctions continues.

Tout le monde connaît la célèbre formule

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots + (-1)^p \frac{x^{2p+1}}{(2p+1)!} + o(x^{2p+2})$$

C'était d'ailleurs un des exercices favoris dans les premiers cours de maths de devoir calculer de tels développements en série pour diverses fonctions. Il fallait notamment en préciser le rayon de convergence autour de $x_0=0$.

Ce type de formule rappelle donc que la base de polynômes permet de décrire un grand nombre de fonctions dans une base finie, avec une zone de convergence plus ou moins grande suivant la taille de la base.

Cette fois c'est l'algorithme d'identification qui va se charger des calculs.

Le système sera défini par:

$$a(k) = \sin Tk$$

$$b(k) = \cos Tk \qquad \text{(B.46)}$$

Les figures suivantes donnent les résultats obtenus avec l'algorithme à facteur d'oubli et trace constante (α=0.8), et 10 fonctions de la base des polynômes.

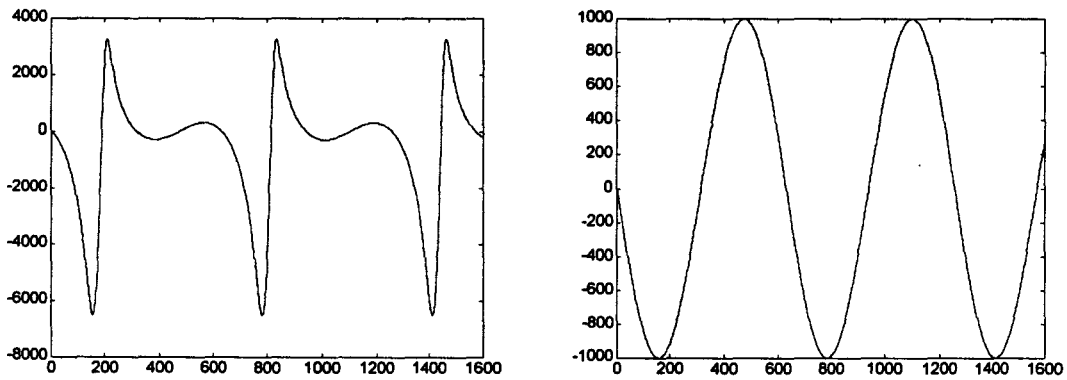

Fig. B.33. y(k) et u(k) avec le système défini par B.46.

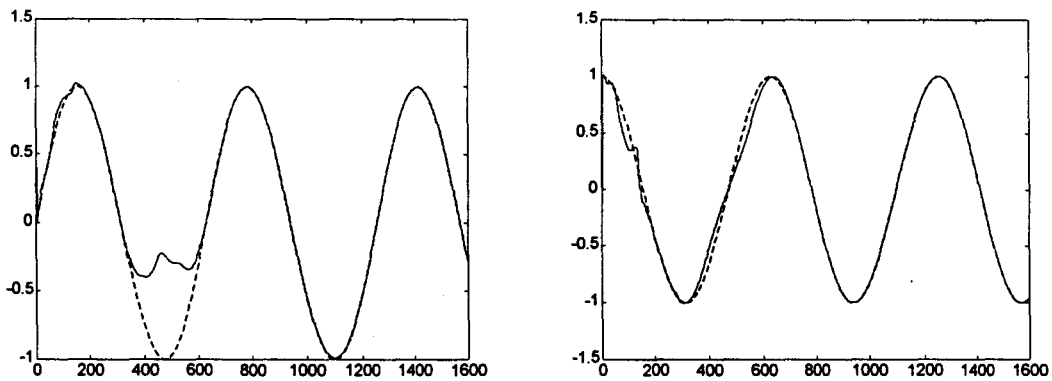

Fig. B.34 a(k) et b(k) défini par B.46 en '----', et leurs estimés en continu.

La simulation a été effectuée sur 1600 échantillons (T=1/100) au lieu de 800 pour vérifier que a(k) était bien reconstruit, à part l'erreur au voisinage de k=500.

L'identification a donc parfaitement fonctionné.

Pour conclure ces simulations, il faudrait comparer les résultats obtenus ici avec les résultats obtenus en utilisant uniquement des modèles linéaires, mais d'ordre plus élevé, pour compenser l'utilisation d'un grand nombre de paramètres supplémentaires avec la base de fonctions.

Un certain nombre de vérifications furent effectuées, mais les résultats ont toujours été décevants.

Ce terme lui même nécessite un commentaire. En effet pour juger les résultats, il faut un critère. Le critère de l'erreur de prédiction a priori s'est révélé inadapté, au moins sur les exemples étudiés, puisque les précédentes simulations ont montré qu'il n'était que faiblement corrélé avec la qualité du modèle.

Dans la littérature sont cités des méthodes qui permettent du juger la qualité de modèles d'ordres diffèrents, de structures différentes, mais avec un algorithme adaptatif basé sur le maximum de vraisemblance. Cela n'est pas facilement transposable avec la méthode des moindres carrés.

Aussi deux critères ad hoc ont été développés.

Le premier revient à forcer la traduction d'un modèle d'ordre élevé en un modèle d'ordre inférieur, en l'occurrence du premier ordre des exemples retenus

.

Pour passer du modèle d'ordre n, donné par (l'indice du haut se réfère à l'ordre du modèle)

$$\hat{y}_{n+k} = -a_1^n y_{n+k-1} - \ldots - a_n^n y_k + b_0^n u_{n+k} + \ldots + b_n^n u_k \tag{B.47}$$

au modèle d'ordre 1:

$$\hat{y}_{n+k} = -a_1^1 y_{n+k-1} + b_0^1 u_{n+k} \tag{B.48}$$

il suffit d'écrire

$$a_1^1(k) = a_1^n + a_2^n \frac{y_{n+k-2}}{y_{n+k-1}} + \ldots + a_n^n \frac{y_k}{y_{n+k-1}} \tag{B.49}$$

$$b_0^1(k) = b_0^n + b_1^n \frac{u_{n+k-1}}{u_{n+k}} + \ldots + b_n^n \frac{u_k}{u_{n+k}} \tag{B.50}$$

Le deuxième critère revient à étudier la stabilité du modèle obtenu, en calculant la vitesse d'évolution du vecteur Θ.

Il s'agit en fait de l'équation B.34 (avec la norme sup):

$$\text{vit} = \max\left(\left| \frac{a_{0id} - a_{0if}}{a_{0id}} \right|, \ldots, \left| \frac{a_{nid} - a_{nif}}{a_{nid}} \right|, \left| \frac{b_{1id} - b_{1if}}{b_{1id}} \right|, \ldots, \left| \frac{b_{nid} - b_{nif}}{b_{nid}} \right| \right) \tag{B.51}$$

L'indice id se réfère au modèle courant, et l'indice if au modèle précèdent

Pour cet exemple, des modèles linéaires d'ordre divers furent testés.

Pour une base de fonction de taille 10, les modèles testés furent d'ordres compris entre 1 et 12, ainsi les deux approches, linéaire et évolutive, utilisaient le même nombre de paramètres.

En général les résultats furent très similaires.
L'erreur de prédiction instantanée s'est toujours révélée faible. La vitesse d'adaptation vit donnée par B.51 est en moyenne faible. Cependant les coeffcients du modèle équivalent d'ordre 1, donnés par B.49 et B.50, ne correspondaient pas du tout aux vraies valeurs.

Les figures suivantes donnent par exemple les résultats avec un modèle linéaire d'ordre 10:

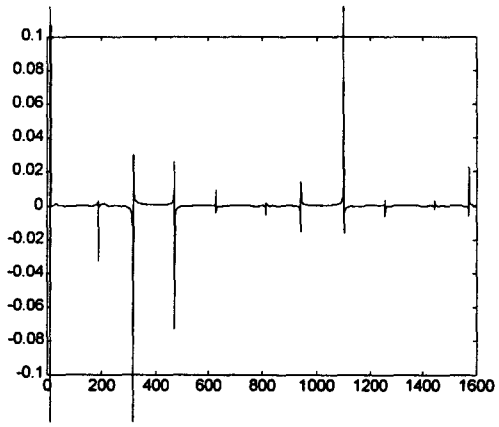

Fig. B.35. erreur de prédiction normalisée avec un modèle linéaire d'ordre 10.

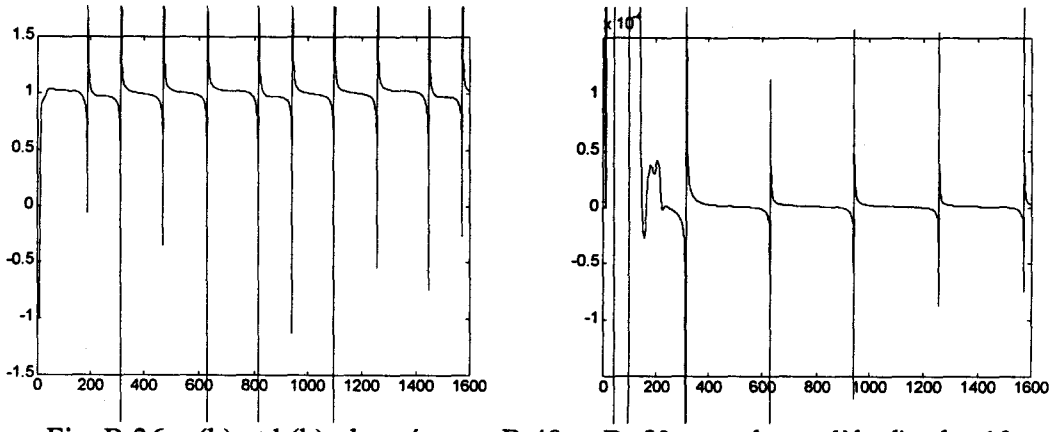

Fig. B.36. a(k) et b(k), données par B.49 et B. 50, pour le modèle d'ordre 10.

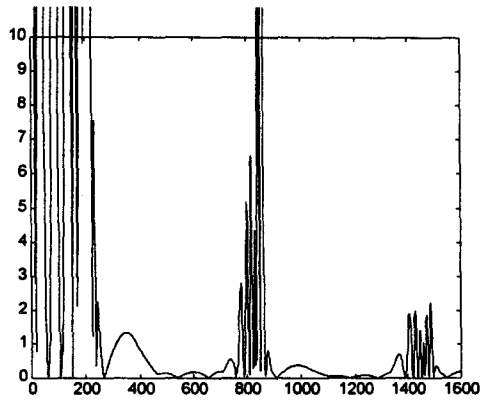

Fig. B.37. Vitesse d'évolution de Θ en % pour le modèle linéaire d'ordre 10.

Cet exemple est presque caricatural, car l'erreur de prédiction est inférieure en moyenne à 1/100 de pour-cent, alors que les coefficients a(k) et b(k) n'ont aucun sens. Les résultats obtenus avec un modèle linéaire d'ordre 1 furent du même ordre.

De manière plus générale, le choix de la structure d'un modèle relève de la propriété de discernabilité [Walter et Pronzato, 1994] des systèmes. Les systèmes discernables permettent en effet de tester plusieurs structures différentes et de les rejeter si elles sont mauvaises. Cette propriété est différente de l'identifiabilité, qui, étant donné la bonne structure, permet de garantir que les bons paramètres peuvent être obtenus.
Walter et Pronzato introduisent aussi la différence, pour un modèle donné, entre la capacité d'approximation, et la capacité de prédiction.

En résumé de ces comparaisons, il semble donc que les méthodes évolutives, par l'introduction de la base de fonctions, diffèrent complètement des méthodes purement linéaires.

B.4.4 Conclusion sur l'identification avec base de fonctions

Les algorithmes d'identification avec base de fonctions sont beaucoup moins utilisés que ceux purement adaptatifs, comme le filtre de Kalman Etendu. Les résultats précédents montrent qu'ils peuvent constituer une autre solution pour identifier les signaux non stationnaires.

Si les coefficients s'écrivent parfaitement dans la base, alors il n'y a aucun problème. Si les signaux sont différents, il faut mettre au point une base de fonctions alliée à un algorithme adaptatif pour pouvoir permettre à l'algorithme d'identification de converger localement. Les résultats sont dans ce cas moins bons que si les signaux s'écrivaient parfaitement dans la base de référence, mais ils sont suffisamment corrects pour donner une bonne idée des plages de variations des paramètres.

La mise au point d'un tel système d'identification est cependant complexe. Il ne faut pas que les termes non stationnaires évoluent trop vite, sinon l'algorithme d'identification n'aura pas le temps de converger. Ce dernier doit être suffisamment adaptatif pour s'accommoder des erreurs d'approximation et les corriger lorsqu'elles augmentent. Il ne doit cependant pas l'être trop, sinon il devient instable. Il existe donc beaucoup de paramètres à régler.

De plus, il faut relever la fréquence fondamentale de la base de fonctions de Fourrier. Il est ainsi très intéressant de lancer l'algorithme d'identification avec une base de Fourrier pour modéliser un signal qui évolue plus lentement que le terme le plus lent de la base, par exemple $\sin\tau 0.5k$. Les zones de convergence sont très petites, et il existe d'énormes zones où le signal reconstruit évolue trop vite, ou est retardé... Dans le cas général il faudrait mettre au point un système qui puisse estimer la période de variation des signaux non stationnaires, lorsqu'ils sont périodiques.

Un autre reproche qui peut être fait à l'encontre des méthodes évolutives est qu'elles requièrent beaucoup de paramètres, puisque un système d'ordre n sur une base m est traduit en un système stationnaire d'ordre $n \times m$. Ceci constitue un véritable problème. Cependant une telle difficulté est inhérente à la méthode. Le fait de pouvoir se ramener à un modèle linéaire à coefficients constants a pour contrepartie une importante augmentation de l'ordre. Toutefois

les exemples précédents ont montré que, parfois, alors que les méthodes purement adaptatives ne convergeaient pas, les méthodes évolutives permettaient une modélisation raisonable du processus.