République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Constantine 1
Faculté des Sciences de la Technologie

Thèse

Présentée par

Talbi Nesrine

Pour l'obtention du Diplôme de

Doctorat en Science

En **Electronique**

***Option* : Contrôle des Systèmes**

Thème

Conception des Systèmes d'Inférence Floue par des Approches Hybrides : Application pour la Commande et la Modélisation des Systèmes Nonlinéaires

Soutenue le 25 /02 / 2014 Devant le jury composé de :

Président :	**Salim Filali**	**Prof. Université de Constantine 1**
Rapporteur :	**Khaled Belarbi**	**Prof. Université de Constantine 1**
Examinateurs :	**Salim Labiod**	**Prof. Université de Jijel**
	Brahim Boutamina	**Maître de conférences A, Université de Constantine 1**
	Hamid Boubartakh	**Maître de conférences A, Université de Jijel**

Remerciements

L'ensemble des travaux présentés dans cette thèse a été effectué au Laboratoire d'Automatique et de Robotique (L.A.R.) de l'université de Constantine 1.

Je tiens à exprimer ma profonde gratitude à mon directeur de thèse Monsieur Khaled Belarbi, Professeur à l'Université de Constantine 1, pour la proposition du sujet de cette thèse et pour ses conseils et son soutient tout au long de ce travail. C'est l'occasion de lui remercier pour les connaissances qui nous a donné durant ma formation : Graduation & Post graduation au sein du département d'électronique de l'université de Constantine 1.

C'est un grand honneur & plaisir que Monsieur Salim Filali, Professeur à l'université de Constantine 1 préside le jury de ce modeste travail.

Je veux remercier d'une manière toute particulière Monsieur Salim Labiod, Professeur à l'université de Jijel pour l'intérêt qu'il a porté à ce travail en acceptant de l'examiner.

Mes remerciements s'adressent aussi à Monsieur Brahim Boutamina, Maître de Conférences classe A à l'université de Constantine 1, d'avoir accepté d'examiner cette thèse.

C'est un grand honneur que Monsieur Hamid Boubartakh, Maître de Conférences classe A à l'Université de Jijel soit membre de jury de cette thèse.

Enfin, mes remerciements vont à toute ma famille, qui m'a accompagné tout au long de mes études par son amour inconditionnel et son soutien constant.

Dédicaces

Je dédie cette thèse

A la mémoire de mon père

A ma très chère mère

A mon mari

A mes enfants

A toute ma famille,

A tous mes collègues.

ملخص

العمل المقدم في هذه الأطروحة يتمحور أساسا حول المجالات الرئيسية للنمذجة و المراقبة الغامضة من نوع Takagi-Sugeno رتبة صفر من أجل أنظمة ديناميكية معقدة و غير الخطية للغاية.

أولا ، تم اقتراح قانون التحكم باستخدام عدة خوارزميات هجينة على أساس مزيج من خوارزميات البحث الشاملة،الخوارزميات الجينية (GA) والتحسين من قبل سرب الجسيمات الأمثل(PSO)، و خوارزمية البحث المحلية، البحث تابو(TS) .التهجين هو الجمع بين خصائص هذه الأساليب (GA-TS و PSO-TS) لإيجاد حل جيد في الحد الأدنى من وقت الحساب، و ضمان الاستقرار والدقة و المتانة لأنظمة التحكم. لكل بنية تحكم ، قدمت أمثلة على التطبيقات محاكاة لتبرير صلاحية النهج المقترح. وتمت مقارنة هذه النتائج مع غيرها من التقنيات المذكورة في المراجع.

في المرحلة الثانية ، تم إجراء استبانة نماذج غامضة من نوع Takagi-Sugeno رتبة صفر من أجل تقريب للأنظمة غير الخطية باستخدام اثنين من الخوارزميات الهجينة (GA-TS) و (PSO-TS) . وقد تمت مقارنة نتائج النمذجة مع تلك الموجودة في المراجع من خلال معيار الأداء.

كلمات البحث: قاعدة من القواعد الغامضة ، البحث تابو، الخوارزميات الجينية ، سرب الجسيمات الأمثل ، وحدة التحكم الغامض ، نظام أحادي المتغير ، ونظام متعدد المتغيرات ، والنمذجة الغامضة .

Abstract

The work presented in this thesis revolves mainly around the main areas of modeling and fuzzy control of Takagi-Sugeno zero order for dynamic, complex and highly nonlinear systems.

First, a control law has been proposed using several hybrid algorithms based on the combination of global search algorithms, Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) and the local search algorithm the Tabu Search (TS). The hybridization consists to combine the characteristics of these methods (GA -TS and PSO-TS) in order to reap the benefits of success for a good solution in a reduced calculation time, all ensuring the stability, accuracy and robustness controlled systems. For each control structure, examples of simulation applications were presented to justify the validity of the proposed approaches. These results were compared with other techniques cited in references.

In the second phase, the identification of fuzzy models of Takagi-Sugeno zero-order approximation for the nonlinear systems was performed using the two hybrid algorithms (GA -TS) and (PSO-TS). The modeling results were compared with those existed in the literature through a performance criterion.

Keywords: Fuzzy rule base, Tabu Search, Genetic Algorithms, Particle Swarm Optimization, fuzzy controller, monovariable system, multivariable system, fuzzy modeling.

Résumé

Les travaux présentés dans cette thèse s'articulent, essentiellement, autour des principaux axes de la modélisation et de la commande floues du type Takagi-Sugeno d'ordre zéro pour des systèmes dynamiques, complexes et fortement nonlinéaires.

En premier lieu, une loi de commande a été proposée en utilisant plusieurs algorithmes hybrides basés sur la combinaison des algorithmes de recherche globale, les Algorithmes Génétiques (AG) et l'Optimisation par Essaim Particulaire (OEP), et l'algorithme de recherche locale, la recherche Tabou (RT). L'hybridation consiste à combiner les caractéristiques de ces méthodes (AG-RT et OEP-RT) pour tirer des avantages permettant d'aboutir à une bonne solution en un temps de calcul réduit, tous en garantissant la stabilité, la précision et la robustesse des systèmes commandés. Pour chaque structure de commande, des exemples d'applications en simulation ont été présentés pour justifier la validité des approches proposées. Ces résultats ont été comparés avec d'autres techniques citées en références.

En deuxième phase, l'identification des modèles flous de type Takagi-Sugeno d'ordre zéro pour l'approximation des systèmes non linéaires a été réalisée en utilisant les deux algorithmes hybrides (AG-RT) et (OEP-RT). Les résultats de modélisation ont été comparés avec ceux existés dans la littérature via un certain critère de performance.

***Mots-clés* :** Base de règles floues**,** recherche Tabou, Algorithmes Génétiques, Optimisation par Essaim Particulaire, contrôleur flou, système monovariable, système multivariable, modélisation floue.

Table des matières

Liste des travaux scientifiques réalisés

- **Publications internationales avec comité de lecture international**

[1] **N. Talbi and K. Belarbi**, "Fuzzy Takagi Sugeno System Optimization using Hybrid Particle Swarm Optimization and Tabu Search Learning Algorithm", *International Journal of Tomography and Simulation,Ceser Publications*, vol.22, n°1, 2013, pp. 4-16.

[2] **N. Talbi and K. Belarbi**, "Designing Fuzzy Controllers for a Class of MIMO Systems using Hybrid Particle Swarm Optimization and Tabu Search", *International Journal of Hybrid Intelligent Systems, IOS Press,* vol. 10, n°1,2013,pp. 1–9.

[3] **N. Talbi and K. Belarbi**, "Designing fuzzy rule base using hybrid elite genetic algorithm and tabu search: Application for control and modeling", *International Journal of Hybrid Intelligent Systems, IOS Press,* vol. 10, n°2,2013,pp. 205–214.

[4] **N. Talbi and K. Belarbi**, "Automatic Generation of fuzzy rule base by a hybrid Approach: Application to Control and modeling", *International Review of Automatic Control (I.RE.A.CO.)*, Vol. 5, N. 2, March 2012, pp. 284-291.

[5] **N. Talbi and K. Belarbi**, "Fast Hybrid PSO and Tabu Search Approach for Optimization of a Fuzzy Controller", *IJCSI International Journal of Computer Science Issues,* (*Electronic Journal*), Vol. 8, Issue 5, No 2, September 2011, pp. 215-219.

- **Communications internationales avec comité de lecture international**

[6] **N. Talbi and K. Belarbi**, "Reduced Fuzzy Rule Base Design using Hybrid Elite Genetic Algorithm and Tabu Search", *in Proc. IEEE, International conference on Hybrid Intelligent Systems (HIS), India,* 2012, pp. 496 – 501.

[7] **N. Talbi and K. Belarbi**, "Optimization of Fuzzy Controller using Hybrid Tabu Search and Particle Swarm Optimization", *in Proc. IEEE, 11th International Conference on Hybrid Intelligent Systems (HIS)*,Malaysia, 2011, pp. 561-565.

[8] N. Talbi and K. Belarbi, "Fuzzy Rule Base Optimization of Fuzzy Controller using Hybrid Tabu Search and Particle Swarm Optimization Learning Algorithm",*in Proc. IEEE, World Congress on Information and Communication Technologies (WICT)*, India, 2011, pp. 1139-1143.

[9] N. Talbi and K. Belarbi, "A Self Organized Fuzzy PD Controller using Tabu Search",*in Proc. IEEE*, *International Symposium onInnovations in Intelligent Systems and Applications (INISTA)*, Turkie, 2011, pp. 460-464.

[10] N. Talbi and K. Belarbi,, "Evolving Fuzzy Inference System by Tabu Search Algorithm and its application to control", *in Proc. IEEE, International Conference on Multimedia Computing andSystems (ICMCS)*, Moroc, 2011, pp. 1-6.

- **Communications internationales avec comité de lecture national**

[11] N. Talbi and K. Belarbi,, "Tuning Fuzzy PD controller using Tabu Search ", *International Conference on Electronics & Oil :From Theory to Applications (ICEO),* Ouargla, 2011, Algeria.

[12] N. Talbi and K. Belarbi,, "Evolving Fuzzy Inference System by Tabu Search Algorithm: an application to Inverted Pendulum Control", Béjaïa, 2010, Algeria.

Liste des tableaux

Liste des figures

Acronymes

La thèse contient un certain nombre d'acronymes, d'usage courant, employés volontairement le long de ce travail. Ces abréviations sont, explicitement, définies ci-dessous. Afin de faciliter la tâche du lecteur, le repérage de ces acronymes est établi par ordre alphabétique.

ACO	Ant Colony Optimization
AG	Algorithme Génétique
AG-RT	L'algorithme hybride des AG et la Recherche Tabou
AGE	Algorithme génétique avec élitisme
CF	Contrôleur flou
EGA	Elite Genetic Algorithm
EQM	Erreur Quadratique Moyenne
HGAPSO	Hybrid Genetic Algorithm and Particle Swarm Optimization
MIMO	Multi Input Multi Output
MISO	Multi Input Single Output
OEP	Optimisation par essaim particulaire
OEP-RT	L'algorithme hybride d'optimisation de l'OEP et la recherche Tabou
PSO-CREV	Particle Swarm Optimisation with Convergence guaranteed
RCACO	Rule base design using Continuous Ant Colony Optimization
REQM	Racine de l'Erreur Quadratique Moyenne
RT	Recherche Tabou
SAE	Somme des valeurs Absolues des Erreurs
SIF	Système d'Inférence Floue
SISO	Single Input Single Output
TS	Takagi-Sugeno

Introduction Générale

A. Etude préliminaire

La plupart des systèmes non linéaires sont modélisables sous des hypothèses parfois assez restrictives qui peuvent rendre difficiles la mise en œuvre des schémas de commande résultants. Il est donc nécessaire de prendre en compte toutes les informations imprécises et incertaines relatives au système. La théorie des sous-ensembles flous développée par Lotfi A. Zadeh en 1965 [1], a permis de traiter les imprécisions et les incertitudes. De nombreuses applications sont alors développées dans divers domaines, là où aucun modèle déterministe n'existe ou est difficile à obtenir.

L'avantage d'un système d'inférence floue (SIF) est que seules les connaissances du comportement du procédé à commander sont suffisantes pour la synthèse de la loi de commande et ils soulèvent un large intérêt, tant théorique que pratique, dans l'identification et la commande des processus complexes et non linéaires. Cela est dû au fait que d'une part les SIF n'exigent pas l'existence d'un modèle analytique du processus à contrôler et peu d'informations sont suffisantes pour mettre en œuvre la boucle de commande et d'autre part, les SIF sont des systèmes non linéaires et sont ainsi plus adaptés à la commande des processus non linéaires.

La conception d'un système d'inférence flou implique la détermination de sa structure et de ses paramètres. Dans beaucoup de cas, la structure est déterminée, empiriquement, en choisissant à priori le type de raisonnement approximatif linguistique ou relationnel, le nombre des sous-ensembles flous pour chaque variable d'entrée et en prenant toutes les combinaisons possibles pour construire la base de règles. Le réglage par essais successifs des nombreux paramètres est assez long et fastidieux. Diverses techniques d'autoréglage, d'optimisation et d'apprentissage des systèmes flous ont été développées. En 1990, les réseaux de neurones ont été utilisés pour le réglage de paramètres des fonctions d'appartenance des prémisses [2] et de conséquences de règles floues [3]-[7]. La première méthode d'autoréglage des paramètres de prémisses et de conséquences d'un SIF par la descente du gradient était par Nomura et *al*. [8], [9]. Une approche différente d'apprentissage supervisé des SIFs est présentée en 1993 par Jang où il propose un système neuro-flou adaptatif (ANFIS) [10]. Après, quelques améliorations ont été effectuées sur les algorithmes neuro-flou par la suite [11]-[13].

D'autres techniques de réglage des systèmes flous utilisent les métaheuristiques. Ces méthodes d'optimisation sont des heuristiques à large domaine d'application par opposition aux heuristiques simples développées pour résoudre un problème particulier. Plusieurs métaheuristiques s'inspirent des systèmes biologiques. Les algorithmes évolutionnaires (AE) dont les plus connus sont les algorithmes génétiques (AG) [14]. Leurs principes de base s'inspirent de la capacité de populations d'organismes vivants à s'adapter à leur environnement à l'aide de mécanismes de sélection et d'héritage génétique. Thrift [15] était le premier à décrire une méthode d'optimisation des règles floues par algorithme génétique standard, en utilisant trois bits pour coder chaque règle. A la même époque, Karr [16] propose une méthode permettant de faire intervenir dans le contrôleur flou à la fois des règles spécifiées par un expert humain et des règles optimisées par un algorithme génétique. Quelques améliorations ont été apportées par la suite [17]-[20].

Un peu plus tard, d'autres métaheuristiques ont été appliquées pour le réglage des systèmes d'inférence floue. Par exemple, l'optimisation par essaim particulaire (OEP), développée par Kennedy et Eberhart [21] a été utilisée pour optimiser dans [22] et [23] un réseau neuro-flou et dans [24] un contrôleur flou adaptatif.

B. Problématique

La structure de la base de règles floues est un facteur important qui affecte la performance de la commande et la modélisation. Lorsque le nombre de règles est grand, la précision est bonne mais le modèle flou devient complexe. D'où la nécessité de construction d'une base de règles simple, réduite et qui satisfait les performances désirées. L'objectif de ce travail est l'utilisation de métaheuristiques pour la conception de contrôleurs flous simples pour des systèmes dynamiques, complexes et fortement non linéaires en respectant les performances recherchées.

Une partie de cette thèse est consacrée à la conception de contrôleurs flous des systèmes dynamiques, complexes et fortement non linéaires en utilisant plusieurs algorithmes hybrides basés sur la combinaison des algorithmes de recherche globale, les AG et l'OEP, et l'algorithme de recherche locale, la recherche tabou. L'hybridation consiste à combiner les caractéristiques de ces méthodes (AG-RT et OEP-RT) pour tirer des avantages permettant d'aboutir à une bonne solution en un temps de calcul réduit, tous en garantissant la stabilité, la précision et la robustesse de ces systèmes.

Par ailleurs, l'identification et la modélisation floues à partir de données sont des outils efficaces pour approximer les systèmes non linéaires [25] grâce à leur capacité d'approximation universelle [26]. La modélisation floue est donc la représentation du comportement du système en utilisant les concepts de la logique floue. Le système flou de Takagi-Sugeno (TS) [27] est le plus utilisé pour modéliser les systèmes non linéaires. Un bon modèle doit être aussi simple que possible et avoir un domaine de validité aussi large que possible. Le problème d'identification en ligne d'un modèle flou pour l'approximation des systèmes non linéaires avec une base de règles floues auto-organisable, simple et réduite est donc posé.

L'autre partie de la thèse est donc consacrée à l'identification des modèles flous de type TS d'ordre zéro pour l'approximation des systèmes non linéaires. L'optimisation de paramètres de fonctions d'appartenances des entrées et les conclusions de règles floues est réalisé par les algorithmes hybrides (AG-RT) et (OEP-RT), en respectant les performances recherchées.

C. Objectifs

Les travaux présentés dans cette thèse ont pour objectifs de montrer les capacités de la combinaison des métaheuristiques pour l'optimisation des SIFs. Il s'agit alors de :

- Trouver une représentation paramétrique adéquate pour un SIF,
- Etablir un jeu de contraintes permettant de préserver la sémantique des règles floues tout le long du processus d'optimisation,
- Choisir l'algorithme d'optimisation correspondant aux données disponibles ainsi que le critère à optimiser.

Cette thèse propose différentes méthodes hybrides d'optimisation des systèmes d'inférence floue qui ont comme but de construire des SIFs capables d'une part de commander des systèmes complexes, dynamiques et fortement non linéaires et assurer leurs stabilités et robustesses et d'autre part modéliser les dynamiques des systèmes non linéaires.

D. Organisation du travail

Cette thèse est organisée comme suit :

Le premier chapitre rappelle les principales notions théoriques des systèmes d'inférence floue et les méthodes d'optimisation utilisées dans cette thèse. Après avoir introduit les concepts de base concernant la structure générale et les différents types de contrôleurs et modèles flous, les caractéristiques structurelles et paramétriques sont présentés. Par la suite, les métaheuristiques pour optimisation sont présentées. La méthode de recherche locale « la recherche Tabou », les algorithmes génétiques et la méthode d'optimisation par essaim particulaire y sont décrites en détails.

Le deuxième chapitre est consacré à la description d'une méthodologie d'optimisation d'une loi de commande floue simple et stable pour le contrôle des systèmes, fortement, non linéaires et complexes. La stratégie de conception proposée dans ce chapitre repose sur le concept d'hybridation des algorithmes génétiques et la recherche tabou. Le contrôleur flou optimisé est de type Takagi-Sugeno d'ordre zéro. Les détails de la mise en œuvre et des

exemples d'application pour les systèmes monovariables (le pendule inversé, la température d'un bain d'eau) et les systèmes multivariables (simulateur d'hélicoptère, le double pendule inversé) sont utilisés pour justifier la validité de l'approche proposée.

Le troisième chapitre de cette thèse décrit une autre méthode d'optimisation des contrôleurs flous qui se base sur l'hybridation de l'algorithme d'optimisation par essaim particulaire et la recherche Tabou. Notre objectif est de combiner les avantages des deux méthodes pour optimiser une base de règles floues pour un contrôleur flou de type Takagi-Sugeno d'ordre zéro. Le principe de fonctionnement de l'algorithme hybride et des exemples d'application similaires à ceux du chapitre précédent justifient la validité de l'algorithme proposé.

Enfin, le quatrième chapitre concerne une application des algorithmes hybrides décrits dans le deuxième et troisième chapitre pour la conception des modèles flous. La modélisation floue est un autre domaine d'application de la logique floue. Le problème d'identification floue consiste à choisir une forme de règles floues appropriée puis à concevoir des lois d'ajustement de paramètres afin que, pour une même entrée, la sortie du modèle flou identifié approche suffisamment de la sortie du procédé. Des exemples d'application sont utilisés pour justifier la validité des approches proposées dans le domaine de la modélisation floue.

Finalement, et pour clôturer cette thèse, quelques conclusions et discussions des perspectives et propositions envisagées pour poursuivre cette recherche sont présentées.

Chapitre 1

Les Systèmes d'Inférence Floue & Les Méthodes d'Optimisation

1.1 Introduction

Les outils intelligents sont de plus en plus utilisés dans la conception, la modélisation et la commande de systèmes complexes tels que les robots, les procédés biologiques, les véhicules routiers, etc. On entend par outils intelligents les techniques du *soft computing* tels que la logique floue, les réseaux de neurones artificiels et les métaheuristiques. La logique floue introduite par Zadeh dans les années soixante constitue un outil très puissant pour la représentation des termes et des connaissances vagues [1]. Les métaheuristiques, comprenant notamment la méthode du recuit simulé, les algorithmes évolutionnaires, la méthode de recherche tabou, les algorithmes de colonies de fourmis... sont apparues à partir des années 80. Ce sont des algorithmes d'optimisation de type stochastiques et progressant vers un optimum par échantillonnage d'une fonction objective dont le but est la résolution de problèmes d'optimisation difficile.

Depuis le début des années 1990, ces techniques intelligentes font leur entrée dans les sciences de l'ingénieur. Le but visé par les chercheurs est de concevoir des systèmes artificiels qui retiennent les mécanismes importants des systèmes naturels.

Ce chapitre est consacré à la description des éléments de base de la théorie des systèmes flous et des métaheuristiques. Ces deux concepts constituent la plateforme pour les différents travaux exposés dans cette thèse.

1.2 Les Systèmes d'Inférence Floue

1.2.1 Introduction

Les systèmes d'inférence floue (SIF) peuvent être considérés comme des systèmes logiques qui utilisent des règles linguistiques pour établir des relations entre leurs variables d'entrée et de sortie. Aujourd'hui, les applications des SIF sont très nombreuses outre la commande, ils sont largement utilisés pour la commande, la modélisation, le diagnostic et la reconnaissance de formes. Pour une meilleure compréhension, nous présentons quelques notions de base de ces systèmes ainsi que leurs types et leurs caractéristiques.

1.2.2 Les Ensembles Flous

La notion d'ensemble flou introduit un caractère graduel de l'appartenance d'un élément à un ensemble donné. Cela permet une meilleure représentation des termes et des connaissances vagues que nous, les humains, manipulons au quotidien. Mathématiquement, un ensemble flou A d'un univers de discours U, est caractérisé par une fonction d'appartenance, notée μ_A, à valeur dans l'intervalle [0,1] et qui associe à chaque élément x de U un degré d'appartenance $\mu_A(x)$ indiquant le niveau d'appartenance de x à A. $\mu_A(x) = 1$ et $\mu_A(x) = 0$ correspondent respectivement à l'appartenance et la non-appartenance.

Un ensemble flou A peut également être décrit par un certain nombre de caractéristiques comme [28 ,29] :

- **Son support :** qui est l'ensemble des éléments de U qui appartiennent au moins un peu à A. Il est définit par :

$$supp(A) = \{x \in U / \mu_A(x) > 0\} \quad (1.1)$$

Un ensemble flou dont le support est un singleton flou dans U avec $\mu_A(x) = 1$ est appelé « singleton flou ».

- **Sa hauteur :** qui est sa plus grande valeur prise par sa fonction d'appartenance. Elle est définit par :

$$h(A) = sup_{x \in U}.\mu_A(x) \quad (1.2)$$

- **Son noyau :**qui est l'ensemble des éléments de U pour lesquels la fonction d'appartenance de A vaut 1. Il est définit par :

$$noy(A) = \{x \in U / \mu_A(x) = 1\} \quad (1.3)$$

Lorsque le noyau est réduit à un point, celui-ci est appelé « valeur modale ». Pour le triangle, elle correspond à la valeur du sommet.

1.2.2.1 Variable linguistique

La notion de variable linguistique permet de modéliser les connaissances imprécises ou vagues sur une variable dont la valeur précise est inconnue. Une variable linguistique, ou variable floue, est donc une variable dont les valeurs floues appartiennent à des ensembles flous pouvant représenter des mots du langage naturel. Ainsi une variable floue peut prendre simultanément plusieurs valeurs linguistiques [28]. Le domaine sur lequel ces termes et ces variables sont définis, constitue l'univers de discours. Le découpage de cet univers de discours par les termes flous est appelé une partition floue. Lorsque l'univers de discours est totalement recouvert par les termes flous, et que pour toutes valeurs, la somme des degrés d'appartenance est égale à 1, on parle alors de *partition floue forte*.

La variable linguistique peut être représentée par un triplet $(x, T(x), U)$ dans lequel x est le nom de la variable linguistique, $T(x)$ l'ensemble des valeurs linguistiques de x et U l'univers de discours. La figure 1.1 illustre un exemple de variable linguistique *'vitesse'* avec trois termes linguistiques : petite, moyenne et grande.

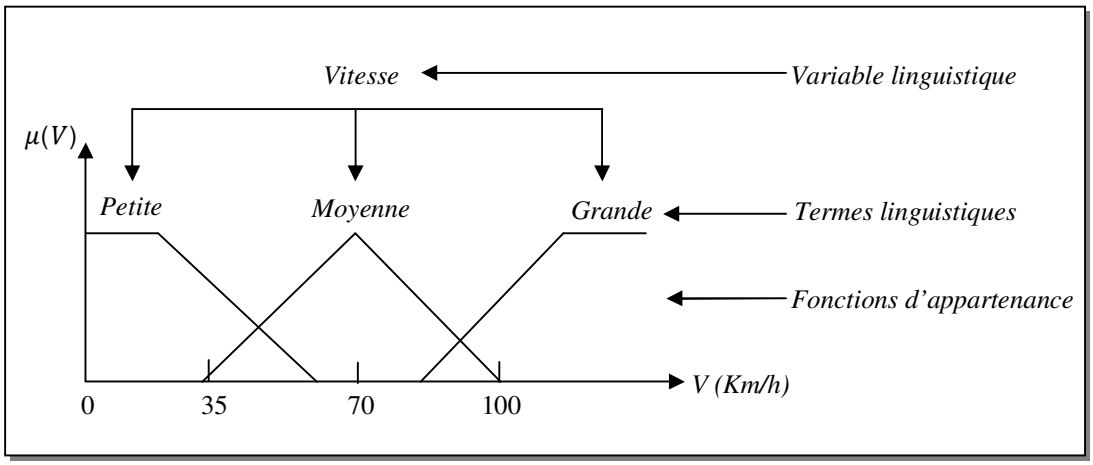


Fig. 1.1 : Variable linguistique

1.2.2.2 Fonction d'appartenance

Soit un ensemble$E$ et un sous-ensemble A de E $(A \subset E)$, et x un élément de E appartenant à A $(x \in A)$. Pour illustrer cette caractéristique, on utilise la fonction d'appartenance $\mu_A(x)$ compris entre 0 et 1, qui représente **le degré d'appartenance** de x à l'ensemble flou A. Le plus souvent, la fonction d'appartenance est déterminée par l'une des fonctions suivantes (figure 1.2) **:**

- **Fonction triangulaire**

Elle est définie par trois paramètres$\{a, b, c\}$, qui déterminent les coordonnées des trois sommets (figure 1.2-a).

$$\mu(x) = max\left(min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \quad (1.4)$$

- **Fonction trapézoïdale**

Elle est définie par quatre paramètres$\{a, b, c, d\}$, (figure 1.2-b) :

$$\mu(x) = max\left(min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \quad (1.5)$$

- **Fonction gaussienne**

Elle est définie par deux paramètres$\{\sigma, m\}$,(figure1.2-c) :

$$\mu(x) = exp\left(-\frac{(x-m)^2}{2\sigma^2}\right) \quad (1.6)$$

- **Fonction sigmoïde**

Elle est définie par deux paramètres$\{a, c\}$,(figure 1.2-d).

$$\mu(x) = \frac{1}{1+exp(-a(x-c))} \quad (1.7)$$

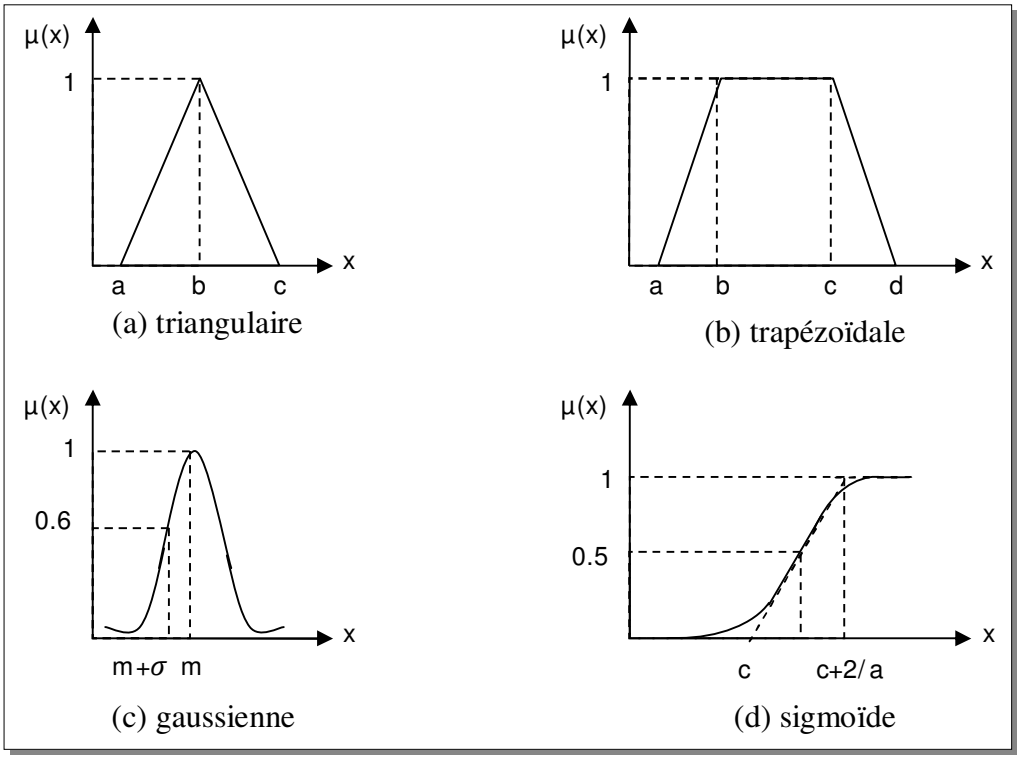


Fig.1.2 : Formes usuelles des fonctions d'appartenance

1.2.2.3 Opérations sur les ensembles flous

Soient A et B deux ensembles flous dans U ayant respectivement μ_A, μ_B comme fonctions d'appartenance. L'union, l'intersection et la complémentation des ensembles flous sont définis à l'aide de leur fonction d'appartenance.

- **Union :** L'union de deux sous-ensembles flous A et B de E est un sous- ensemble flou $A \cup B$ qui est défini par le plus grand sous-ensemble flou qui contient A et qui contient B. Sa fonction d'appartenance est donnée par **:**

$$\mu_{A\cup B}(x) = \max[\mu_A(x), \mu_B(x)] \; ; \; \forall x \in E. \tag{1.8}$$

➢ **Intersection :** L'intersection de deux sous-ensembles flous A et B de E est un sous-ensemble flou $A \cap B$ qui est défini par le plus petit sous-ensemble contenu à la fois dans A et dans B. Sa fonction d'appartenance est donnée par :

$$\mu_{A\cap B}(x) = \min[\mu_A(x), \mu_B(x)] \; ; \; \forall x \in E. \tag{1.9}$$

➢ **Complément :** Le complément d'un sous-ensemble flou A de E est un ensemble flou dénoté par $\bar{A}$ dont la fonction d'appartenance est donnée par **:**

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x); \; \forall x \in E. \tag{1.10}$$

➢ **Inclusion :** Soit les deux sous-ensembles flous A et B de l'ensemble E, on dira que A est inclus dans$B$ si :

$$\forall\, x \in E: \mu_A(x) \leq \mu_B(x). \tag{1.11}$$

Et on notera alors : $A \subset B$

1.2.3 La Commande Floue

Le succès de la commande floue trouve aussi en grande partie son origine dans sa capacité à traduire une stratégie de contrôle d'un opérateur qualifié en un ensemble de règles linguistiques « si… alors » facilement interprétables. L'utilisation de la commande floue est particulièrement intéressante lorsqu'on ne dispose pas de modèle mathématique précis du processus à commander ou lorsque ce dernier présente de trop fortes non linéarité ou imprécisions.

1.2.3.1 Description générale d'un contrôleur flou [28,29,30]

En général, les contrôleurs flous sont utilisés dans les structures de commande en boucle fermée des processus (figure 1.3).

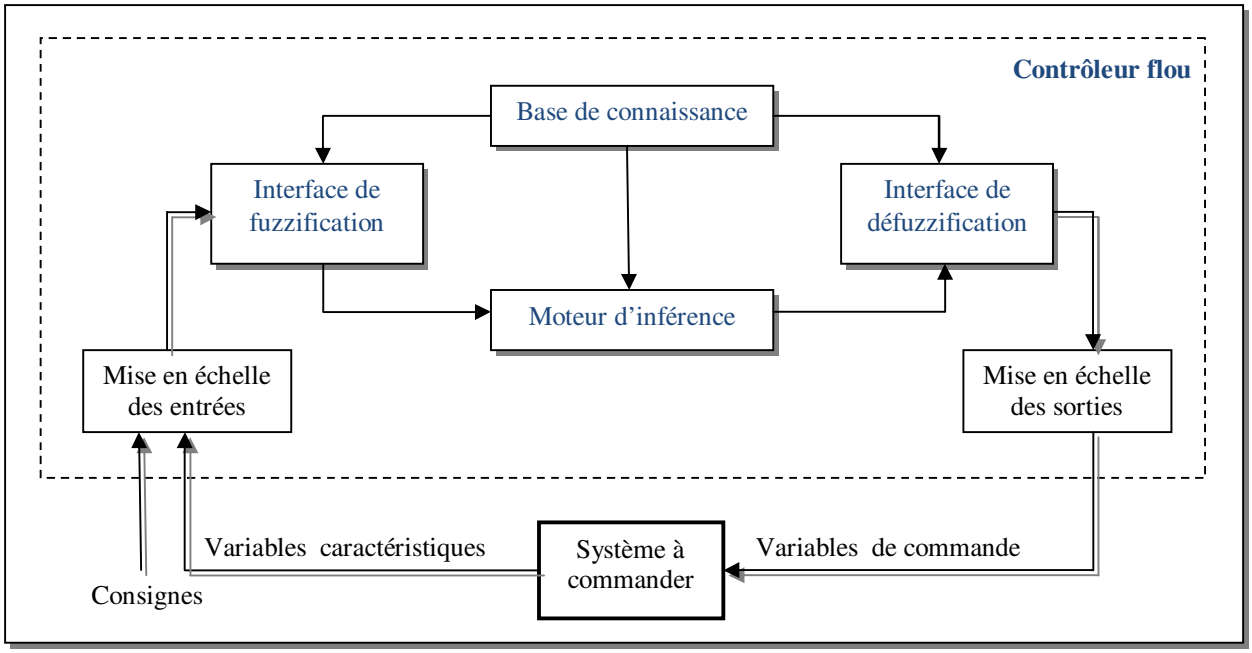


Fig. 1.3 : Structure de base d'un régulateur flou

Les variables caractéristiques du système à commander et les consignes définissent les variables d'entrées du contrôleur flou. Les variables caractéristiques sont, en général, les grandeurs de sortie du processus et, le cas échéant, d'autres mesures déterminantes pour saisir l'évolution dynamique du système. Les variables de sortie du contrôleur flou sont les commandes appliquées au processus. La Mise en échelle (normalisation/dénormalisation) des grandeurs d'entrées/sorties permet d'adapter le traitement des signaux d'entrées/sorties. Par convention, la plage de variation des variables d'entrées/sorties est comprise entre -1 et +1. Les opérations de normalisation et dénormalisation sont optionnelles.

La configuration de base du contrôleur flou comprend quatre parties :

- Interface de fuzzification,
- Base de connaissances,
- Moteur d'inférence floue,
- Interface de défuzzification.

1.2.3.1.a Interface de fuzzification

La fuzzification de la valeur précise d'une variable consiste à caractériser le degré avec lequel cette mesure appartient à un sous-ensemble flou donné, c'est-à-dire le passage d'une grandeur précise à une variable floue.

Le choix des formes des fonctions d'appartenance est arbitraire. Des études comparatives ont montré qu'avec les différentes formes des fonctions d'appartenance, les résultats sont pratiquement similaires en boucle fermée. La forme la plus fréquemment utilisée en commande floue est la forme triangulaire. Le nombre de fonctions d'appartenance est généralement impair et se répartissent autour de zéro. En général, on introduit pour une variable linguistique trois, cinq ou sept ensembles flous. Le choix du nombre dépend de la précision souhaitée. Les fonctions d'appartenance peuvent être symétriques, non symétriques et équidistantes ou non équidistantes.

1.2.3.1.b Base de connaissance

La conception d'une base de connaissances représente la phase dans la conception des systèmes experts. Elle comprend la base de données et la base des règles floues.

- ***La base de données***

Contient la définition des ensembles flous, les facteurs d'échelle pour la normalisation des ensembles de référence et la partition de l'espace flou d'entrée et sortie.

- ***La base des règles floues***

Elle rassemble l'ensemble des règles floues de type « **Si-Alors** » décrivant en termes linguistiques basés sur la connaissance d'un expert le comportement dynamique du système :

$$R_l : \boldsymbol{si}\ x_1 est\ A_1^l et \ldots et\ x_n est\ A_n^l\ \boldsymbol{Alors}\ u_l\ est\ B^l \qquad (1.12)$$

Avec : [x_1 ,…, x_n] : les entrées du régulateur.

u_l : la sortie du régulateur.

Chaque régulateur activé donne un sous-ensemble flou de sortie.

D'une manière générale, la base de règles d'un système flou doit respecter certaines conditions afin d'assurer le bon fonctionnement de ce dernier. Parmi celles-ci citons:

- La complétude : une base de règles d'un système flou est dite complète si, pour chaque vecteur d'entrée, il existe au moins une règle floue activée. Afin d'assurer cette propriété, les fonctions d'appartenance doivent couvrir toutes les plages possibles des

variables d'entrée. L'utilisation de fonctions d'appartenance triangulaires régulièrement réparties respecte la propriété de complétude.

- La consistance : une base de règles d'un système flou est dite inconsistante, s'il existe deux règles floues ayant la même prémisse mais des conclusions différentes. La propriété de consistance permet d'éviter les contradictions dans une base de règles.

1.2.3.1.c Moteur d'inférence floue

C'est un mécanisme de décision. Il permet à partir d'un fait observé de la base des règles floues une décision en exploitant le raisonnement approximatif. Dans les inférences de régulateur par logique floue interviennent les opérateurs ET et OU. L'opérateur ET s'applique aux variables à l'intérieur d'une règle tandis que l'opérateur OU lie les différentes règles.

A cause de l'empiétement des fonctions d'appartenances, en générale deux ou plusieurs règles sont activées en même temps. Ce fait doit être pris en considération lors de la réalisation de l'opérateur OU. Il existe plusieurs possibilités pour réaliser ces opérateurs qui s'appliquent aux fonctions d'appartenances. On introduit alors la notion de méthode d'inférence. Elle détermine la réalisation des différents opérateurs dans une inférence, permettant ainsi un traitement numérique de cette dernière.

Pour le réglage par logique floue, on utilise en générale une des méthodes suivantes :

- Méthode d'inférence max-min (Mamdani)
- Méthode d'inférence max-prod (Larsen)
- Méthode d'inférence somme-prod (Sugeno)

➢ ***Méthode d'inférences max-min***

La méthode d'inférences max-min réalise, au niveau de la condition l'opérateur OU par la formation du maximum et l'opérateur ET par la formation du minimum. La conclusion dans chaque règle, introduite par ALORS, lit le facteur d'appartenance de la condition avec la fonction d'appartenance de la variable de sortie par l'opérateur ET, réalisé dans le cas présent par la formation du minimum. Enfin l'opérateur OU qui lie les différentes règles est réalisé par la formation du maximum.

A noter que désignation de la méthode d'inférence (max-min dans le cas présent) se rapporte à la réalisation de OU liant les règles (max) et la réalisation de ALORS (min).

Pour chaque règle, on obtient la fonction d'appartenance partielle par la relation :

$$\mu_{Ri}(x_R) = \min[\mu_{ci}, \mu_{oi}(x_R)]; \; i = 1,2,\ldots,m. \tag{1.13}$$

Où $\mu_{oi}(x_R)$ est la fonction d'appartenance liée à l'opération imposée par la règle R_i,μ_{ci} est le facteur d'appartenance. La fonction d'appartenance résultante est alors donnée par :

$$\mu_{Res}(x_R) = \max[\mu_{R1}(x_R), \mu_{R2}(x_R), \ldots, \mu_{Rm}(x_R)]. \tag{1.14}$$

- ***Méthode d'inférence max-prod***

La méthode d'inférences max-prod réalise en générale, au niveau de la condition, l'opérateur OU par la formation du maximum et l'opérateur ET par la fonction du minimum. Par contre, la conclusion dans chaque règle, introduite par ALORS, qui lie le facteur d'appartenance de la condition avec la fonction d'appartenance de la variable de sortie par l'opérateur ET est réalisée cette fois-ci par la formation du produit. L'opérateur OU qui lie les différentes règles est réalisé de nouveau par la formation du maximum.

Comme on le voit, le OU, liant les règles est réalisé par la formation du maximum et la ALORS est réalisé par la formation du produit, d'où la désignation de cette méthode d'inférence par max-prod.

Pour chaque règle, la fonction d'appartenance partielle $\mu_{Ri}(x_R)$ est donnée par la relation :

$$\mu_{Ri}(x_R) = \mu_{ci} \times \mu_{oi}(x_R); i = 1,2,\ldots,m. \tag{1.15}$$

Pour la fonction d'appartenance résultante, on obtient :

$$\mu_{Res}(x_R) = \max[\mu_{R1}(x_R), \mu_{R2}(x_R), \ldots, \mu_{Rm}(x_R)]. \tag{1.16}$$

- ***Méthode d'inférence somme-prod***

Par opposition aux méthodes d'inférence précédentes, la méthode d'inférence somme-prod réalise, au niveau de la condition, l'opérateur OU par la formation de la somme, plus précisément par la valeur moyenne, tandis que l'opérateur ET est réalisé par la formation du produit. La conclusion de chaque règle, précédée par ALORS, liant le facteur d'appartenance de la condition avec la fonction d'appartenance de la variable de sortie par l'opérateur ET, est réalisée par la formation du produit. L'opérateur OU qui lie les différentes règles est réalisé par la formation de la somme, donc de la valeur moyenne.

En toute généralité, on obtient la fonction d'appartenance partielle $\mu_{Ri}(x_R)$ de chaque règle par la relation :

$$\mu_{Ri}(x_R) = \mu_{ci} \times \mu_{oi}(x_R);\ i = 1,2,\ldots,m. \quad (1.17)$$

La fonction d'appartenance résultante est donnée par :

$$\mu_{Res}(x_R) = \sum[\mu_{R1}(x_R), \mu_{R2}(x_R), \ldots, \mu_{Rm}(x_R)]/\, m. \quad (1.18)$$

Où m est le nombre de règles intervenant dans l'inférence.

1.2.3.1.d Interface de défuzzification

Les méthodes d'inférence fournissent une fonction d'appartenance résultante, $\mu_{Res}(x_R)$ pour la variable de sortie x_R, il s'agit donc d'une information floue. Par cette étape, se fait alors le retour aux grandeurs de sortie réelles. Il s'agit à cet effet, de calculer à partir des degrés d'appartenance à tous les ensembles flous de la variable de sortie, l'abscisse qui correspond à la valeur de cette sortie. Cette transformation est appelée défuzzification. Dans la commande en temps réel, un critère de choix de la commande de défuzzification est la simplicité de calcul.

Plusieurs stratégies de défuzzification existent, les plus utilisées sont:

- Méthode de centre de gravité.
- Méthode du maximum.
- Méthode de la moyenne des maxima.

➢ ***Méthode de centre de gravité :***

La commande résultante $u_r$représente le centre de gravité de l'ensemble flou inféré:

Dans le cas discret :

$$u_r = \frac{\sum_{i=1}^{n} u_i\, \mu_{Res}(u_i)}{\sum_{i=1}^{n} \mu_{Res}(u_i)} \quad (1.19)$$

n : Le nombre de niveaux de quantification de la sortie du contrôleur flou.

Dans le cas continu:

$$u_r = \frac{\int u \cdot \mu_{Res}(u)\, du}{\int \mu_{Res}(u)\, du} \quad (1.20)$$

- ***Méthode du maximum***

Cette méthode, s'applique uniquement dans le cas où la fonction d'appartenance associée à l'ensemble de sortie n'admet qu'un seul maximum. On choisit comme sortie l'abscisse $u_r$correspondant à ce maximum :

$$\mu_A(u_r) = \max(\mu_A(u)) \tag{1.21}$$

- ***Méthode de la moyenne des maxima***

Dans cette méthode, la valeur de sortie est estimée par l'abscisse du point correspondant au centre de l'intervalle M pour lequel la fonction d'appartenance est maximale. Cette valeur est fournie par l'expression :

$$u_r = (\inf(M) + \sup(M))/2 \tag{1.22}$$

Où $\inf(M)$ et $\sup(M)$ sont respectivement les bornes inférieure et supérieure de l'intervalle M.

1.2.3.2 Différents types de systèmes d'inférence floue

1.2.3.2.a Système d'inférence floue de type Mamdani

En 1974, E.H Mamdani avait présenté, pour la première fois, la technique de commande par logique floue [31]. Celle-ci consiste à déterminer un ensemble de règles qui maîtrise le comportement dynamique du système à commande. L'obtention de ces règles est facile auprès des experts qui connaissent bien le système. Il avait utilisé des règles à prémisses et conclusions symboliques, l'inférence (max, min), et la défuzzification par centre de gravité. La forme de l'implication floue définie par ce type de contrôleur est de la forme :

$$\text{« } \boldsymbol{Si}\ x_1\ est\ A\ et\ x_2\ est\ B\ \boldsymbol{Alors}\ y\ est\ C \text{ »} \tag{1.23}$$

La conséquence de ce type de système est une valeur floue.

1.2.3.2.b Système d'inférence flou de type Takagi-Sugeno

Dans ces systèmes, les prémisses des règles sont exprimées symboliquement et les conclusions sont par des fonctions linéaires [27].

Notons par $x = [x_1, x_2, \ldots x_n]^T$ les entrées du contrôleur flou, et par y sa sortie. Pour chaque $x_i$est associé m_i ensemble flou F_i^j dans X_i tel que $x_i \in X_i$. La base de règles complète du contrôleur flou comporte $N = \prod_{i=1}^{n} m_i$ règles floues de la forme:

$$R_k: \textbf{Si} x_1 est\ F_1^k\ et\ x_2 est\ F_2^k\ et \ldots et\ x_n est\ F_n^k \textbf{ Alors } y = f_k(x); \quad k = 1,2,\ldots,n \qquad (1.24)$$

En général $f_k(x)$ est une fonction polynomiale en fonction des variables d'entrées :

$$f_k(x) = a_0^k + \sum_{i=1}^{n} a_i^k x_i \qquad (1.25)$$

Alors on a affaire à un contrôleur flou de type Takagi-Sugeno d'ordre un. Si par contre $f_k(x)$ est une constante :

$$f_k(x) = a_0^k \qquad (1.26)$$

On a donc un contrôleur flou de type Takagi-Sugeno d'ordre zéro. Etant donné que chaque règle possède une conclusion numérique, et de cette manière, le temps consommé par la procédure de défuzzification est évité. En fait, la sortie du contrôleur flou est donnée par la relation suivante :

$$y(x) = \frac{\sum_{k=1}^{N} \mu_k(x) \cdot f_k(x)}{\sum_{k=1}^{N} \mu_k(x)} \qquad (1.27)$$

Avec :

$\mu_k(x) = \prod_{i=1}^{n} F_i^k$, $F_i^k \in \{F_i^1, F_i^2, \ldots, F_i^{m_i}\}$ représente le degré d'activation de la règle R_k.

1.2.4 Caractéristiques des systèmes d'inférence floue

Comme il est noté précédemment, un système flou est entout premier lieu caractérisé par son type (Mamdani, Takagi-Sugeno,…) et par les partitions floues qu'il met en œuvre. Cependant, la définition totale d'un système flou passe par la spécification d'un ensemble de caractéristiques dites structurelles et paramétriques.

1.2.4.1 Les caractéristiques structurelles

Ces caractéristiques spécifient tous les éléments du SIF qui influent sur sa structure. Ces éléments sont constitués par :

- le type de fonction d'appartenance utilisé(triangle, trapèze, TPE, forme de cloche,…) pour chaque terme linguistique,
- le nombre de termes linguistiques pour chaque variable,
- le nombre optimal de règles,
- les variables principales à ces règles, et
- les opérateurs de conjonction, de disjonction et d'implication, et la technique de défuzzification.

1.2.4.2 Les caractéristiques paramétriques

Une fois la structure du SIF est choisie, le problème est alors le placement optimal des fonctions d'appartenance d'entrées et de sorties ou des singletons de sorties. Les caractéristiques paramétriques se situent au plus bas niveau de spécification d'un SIF.Elles représentent en fait l'aspect purement numérique du système flou et définissent les sous-ensembles qui le constituent :

- les paramètres des fonctions d'appartenance des variables d'entrée (point modal, base, écart type, …),
- les paramètres des fonctions d'appartenance des variables de sortie (conclusions de règles floues) pour les SIF de type Takagi-Sugeno.

1.2.5 Conclusion

D'un point de vue mathématique, un système flou définit une relation non linéaire d'un espace d'entrée vers un espace de sortie, et d'un point de vue logique, un système flou est une machine de prise de décision composée de quatre parties essentielles : la fuzzification, base de connaissance et moteur d'inférence floue et la défuzzification. L'architecteur d'un système flou est déterminée par une meilleure compréhension des ensembles flous et des opérateurs flous. Nous avons constaté qu'il n'existe pas un seul type de système flou, mais il y en a plusieurs. Un utilisateur des systèmes flous doit décider sur la méthode de défuzzification, le type des fonctions d'appartenance, le type des règles floues, la méthode du raisonnement flou et la stratégie de défuzzification. Les SIF exigent de préférence la disponibilité d'une

expertise humaine ; par conséquent, les performances de ces derniers sont étroitement liées aux techniques d'acquisition de connaissances et la justesse des informations acquises.

Cependant, pour réaliser un SIF, il va falloir :

- Interpréter les différents termes linguistiques, c'est-à-dire placer les différentes fonctions d'appartenance.
- Définir le type du SIF puis déterminer sa structure de manière empirique en choisissant *a priori* le nombre de termes linguistiques (sous-ensembles floues) pour chaque entrée et en prenant toutes les combinaisons possibles.
- Choisir une méthode d'inférence.
- Ajuster les différents paramètres au vu des résultats.

Le choix des différents paramètres aura des conséquences immédiates sur les résultats. L'obtention d'un SIF optimal par un réglage manuel n'étant pas systématique ; d'où la nécessité d'utilisation de méthodes d'extraction automatiques de connaissance, notamment, des méthodes d'intelligence artificielle (métaheuristiques).

1.3 Les Méthodes d'Optimisation

1.3.1 Introduction

Les métaheuristiques sont apparues dans les années 1980 et forment une famille d'algorithmes d'optimisation visant à résoudre des problèmes d'optimisation difficile, pour lesquels on ne connaît pas de méthode classique plus efficace. Elles sont généralement utilisées comme des méthodes génériques pouvant optimiser une large gamme de problèmes différents, sans nécessiter de changements profonds dans l'algorithme employé.

L'un des intérêts majeurs des métaheuristiques est leur facilité d'utilisation dans des problèmes concrets sans nécessité de connaissances particulières sur le problème d'optimisation à résoudre [32].

1.3.2 Organisation générale des métaheuristiques

Les métaheuristiques sont souvent inspirées par des systèmes naturels, qu'ils soient pris en physique (les méthodes de voisinage comme le recuit simulé et la recherche tabou), en biologie de l'évolution (les algorithmes évolutifs comme les algorithmes génétiques) ou encore en éthologie (les algorithmes de colonies de fourmis et d'optimisation par essaim particulaire). Dans ce qui suit, nous décrivons les métaheuristiques utilisées dans cette thèse.

1.3.3 La Recherche Tabou

1.3.3.1 Principe de fonctionnement

La recherche Tabou (RT) est une métaheuristique originalement développée par Glover [33, 34, 35]. C'est une procédure itérative qui, partant d'une solution initiale, tente de converger vers la solution optimale en exécutant, à chaque pas, un mouvement dans l'espace de recherche. Chaque pas consiste d'abord à engendrer un ensemble de solutions voisines dela solution courante pour ensuite en choisir la meilleure, même si ce choix entraîne une augmentation de la fonction objective à minimiser. En acceptant de détériorer la valeur de la solution courante, le minimum local peut être évité mais, en contrepartie, des parcours répétitifs sont déplorés. Aussi, pour palier l'inconvénient majeur des méthodes de recherche locale, la recherche tabou a pour but d'améliorer à chaque étape, la valeur de la fonction objective, en utilisant une mémoire afin de conserver les informations sur les solutions déjà visitées. Cette mémoire constitue la *liste Tabou* qui va servir à interdire l'accès aux dernières solutions visitées. Lorsqu'un optimum local est atteint, il y a interdiction de revenir sur le même chemin. Un critère d'aspiration, est également utilisé pour lever l'interdiction l'utilisation d'un mouvement si ce dernier conduit à une meilleure solution.

1.3.3.2 Eléments de la RT

Les éléments de base de la RT sont brièvement indiqués et définis comme suit :

- ***Solution actuelle***: il s'agit d'un ensemble de valeurs de paramètres optimisé à chaque itération. Il joue un rôle central dans la génération des solutions voisines.

- ***Mouvements***: ils caractérisent le processus de génération de solutions d'essais qui sont liés à la solution actuelle.
- ***Ensemble de mouvements***: c'est l'ensemble de tous les mouvements possibles ou des solutions d'essais dans le voisinage de la solution actuelle.
- ***Les restrictions taboues***: sont certaines conditions imposées sur les mouvements interdits. Ces derniers sont enregistrés dans une liste d'une certaine taille appelée *liste tabou* .En général, la taille de la liste tabou devrait croître avec la taille du problème donné.

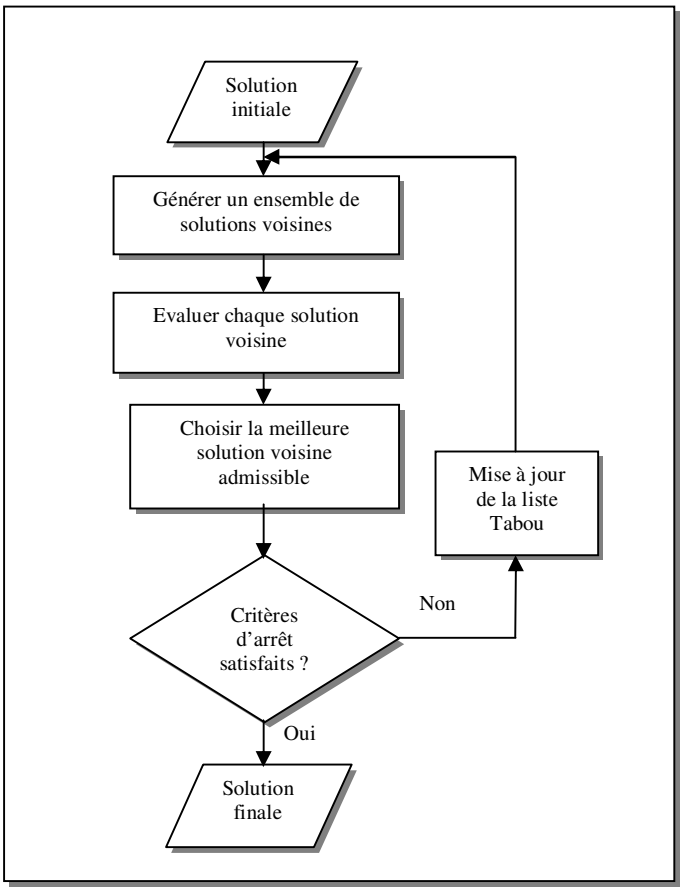


Fig. 1.4 : L'organigramme général de l'algorithme de la recherche tabou standard.

- ***Critère d'aspiration*** : permet d'éliminer le statut tabou d'une solution. Nous considérons que si une solution taboue est la meilleure solution rencontrée depuis le début de la recherche, alors cette solution deviendra la solution initiale suivante dans le processus itératif de recherche. L'importance d'utilisation du critère d'aspiration est

d'ajouter une certaine souplesse à l'algorithme en l'orientant vers les mouvements attractifs.

- ***Critères d'arrêt*** : ce sont les conditions dans lesquelles le processus de recherche se termine. La recherche prendra fin si l'un des critères suivants est satisfait :**(a)** le nombre d'itérations depuis le dernier changement de la meilleure solution est supérieur à un nombre prédéterminé, ou **(b)** le nombre d'itérations atteint le nombre maximum admissible.

L'organigramme général de l'algorithme RT standard est représenté dans la figure 1.4.

1.3.3.3 Amélioration de la RT

Plusieurs stratégies ont été proposées récemment afin d'améliorer l'efficacité de la méthode tabou présenté [35]. L'intensification et la diversification de la recherche sont deux d'entre elles. L'intensification consiste à explorer en détail une région de X jugée prometteuse. Sa mise en œuvre réside le plus souvent en un élargissement temporaire du voisinage de la solution courante dans le but de visiter un ensemble de solutions partageant certaines propriétés. La diversification est une technique complémentaire à l'intensification. Son objectif est de diriger la procédure de recherche vers des régions inexplorées de l'espace X. La stratégie de diversification la plus simple consiste à redémarrer périodiquement le processus de recherche à partir d'une solution générée aléatoirement ; ou choisir judicieusement dans une région non encore visitée de l'ensemble des solutions admissibles.

1.3.4 Les Algorithmes Génétiques

1.3.4.1 Introduction

Les algorithmes génétiques développés par J. Holland [36] présentent des qualités intéressantes pour la résolution de problèmes d'optimisation complexes. Leurs fondements théoriques furent exposés par Goldberg [37]. Ils tentent de simuler le processus d'évolution des espèces dans leur milieu naturel: soit une transposition artificielle de concepts basiques de la génétique et des lois de survie énoncés par Darwin.

1.3.4.2 Principe de fonctionnement des algorithmes génétiques

Un algorithme génétique est construit de manière tout à fait analogue au processus d'évolution d'une population. Dans l'ensemble des solutions d'un problème d'optimisation, une population de taille N est constituée de N solutions (les individus de la population) convenablement marquées par un codage qui les identifie complètement. Une procédure d'évaluation est nécessaire à la détermination de la force de chaque individu de la population. Viennent ensuite une phase de sélection (en sélectionnant les individus selon leur force) et une phase de recombinaison (opérateurs artificiels de croisement et de mutation) qui génèrent une nouvelle population d'individus, qui ont de bonnes chances d'être plus forts que ceux de la génération précédente. De génération en génération, la force des individus de la population augmente et après un certain nombre d'itérations, la population est entièrement constituée d'individus tous forts, soit de solutions quasi-optimales du problème posé.

1.3.4.3 Structure de l'algorithme génétique

L'implémentation d'un AG est spécifique au problème à résoudre. Pour l'utiliser, il faut disposer des cinq éléments suivants [37,38] :

- Un principe de codage de l'élément de population : La qualité du codage des données conditionne le succès des algorithmes génétiques. Les codages binaires ont été très utilisés à l'origine. Les codages réels sont désormais largement utilisés, notamment dans les domaines applicatifs pour l'optimisation de problèmes à variables réelles.
- Un mécanisme de génération de la population initiale. Ce mécanisme doit être capable de produire une population d'individus non homogène qui servira de base pour les générations futures.
- Une fonction à optimiser. Celle-ci retourne une valeur appelée *fitness* ou fonction d'évaluation de l'individu.
- Des opérateurs permettant de diversifier la population au cours des générations et d'explorer l'espace d'état. L'opérateur de croisement recompose les gènes d'individus existant dans la population ainsi que l'opérateur de mutation a pour but de garantir l'exploration de l'espace d'états.
- Des paramètres de dimensionnement: taille de la population, nombre total de générations, probabilités d'application des opérateurs de croisement et de mutation.

L'organigramme fonctionnel de la figure 1.5 illustre la structure de l'algorithme génétique. Les diverses phases et les mécanismes associés à chacune d'entre elles seront présentés dans les sections suivantes.

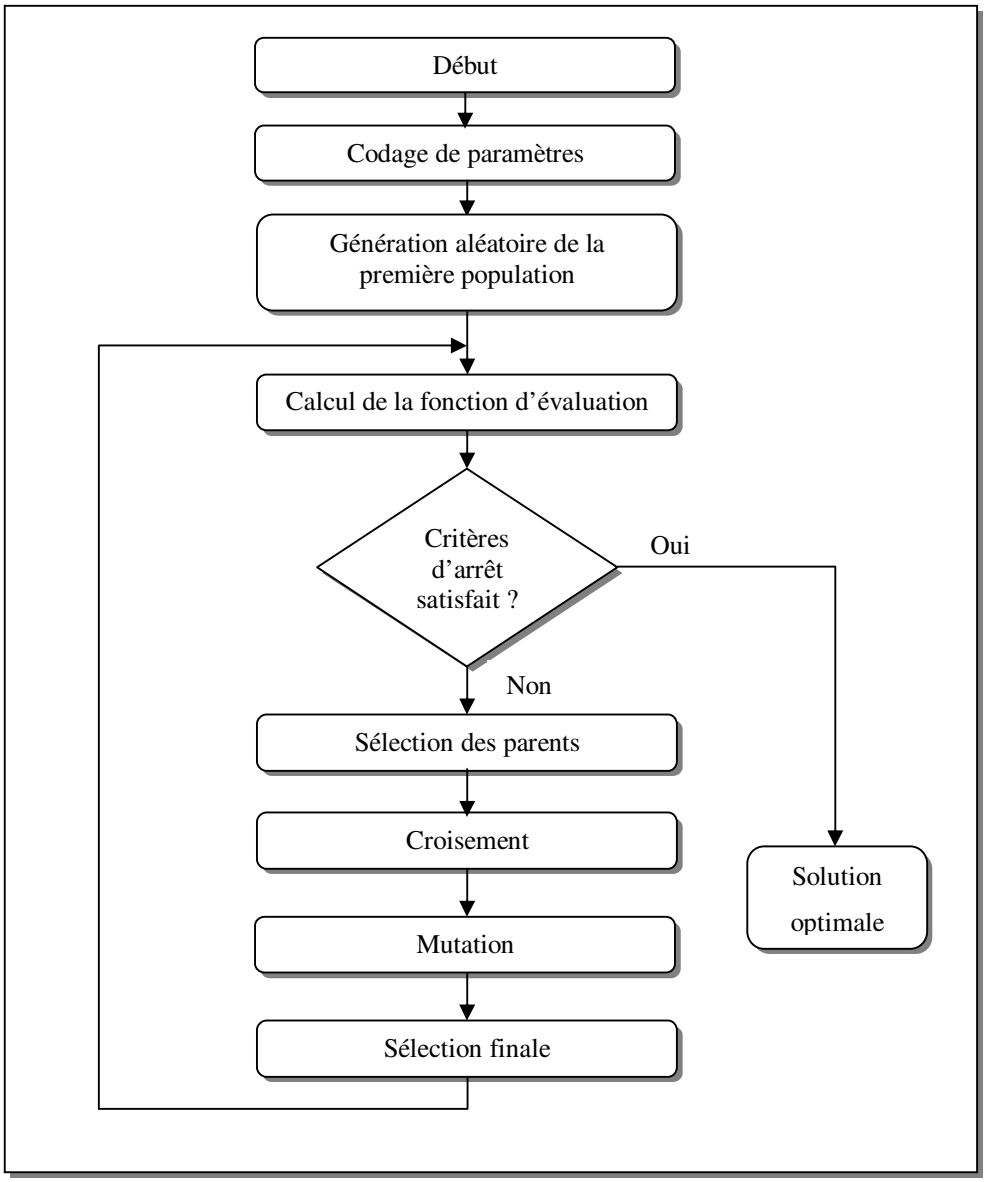


Fig. 1.5 : Organigramme général de l'AG.

1.3.4.4 Le codage

Chaque paramètre d'une solution est assimilé à un gène, toutes les valeurs qu'il peut prendre sont les allèles de ce gène. Un chromosome est une suite de gènes, on peut choisir de

regrouper les paramètres similaires dans un même chromosome et chaque gène sera repérable par sa position. Chaque individu est représenté par un ensemble de chromosomes.

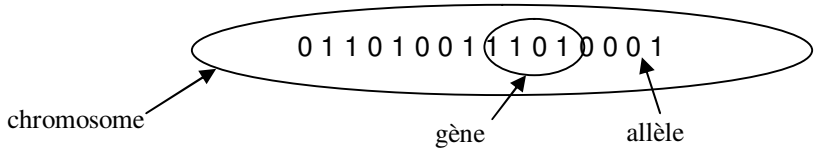


Fig. 1. 6 : Structure d'un chromosome

Il existe principalement deux types de codage : le codage binaire, le codage réel :

1.3.4.4.a Le codage binaire

Chaque gène dispose du même alphabet binaire {0, 1}. Chaque paramètre x_i situé dans un intervalle $[x_{imin}, x_{imax}]$, est associé à une chaîne binaire $b_0, b_1, \ldots, b_{L_{xi}-1}$ définie sur L_{xi} bits. A chaque chaîne correspond une valeur entière naturelle [59] :

$$g_i = \sum_{i=0}^{L_{xi}-1} 2^i . b_i \qquad (1.30)$$

Cette chaîne doit être décodée pour pouvoir calculer la valeur de la fonction coût (fonction d'évaluation) qui lui est associée. Le paramètre réel x_i de l'espace de recherche relatif à g_i est obtenu par interpolation linéaire :

$$x_i = x_{imin} + \frac{x_{imax} - x_{imin}}{2^{L_{xi}} - 1} \cdot g_i \qquad (1.31)$$

1.3.4.4.b Le codage réel

Cela peut-être utile notamment dans le cas où l'on recherche le maximum d'une fonction réelle. La représentation des gènes par des nombres réels arrive naturellement dans le cas d'optimisation de paramètres avec des variables comprises dans des domaines continus. La taille des chromosomes est la même que la taille du vecteur qui sera la solution du problème. Suite à cette approche, chaque gène représente une variable du problème.

L'utilisation de paramètres réels donne la possibilité d'utiliser des domaines étendus pour les variables, ce qui est difficile à atteindre si le codage binaire est utilisé. Dans le codage binaire, l'utilisation de domaines vastes amène à une perte de précision des solutions. Un

autre avantage de l'utilisation des nombres réels est leur capacité à exploiter la gradualité (les changements survenus au niveau des variables impliquent des changements au niveau de la fonction) des fonctions qui ont des variables continues. Suite à cette idée, un avantage très important est la capacité de réglage local de solutions, qui se produit beaucoup plus rapidement que dans le cas du codage binaire des algorithmes génétiques.

1.3.4.5 Génération de la population initiale

Le choix de la population initiale d'individus conditionne fortement la rapidité de convergence de l'algorithme. Si la position de l'optimum dans l'espace d'état est totalement inconnue, il est naturel de générer aléatoirement des individus en faisant des tirages uniformes dans chacun des domaines associés aux composantes de l'espace d'état en veillant à ce que les individus produits respectent les contraintes. Si des informations a priori sur le problème sont disponibles, il est naturel de générer les individus dans un sous domaine particulier afin d'accélérer la convergence.

1.3.4.6 Fonction d'évaluation

En raison de l'analogie avec la théorie de l'évolution (survie des individus les mieux adaptés à leur environnement), l'algorithme génétique est naturellement formulé en terme de maximisation. Etant donné une fonction f réelle à une ou plusieurs variables, le problème d'optimisation sur l'espace de recherche E s'écrit de la manière suivante :

$$\max_{x \in E} f(x) \tag{1.32}$$

Dans beaucoup de problèmes, l'objectif est exprimé sous forme de minimisation d'une fonction coût h :

$$\min_{x \in E} h(x) \tag{1.33}$$

Le passage du problème de minimisation à un problème de maximisation est obtenu par transformation de la fonction h selon la relation suivante :

$$f(x) = 1/(1 + h(x)) \tag{1.34}$$

1.3.4.7 Les mécanismes d'un AG

A partir d'une première population d'individus créée aléatoirement, les AG génèrent de nouveaux individus plus performants que leurs prédécesseurs en effectuant des opérations génétiques. Les AG utilisent des outils tels que la reproduction, le croisement et la mutation. Ces outils sont basés sur des processus aléatoires. La reproduction est une version artificielle de la sélection naturelle, c'est un processus dans lequel chaque individu est copié en fonction des valeurs de la fonction d'évaluation. Le croisement est l'opérateur le plus dominant dans un AG, il permet à deux chaînes d'échanger des portions de leurs structures produisant ainsi de nouvelles chaînes. La mutation est un opérateur local qui est appliqué avec une très faible probabilité.

1.3.4.7.a La sélection

La sélection est un mécanisme qui fixe à partir de la génération Précédente, quels individus pourront de reproduire pour former la génération suivante. Lors de cette phase, les individus les plus forts sont généralement dupliqués et forment les parents de la génération en cours, alors que les faibles disparaissent sans avoir la possibilité de se produire.

On trouve essentiellement trois types de méthodes de sélection différentes :

- La méthode de la "loterie biaisée",
- La méthode "élitiste",
- La sélection par tournois.

- ***Sélection par roue de loterie biaisée***

Avec cette méthode chaque individu a une chance d'être sélectionné proportionnelle à sa performance *'fitness'*, donc plus les individus sont adaptés au problème, plus ils ont de chances d'être sélectionnés. Ainsi, dans le cas d'un codage binaire, la fonction d'évaluation d'un chromosome particulier ch_i étant $f(ch_i)$, la probabilité avec laquelle il sera réintroduit dans la nouvelle population de taille *N* est :

$$p_s(c_i) = \frac{f(ch_i)}{\sum_{j=1}^{N} f(ch_j)} \qquad (1.35)$$

Plus la performance d'un individu est élevée par rapport à celle des autres, plus il a une chance d'être reproduit dans la population. Les individus ayant une grande *fitness* relative ont donc plus de chance d'être sélectionnés. On parle alors de sélection proportionnelle.

- ***La méthode élitiste***

La stratégie élitiste consiste à conserver le meilleur individu à chaque génération. Ainsi l'élitisme empêche l'individu le plus performant de disparaître au cours de la sélection ou que ses bonnes combinaisons soient affectées par les opérateurs de croisement et de mutation. Après chaque évaluation de la performance des individus à une génération g donnée, le meilleur individu de la génération précédente $(g-1)$ est réintroduit dans la population si aucun des individus de la génération g n'est meilleur que lui. Par cette approche, la performance du meilleur individu de la population courante est monotone de génération en génération.

- ***La sélection par tournois***

Le principe de cette méthode est le suivant : on effectue un tirage avec remise de deux individus de la population courante, et on le fait "combattre". Celui qui a la *fitness* la plus élevée l'emporte avec une probabilité p_s comprise entre 0.5 et 1. On répète ce processus *n* fois de manière à obtenir les *n* individus de la nouvelle population qui serviront de parents.

1.3.4.7.b Le croisement

Le croisement est l'échange d'un certain nombre de bits entre deux chromosomes représentant deux individus de la population. En choisissant aléatoirement deux individus parents, l'opération de croisement donne naissance à deux nouveaux individus.

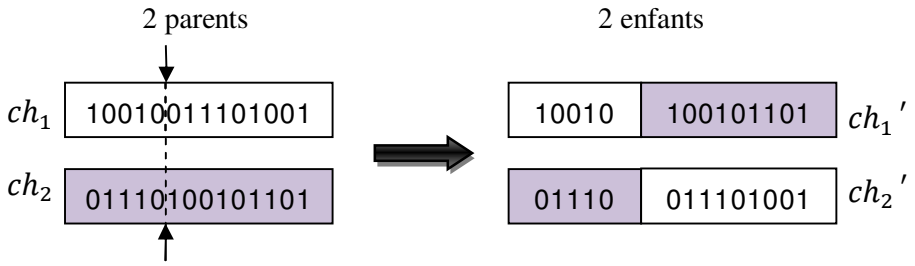
Il existe différents types de croisements pour un algorithme génétique classique. Sur la figure 1.7, les trois principaux types de croisement sont présentés (en un point, en deux points et uniforme). Cette opération est contrôlée par une probabilité p_c souvent supérieure à 0.5.

- ***Croisement en un point***

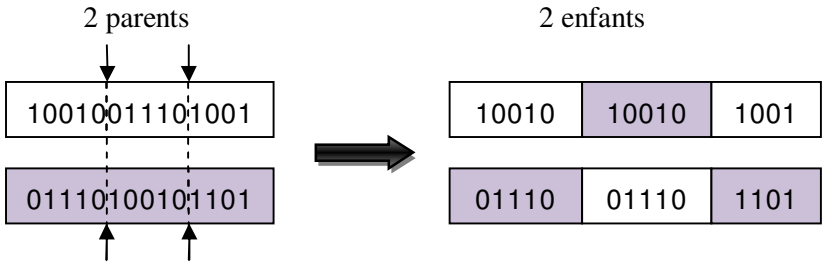
La population courante est divisée en deux sous populations de même taille $(N/2)$ et chaque couple formé par un membre provenant de chaque sous population participe à un

croisement avec une probabilité p_c. Si le croisement a eu lieu entre deux chromosomes parents (ch_1 et ch_2), constitués de l gènes, on tire aléatoirement une position de chacun des parents. On échange ensuite les deux sous chaînes terminales de chacun des chromosomes, ce qui produit deux enfants ch'_1 et ch'_2 comme indiqué sur la figure 1.7.a .

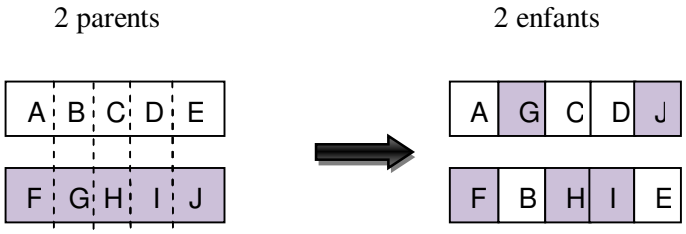
On peut noter que le nombre de points de croisements ainsi que la probabilité de croisement *pc* permettent d'introduire plus ou moins de diversité. En effet, plus le nombre de points de croisements sera grand et plus la probabilité de croisement sera élevée plus il y aura d'échange de segments, donc d'échange de paramètres, d'information, et plus le nombre de points de croisements sera petit et plus la probabilité de croisement sera faible, moins le croisement apportera de diversité.



(a) croisement en un point



(b) : croisement en deux points



(c) : croisement uniforme

Fig. 1.7 : Exemples d'opérations de croisement

- ***Croisement uniforme***

Il consiste à définir de manière aléatoire un "masque", c'est-à-dire une chaîne de bits de même longueur que les chromosomes des parents sur lesquels il sera appliqué. Ce masque est destiné à savoir, pour chaque locus, de quel parent le premier fils devra hériter du gène s'y trouvant; si face à un locus le masque présente un 0, le fils héritera le gène s'y trouvant du parent #1, s'il présente un 1 il en héritera du parent #2. La création du fils #2 se fait de manière symétrique : si pour un gène donné le masque indique que le fils #1 devra recevoir celui-ci du parent #1 alors le fils #2 le recevra du parent #2, et si le fils #1 le reçoit du parent #2 alors le fils #2 le recevra du parent #1.

- ***Croisement barycentre***

Deux gènes $ch_1(i)$ et $ch_2(i)$ sont sélectionnés dans chacun des parents à la même position$i$. Ils définissent deux nouveaux gènes $ch'_1(i)$ et $ch'_2(i)$ par combinaison linéaire [39]:

$$\begin{cases} ch'_1(i) = \alpha\, ch_1(i) + (1-\alpha)\, ch_2(i) \\ ch'_2(i) = \alpha\, ch_2(i) + (1-\alpha)\, ch_1(i) \end{cases} \tag{1.36}$$

Où α est un coefficient de pondération aléatoire adapté au domaine d'extension des gènes qui prend généralement ses valeurs dans l'intervalle $[-0.5, 1.5]$.

1.3.4.7.c La mutation

Une mutation consiste simplement en l'inversion d'un bit se trouvant en un locus bien particulier avec une probabilité $p_m$très faible. L'opérateur de mutation modifie donc de manière complètement aléatoire les caractéristiques d'une solution, ce qui permet d'introduire et de maintenir la diversité au sein de la population de solutions. Cet opérateur joue le rôle d'un "élément perturbateur" ; il introduit du "bruit" au sein de la population.

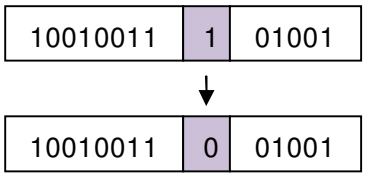


Fig. 1.8 : Exemple d'une opération de mutation

Pour ne pas détériorer les performances de l'AG, Goldberg conseille une fréquence de mutation tous les 1000 bits ; certains préconisent plutôt la valeur suivante pour p_m [37] :

$$p_m = 1/L\ . \tag{1.37}$$

Où L est la longueur du chromosome.

Il existe plusieurs méthodes de mutation parmi lesquels on trouve :

- ***Mutation binaire***

Dans le cas du codage binaire, chaque bit est remplacé par son complément. Il existe une variante où plusieurs bits peuvent muter au sein d'un même chromosome. Un test sous le taux de mutation est effectué non plus pour le chromosome mais pour chacun de ses bits : en cas de succès, un nouveau bit tiré au hasard remplace l'ancien.

- ***Mutation réelle***

La mutation réelle ne se différencie de la mutation binaire que par la nature de l'élément qu'elle altère : ce n'est plus un bit qui est inversé, mais une variable réelle qui est de nouveau tirée au hasard sur son intervalle de définition.

Dans le cas d'un codage réel, on utilise principalement deux opérateurs de mutation: la mutation uniforme et la mutation non uniforme [39].

- ***Mutation uniforme***

En supposant fixée la probabilité de mutation p_m, un tirage au sort pour chaque gène x_k d'un chromosome ch permet de décider si ce gène doit être ou non modifié. Le gène x_k sélectionné est remplacé par une valeur quelconque x'_k tirée aléatoirement dans l'intervalle $\left[x_k^{min}, x_k^{max}\right]$.

- ***Mutation non uniforme***

Le calcul de la nouvelle valeur d'un gène est un peu plus complexe. Le gène x_k subit des modifications importantes durant les premières générations, puis graduellement décroissantes au fur et à mesure que l'on progresse dans le processus d'optimisation. Pour une génération t,

on tire au sort une valeur binaire qui décidera si le changement doit être positif ou négatif. La nouvelle valeur x'_k du gène x_k est donnée par :

$$x'_k = \begin{cases} x_k + \Delta(t, x_k^{max} - x_k) & si \quad r = 0 \\ x_k - \Delta(t, x_k - x_k^{min}) & si \quad r = 1 \end{cases} \tag{1.38}$$

Où, $\Delta(t, y)$ est une fonction qui définit l'écart entre la nouvelle valeur et la valeur initiale à la génération t et r est un nombre aléatoire qui prend les valeurs 0 ou 1.

1.3.4.8 La sélection des individus d'une nouvelle génération

A la suite de la recombinaison génétique (croisement et mutation), la population compte $2N$ individus (N parents et N enfants) ; N étant la taille de la population. Il faut donc éliminer N individus pour constituer la génération suivante. C'est le rôle de la sélection finale qui agit sur les populations de parents et d'enfants d'une génération courante pour créer une nouvelle génération. Plusieurs stratégies sont possibles pour effectuer une telle sélection :

1.3.4.8.a La sélection par descendance

Il n y a aucune compétition entre parents et enfants. La population de la nouvelle génération est obtenue par descendance, les enfants remplaçants automatiquement leurs parents quel que soit leur adaptation.

1.3.4.8.b La sélection par compétition

Une compétition a lieu entre parents et enfants pour déterminer les *survivants* de la génération. Ainsi, les enfants peuvent être insérés dans la population si et seulement si leur performance est supérieure à celle de leur parents à rang équivalent.

1.3.4.8.c La sélection élitiste

L'élitisme est une façon de protéger la rémanence de bonnes solutions et d'assurer leur survie tout au long de la recherche. Ici, la population de la prochaine génération est choisie à partir de la population des enfants et de la population parents ensemble. Cette sélection a l'avantage de permettre une convergence (plus) rapide des solutions, mais au détriment de la diversité des individus.

1.3.4.9 Critère d'arrêt

Le cycle de génération et de sélection de population est répété jusqu'à ce qu'un critère d'arrêt soit satisfait ; ce critère peut être notamment un nombre maximum de générations, un temps maximal de calcul, une valeur de fitness minimale, ou/et une convergence vers une solution satisfaisante.

1.3.5 L'algorithme d'Optimisation par Essaim Particulaire

1.3.5.1 Principe de fonctionnement

La méthode d'Optimisation par Essaim Particulaire (OEP) a été proposée en 1995 par James Kennedy et Russel Eberhart [40] qui cherchaient à simuler la capacité des oiseaux à voler de façon synchrone et leur aptitude à changer brusquement de direction, tout en restant en formation optimale. Le fonctionnement de l'OEP fait qu'elle peut être classée parmi les méthodes itératives (approche progressive de la solution) et stochastiques (faisant appel au hasard) dans le but d'améliorer la situation existante en se déplaçant partiellement au hasard et partiellement selon des règles prédéfinies, en vue d'atteindre la solution globale souhaitée.

La méthode d'optimisation par essaim particulaire, est une procédure de recherche basée sur une population d'individus, appelés particules, qui changent leur position (état) avec le temps. Dans un système d'OEP, les particules se déplacent à l'intérieur d'un espace de recherche. Pendant le déplacement, chaque particule ajuste sa position selon sa propre expérience, et selon l'expérience des particules voisines, se servant de sa meilleure position produite et de celle de ses voisines. Ce comportement est semblable à celui du comportement humain consistant à prendre des décisions où les individus considèrent leur expérience antérieure et celle des personnes qui les entourent.

1.3.5.2 Les éléments de l'OEP [40, 41]

Pour appliquer l'OEP, il faut définir un espace de recherche constitué de particules et une fonction coût (fonction '*objectif*') à optimiser. Le principe de l'algorithme est de déplacer ces particules afin qu'elles trouvent l'optimum. Chacune de ces particules est dotée :

- D'une position, c'est-à-dire ses coordonnées dans l'ensemble de définition.
- D'une vitesse qui permet à la particule de se déplacer. De cette façon, au cours des itérations, chaque particule change de position. Elle évolue en fonction de son meilleur voisin, de sa meilleure position, et de sa position précédente. C'est cette évolution qui permet de tomber sur une particule optimale.
- D'un voisinage, c'est-à-dire un ensemble de particules qui interagissent directement sur la particule, en particulier celle qui a le meilleur critère.

Dans le cas d'un problème d'optimisation, la qualité d'un site de l'espace de recherche est déterminée par la valeur de la fonction coût en ce point.

1.3.5.3 Principe fondamental

L'algorithme de base de l'OEP travaille sur une population appelée *essaim* de solutions possibles, elles-mêmes appelées *particules*. Ces particules sont placées aléatoirement dans l'espace de recherche de la fonction *objectif*.

A chaque itération, chaque particule se déplace en prenant en compte sa meilleure position (p_{best}) ainsi que la meilleure position de son voisinage (g_{best}). On calcule alors la nouvelle vitesse de chaque particule par la formule (1.39), La nouvelle position sera déterminée par la somme de la position précédente et la nouvelle vitesse comme l'indique l'équation suivante:

$$V(t+1) = w * V(t) + c_1 * R_1 * \left(p_{best}(t) - p(t)\right) + c_2 * R_2 * (g_{best}(t) - p(t)) \quad (1.39)$$

$$p(t+1) = p(t) + V(t+1) \quad (1.40)$$

Où V est la vitesse de la particule, w est en général une constante appelée, *coefficient d'inertie*, p est la position (solution) actuelle, $c_1$et c_2 sont deux constantes, appelées *coefficients d'accélération*, R_1 et R_2 sont deux nombres aléatoires tirés uniformément dans l'intervalle [0,1]. Pour éviter que les particules se déplacent trop rapidement d'une région à une autre dans l'espace de recherche, la vitesse de déplacement pendant une itération est serrée dans l'intervalle $[-V_{max}, V_{max}]$ donnée par l'équation (1.41) et de même pour son déplacement donné par l'équation (1.42) [42] :

$$V(t) = sign\big(V(t)\big) \cdot \min\left(|V(t)|, V_{max}\right) \quad (1.41)$$

$$p(t) = sign\big(p(t)\big) \cdot \min(|p(t)|, p_{max}) \quad (1.42)$$

Une étude sur le comportement de l'OEP suivant les valeurs de V_{max} est disponible dans [43].

1.3.5.4 Algorithme de base

L'OEP est un algorithme à population. Il commence par une initialisation aléatoire de l'essaim dans l'espace de recherche (N est le nombre de particules de l'essaim). A chaque itération de l'algorithme, chaque particule est déplacée suivant (1.39) et (1.40). Une fois le déplacement des particules effectué, les nouvelles positions sont évaluées. Les p_{best} ainsi que g_{best} sont alors mis à jour. Cette procédure est résumée par l'Algorithme suivant :

1. **Initialisation** aléatoire des positions et des vitesses de chaque particule
2. **Pour** chaque particule i, $p_{best}^i = p^i$
3. **Tant que** le critère d'arrêt n'est pas atteint **faire**

   Pour i = 1 à N **faire**

   Déplacement de la particule à l'aide de (1.39) et (1.40)

   Modification de la position et la vitesse de chaque particule par (1.41) et (1.42)

   Evaluation des positions

   Si $f(p^i) < f(p_{best}^i)$

   $p_{best}^i = p^i$

   Fin Si

   Si $f(p_{best}^i) < f(g_{best})$

   $g_{best} = p_{best}^i$

   Fin Si

   Fin Pour

Fin Tant que

Le critère d'arrêt peut être différent suivant le problème posé. Si l'optimum global est connu a priori, on peut définir une erreur acceptable comme critère d'arrêt. Sinon, il est commun de fixer un nombre maximum d'évaluations de la fonction *objectif* ou un nombre maximum d'itérations comme critère d'arrêt.

1.4 Conclusions

Dans ce chapitre, les fondations nécessaires à la maîtrise des techniques intelligentes sont établies. Après avoir introduit des concepts de base sur lesquelles reposent les systèmes d'inférence floue, la structure générale d'un contrôleur flou ainsi que ses différents composants ont été décrits. La notion de modélisation floue ainsi que les différents types de modèles flous ont été présentés.

Les principaux avantages des techniques floues sont l'approche naturelle de la modélisation et la bonne interprétabilité de la description tout en employant des règles linguistiques.

Les métaheuristiques, comprenant notamment la méthode de recherche tabou, les algorithmes génétiques et la méthode d'optimisation par essaim particulaire sont des algorithmes d'optimisation de type stochastiques qui progressant vers un optimum par échantillonnage d'une fonction coût dont le but est la résolution de problèmes d'optimisation difficile. Grâce à la simplicité et la souplesse de leurs principes, ils peuvent être un outil d'optimisation et de conception des systèmes de contrôle des processus complexes dont la dynamique n'est pas encore maîtrisée.

Chapitre 2

Optimisation des Contrôleurs Flous par l'Approche Hybride AG-RT

2.1 Introduction

Une loi de commande peut être construite à partir d'une approche système basée sur un modèle mathématique linéaire ou non linéaire, lorsqu'un tel modèle est disponible. Cependant, cela n'est pas toujours possible, en effet, il existe des situations où le modèle mathématique, pour une raison ou une autre, n'est pas disponible. Les approches conventionnelles sont alors inadaptées et les contrôleurs basés sur les systèmes d'inférence floue et les réseaux de neurones artificiels sont tout indiquées pour ce genre de situations.

2.2 Conception des Contrôleurs Flous

Les premiers développements en commande floue ont été initialisés par Mamdani [31, 44]. L'idée de base consistait à exploiter l'expérience des opérateurs humains pour construire une

loi de commande. Un jeu de règles floues traduit alors le comportement des opérateurs en termes de stratégie de commande. Une base de règle anti-diagonale, dite de Mac Vicar-Whelan peut être utilisée [45]. De nombreuses études ont permis de justifier l'écriture des règles par analogie au régime glissant [46] ou par une analyse qualitative du comportement dans le plan de phase [47]. Ces structures de commande sont des équivalents structuraux des contrôleurs classiques [48].

De par leur structure, les systèmes flous de Takagi Sugeno fournissent une expression analytique simple de la sortie générée en fonction des entrées considérées. Cette propriété permet alors d'exploiter des mécanismes d'optimisation numérique pour la synthèse de contrôleurs flous. Ainsi, Sugeno et Takagi [27] utilisent les algorithmes des moindres carrés alors que Bersini [49] utilise une méthode de descente de gradient pour minimiser un critère quadratique. Par la suite, de nombreux travaux ont été publiés sur l'utilisation des contrôleurs flous de type TS dans la commande des procédés non linéaires.

Les métaheuristiques, particulièrement les algorithmes évolutionnaires tels que les algorithmes génétiques ont été largement utilisés pour la conception des contrôleurs flous voir par exemple [50]-[54]. Cependant, les algorithmes génétiques sont coûteux en temps de calcul, car ils gèrent simultanément plusieurs solutions comme ils peuvent être piégés dans des minima locaux. Afin d'améliorer la qualité des solutions et d'accélérer la convergence des AG, il est intéressant d'hybrider les AG avec un algorithme de recherche locale. Le rôle de la méthode de recherche locale est d'explorer en profondeur un domaine particulier pour guider la recherche de l'algorithme génétique dans ce domaine.

Par rapport aux AG, la recherche Tabou (RT) présente une particularité intéressante. Elle a une mémoire appelée *liste tabou*, qui stocke les bonnes solutions et elle est caractérisé par sa capacité à éviter les solutions locales ainsi d'éviter le recyclage en utilisant la mémoire flexible de l'historique de recherches (cf. chapitre 1).

Dans ce chapitre, les algorithmes génétiques seront combinés avec la recherche tabou afin d'optimiser des contrôleurs flous de type Takagi-Sugeno (TS) à conclusion constante pour des systèmes monovariables et des systèmes multivariables. Les résultats seront comparés avec ceux des méthodes de la littérature.

2.3 Structure du Contrôleur Flou à Optimiser

On considère un contrôleur flou de type TS d'ordre zéro à deux entrées, l'erreur $e(t)$ et sa variation $\Delta e(t)$ et une sortie $u(t)$, la commande appliquée au système (figure 2.1):

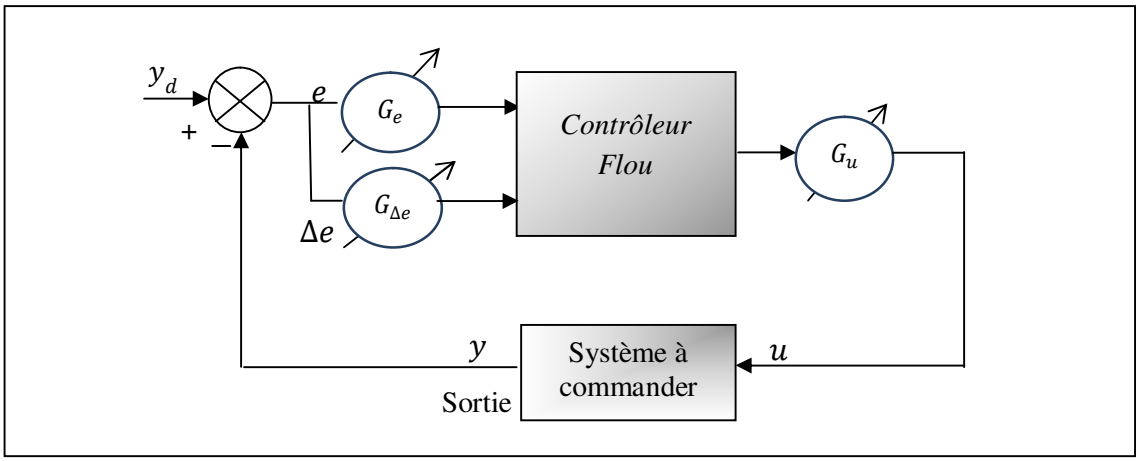


Fig. 2.1 : Structure du contrôleur flou à optimiser

Où, $e(t) = y_d(t) - y(t)$ est la différence entre la sortie désirée $y_d(t)$ et la sortie mesurée du système commandé $y(t)$. Pour avoir une flexibilité dans l'implantation du régulateur, les univers de discours des entrées et de sortie sont limités à un intervalle [-1, 1] déterminé par la normalisation des entrées et de sortie pour ce faire, des gains d'adaptations (facteurs d'échelles) sont utilisés pour avoir la dynamique désirée.

Pour l'optimisation paramétrique d'un contrôleur flou, on doit alors définir :

- Le type et le nombre des ensembles flous pour chaque variable d'entrée et de sortie,
- l'ensemble des paramètres des fonctions d'appartenance qui constituent la partition floue de la variable linguistique,
- La structure des règles floues,
- Le type de mécanisme d'inférence, les opérateurs de connections et la méthode de défuzzification.

Dans cette thèse, la structure de fonction d'appartenance adoptée a seulement trois ensembles flous pour chaque entrée, et un univers de discours normalisé sur [-1, 1], comme indiqué sur la figure 2.2.

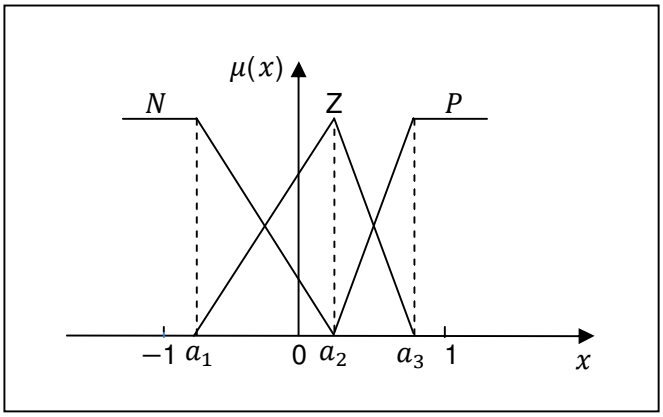


Fig. 2.2 : Partition des entrées du contrôleur flou

Où, x est la variable à fuzzifier. *N, Z* et *P* sont des ensembles flous de la variable d'entrée ; leur signification est successivement Négative, Zéro et Positive. Puisque le contrôleur flou est de type TS d'ordre zéro, les conclusions de règles floues sont des constantes.

2.3.1 La base de règles floues

Sur la base de description du système à régler avec des variables linguistiques, et de la définition des fonctions d'appartenance pour les variables d'entrée et de la sortie, on peut établir les règles d'inférence.

La $i^{\text{ème}}$ règle, notée R_i , d'un contrôleur flou de type TS d'ordre zéro est présentée sous la forme suivante :

$$R_i : Sie(k)\ est\ A_i\ Et\ \Delta e(k)\ est\ B_i\ Alors\ u(k)\ est\ s_i \tag{2.3}$$

Où, $e(k)$ et $\Delta e(k)$ sont les variables d'entrée, k est le temps, $u(k)$ est la sortie du contrôleur flou, A_i et B_i sont les ensembles flous des variables d'entrée et s_i sont des valeurs réelles.

La décision du nombre de règles floues est un enjeu très important car il joue un rôle clé dans la commande et la modélisation floue des systèmes. Il n'existe généralement pas de procédure systématique et efficace pour choisir le nombre de règles le plus approprié. Un nombre raisonnable de règles floues, sans perdre trop d'informations sur le système à commander ou à modéliser doit être soigneusement obtenue.

Dans cette thèse, On a proposé seulement trois règles exprimées comme suit:

$R_1 : Si\ e(k)\ est\ N(-1, a_1, a_2)\ ET\ \Delta e(k)\ est\ N(-1, b_1, b_2)\ Alors\ u(k) = s_1.$

$R_2 : Si\ e(k)\ est\ Z(a_1, a_2, a_3)\ ET\ \Delta e(k)\ est\ Z(b_1, b_2, b_3)\ Alors\ u(k) = s_2.$

$R_3 : Si\ e(k)\ est\ P(a_2, a_3, 1)\ ET\ \Delta e(k)\ est\ P(b_2, b_3, 1)\ Alors\ u(k) = s_3.$

Où s_1, s_2, s_3 sont des valeurs réelles qui déterminent successivement les singletons flous N, Z, P de la sortie du contrôleur flou.

2.3.2 Le moteur d'inférence et la défuzzification

Dans le mécanisme d'inférence, Le *ET* dans la règle floue est implémenté par le produit algébrique dans la théorie de la logique floue (selon Larsen). Ainsi, étant donné un ensemble de données d'entrée $\vec{x} = (e, \Delta e)$, le degré d'activation $\beta_i(\vec{x})$ de la règle i est calculée par :

$$\beta_i(\vec{x}) = \mu_{A_i}(e(k)) \cdot \mu_{B_i}(\Delta e(k)) \tag{2.2}$$

S'il y a r règles dans un CF, la sortie résultante de l'ensemble des règles est donnée par la moyenne des sorties individuelles pondérées par le degré d'activation des règles, soit :

$$u = \frac{\sum_{i=1}^{r} \beta_i(\vec{x}) s_i}{\sum_{i=1}^{r} \beta_i(\vec{x})} \tag{2.3}$$

Où s_i est la valeur de la conclusion de la $i^{ème}$ règle.

L'optimisation d'un CF comprend la détermination de tous les paramètres de chaque règle floue.

2.4 Méthode d'Optimisation

2.4.1 Structure d'optimisation

Le diagramme de la commande en boucle fermée est schématisé par la figure 2.3. Il consiste en quatre blocs principaux:

- Bloc structurel représenté par le contrôleur flou proposé.
- Bloc du système à commander.
- Bloc d'optimisation caractérisé par un algorithme hybride.
- Bloc décisionnel caractérisé par le critère de performances désirées.

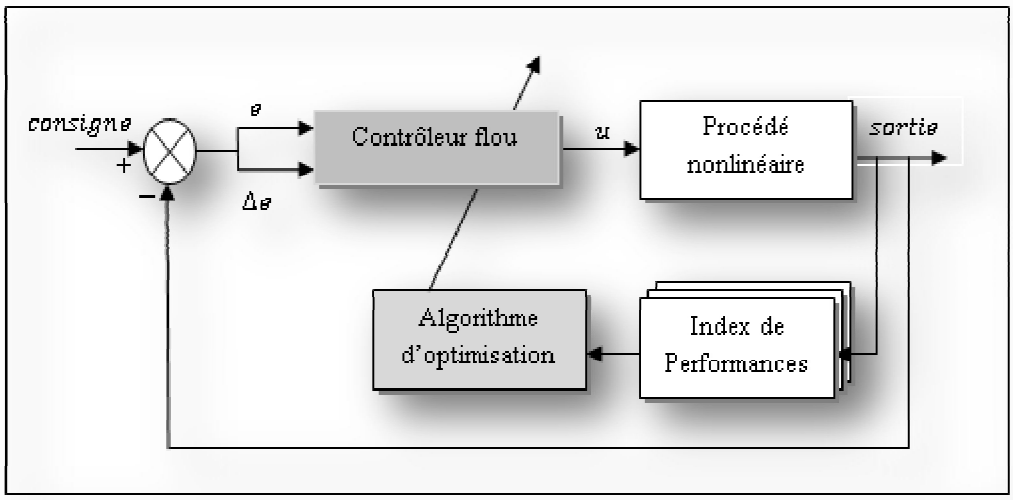


Fig. 2.3: Structure d'optimisation et de contrôle

L'interaction mutuelle entre les différents blocs de la structure de la figure 2.3 est illustrée par la procédure suivante :

1. Génération d'une population initiale des chromosomes caractérisant les paramètres du contrôleur.
2. Pour tous les chromosomes :
   - Evaluer la fonction objective.
   - Classifier les chromosomes selon leur aptitude.
   - Construction d'une nouvelle population par application des opérateurs génétiques adaptés au codage des chromosomes.

L'étape 2 est répétée jusqu'à ce qu'un nombre maximum de générations soit effectué. Après le processus d'évolution, la génération finale de l'algorithme se compose des individus bien adaptés et qui fournissent des solutions 'optimales' ou proches.

L'inclusion des contraintes de conception dans le processus d'optimisation permet de préserver la sémantique des règles floues. Pour cela, des contraintes sur les limites du vecteur des paramètres à identifier, et des limites sur les grandeurs de commande sont imposées.

2.4.2 Représentation du chromosome

Pour appliquer les AG directement ou couplée avec d'autres métaheuristiques, le problème de la représentation du chromosome est posé. La première décision que le concepteur doit faire est de savoir comment représenter une solution dans une structure chromosomique. Selon la figure 2.4, le chromosome proposé est composé de neuf paramètres. Ces paramètres représentent les emplacements de point de départ, point de sommet et le point de fin pour une fonction d'appartenance en forme de triangle pour les entrées d'un CF et les singletons flous pour sa sortie , tout en respectant la contrainte suivante:

$$\begin{cases} a_1 < a_2 < a_3 \\ b_1 < b_2 < b_3 \\ s_1 < s_2 < s_3 \end{cases} \tag{2.4}$$

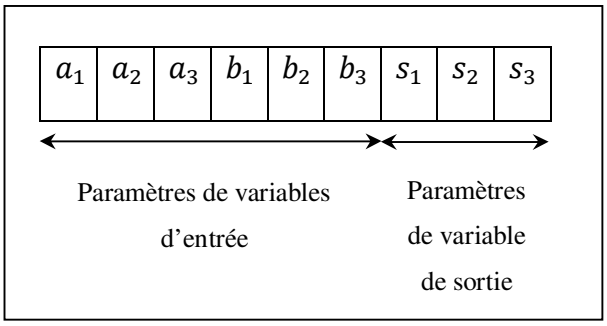


Fig. 2.4 : Structure du chromosome

2.4.3 L'algorithme principal des AG

Afin de construire un algorithme d'optimisation à base des AG, il faut bien préparer la plateforme suivante :

- Définir la configuration initiale (taille de la population, critère d'arrêt, probabilités de croisement et de mutation, …etc.).
- Mode du codage approprié (binaire, réel, …etc.).

- Méthode de génération de la population initiale.
- Définir la fonction objective.
- Adapter les opérateurs génétiques au mode de codage choisi.

Les AG utilisés dans cette thèse ont les caractéristiques fondamentales suivantes :

- Le codage utilisé pour le chromosome est le codage réel, puisque la résolution du problème d'optimisation sera plus efficace avec les AG à codage réel.
- La population initiale sera générée aléatoirement pour couvrir tous l'espace de recherche.
- La méthode de la "loterie biaisée" est choisie pour l'opération de sélection, pour que Les individus qui ont une grande *fitness* aient plus de chance d'être sélectionnés.
- Le croisement en 2 points est utilisé pour permettre d'introduire plus ou moins de diversité. En effet, plus le nombre de points de croisements sera grand et plus la probabilité de croisement sera élevée plus il y aura d'échange de segments, donc d'échange de paramètres et d'information.
- La mutation uniforme est choisie car elle est la plus simple et la plus adaptée au codage réel.
- La sélection finale pour la nouvelle génération sera par compétition, pour que les meilleurs individus (enfants et parents) participent dans la nouvelle génération.

L'objectif global du système de commande est de réduire au minimum l'erreur à chaque instant k entre la réponse réelle du système $y(k)$ et le point de consigne $y_d(k)$. Il existe plusieurs indices de performance permettant de mesurer la qualité d'un réglage donné, ceux utilisés dans cette thèse sont:

- Critère de l'erreur quadratique moyenne définie par :

$$Fit = EQM = \left(\frac{1}{nT}\right) \sum_{k=1}^{n} e^2(k) \tag{2.5}$$

Où, n est le nombre total d'échantillons, $e(k) = y_d(k) - y(k)$ et T est le temps d'échantillonnage.

- Critère de la somme de valeurs absolues de l'erreur défini par :

$$Fit = SAE = \sum_{k=1}^{n} |e(k)| \tag{2.6}$$

Les algorithmes génétiques permettent de déterminer le maximum d'une fonction, alors la fonction coût de l'AG est calculée comme suit :

$$f_c = 1/(1 + Fit) \tag{2.7}$$

2.4.4 Caractéristiques de l'algorithme de la RT

Avant de construire un algorithme d'optimisation de la recherche tabou, il faut d'abord définir:

- la configuration initiale (taille de la liste Tabou, nombre de solutions voisines, critère d'arrêt).
- la fonction objective.
- La méthode de génération de la solution initiale et son voisinage.

Les caractéristiques fondamentales de l'algorithme de la RT utilisé pour la construction de l'algorithme hybride sont :

- Génération d'une solution initiale et solutions voisines aléatoires, en prenant en compte tous l'espace de recherche respectant ainsi les critères d'intensification et de diversification.
- La liste tabou est une mémoire FIFO (*First Input First Output*), sa taille est souvent proportionnel au nombre d'itérations.
- La fonction objective (*fitness function*) da la recherche Tabou va automatiquement suivre le choix du critère de performance des AG calculé soit par la formule (2.5) ou par (2.6).

2.4.5 Principe de fonctionnement de l'algorithme hybride AG-RT

Afin d'améliorer la convergence des algorithmes génétiques quatre algorithmes sont proposés:

- Dans le premier algorithme AG-RT1, la RT est intégrée après l'opération de croisement,

- Dans le deuxième algorithme AG-RT2, la RT est intégrée après l'opération de mutation,
- Dans le troisième algorithme AG-RT3, la RT est intégrée après l'opération de sélection,
- Dans le quatrième algorithme AGE-RT, la RT est intégrée après la sélection finale de la nouvelle génération.

Le tableau 2.1 définie les acronymes utilisés dans les 4 algorithmes.

Paramètre	Signification
$\max\_gen$	Nombre maximum de générations de l'AG
N	Taille de la population
p_c	Probabilité de croisement
p_m	Probabilité de mutation
ch	Chromosome
s_{opt}	Solution optimale
$Elit$	Solution élitiste
$Elit_{best}$	Meilleure solution élitiste
s_{init}	Solution initiale de la RT
k	Nombre maximum d'itérations de la RT
N_v	Nombre de solutions voisines
T	Taille de la liste tabou

Tab. 2.1 : Paramètres de l'algorithme AG-RT

2.4.5.1 L'algorithme hybride AG-RT1

A chaque génération de l'algorithme génétique, la recherche Tabou est introduite après l'opération de croisement pour chercher le meilleur voisin de chaque solution trouvée. Le résultat sera une nouvelle population optimisée qui va continuer son évolution par les autres opérations génétiques pour produire des nouvelles générations plus adaptées au problème.

Le pseudo-code de la procédure AG-RT1 est le suivant :

Début *Algorithme Génétique*

Définir max $\_gen$ p_c, p_m, N

Générer la population initiale

Pour $i = 1$ à max $\_gen$ **faire**

Evaluer chaque individu de la population par l'eq. (2.7)

Sélectionner les parents de la population

Produire les enfants des parents sélectionnés par l'opération de croisement

Début *Recherche Tabou*

Définir k, N_v, T

Initialiser la liste Tabou $T = \emptyset$, $i = 0$

Tant que $i < k$

Pour tout ch de la population produite **faire**

Générer N_v solutions voisines de ch

Evaluer chaque solution voisine

Prendre la solution voisine la plus adaptée

Mettre à jour la liste Tabou

Fin Pour

$i = i + 1$

Fin Tant que

Fin

Muter les individus de la population résultante

Sélectionner les enfants et les parents les plus adapté pour la nouvelle génération

Fin Pour

Retourner la meilleure solution s_{opt}

Fin

La solution optimale s_{opt} représente le chromosome qui contient les paramètres du contrôleur flou optimisé.

2.4.5.2 L'algorithme hybride AG-RT2

Dans cet algorithme, la recherche Tabou est introduite après l'opération de mutation pour chercher le meilleur voisin de chaque solution trouvée. Le pseudo-code de la procédure AG-RT2 est le suivant :

Début *Algorithme Génétique*

Définir max $\_gen, p_c, p_m, N$

Générer la population initiale

Pour $i = 1$ à max $\_gen$ **faire**

Evaluer chaque individu de la population par l'eq. (2.7)

Sélectionner les parents de la population

Produire les enfants des parents sélectionnés par l'opération de croisement

Muter les individus de la population résultante

Début *Recherche Tabou*

Définir k, N_v, T, Initialiser la liste tabou $T = \emptyset$, $i = 0$

Tant que $i < k$

Pour tout ch de la population produite **faire**

Générer N_v solutions voisines de ch

Evaluer chaque solution voisine

Prendre la solution voisine la plus adaptée

Mettre à jour la liste Tabou

Fin Pour

$i = i + 1$

Fin Tant que

Fin

Sélectionner les enfants et les parents les plus adapté pour la nouvelle génération

Fin Pour

Retourner la meilleure solution s_{opt}

Fin

2.4.5.3 L'algorithme hybride AG-RT3

Dans cet algorithme, la recherche Tabou est introduite après l'opération de sélection des parents les plus adaptés pour améliorer les individus parents de la population. La RT va chercher le meilleur voisin de chaque individu de la population sélectionnée :

Début *Algorithme Génétique*

Définir max $\_gen, p_c, p_m, N$

Générer la population initiale

Pour $i = 1$ à max $\_gen$ **faire**

Evaluer chaque individu de la population par l'eq. (2.7)

Sélectionner les parents de la population

Début *Recherche Tabou*

Définir k, N_v, T, Initialiser la liste tabou $T = \emptyset$, $i = 0$

Tant que $i < k$

Pour tout ch de la population produite **faire**

Générer N_v solutions voisines de ch

Evaluer chaque solution voisine

Prendre la solution voisine la plus adaptée

Mettre à jour la liste Tabou

Fin Pour

$i = i + 1$

Fin Tant que

Fin

Produire les enfants des parents sélectionnés par l'opération de croisement

Muter les individus de la population résultante

Sélectionner les enfants et les parents les plus adapté pour la nouvelle génération

Fin Pour

Retourner la meilleure solution s_{opt}

Fin

2.4.5.4 L'algorithme hybride AGE-RT

Dans cet algorithme, la recherche Tabou est introduite après la sélection finale de la nouvelle génération. Le pseudo-code de la procédure AGE-RT est le suivant :

Début *Algorithme Génétique*

Définir max $\_gen, p_c, p_m, N$

Générer la population initiale

Pour $i = 1$ à max $\_gen$ **faire**

Evaluer chaque individu de la population par l'eq. (2.7)

Sélectionner les parents de la population

Produire les enfants des parents sélectionnés par l'opération de croisement

Muter les individus de la population résultante

Sélectionner les enfants et les parents les plus adapté pour la nouvelle génération et enregistrer le meilleur individu comme $Elit$

Début *Recherche Tabou*

Définir k, N_v, T, Initialiser la liste tabou $T = \emptyset$, $i = 0$

$s_{init} = Elit$

Tant que $i < k$

Générer N_v solutions voisines de s_{init}

Evaluer chaque solution voisine

Prendre la solution voisine la plus adaptée comme $Elit_{best}$

Mettre à jour la liste Tabou

$i = i + 1$

Fin Tant que

Fin

Enregistrer le meilleur individu comme $Elit_{best}$ et insérer le dans la nouvelle génération

Fin Pour

Retourner la meilleure solution s_{opt}

Fin

Le meilleur individu de la nouvelle génération, $Elit$, est introduit dans l'algorithme de la RT pour améliorer sa qualité en cherchant dans son voisinage une autre meilleure solution $Elit_{best}$ plus adaptée.

L'élitisme est un mécanisme introduit pour conserver les bonnes solutions lors du passage de la génération courante à la prochaine génération. Conserver ces solutions pour les générations futures permet d'améliorer les performances des algorithmes sur certains problèmes.

2.5 Simulations

Dans le but de tester et évaluer les performances des algorithmes proposés nous avons effectué des applications en simulation sur deux systèmes non linéaires monovariables, le pendule inversé et un bain d'eau et deux systèmes non linéaires multivariables, le simulateur d'hélicoptère et le double pendule inversé. A chaque type de système non linéaire est appliquée une forme spécifique de loi de commande développée. Cette différence est due, principalement, à la nature et à la complexité du procédé à commander. L'objectif de commande est de permettre aux états du système de suivre une trajectoire de référence prédéterminée.

2.5.1 Commande floue des systèmes monovariables

2.5.1.1 Commande du pendule inversé

2.5.1.1.a Modèle mathématique du pendule inversé

Le pendule inversé est un système académique simple permettant d'étudier la dynamique de systèmes instables plus complexes, comme celle d'une fusée sur son pas de tir ou d'un bateau. En exerçant une force horizontale F sur le chariot, il résulte une translation de x mètres de celui-ci ainsi qu'une rotation de θ radian du pendule. Dans cette étude, nous considérons le système de pendule inversé comme un système SISO (single input, single

output) ; seulement la régulation de l'angle θ est prise en compte. La figure 2.5 montre la structure du pendule inversé étudié. La dynamique du système est caractérisée par l'expression (2.8) [55] :

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = f(x_1, x_2) + b((x_1, x_2)\, u \tag{2.8}$$

Avec :

$$f(x_1, x_2) = \frac{g\, sin(x_1) - (m\, l\, x_2^2 \cos(x_1)\ \sin(x_1)) / (m_c + m)}{l\left(\frac{4}{3} - m\, cos^2(x_1)\right) / (m_c + m)}$$

$$b(x_1, x_2) = \frac{\cos(x_1) / (m_c + m)}{l\left(\frac{4}{3} - m\, cos^2(x_1)\right) / (m_c + m)}$$

Où, u indique la force appliquée (la commande) [N], x_1 est l'angle que fait le pendule avec la verticale [rad], x_2 est la vitesse angulaire [rad/s], g est l'accélération gravitationnelle = 9.8 m/s^2, m est la masse du segment = 0.1 Kg, l indique la demi-longueur du pendule = 0.5 m, m_c est la masse du chariot soutenant le pendule = 1 Kg.

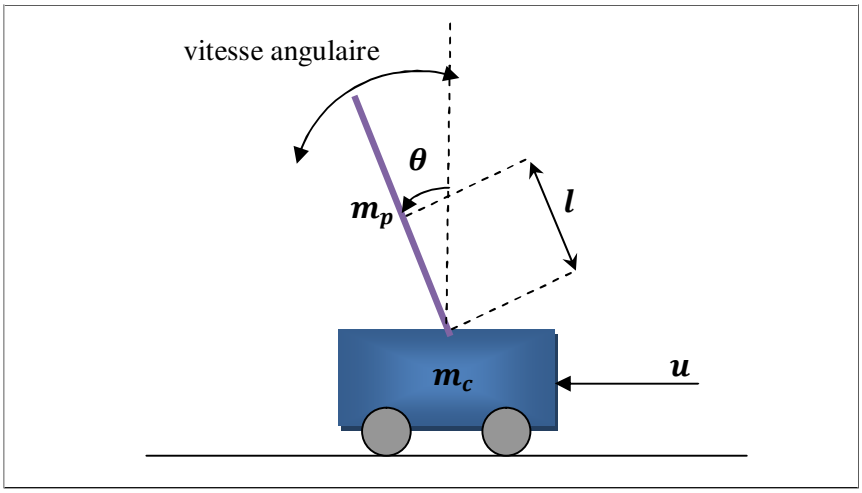


Fig. 2.5 : Structure du pendule inversé

Les valeurs spécifiques de chaque algorithme hybride AG-RT pour l'optimisation du CF du pendule inversé ainsi que les contraintes imposées sont données par le tableau 2.2. Pour illustrer les performances de la méthode proposée, nous considérons la commande en régulation du système de pendule inversé dont la dynamique est donnée par l'expression (2.8).

Caractéristique	Valeur
Taille de la population (N)	10
Nombre de générations ($\max\_gen$)	10
Probabilité de croisement (p_c)	0.9
Probabilité de mutation (p_m)	0.07
Facteurs d'échelle ($G_e, G_{\Delta e}, G_u$)	(1.5, 0.4, 170)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[-50[N], 50[N]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	4
Nombre d'itérations de la RT (k)	5

Tab. 2.2 : valeurs spécifiques de l'algorithme hybride d'optimisation AG-RT

2.5.1.1.b Commande en régulation

Le contrôleur flou optimisé doit ramener et maintenir le pendule dans sa position d'équilibre correspondante à un angle $\theta = 0$ [rad] et une vitesse angulaire $\dot{\theta} = 0$ [rad/s]. Le chariot peut se déplacer à droite ou à gauche, sans prise en compte de sa position. Les conditions initiales du système sont générées, aléatoirement. Les valeurs considérées sont (0.17, 0.22, 0.5, 0.8 [rad]) pour l'angle. Dans le cas illustré par la suite, on a $(x_1(0), x_2(0)) =$ (0.22 [rad], 0[rad/s]). L'indice de performance, Fit dans ce cas est l'erreur quadratique moyenne calculée par l'équation 2.5.

Les algorithmes génétiques permettent de déterminer le maximum d'une fonction, alors la fonction coût, f_c de l'AG est calculée par l'équation 2.7.

- ***Optimisation Par l'algorithme AG-RT1***

En suivant la stratégie d'optimisation décrite précédemment, l'évolution de la fonction coût au cours de l'exécution de l'algorithme hybride AG-RT1 est illustrée par la figure 2.6. La base de règles floues obtenue après optimisation est comme suit :

$R_1 : Si\ e\ est\ N(-1,-0.76,0.23)\ ET\ \Delta e\ est\ N(-1,-0.53,-0.50)\ Alors\ u = -0.92.$

$R_2 : Si\ e\ est\ Z(-0.76,0.23,0.76)\ ET\ \Delta e\ est\ Z(-0.53,-0.50,\ 0.59)\ Alors\ u = 0.1.$

$R_3 : Si\ e\ est\ P(0.23,0.76,1)\ ET\ \Delta e\ est\ P(-0.50,\ 0.59,1)\ Alors\ u = 0.66.$

Les réponses du pendule soumis au contrôleur flou optimisé obtenues pour les paramètres nominaux du système et différentes conditions initiales sont présentées sur la figure 2.7. Les résultats montrent la convergence de l'angle et de la vitesse angulaire vers l'état d'équilibre (0 [rad], 0 [rad/s]) à partir d'un état initial quelconque.

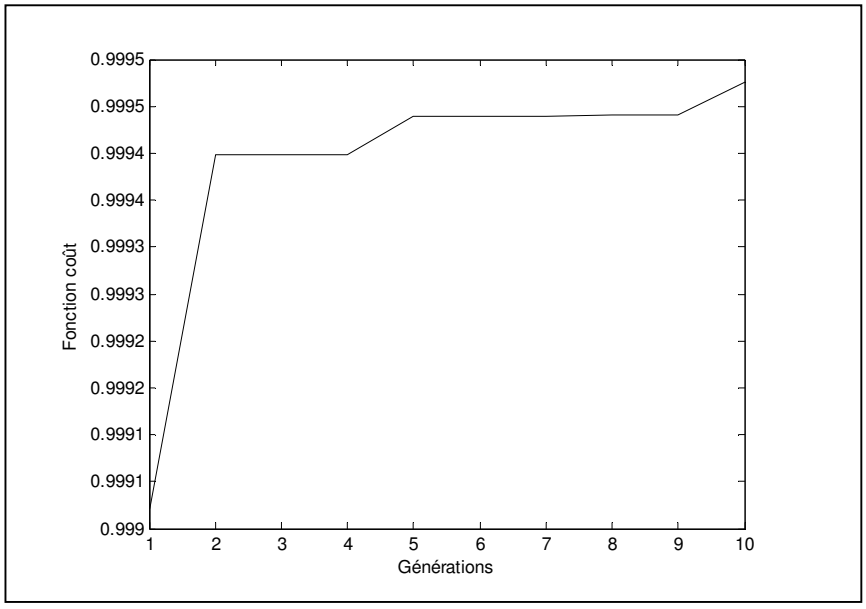


Fig. 2.6 : Evolution de la fonction coût

La robustesse du contrôleur est justifiée par sa capacité de bien face au changement des conditions initiales de fonctionnement ainsi que le changement des paramètres du pendule. Ce dernier est illustré par un changement du poids du pendule (voir la figure 2.8). Les valeurs considérées du poids sont (0.05, 0.1, 0.2, 0.25, 0.35, 0.45 [Kg]). D'après les figures 2.7 et 2.8, il est clair que la stratégie proposée de conception du contrôleur flou accomplit efficacement les performances désirées.

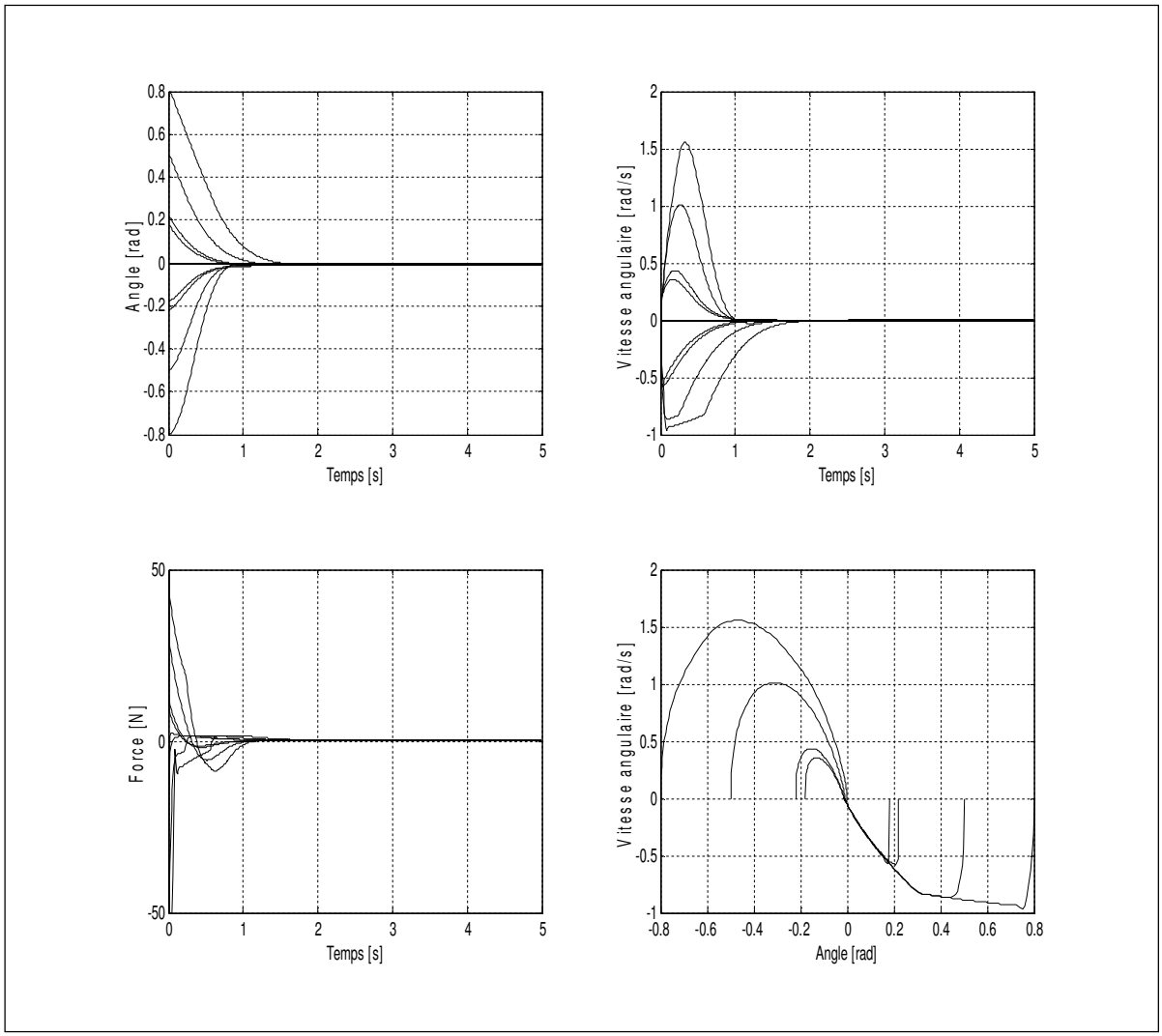


Fig. 2.7 : Réponses du système pour différentes conditions initiales.

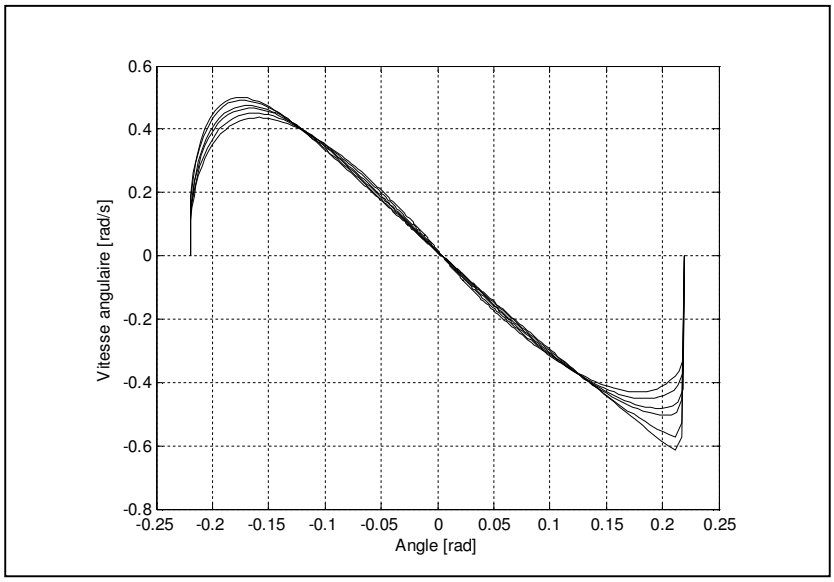


Fig. 2.8 : Réponses aux changements de poids du segment : Plan de phase

- ***Optimisation par l'algorithme AG-RT2***

En appliquant la même procédure du premier algorithme, l'évolution de la fonction coût est donnée par la figure 2.9. La base de règles floues obtenue à la fin de l'exécution de l'algorithme AG-RT2 est comme suit :

$R_1 : Si\ e\ est\ N(-1,-0.93,0.44)\ ET\ \Delta e\ est\ N(-1,-0.73,0.42)\ Alors\ u = -0.95.$

$R_2 : Si\ e\ est\ Z(-0.93,0.44,0.51)\ ET\ \Delta e\ est\ Z(-0.73,-0.42,\ 0.92)\ Alors\ u = -0.02.$

$R_3 : Si\ e\ est\ P(0.44,0.51,1)\ ET\ \Delta e\ est\ P(-0.42,\ 0.92,1)\ Alors\ u = 0.61.$

Les réponses du pendule soumis au contrôleur flou optimisé obtenues pour les paramètres nominaux du système et différentes conditions initiales sont présentées sur la figure 2.10. Les résultats montrent la convergence de l'angle et de la vitesse angulaire vers l'état d'équilibre (0 [rad], 0 [rad/s]) à partir d'un état initial quelconque.

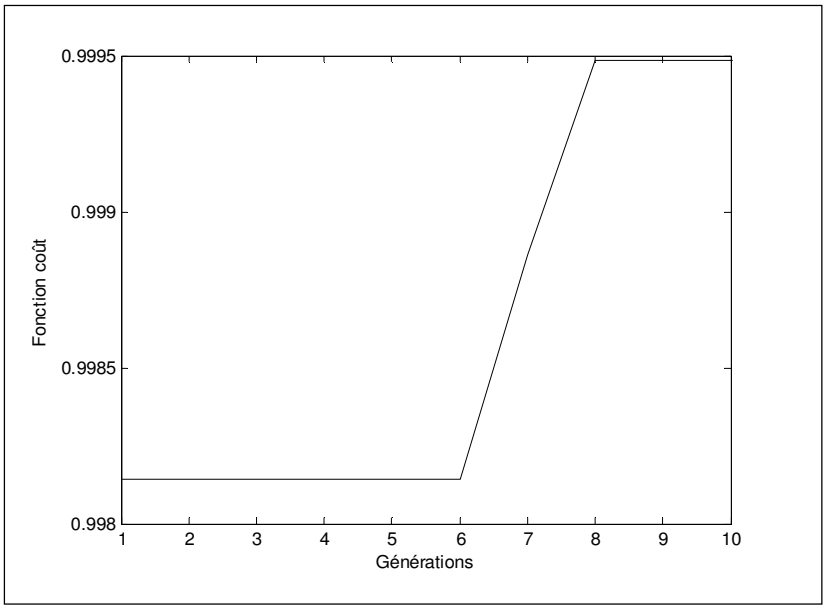


Fig. 2.9 : Evolution de la fonction coût

La robustesse du contrôleur est testée en changeant les conditions initiales de fonctionnement (figure 2.10), ainsi que le changement du poids du pendule (figure 2.11). D'après les figures 2.10 et 2.11, on conclue que la stratégie proposée de conception du contrôleur flou accomplit efficacement les performances désirées.

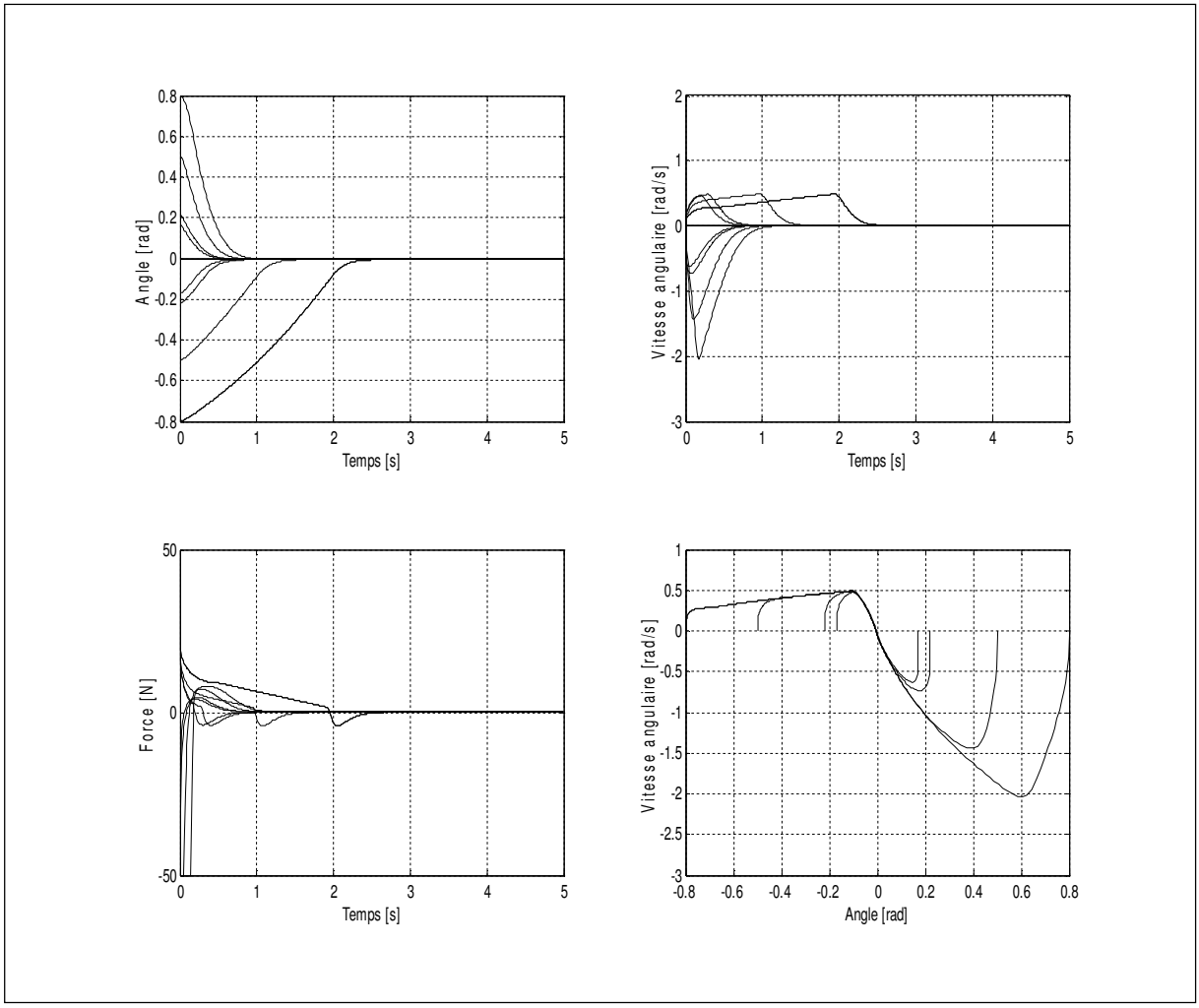


Fig. 2.10 : Réponses du système pour différentes conditions initiales.

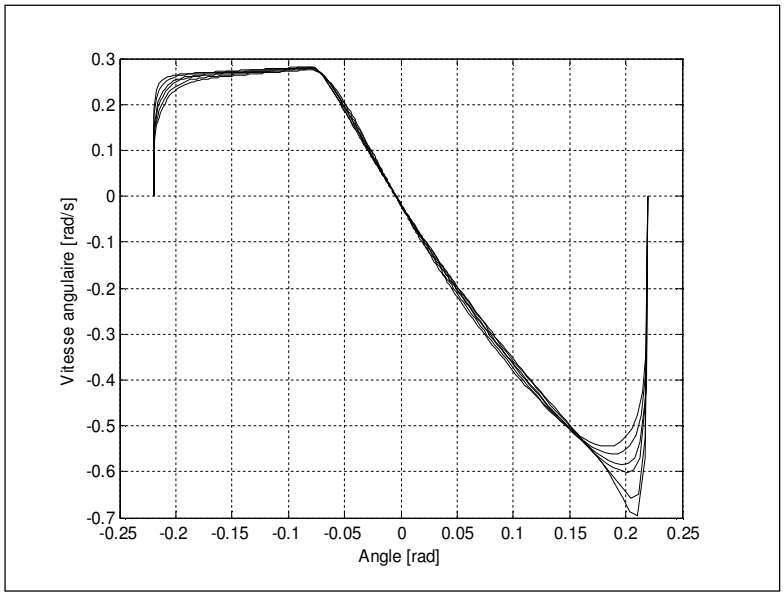


Fig. 2.11 : Réponses aux changements de poids du segment : Plan de phase

- ***Optimisation par l'algorithme AG-RT3***

L'évolution de la fonction coût est donnée par la figure 2.12. La base de règles floues obtenue à la fin de l'exécution de l'algorithme AG-RT3 est comme suit :

$R_1 : Si\ e\ est\ N(-1,-0.57,0.16)\ ET\ \Delta e\ est\ N(-1,-0.74,0.45)\ Alors\ u = -0.59.$

$R_2 : Si\ e\ est\ Z(-0.57,0.16,0.63)\ ET\ \Delta e\ est\ Z(-0.74,0.45,\ 0.81)\ Alors\ u = 0.00.$

$R_3 : Si\ e\ est\ P(0.16,0.63,1)\ ET\ \Delta e\ est\ P(0.45,\ 0.81,1)\ Alors\ u = 0.95.$

Les réponses du pendule soumis au contrôleur flou optimisé obtenues pour les paramètres nominaux du système et différentes conditions initiales sont présentées sur la figure 2.13. Les résultats montrent la convergence de l'angle et de la vitesse angulaire vers l'état d'équilibre à partir d'un état initial quelconque.

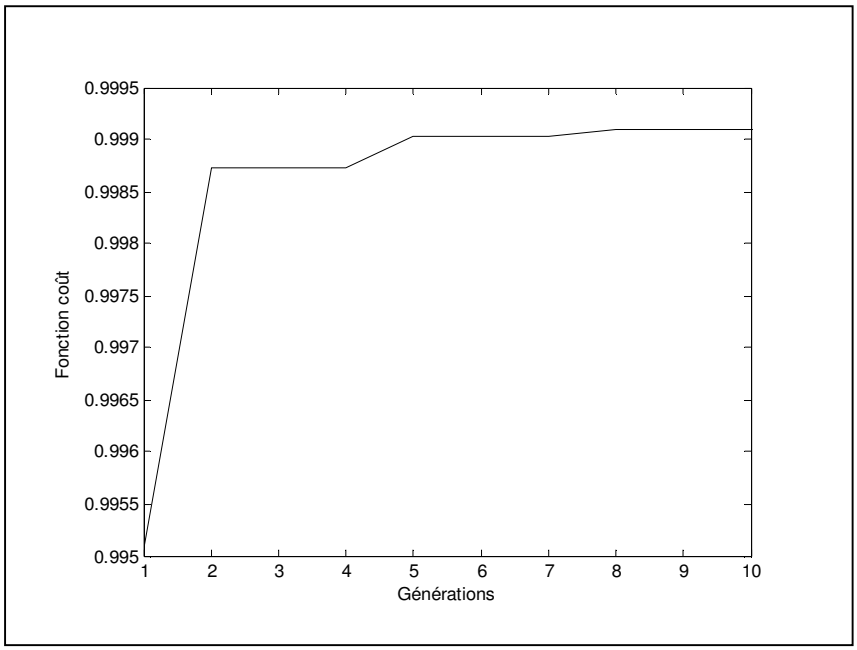


Fig. 2.12 : Evolution de la fonction coût

La robustesse du contrôleur est testée en changeant les conditions initiales de fonctionnement ainsi que le changement du poids du pendule. D'après les figures 2.13 et 2.14, on conclue que la stratégie proposée de conception du contrôleur flou accomplit efficacement les performances désirées.

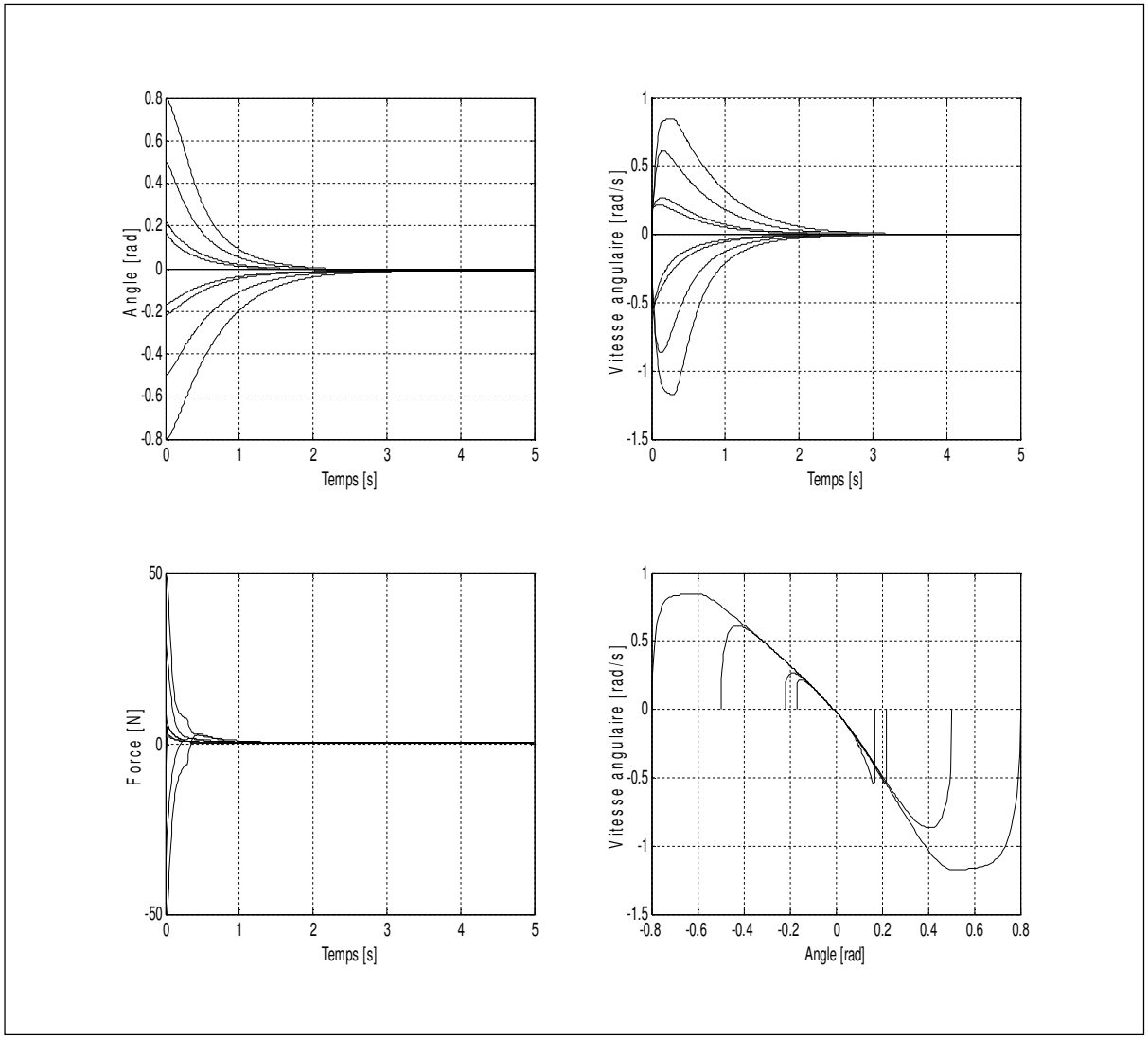


Fig. 2.13 : Réponses du système pour différentes conditions initiales.

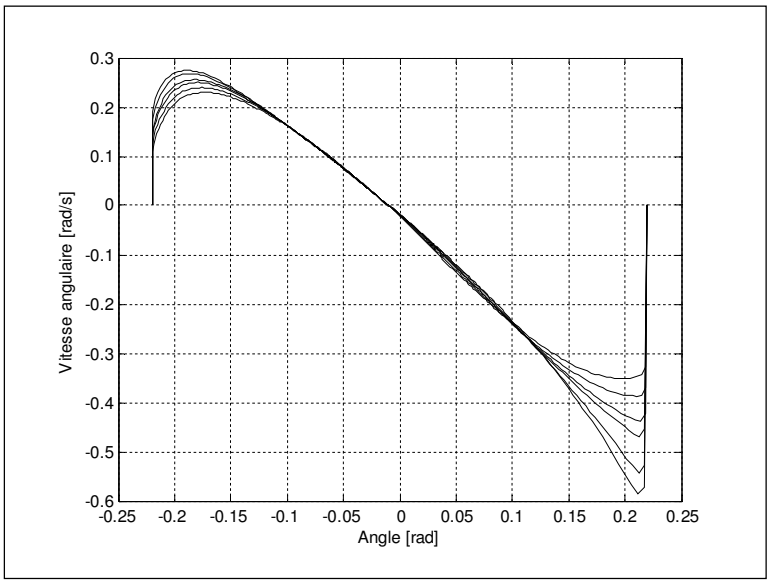


Fig. 2.14 : Réponses aux changements de poids du segment : Plan de phase

- ***Optimisation par l'algorithme AGE-RT***

L'évolution de la fonction coût est donnée par la figure 2.15. La base de règles floues obtenue à la fin de l'exécution de l'algorithme AGE-RT est comme suit :

$R_1 : Si\ e\ est\ N(-1,-0.53,0.00)\ ET\ \Delta e\ est\ N(-1,-0.63,-0.19)\ Alors\ u = -0.90.$

$R_2 : Si\ e\ est\ Z(-0.53,0.00,0.63)\ ET\ \Delta e\ est\ Z(-0.63,-0.19,\ 0.94)\ Alors\ u = 0.17.$

$R_3 : Si\ e\ est\ P(0.00,0.63,1)\ ET\ \Delta e\ est\ P(-0.19,\ 0.94,1)\ Alors\ u = 0.61.$

Les réponses du pendule soumis au contrôleur flou optimisé obtenues pour les paramètres nominaux du système et différentes conditions initiales sont présentées sur la figure 2.16. Les résultats montrent la convergence de l'angle et de la vitesse angulaire vers l'état d'équilibre à partir d'un état initial quelconque.

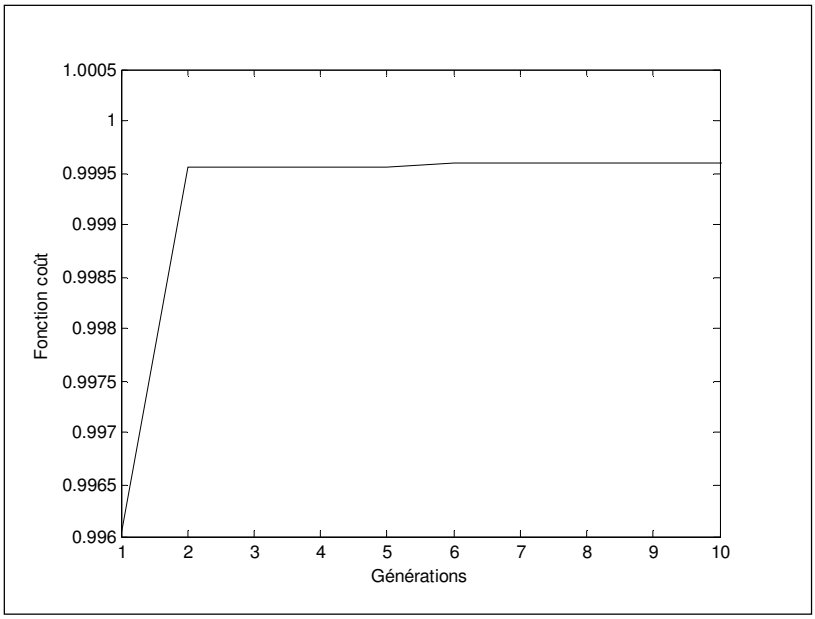


Fig. 2.15 : Evolution de la fonction coût

La robustesse du contrôleur est testée en changeant les conditions initiales de fonctionnement ainsi que le changement du poids du pendule. D'après les figures 2.16 et 2.17, on conclue que la stratégie proposée de conception du contrôleur flou accomplit efficacement les performances désirées.

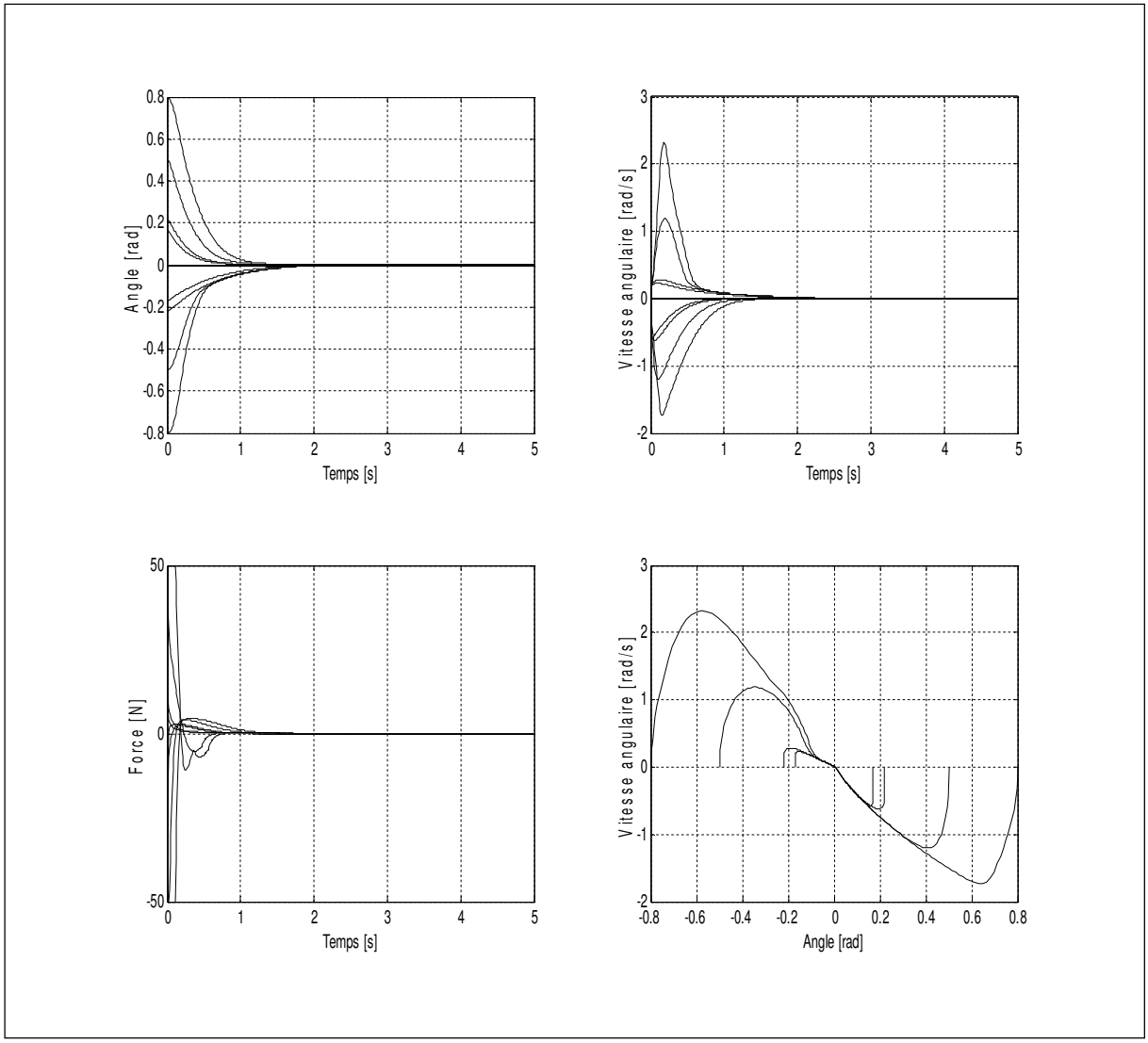


Fig. 2.16 : Réponses du système pour différentes conditions initiales

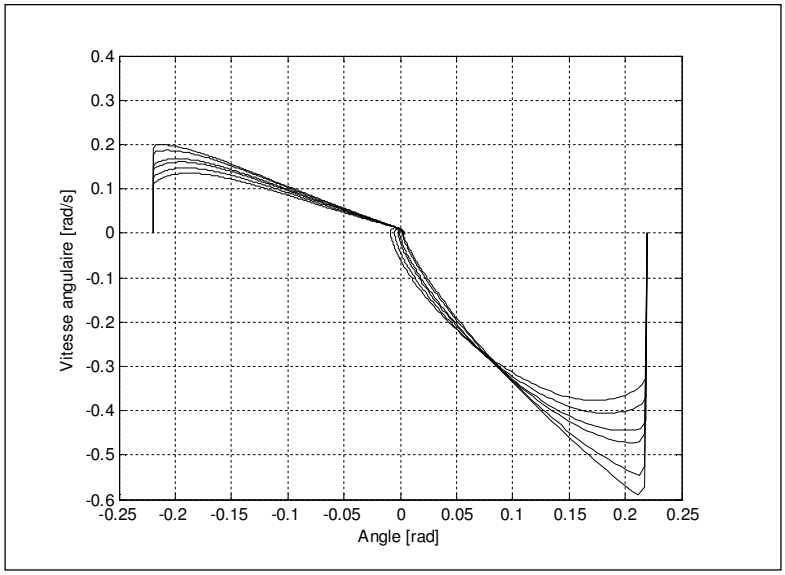


Fig. 2.17 : Réponses aux changements de poids du segment : Plan de phase

2.5.1.1.c Interprétation des résultats et discussion

Le tableau 2.3 présente les résultats obtenus pour la commande du système du pendule inversé en utilisant différentes méthodes d'optimisation. Ces résultats sont obtenus après exécution de 10 itérations de l'algorithme d'optimisation et pour la même base de règle (3 règles floues). D'après les figures et résultats obtenus, on remarque :

- ☞ Les contrôleurs flous donnés par quatre algorithmes arrivent à stabiliser le pendule dans un temps de réponse moins de 2.5 s et donnent une bonne robustesse pour le changement des paramètres du pendule.
- ☞ La fonction coût optimale est celle du quatrième algorithme AGE-RT, f_c = 0.9996 pour une erreur quadratique moyenne égale à 4×10^{-4}.
- ☞ Selon le tableau 2.3, le minimum temps de calcul est celui de l'algorithme AGE-RT, cela dû au principe de fonctionnement de l'algorithme. L'algorithme de la RT dans AGE-RT va trouver seulement le meilleur voisin de l'élitisme pour lui introduire dans la nouvelle génération, par contre dans les trois autres algorithmes, la RT va chercher le meilleur voisin de chaque individu de la population.

Méthode	EQM	Temps de calcul [s]
AG à codage réel [57]	0.00196	4.60
RT [57], [58]	0.00051	3.88
AG-RT1 [56]	0.00050	89.1
AG-RT2 [56]	0.00052	41.7
AG-RT3 [56]	0.00060	84.2
AGE-RT	0.00040	9.1

Tab. 2.3 : Comparaison des méthodes d'optimisation pour la commande du pendule inversé

D'après les résultats du tableau 2.3, on constate qu'il n y a pas de méthode d'optimisation qui garantit une bonne précision et un minimum temps de calcul simultanément. L'algorithme AGE-RT donne un bon compromis précision-minimum temps de calcul.

2.5.1.2 Commande de la température d'un bain d'eau

Le but de cette section est de commander la température d'un bain d'eau. Le modèle récurrent du système est donné par l'expression suivante [59, 60, 61] :

$$y(k+1) = e^{-\alpha T_s}\, y(k) + \frac{\frac{\beta}{\alpha}(1 - e^{-\alpha T_s})}{1 + e^{0.5y(k)-40}}\, u(k) + (1 - e^{-\alpha T_s}) Y_0 \qquad (2.9)$$

Où $y(k)$ est la température du système en °C, $u(k)$ et le signal de commande à appliquer. Y_0 est la température ambiante. T_s est la période d'échantillonnage. α et β sont des valeurs constantes décrivant le système. Les paramètres du système sont ceux d'un modèle réel [60] ; ils sont donnés par le tableau 2.4.

Coefficient	Valeur
α	$1.0015.10^{-4}$
β	$8.6793.10^{-3}$
T_s[s]	30
Y_0[°C]	25

Tab. 2.4 : Paramètres du système

L'algorithme hybride AGE-RT est appliqué pour optimiser le contrôleur flou de type TS d'ordre zéro décrit dans la section 2.3 pour commander la température du bain d'eau. L'objectif donc, est de générer une séquence d'actions $0[\text{v}] \leq u(k) \leq 5[\text{v}]$ permettant au système de suivre la trajectoire de référence donnée par [61] :

$$y_d(k) = \begin{cases} 35[°C] & si \quad k \leq 40 \\ 55[°C] & si \quad 40 < k \leq 80 \\ 75[°C] & si \quad 80 < k \leq 120 \end{cases} \qquad (2.10)$$

Les valeurs spécifiques de l'algorithme d'optimisation AGE-RT du contrôleur flou ainsi que les contraintes imposées sont données par le tableau 2.5.

Pour qu'on puisse comparer l'approche proposée avec d'autre méthodes d'optimisation dans la littérature [62], on a choisi le même indice de performance Fit, qui est la somme des valeurs absolues des erreurs; ainsi que le même nombre de générations qui égal à 100.

La figure 2.18 donne l'évolution de l'indice de performance durant l'exécution des 4 algorithmes AG-RT. Les 4 algorithmes donnent une bonne convergence vers la trajectoire désirée mais la meilleure fonction coût est celui de l'AGE-RT.

Caractéristique	Valeur
Taille de la population (N)	10
Nombre de générations ($\max\_gen$)	100
Probabilité de croisement (p_c)	0.8
Probabilité de mutation (p_m)	0.07
Facteurs d'échelle ($G_e, G_{\Delta e}, G_{\Delta u}$)	(30, 18, 1.1)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[0[v], 5[v]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	4
Nombre d'itérations de la RT (k)	5

Tab. 2.5 : Valeurs spécifiques de l'algorithme hybride d'optimisation AGE-RT

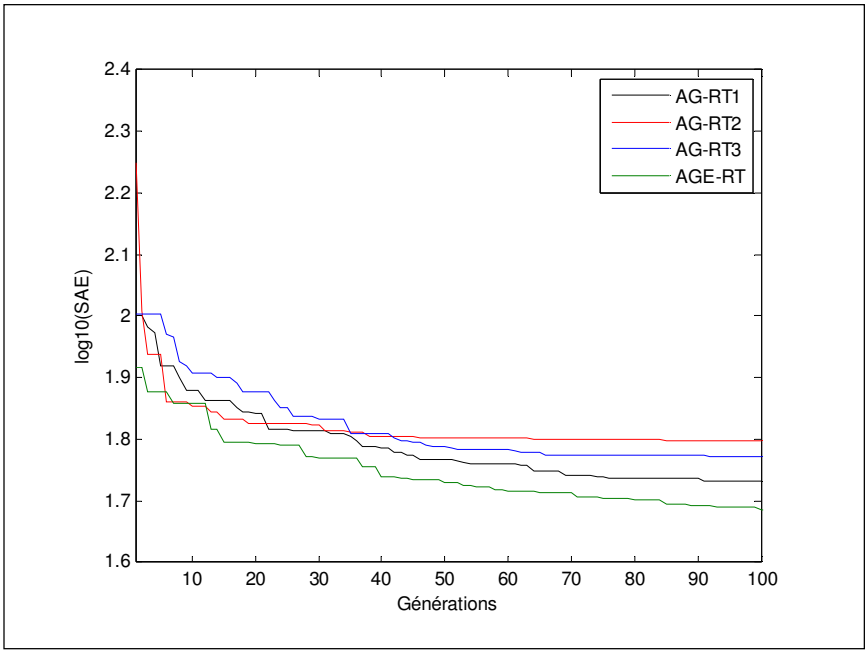


Fig. 2.18 : Evolution de la fonction coût

La base de règles floues obtenue à la fin de l'exécution de l'algorithme AGE-RT est comme suit :

$R_1 : Si\ e\ est\ N(-1,-0.90,-0.03)\ ET\ \Delta e\ est\ N(-1,-0.92,0.29)\ Alors\ \Delta u = -0.98.$
$R_2 : Si\ e\ est\ Z(-0.90,-0.03,0.91)\ ET\ \Delta e\ est\ Z(-0.92,0.29,\ 0.84)\ Alors\ \Delta u = -0.01.$
$R_3 : Si\ e\ est\ P(-0.03,0.91,1)\ ET\ \Delta e\ est\ P(0.29,\ 0.84,1)\ Alors\ \Delta u = 0.98.$

La commande du système sera calculée par la formule suivante:

$$u(k) = u(k-1) + G_{\Delta u}\,.\Delta u(k) \tag{2.11}$$

Où Δu est l'incrément de commande, $G_{\Delta u}$ est le facteur d'échelle de la variation de commande.

La figure 2.19 montre la réponse du système ainsi que la commande générée. On remarque une convergence de la réponse vers le modèle de référence.

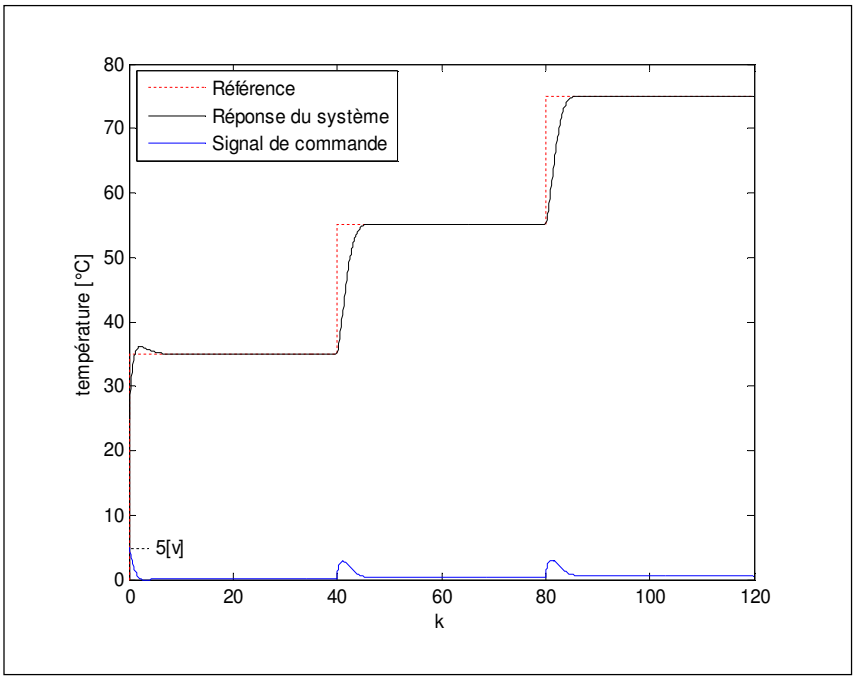


Fig. 2.19 : Réponse du système de bain d'eau pour le contrôleur flou optimisé

La robustesse du contrôleur est justifiée par sa capacité de bien réagir lorsqu'on s'écarte des conditions de fonctionnement nominales. Dans ce cas, la robustesse est testée par le changement de la trajectoire de référence. La nouvelle trajectoire est définie comme suit:

$$y_d(k) = \begin{cases} 34\,[°C] & si \quad k \leq 30 \\ \left(34 + 0.5 * (k - 30)\right)[°C] & si \quad 30 < k \leq 50 \\ \left(44 + 0.8 * (k - 50)\right)[°C] & si \quad 50 < k \leq 70 \\ \left(60 + 0.5 * (k - 70)\right)[°C] & si \quad 70 < k \leq 90 \\ 70\,[°C] & si \quad 90 < k \leq 120 \end{cases} \tag{2.12}$$

La figure 2.20 montre le test de robustesse du contrôleur AGE-RT. Les résultats obtenus confirment la capacité de cet algorithme pour la conception d'un contrôleur flou optimal pour la poursuite de trajectoire.

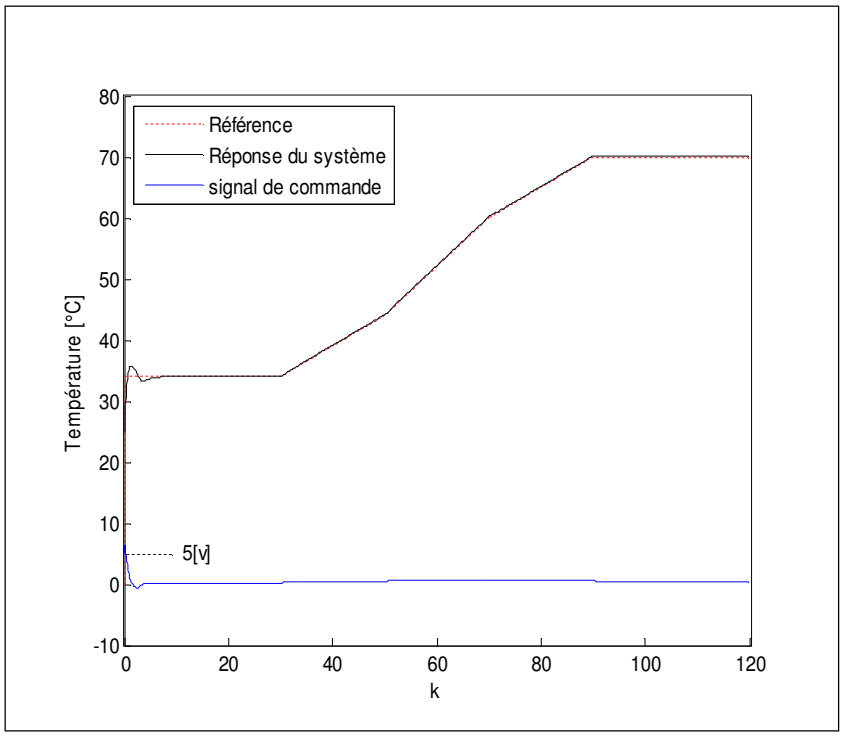


Fig. 2.20 : Test de robustesse en changeant le modèle de référence.

Le tableau 2.6 montre les résultats de différents algorithmes d'optimisation pour la commande de la température du bain d'eau. On note que les 4 algorithmes hybrides AG-RT donnent de bonnes performances pour le système commandé. AGE-RT a optimisé le contrôleur flou avec une fonction coût minimale, ce qui lui qualifie d'être le meilleur algorithme pour commander ce type de système.

Méthode	Nombre de règles floues	SAE
ACO [63]	9	103.2
EGA [62]	9	109.7
HGAPSO [64]	9	98.1
PSO-CREV [65]	9	108.7
RCACO [66]	9	63.6
AG-RT1 [56]	3	53.96
AG-RT2 [56]	3	62.89
AG-RT3 [56]	3	59.27
AGE-RT	3	48.53

Tab. 2.6 : Comparaison de différentes méthodes d'optimisation pour le problème de contrôle de température de bain d'eau.

2.5.2 Commande floue des systèmes multivariables

2.5.2.1 Commande d'un simulateur d'hélicoptère

Le modèle d'hélicoptère CE150 de Humusoft Ltd se compose d'un corps portant deux moteurs à courant continu entraînant des hélices. Le corps présente deux degrés de liberté. Les axes de rotation du corps ainsi que les axes des moteurs sont perpendiculaires (voir figure 2.21). Les deux angles de position du corps, qui sont l'angle d'azimut en horizontal (Φ) et l'angle d'élévation dans le plan vertical (Ψ) sont influencées par les hélices qui tournent en même temps. Le couple qui est produit par l'hélice principale est plus efficace sur (Ψ) ; ainsi que le couple qui est du côté d'hélice est plus efficace sur (Φ) plus que l'autre [67], [68].

Le modèle d'hélicoptère est un système multivariable dynamique avec deux entrées manipulées u_1 et u_2 et deux sorties mesurées Φ et Ψ. Toutes les entrées et sorties sont couplées. Le système est essentiellement non linéaire et au moins du sixième ordre, en fonction de la précision de modélisation [69].

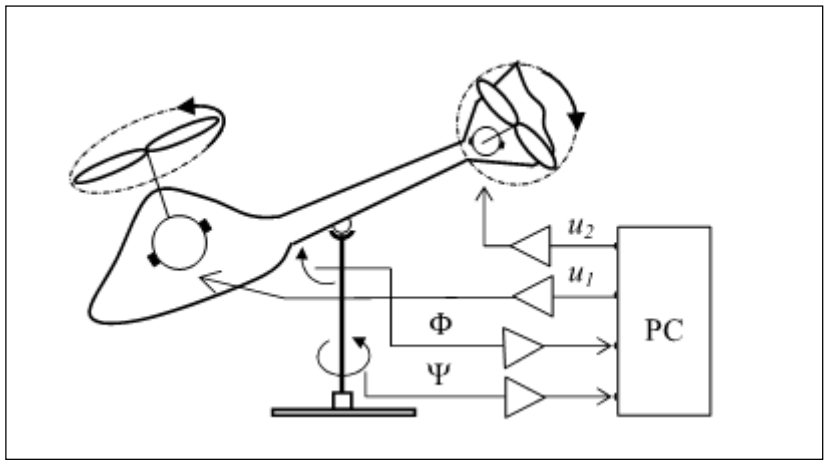


Fig. 2.21 : Modèle d'hélicoptère

Le modèle mathématique du simulateur d'hélicoptère est donné par les équations différentielles suivantes [67] :

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = 0.8764\, x_2\, sinx_1 + 3.4325\, x_4 u_1\, cosx_1 + 0.4211\, x_2 - 0.0035\, x_5^2 - 46.35\, x_6^2 - 0.8076\, x_5\, x_6 - 0.0259\, x_5 - 2.9749\, x_6$$

$$\dot{x}_3 = x_4$$

$$\dot{x}_4 = 21.4010\, x_4 - 31.8841\, x_8^2 - 14.2029\, x_8 - 21.7150\, x_9 + 1.4010\, u_1$$

$$\dot{x}_5 = -6.6667\, x_5 - 2.7778\, x_6 + 2\, u_1 \qquad (2.13)$$

$$\dot{x}_6 = 4\, x_5$$

$$\dot{x}_7 = -8\, x_7 - 4\, x_8 + 2\, u_2$$

$$\dot{x}_8 = 4\, x_7$$

$$\dot{x}_9 = -1.3333\, x_9 + 0.0625\, u_1$$

Où: $x_1 = \Psi$, $x_2 = \frac{d\Psi}{dt}$, $x_3 = \Phi$, $x_4 = \frac{d\Phi}{dt}$ et x_5 à x_9 sont des variables d'états représentants les deux moteurs à courants continus et les effets de couplage.

L'objectif est de stabiliser l'hélicoptère autour d'un point de référence (Ψ_r, Φ_r). Pour cela, deux techniques de commande seront considérées. La première technique consiste à introduire deux régulateurs qui travaillent séparément (commande décentralisée): un contrôleur flou pour le sous-système d'élévation et l'autre pour le sous-système d'azimut (voir la figure

2.22) ; tandis que la deusième technique consiste à introduire un seul régulateur flou multivariable (commande couplée) : un seul contrôleur qui calcule simultanément la commande du système d'élévation et la commande du système d'azimut.

2.5.2.1.a Commande décentralisée

Dans ce cas, chaque contrôleur flou est caractérisé par trois fonctions d'appartenance triangulaires pour les entrées et trois singletons flous pour la sortie et une base de règles composée de trois règles (voir section 2.3). Le chromosome est constitué de paramètres des deux contrôleurs flous, soient 18 paramètres.

L'indice de performance (la somme des erreurs quadratiques moyennes), Fit est choisi comme suit :

$$Fit = \frac{1}{nT}\left(\sum_{k=1}^{n} e_1^2(k) + \sum_{k=1}^{n} e_2^2(k)\right) \tag{2.14}$$

Où e_1 est l'erreur de l'angle d'élévation, e_2 est l'erreur d'angle d'azimut, n est le nombre total d'échantillons et T est le temps d'échantillonnage.

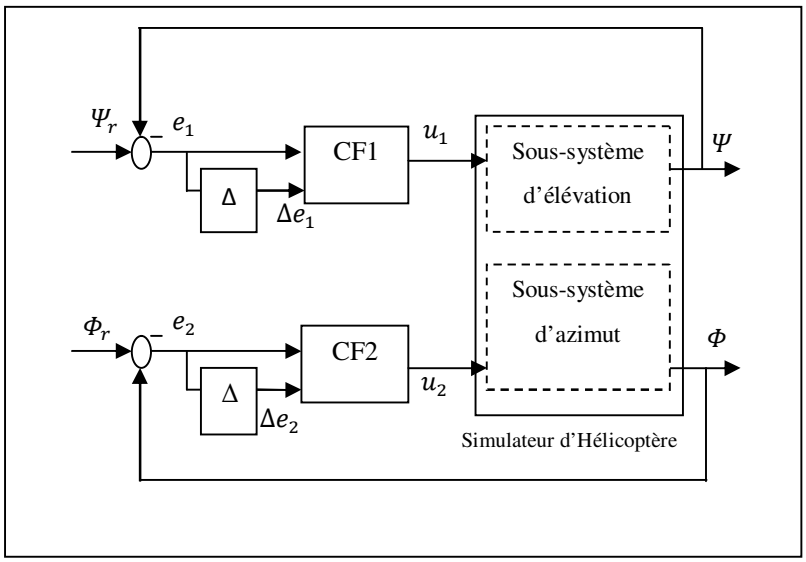


Fig. 2.22 : Structure de commande décentralisée du simulateur d'hélicoptère.

Les valeurs spécifiques de l'algorithme d'optimisation AGE-RT du contrôleur flou ainsi que les contraintes imposées sont données par le tableau 2.7.

Pour illustrer les performances de la méthode proposée, nous considérons la commande en poursuite du système de l'hélicoptère dont la dynamique est donnée par l'expression (2.13).

Caractéristique	Valeur
Taille de la population (N)	10
Nombre de générations ($\max\_gen$)	100
Probabilité de croisement (p_c)	0.9
Probabilité de mutation (p_m)	0.07
Facteurs d'échelle de l'angle d'élévation ($G_{e_1}, G_{\Delta e_1}, G_{u_1}$)	(0.1, 0.05, 1)
Facteurs d'échelle de l'angle d'azimut ($G_{e_2}, G_{\Delta e_2}, G_{u_2}$)	(0.4, 0.2, 0.09)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[-0.09 [v], 0.09 [v]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	4

Tab. 2.7 : Valeurs spécifiques de l'algorithme hybride d'optimisation AGE-RT

Le modèle de référence est définit comme suit: $(\Psi_r, \Phi_r) =$ (1 [rad], 1 [rad]). Les conditions initiales sont choisies de sorte que : $(\Psi(0), \Phi(0)) =$ (0 [rad], 0 [rad]). La figure 2.23 montre l'évolution de la fonction coût f_c et l'indice de performance Fit durant l'exécution de l'algorithme AGE-RT.

La base de règles floues du CF1 obtenue après optimisation est comme suit :

$R_1 : Si\ e_1\ est\ N(-1,-0.5,0.5)\ ET\ \Delta e_1\ est\ N(-1,-0.8,0.16)\ Alors u_1 = -0.56.$

$R_2 : Si\ e_1\ est\ Z(-0.5,0.5,0.66)\ ET\ \Delta e_1\ est\ Z(-0.8,0.16,0.56)\ Alors u_1 = 0.03.$

$R_3 : Si\ e_1\ est\ P(0.5,0.66,1)\ ET\ \Delta e_1\ est\ P(0.16,0.56,1) Alors\ u_1 = 0.73.$

La base de règles floues du CF2 obtenue après optimisation est comme suit :

$R_1 : Si\ e_2\ est\ N(-1,-0.63,0.43)\ ET\ \Delta e_2\ est\ N(-1,-0.50,-0.1)\ Alors\ u_2 = -0.96.$

$R_2 : Si\ e_2\ est\ Z(-0.63,0.43,0.80)\ ET\ \Delta e_2\ est\ Z(-0.50,-0.1,0.66)\ Alors\ u_2 = 0.23.$

$R_3 : Si\ e_2\ est\ P(0.43,0.80,1)\ ET\ \Delta e_2\ est\ P(-0.1,0.66,1)\ Alors\ u_2 = 0.90.$

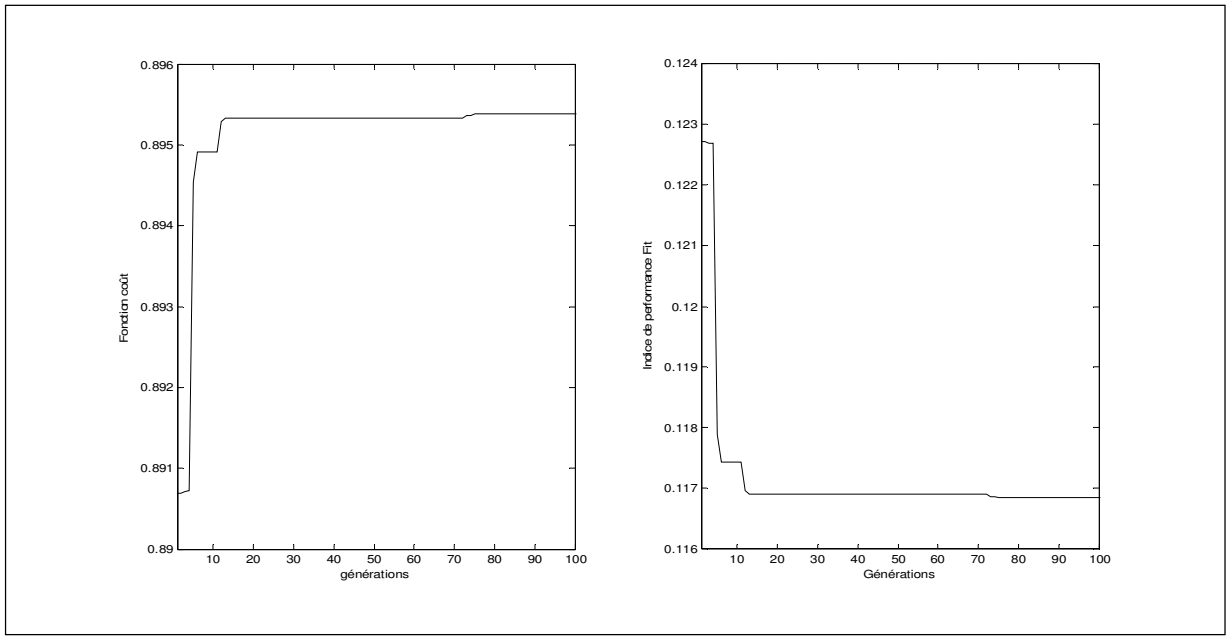


Fig. 2.23 : Evolution de la fonction coût f_c et l'indice de performance.

La figure 2.24 montre les réponses des angles d'élévation et d'azimut du système d'hélicoptère ainsi que leurs signaux de commande optimisés u_1 et u_2. Il est clair que l'algorithme proposé permet la conception des contrôleurs flous capables de stabiliser le simulateur d'hélicoptère dans un temps minimum (moins de 20s) avec une bonne précision (voir la figure 2.23).

D'après la figure 2.24, on conclue que la stratégie proposée de conception des contrôleurs flous pour les sous-systèmes d'élévation et d'azimut accomplit efficacement les performances désirées.

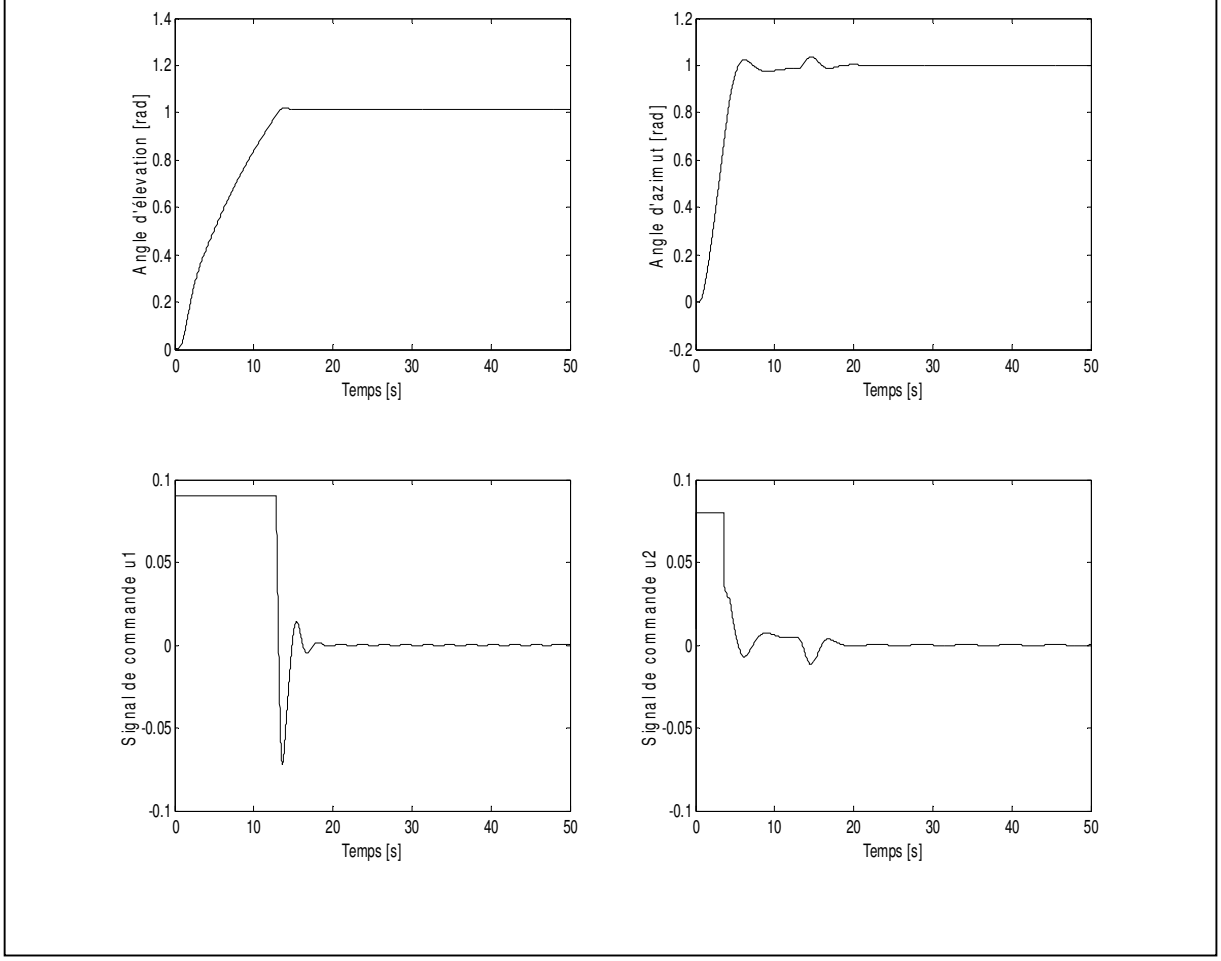


Fig. 2.24 : Réponses du système utilisant les contrôleurs flous optimisés

2.5.2.1.b Commande couplée

Les systèmes non linéaires multivariables se caractérisent par la présence des interactions non linéaires. Le contrôleur utilisé dans ce cas est un contrôleur flou multivariable caractérisé par trois fonctions d'appartenance triangulaires pour chaque entrée et trois singletons flous pour chaque sortie, avec une base de règles composée de neuf règles. Cette dernière est extraite de l'analyse sera exprimées comme suit:

$R_1 : \boldsymbol{Si}\ e_1\ est\ N\ ET\ \Delta e_1\ est\ N\ ET\ e_2\ est\ N\ ET\ \Delta e_2\ est\ N\ \boldsymbol{Alors}\ u_1 =\ s_1\ ET\ u_2 =\ o_1.$

$R_2 : \boldsymbol{Si}\ e_1\ est\ N\ ET\ \Delta e_1\ est\ Z\ ET\ e_2\ est\ N\ ET\ \Delta e_2\ est\ Z\ \boldsymbol{Alors}\ u_1 =\ s_1\ ET\ u_2 =\ o_1.$

$R_3 : \boldsymbol{Si}\ e_1 est\ \ N\ ET\ \Delta e_1 est\ \ P\ ET\ e_2\ est\ N\ ET\ \Delta e_2\ est\ P\ \boldsymbol{Alors}\ u_1 =\ s_1\ ET\ u_2 =\ o_1.$

$R_4 : \boldsymbol{Si}\ e_1\ est\ Z\ ET\ \Delta e_1\ est\ N\ ET\ e_2\ est\ Z\ ET\ \Delta e_2\ est\ N\ \boldsymbol{Alors}\ u_1 =\ s_1\ ET\ u_2 =\ o_1.$

R_5 : **Si** e_1 *est* Z *ET* Δe_1 *est* Z *ET* e_2 *est* Z *ET* Δe_2 *est* Z **Alors** $u_1 = s_2$ *ET* $u_2 = o_2$.

R_6 : **Si** e_1 *est* Z *ET* Δe_1 *est* P *ET* e_2 *est* Z *ET* Δe_2 *est* P **Alors** $u_1 = s_3$ *ET* $u_2 = o_3$.

R_7 : **Si** e_1 *est* P *ET* Δe_1 *est* N *ET* e_2 *est* P *ET* Δe_2 *est* N **Alors** $u_1 = s_3$ *ET* $u_2 = o_3$.

R_8 : **Si** e_1 *est* P *ET* Δe_1 *est* Z *ET* e_2 *est* P *ET* Δe_2 *est* Z **Alors** $u_1 = s_3$ *ET* $u_2 = o_3$.

R_9 : **Si** e_1 *est* P *ET* Δe_1 *est* P *ET* e_2 *est* P *ET* Δe_2 *est* P **Alors** $u_1 = s_3$ *ET* $u_2 = o_3$.

Où $s_1, s_2, s_3, o_1, o_2, o_3$ sont des valeurs réelles qui déterminent successivement les singletons flous N, Z, P des sorties $u_1$et u_2 du contrôleur flou multivariable.

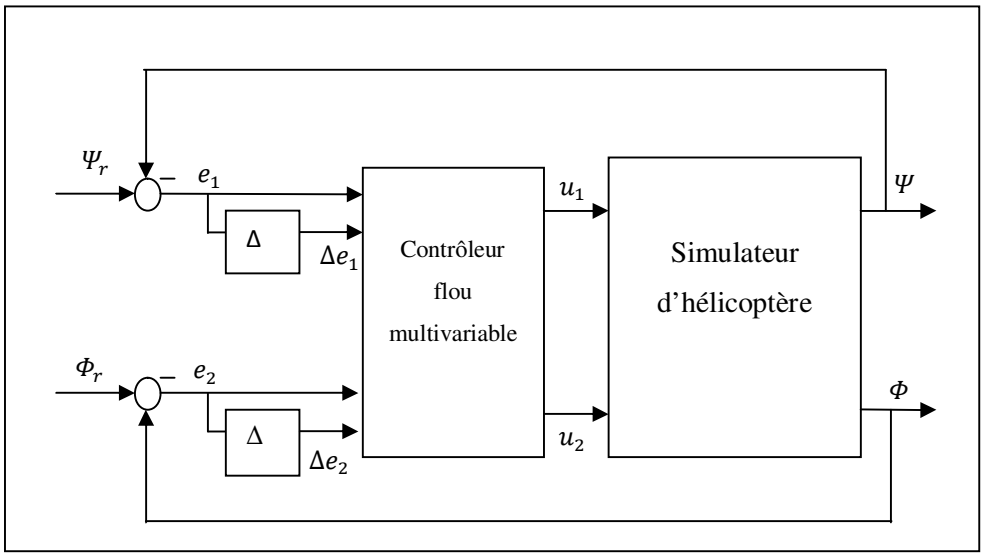


Fig. 2.25 : Structure de la commande couplée du simulateur d'hélicoptère.

Le chromosome est constitué de paramètres des entrées et de sorties du contrôleur flou, soient 15 paramètres. L'indice de performance (la somme des erreurs quadratiques moyennes), Fit est choisi comme la formule (2.14). Les algorithmes génétiques permettent de déterminer le maximum d'une fonction, alors la fonction coût f_c de l'AGE-RT est calculée par l'équation (2.9).

Les valeurs spécifiques de l'algorithme d'optimisation AGE-RT du contrôleur flou multivariable ainsi que les contraintes imposées sont données par le tableau 2.8.

Pour illustrer les performances de la méthode proposée, nous considérons la commande en poursuite du système de l'hélicoptère dont la dynamique est donnée par l'expression (2.13).

Caractéristique	Valeur
Taille de la population (N)	10
Nombre de générations ($\max\_gen$)	100
Probabilité de croisement (p_c)	0.9
Probabilité de mutation (p_m)	0.07
$[u_{min}, u_{max}]$	[-0.08 [v], 0.08 [v]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	1

Tab. 2.8 : Valeurs spécifiques de l'algorithme hybride d'optimisation AGE-RT

Le modèle de référence est définit comme : $(\Psi_r, \Phi_r) = (1\ [\text{rad}], 1\ [\text{rad}])$. Les conditions initiales sont choisies de sorte que : $(\Psi(0), \Phi(0)) = (0\ [\text{rad}], 0\ [\text{rad}])$.

La base de règles floues obtenue du CF multivariable après la mise en échelles à l'entrée et à la sortie est comme suit :

$$R_1 : \textbf{Si}\ e_1\ est\ N(-0.9, -0.62, -0.29)\ ET\ \Delta e_1\ est\ N(-0.06, -0.05, -0.01)\ ET\ e_2\ est\ N(-1.1, -0.7, -0.6)\ ET\ \Delta e_2\ est\ N(-0.2, -0.18, 0.09)\ \textbf{Alors}\ u_1 = -1.57\ ET\ u_2 = -0.05$$

$$R_2 : \textbf{Si}\ e_1\ est\ N(-0.9, -0.62, -0.29)\ ET\ \Delta e_1\ est\ Z(-0.05, -0.01, 0.05)\ ET\ e_2\ est\ N(-1.1, -0.7, -0.6)\ ET\ \Delta e_2\ est\ Z(-0.18, 0.09, 0.15)\ \textbf{Alors}\ u_1 = -1.57\ ET\ u_2 = -0.05.$$

$$R_3 : \textbf{Si}\ e_1\ est\ N(-0.9, -0.62, -0.29)\ ET\ \Delta e_1\ est\ P(-0.01, 0.05, 0.06)\ ET\ e_2\ est\ N(-1.1, -0.7, -0.6)\ ET\ \Delta e_2\ est\ P(0.09, 0.15, 0.2)\ \textbf{Alors}\ u_1 = -1.57\ ET u_2 = -0.05.$$

$$R_4 : \textbf{Si}\ e_1\ est\ Z(-0.62, -0.29, 0.72)\ ET\ \Delta e_1\ est\ N(-0.06, -0.05, -0.01)\ ET\ e_2\ est\ Z(-0.7, -0.6, 0.84)\ ET\ \Delta e_2\ est\ N(-0.2, -0.18, 0.09)\ \textbf{Alors}\ u_1 = -1.57\ ET u_2 = -0.05.$$

$$R_5 : \textbf{Si}\ e_1\ est\ Z(-0.62, -0.29, 0.72)\ ET\ \Delta e_1\ est\ Z(-0.05, -0.01, 0.05)\ ET\ e_2\ est\ Z(-0.7, -0.6, 0.84)\ ET\ \Delta e_2\ est\ Z(-0.18, 0.09, 0.15)\ \textbf{Alors}\ u_1 = 0.11\ ET u_2 = 0.004.$$

$$R_6 : \textbf{Si}\ e_1\ est\ Z(-0.62, -0.29, 0.72)\ ET\ \Delta e_1\ est\ P(-0.01, 0.05, 0.06)\ ET\ e_2\ est\ Z(-0.7, -0.6, 0.84)\ ET\ \Delta e_2\ est\ P(0.09, 0.15, 0.2)\ \textbf{Alors}\ u_1 = 1.8\ ET u_2 = 0.13.$$

$$R_7 : \textbf{Si}\ e_1\ est\ P(-0.29, 0.72, 0.9)\ ET\ \Delta e_1\ est\ N(-0.06, -0.05, -0.01)\ ET\ e_2\ est\ P\ (-0.6, 0.84, 1.1)\ ET\ \Delta e_2\ est\ N\ (-0.2, -0.18, 0.09)\ \textbf{Alors}\ u_1 = 1.8\ ET\ u_2 = 0.13.$$

$$R_8 : \textbf{Si}\ e_1\ est\ P(-0.29, 0.72, 0.9)\ ET\ \Delta e_1\ est\ Z(-0.05, -0.01, 0.05)\ ET\ e_2\ est\ P(-0.05, -0.01, 0.05)\ ET\ \Delta e_2\ est\ Z(-0.18, 0.09, 0.15)\ \textbf{Alors}\ u_1 = 1.8\ ET\ u_2 = 0.13.$$

$$R_9 : \textbf{Si}\ e_1\ est\ P(-0.29, 0.72, 0.9)\ ET\ \Delta e_1\ est\ P\ (-0.01, 0.05, 0.06)\ ET\ e_2\ est\ P(-0.6, 0.84, 1.1)\ ET\ \Delta e_2\ est\ P\ (0.09, 0.15, 0.2)\ \textbf{Alors}\ u_1 = 1.8\ ET\ u_2 = 0.13.$$

La figure 2.26 montre l'évolution de la fonction coût f_c et l'indice de performance Fit durant l'exécution de l'algorithme AGE-RT.

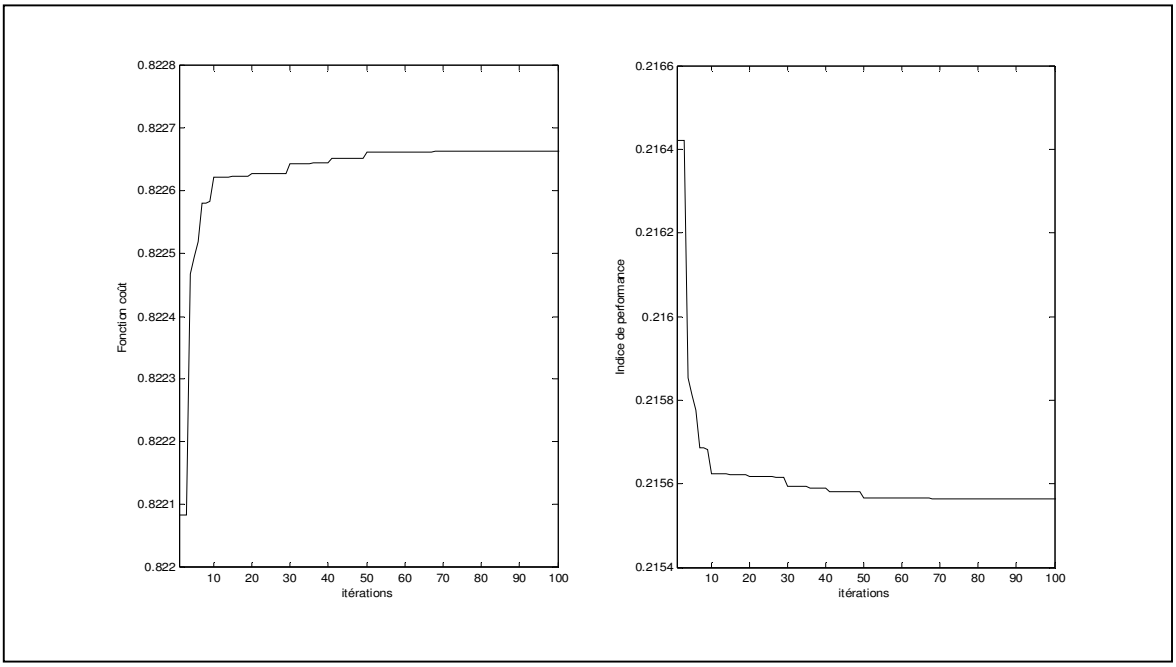


Fig. 2.26 : Evolution de la fonction coût.

La figure 2.27 montre les réponses des angles d'élévation et d'azimut du système d'hélicoptère ainsi que leurs signaux de commande optimisés u_1 et u_2.L'algorithme proposé permet la conception du contrôleur flou multivariable qui assure une bonne poursuite de trajectoire du système d'hélicoptère et qui garantit de bonnes performances pour le système contrôlé.

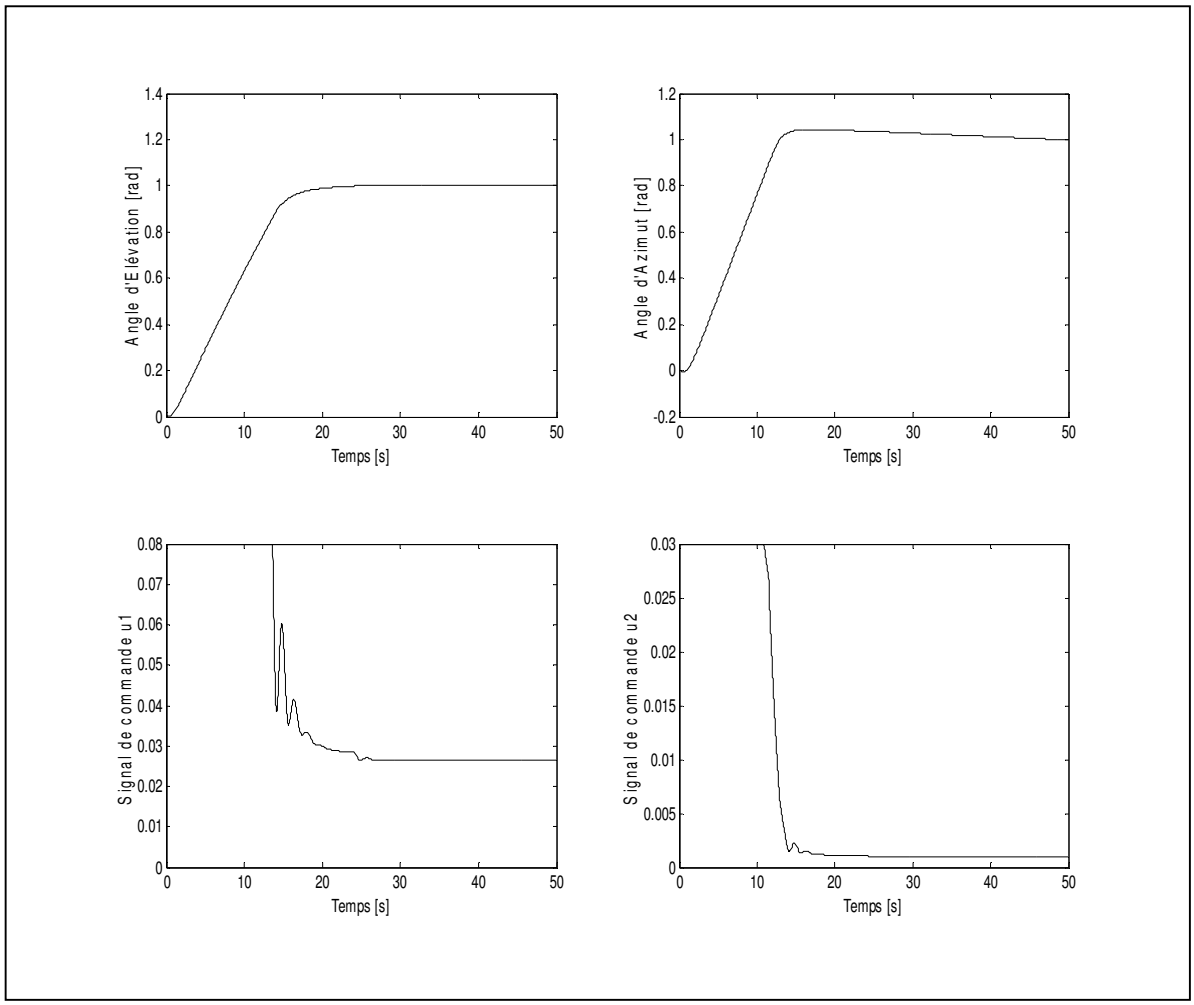


Fig. 2.27 : Réponses du système et commandes générées par le contrôleur flou multivariable optimisé.

2.5.2.2 Stabilisation du double pendule inversé

La structure du système de double pendule inversé est illustrée par la figure 2.28. Le double pendule inversé est connecté par un ressort de rappel, deux couples u_1 et u_2 sont appliqués aux entrées à travers des servomoteurs afin de garantir les positions perpendiculaires.

La dynamique du système est représentée par les équations suivantes [70]:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \theta_1 sinx_1 + \theta_2 sinx_3 + \theta_3 + \frac{u_1}{J_1} \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \theta_4 sinx_3 + \theta_5 sinx_1 + \theta_6 + \frac{u_2}{J_2} \end{cases} \quad (2.15)$$

Où

$$\theta_1 = \frac{m_1 gr}{J_1} - \frac{kr^2}{4J_1}, \theta_2 = \frac{kr^2}{4J_1}, \theta_3 = \frac{kr}{2J_1}(l-b)$$

$$\theta_4 = \frac{m_2 gr}{J_2} - \frac{kr^2}{4J_2}, \theta_5 = \frac{kr^2}{4J_2}, \theta_6 = \frac{kr}{2J_2}(l-b)$$

Les variables d'état sont les positions angulaires et les vitesses angulaires :

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} q_1 \\ \dot{q}_1 \\ q_2 \\ \dot{q}_2 \end{bmatrix}$$

$m_1 = 2\,[kg]$, $m_2 = 2.5[kg]$, $k = 100[N/m]$,$l = 0.5\,[m]$,$r = 0.5\,[m]$, $b = 0.4\,[m]$, $J_1 = 0.5$, $J_2 = 0.625$.

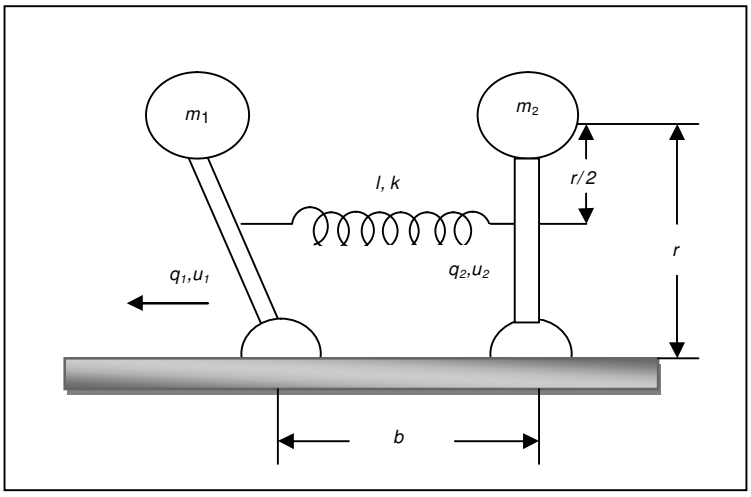


Fig. 2.28 : Structure du double pendule inversé

L'objectif est de définir deux couples pour maintenir l'état d'équilibre défini par $q_1 = 0$ [rad] et $q_2 = 0$ [rad]. Les conditions initiales sont $(q_1(0), q_2(0)) = (0.5\,[rad], -0.5\,[rad])$.

Les valeurs spécifiques de l'algorithme hybride AGE-RT pour l'optimisation des contrôleurs flous du double pendule inversé ainsi que les contraintes imposées sont données par le tableau 2.9.

Pour illustrer les performances de la méthode proposée, nous considérons la commande en régulation du système de double pendule inversé dont la dynamique est donnée par l'expression (2.15). La structure de commande consiste à introduire deux contrôleurs flous qui travaillent séparément : un contrôleur flou pour l'angle q_1 et l'autre pour l'angle q_2.

Caractéristique	Valeur
Taille de la population (N)	10
Nombre de générations ($\max\_gen$)	100
Probabilité de croisement (p_c)	0.8
Probabilité de mutation (p_m)	0.07
Facteurs d'échelle de l'angle d'élévation ($G_{e_1}, G_{\Delta e_1}, G_{u_1}$)	(1, 0.5, 90)
Facteurs d'échelle de l'angle d'azimut ($G_{e_2}, G_{\Delta e_2}, G_{u_2}$)	(5, 2.6, 120)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[-50 [N.m], 50 [N.m]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	4

Tab. 2.9 : Valeurs spécifiques de l'algorithme hybride d'optimisation AGE-RT

La figure 2.29 montre l'évolution de la fonction coût f_c et l'indice de performance Fit durant l'exécution de l'algorithme AGE-RT.

La base de règles floues du CF1 obtenue après optimisation est comme suit :

$R_1 : Si\ e_1\ est\ N(-1,-0.96,0.43)\ ET\ \Delta e_1\ est\ N(-1,-0.83,-0.23)\ Alors\ u_1 = -0.56.$

$R_2 : Si\ e_1\ est\ Z(-0.96,0.43,0.96)\ ET\ \Delta e_1\ est\ Z(-0.83,-0.23,0.53)\ Alors\ u_1 = -0.64.$

$R_3 : Si\ e_1\ est\ P(0.43,0.96,1)\ ET\ \Delta e_1\ est\ P(-0.23,0.53,1)\ Alors\ u_1 = 0.96.$

La base de règles floues du CF2 obtenue après optimisation est comme suit :

$R_1 : Si\ e_2\ est\ N(-1,-0.83,0.43)\ ET\ \Delta e_2\ est\ N(-1,-0.80,-0.50)\ Alors\ u_2 = -0.63.$

$R_2 : Si\ e_2\ est\ Z(-0.83,0.43,0.83)\ ET\ \Delta e_2\ est\ Z(-0.80,-0.50,0.93)\ Alors\ u_2 = 0.16.$

$R_3 : Si\ e_2\ est\ P(0.43,0.83,1)\ ET\ \Delta e_2\ est\ P(-0.50,0.93,1)\ Alors\ u_2 = 0.80.$

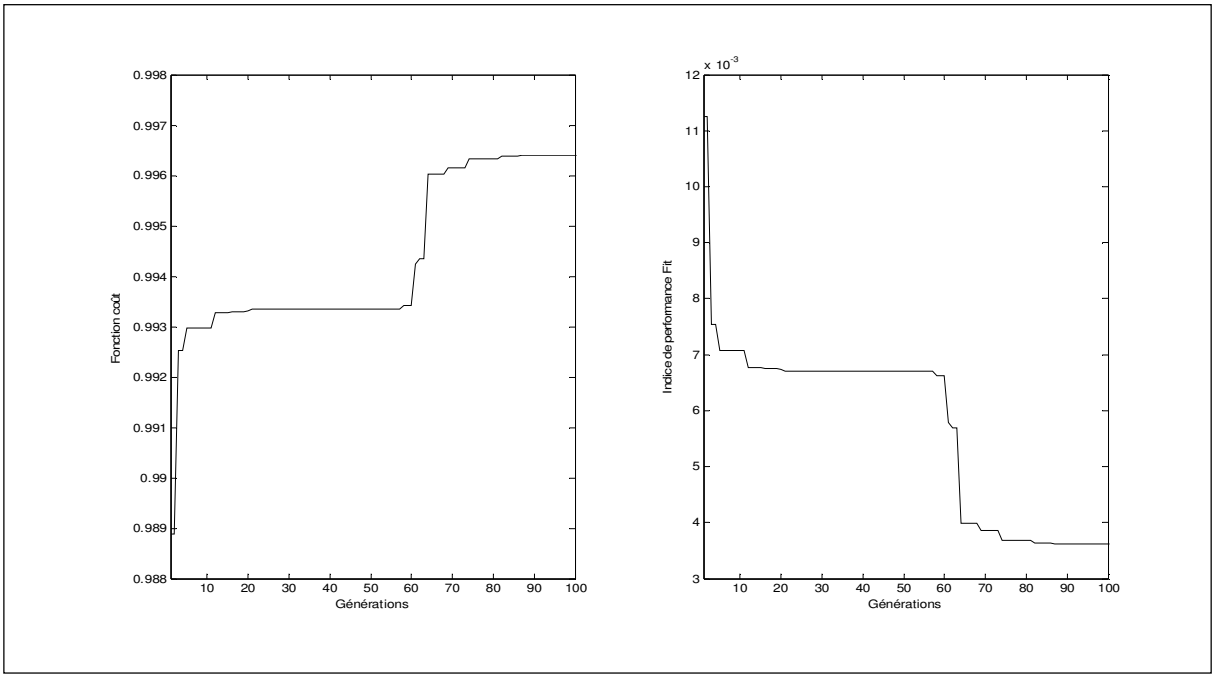


Fig. 2.29 : Evolution de la fonction coût et l'indice de performance

La figure 2.30 montre les réponses des positions angulaires q_1 et q_2, les vitesses angulaires $\dot{q}_1$ et $\dot{q}_2$ ainsi que leurs couples u_1 et u_2. Les résultats montrent la convergence des angles et des vitesses angulaires vers l'état d'équilibre.

La robustesse du contrôleur est justifiée par sa capacité de bien réagir devant le changement des conditions initiales de fonctionnement ainsi que le changement des paramètres du double pendule. Pour le premier cas, les conditions initiales testées sont les suivantes : $(q_1(0), q_2(0)) \in \{\pm 0.3, \pm 0.5, \pm 0.8, \pm 1.2\}$ [rad], avec $(\dot{q}_1(0), \dot{q}_2(0)) =$ (0 [rad/s],0 [rad/s]). La figure 2.31 illustre ces différentes situations. Par contre dans le deuxième cas, la robustesse via le changement de paramètres est illustrée par une augmentation du paramètre $\delta = kr(l-b)$ [71]. La figure 2.32 montre l'évolution des positions angulaires pour 10δ. On remarque une convergence vers l'état d'équilibre avec une faible sensibilité vis-à-vis cette variation.

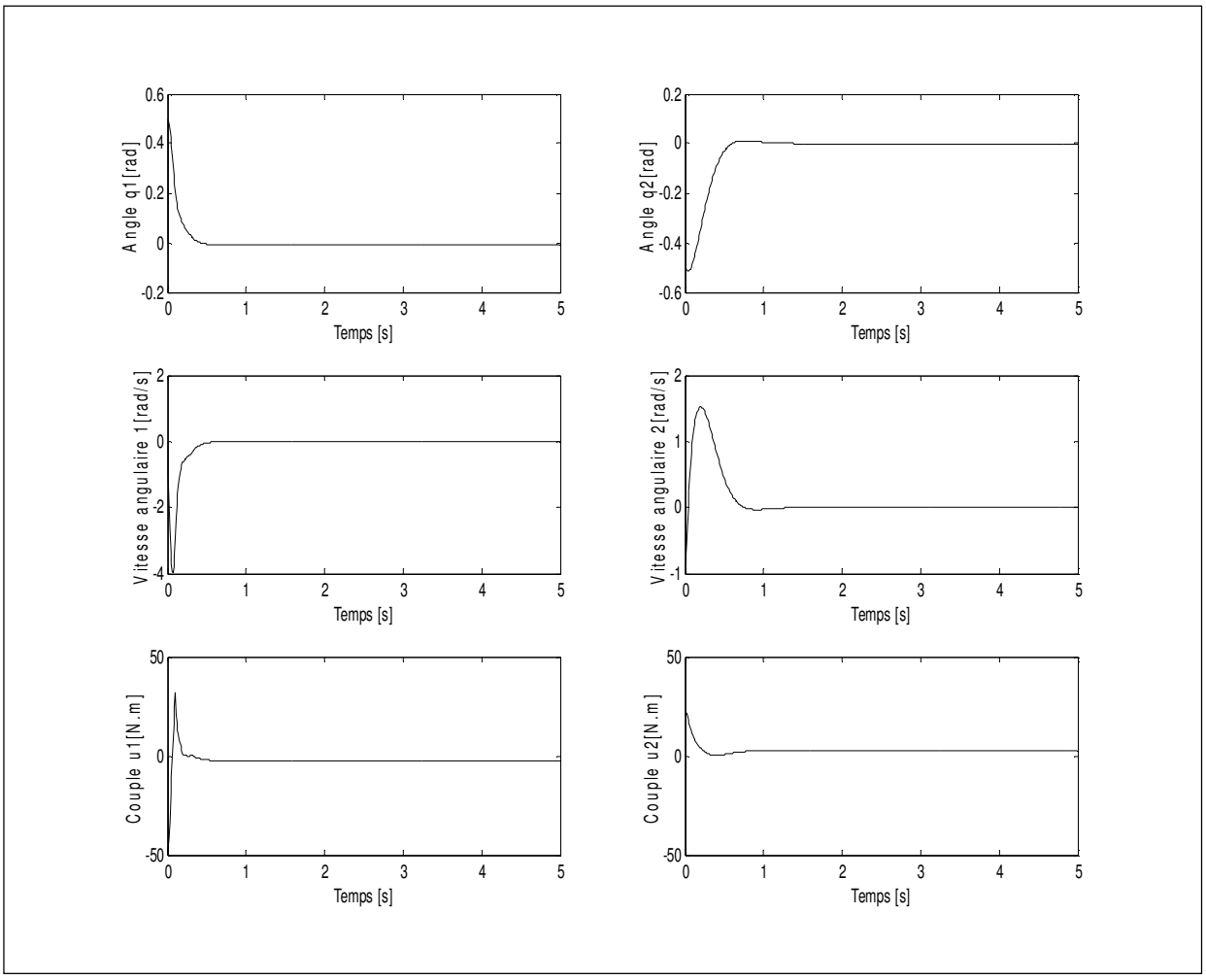


Fig. 2.30 : Les réponses du système soumis aux contrôleurs flous optimisés

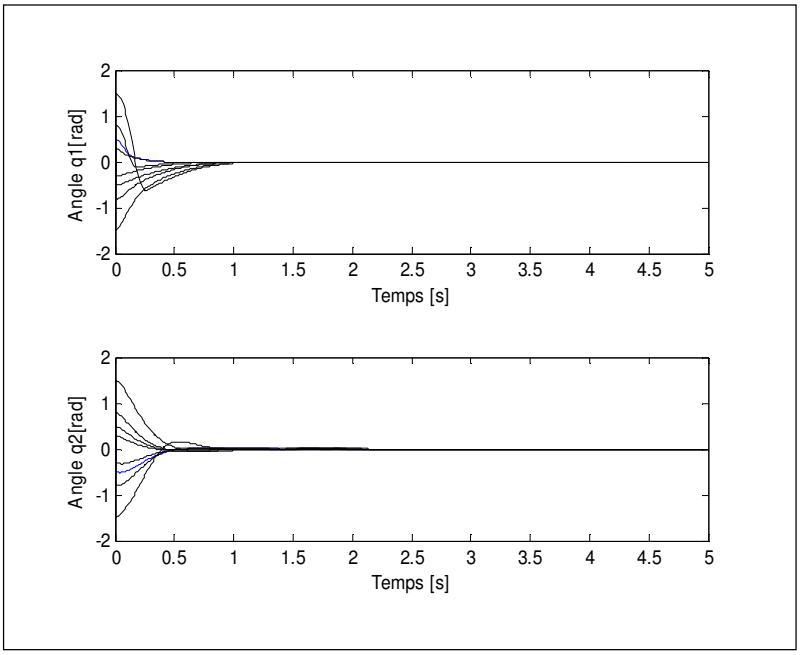


Fig. 2.31 : Réponses du système avec différentes conditions initiales

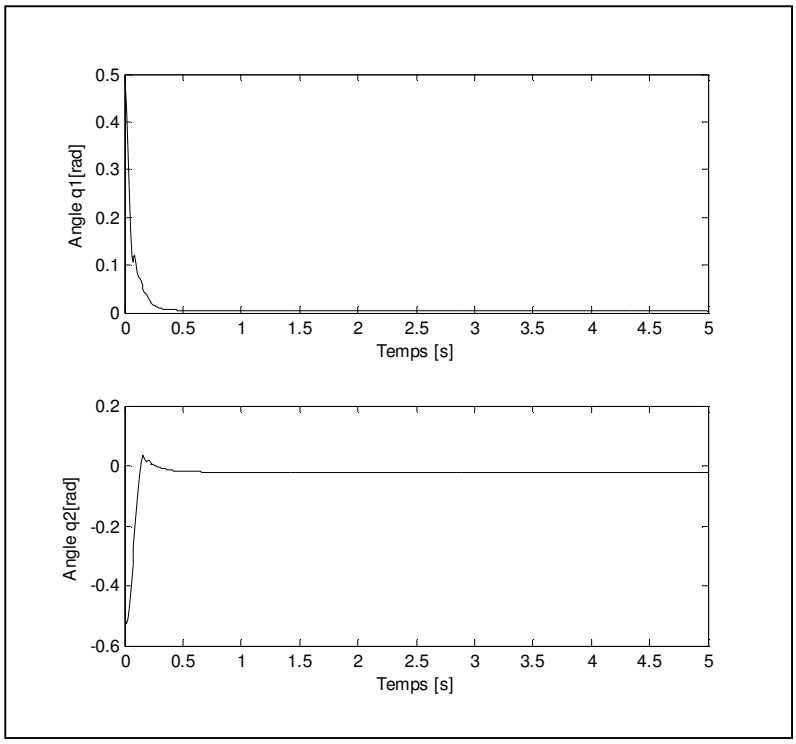


Fig. 2.32 : Test de robustesse via le changement du paramètre δ

D'après les figures 2.30, 2.31 et 2.32, il est visible que l'algorithme de commande optimisée réalise une bonne performance de régulation du double pendule inversé.

2.6 Conclusions

Le problème d'optimisation d'une base de règles floues consiste à choisir une structure appropriée du CF puis à concevoir des lois d'ajustement de paramètres satisfaisant un critère de performances prédéterminé par le concepteur.

Dans ce chapitre, nous avons proposé une méthodologie d'optimisation d'une loi de commande simple, efficace, stable et robuste pour une classe de systèmes non linéaires. La stratégie de commande proposée, dans ce travail, repose sur l'hybridation des algorithmes génétiques et la recherche tabou.

Afin d'améliorer la qualité des solutions et d'accélérer la convergence des AG, l'hybridation des AG et la RT a été réalisée. Selon l'intervention de la RT dans l'algorithme des AG, on a pu construire 4 algorithmes d'optimisation. Ces derniers sont testés pour la

commande du pendule inversé. D'après les résultats, on a choisi de continuer nos tests sur les autres procédés nonlinéaires avec l'algorithme AGE-RT car il donne une bonne précision et un temps de calcul minimal.

Les procédures de tests des contraintes et la recherche des corrections 'solutions' représentent l'inconvénient majeur de l'algorithme. La lourde procédure d'optimisation peut être évitée par une bonne initialisation des paramètres fonctionnels de l'algorithme avec une représentation minimale. Dans ce cas, le savoir-faire du concepteur 'l'expert' peut améliorer le temps de convergence de l'algorithme vers l'objectif visé.

D'après les résultats de simulation des différents procédés non linéaires testés monovariables et multivariables, nous avons pu constater que la stratégie de commande proposée est robuste en présence d'un changement dans les paramètres des systèmes non linéaires ainsi que le changement des conditions de fonctionnement.

Chapitre 3

Optimisation des Contrôleurs Flous par l'Approche Hybride OEP-RT

3.1 Introduction

Les difficultés rencontrées dans la conception des contrôleurs flous (CF), ont guidé les chercheurs ces dernières années à s'orienter vers l'utilisation de l'intelligence par essaim (en anglais : *Swarm Intelligence*), dont la plus connue est l'optimisation par essaim particulaire (OEP). Elle a été proposée en 1995 par Kennedy et Eberhart pour la résolution de problèmes d'optimisation difficiles continus [21]. L'OEP est simple, facile à mettre en œuvre et largement utilisé pour résoudre des problèmes d'optimisation. Toutefois, dans l'algorithme de base de l'OEP toutes les particules peuvent être piégées dans un minimum local dans la phase ultérieure de la convergence. Dans le but de résoudre cet inconvénient, un algorithme qui combine l'OEP et la recherche Tabou (RT) pour optimiser un contrôleur flou est proposé dans ce chapitre.

La première utilisation de l'OEP dans le domaine d'optimisation des systèmes flous était en 2004 où un réseau de neurones flous a été optimisé par une méthode hybride des AG et OEP [22]. Dans la même année, les gains d'un contrôleur PID flou ont été ajustés par l'OEP [73]. Dans [23], l'OEP a été utilisé pour l'optimisation d'un réseau de neurones flous pour le contrôle de trajectoire d'un robot par l'OEP. L'OEP a été utilisé en combinaison avec la théorie de Lyapunov dans [24] pour la conception des CF adaptatifs par l'approche hybride. L'OEP a été combiné avec la recherche Tabou pour la génération automatique des prémisses et de conclusion d'une base de règles floues pour un contrôleur flou de type TS d'ordre zéro [75]-[76].

Dans ce chapitre, l'algorithme hybride d'optimisation par essaim particulaire et la recherche tabou est introduit pour optimiser des contrôleurs flous de type Takagi-Sugeno (TS) à conclusion constante pour des systèmes monovariables et des systèmes multivariables. Les résultats seront comparés avec ceux des méthodes de la littérature.

3.2 Principe de fonctionnement de l'algorithme OEP-RT

3.2.1 Introduction

Dans l'algorithme d'optimisation par essaim particulaire, chaque particule représente une solution potentielle dans l'espace de recherche. La nouvelle position d'une particule est déterminée en fonction de sa propre valeur et celle de ses voisines. Dans cet algorithme, quand une particule trouve un optimum local, toutes les particules se regroupent autour de celui-ci et ne parviennent plus à s'échapper. La méthode de recherche Tabou permet justement de s'échapper des optima locaux en s'éloignant de la meilleure solution actuelle. C'est la raison pour laquelle l'hybridation de l'OEP et la RT a été proposée ; la meilleure position de chaque particule est optimisée par la recherche Tabou en cherchant dans son voisinage une autre meilleure position en minimisant un certain critère de performance. Ceci rend chaque particule active et la capacité de recherche de l'essaim augmente. La méthode proposée est appliquée pour l'optimisation paramétrique d'un contrôleur flou de type Takagi-Sugeno d'ordre zéro.

3.2.2 Représentation de la particule

Pour appliquer l'algorithme OEP directement ou couplé avec la recherche tabou, le problème de représentation de la particule est posé. Selon la figure 3.1, la particule est composée de neuf paramètres. Ces paramètres représentent les points d'appui et le sommet des fonctions d'appartenance triangulaire pour les entrées d'un CF et les singletons flous pour sa sortie, tout en respectant la contrainte suivante:

$$\begin{cases} a_1 < a_2 < a_3 \\ b_1 < b_2 < b_3 \\ s_1 < s_2 < s_3 \end{cases} \quad (3.1)$$

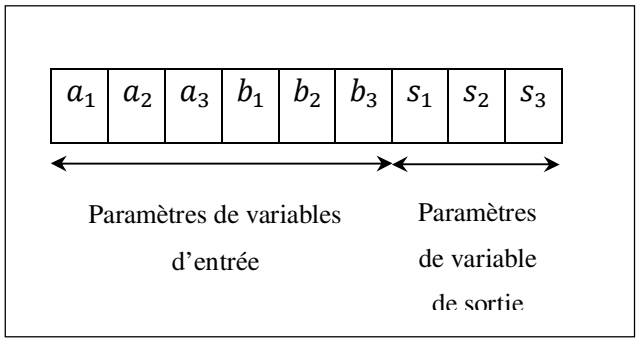


Fig. 3.1 : Structure de la particule

3.2.3 L'algorithme principal de l'OEP

L'OEP utilisé est caractérisé par les points suivants :

- La population initiale sera générée aléatoirement pour couvrir tous l'espace de recherche.
- Les constantes d'accélération c_1 et c_2 ont été choisies dans l'intervalle [1.5, 2]. Quand $c_1 > c_2$, la particule tend à se fonder sur son expérience antérieure plus que l'expérience de ses voisins, et quand $c_1 < c_2$, la particule fait confiance à l'expérience de ses voisins davantage que sa propre expérience [42]. La plupart des implémentations de l'OEP utilisent des valeurs égales de c_1 et c_2. Dans cette thèse, on a choisi $c_1 = c_2 = 1.5$.

- Le moment d'inertie w est typiquement inférieur à la valeur de l'unité, cette constante agit comme amortisseur de la vitesse de la particule. Il a été proposé de commencer l'algorithme par une valeur de poids de 0,9 pour surveiller le paysage, puis de réduire cette valeur pour exploiter la connaissance obtenue [42].

L'indice de performance est l'erreur quadratique moyenne, entre la sortie à commander et sa référence, EQM, calculée par l'équation (2.5) ou la somme de valeurs absolues des erreurs, SAE, définie par l'équation (2.6).

3.2.4 Caractéristiques de l'algorithme de la RT

L'algorithme d'optimisation de la recherche Tabou utilisé pour l'hybridation avec l'OEP est définit par:

- Sa configuration initiale (taille de la liste tabou, nombre de solutions voisines, critère d'arrêt),
- la fonction objective.
- La méthode de génération de la solution initiale et de son voisinage.

Les caractéristiques fondamentales de l'algorithme de la RT utilisé pour la construction de cet algorithme hybride sont :

- Génération d'une solution initiale et solutions voisines aléatoires, en prenant en compte tous l'espace de recherche respectant ainsi les critères d'intensification et de diversification.
- La liste tabou est une mémoire FIFO, sa taille est souvent proportionnelle au nombre d'itérations.
- La fonction objective est choisie comme celle de l'OEP calculée soit par la formule (2.5) ou par (2.6) selon le problème à résoudre.

3.2.5 L'algorithme hybride OEP-RT

Dans l'algorithme hybride OEP-RT, à chaque itération de l'algorithme OEP, la meilleure position (solution) est extraite, la RT est alors invoquée pour explorer l'espace autour de cette position pour l'améliorer.

Le tableau suivant définit les acronymes utilisés dans l'algorithme hybride OEP-RT.

Paramètre	Signification
$\max\_iter$	Nombre maximum d'itérations de l'OEP
N	Nombre de particules
P^i	Particule i
c_1, c_2	Constantes d'accélération
r_1, r_2	Constantes aléatoires
w	Moment d'inertie
p^i	Position actuelle de la particule P^i
p^i_{best}	Meilleure position de la particule P^i
g_{best}	Meilleure position du voisinage de P^i
V	Vitesse de la particule P^i
k	Nombre maximum d'itérations de la RT
N_v	Nombre de solutions voisines
T	Taille de la liste Tabou

Tab. 3.1 : Paramètres de l'algorithme OEP-RT

Le pseudo-code de l'algorithme hybride OEP-RT est le suivant:

Début *OEP*

Définir max $\_iter, N, c_1, c_2, w, p_{min}, p_{max}, V_{min}, V_{max},$

Générer r_1, r_2 aléatoirement

Générer la population initiale aléatoirement

Pour $i = 1$ à max $\_iter$ **faire**

Evaluer chaque particule P^i de la population

Si $f(p^i) < f(p^i_{best})$

$p^i_{best} = p^i$

Fin Si

Si $f(p^i_{best}) < f(g_{best})$

$g_{best} = p^i_{best}$

Fin Si

Calculer la vitesse V de P^i par l'eq. (1.39) puis normaliser la par l'eq. (1.41)

Faire la mise à jour de la position p de chaque particule par l'eq. (1.40) puis normaliser la par l'eq. (1.42)

Début *Recherche Tabou*

Définir k, N_v, T, Initialiser la liste tabou $T = \emptyset, j = 0$

Tant que $j < k$

Pour toute P^i de la population **faire**

Générer N_v solutions voisines
Evaluer chaque solution voisine
Prendre la solution voisine la plus adaptée
Mettre à jour la liste tabou

Fin Pour

$i = i + 1$

Fin Tant que

Fin

Fin Pour

Fin

Retourner la meilleure particule trouvée.

3.3 Simulations

Pour tester et évaluer les performances de l'algorithme proposé pour l'optimisation d'une loi de commande, nous avons effectué des simulations sur quelques systèmes non linéaires monovariables et multivariables.

La configuration du contrôleur flou à optimiser est la suivante :

- A chaque variable d'entrée sont associés 3 sous-ensembles flous (voir la figure 2.4), et à chaque variable de sortie sont associés 3 singletons flous.
- La base de règles floues pour un CF à deux entrées et une sortie est composée de 3 règles.

3.3.1 Commande floue des systèmes monovariables

3.3.1.1 Commande du pendule inversé

Pour illustrer les performances de la méthode proposée, nous considérons la commande en régulation du système de pendule inversé dont la dynamique est donnée par l'expression (2.8). Le contrôleur flou optimisé doit ramener et maintenir le pendule dans sa position d'équilibre correspondant à un angle $\theta = 0$ [rad] et une vitesse angulaire $\dot{\theta} = 0$ [rad/s].

Le chariot peut se déplacer à droite ou à gauche, sans prise en compte de sa position. Les conditions initiales du système sont générées, aléatoirement. Les valeurs considérées sont (0.17, 0.22, 0.5, 0.8 [rad]) pour l'angle. Dans le cas illustré par la suite (test de robustesse via le changement de paramètres), on a $(x_1(0), x_2(0)) =$ (0.22 [rad], 0 [rad/s]).

Les valeurs spécifiques de l'algorithme hybride OEP-RT pour l'optimisation du CF du pendule inversé ainsi que les contraintes imposées sont données par le tableau 3.2.

Caractéristique	Valeur
Nombre de particules (N)	5
Nombre d'itérations (max $\_iter$)	10
c_1, c_2	1.5
w	0.9
Facteurs d'échelle ($G_e, G_{\Delta e}, G_u$)	(1.6, 0.53, 140)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[-50[N], 50[N]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	16
Nombre d'itérations de la RT (k)	1

Tab.3.2 : valeurs spécifiques de l'algorithme hybride d'optimisation OEP-RT

L'évolution de la fonction coût au cours de l'exécution de l'algorithme hybride OEP-RT est illustrée par la figure 3.2. La base de règles floues obtenue après optimisation est comme suit :

$R_1 : Si\ e\ est\ N(-1,-0.76,0.23)\ ET\ \Delta e\ est\ N(-1,-0.53,-0.50)\ Alors\ u = -0.96.$
$R_2 : Si\ e\ est\ Z(-0.66,0.04,0.73)\ ET\ \Delta e\ est\ Z(-0.83,-0.11,0.65)\ Alors\ u = -0.03.$
$R_3 : Si\ e\ est\ P(0.23,0.76,1)\ ET\ \Delta e\ est\ P(-0.50,\ 0.59,1)\ Alors\ u = 0.92.$

Les réponses du pendule soumis au contrôleur flou optimisé obtenues pour les paramètres nominaux du système et différentes conditions initiales sont présentées sur la figure 3.3. Les résultats montrent la convergence de l'angle et de la vitesse angulaire vers l'état d'équilibre (0 [rad], 0 [rad/s]) à partir d'un état initial quelconque.

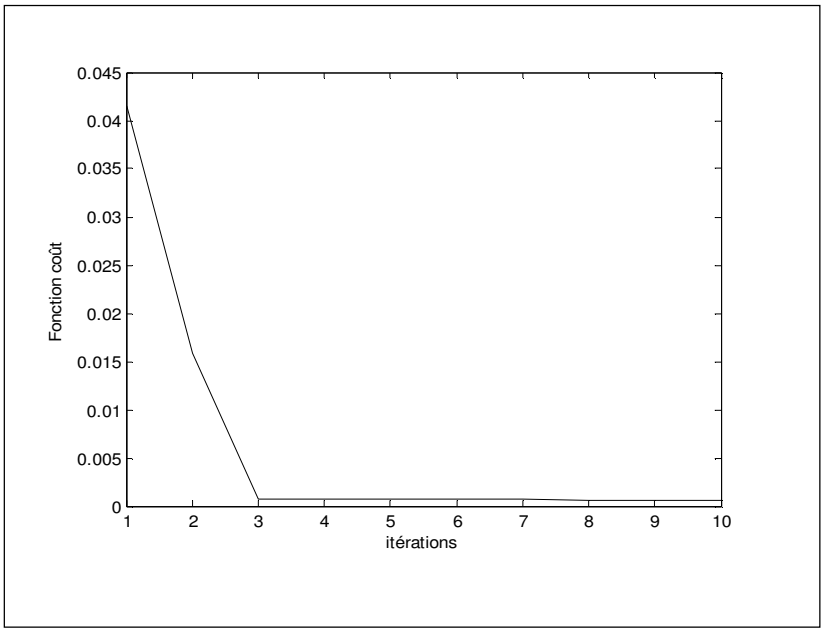


Fig. 3.2 : Evolution de la fonction coût.

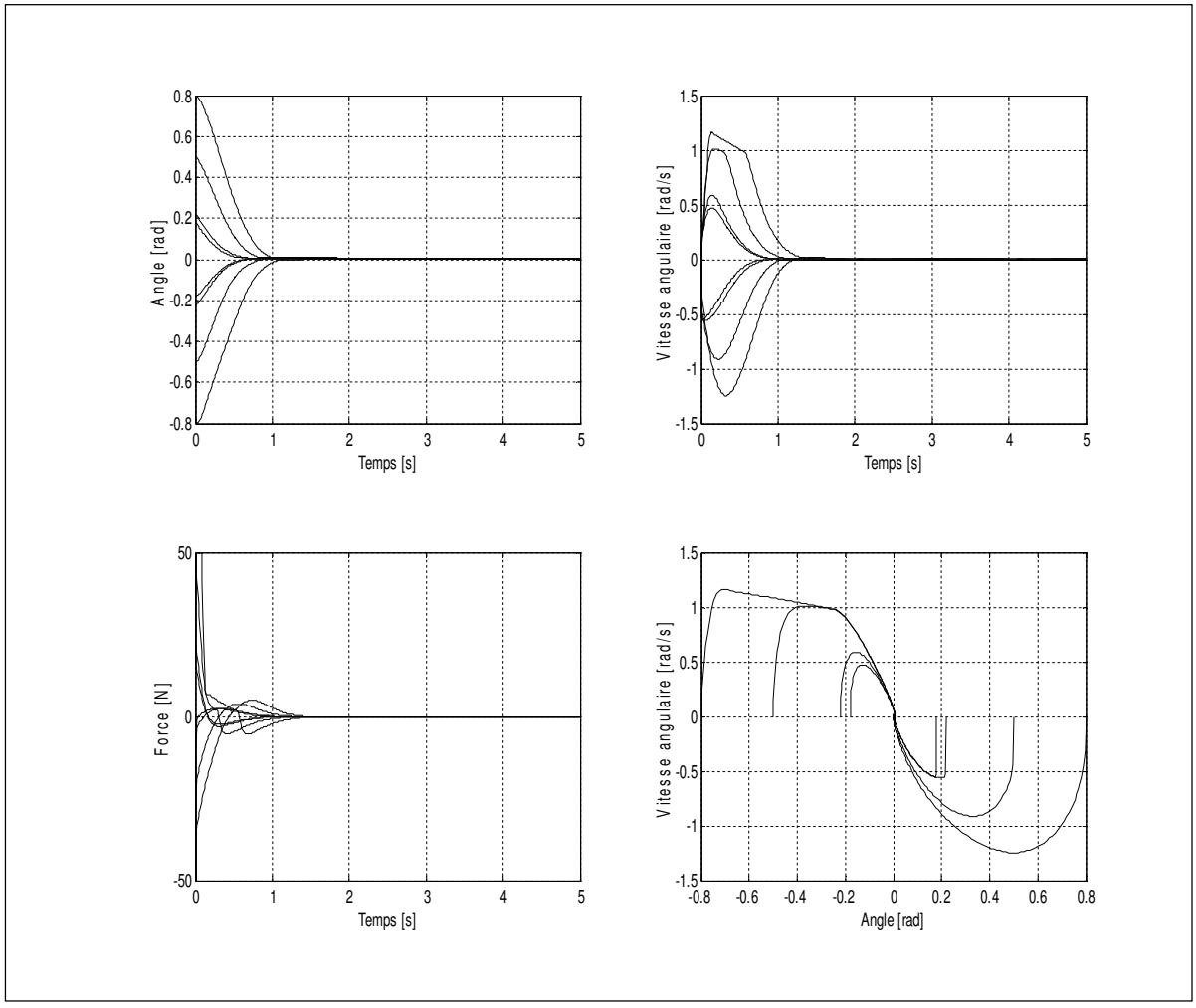


Fig. 3.3 : Réponses du système pour différentes conditions initiales.

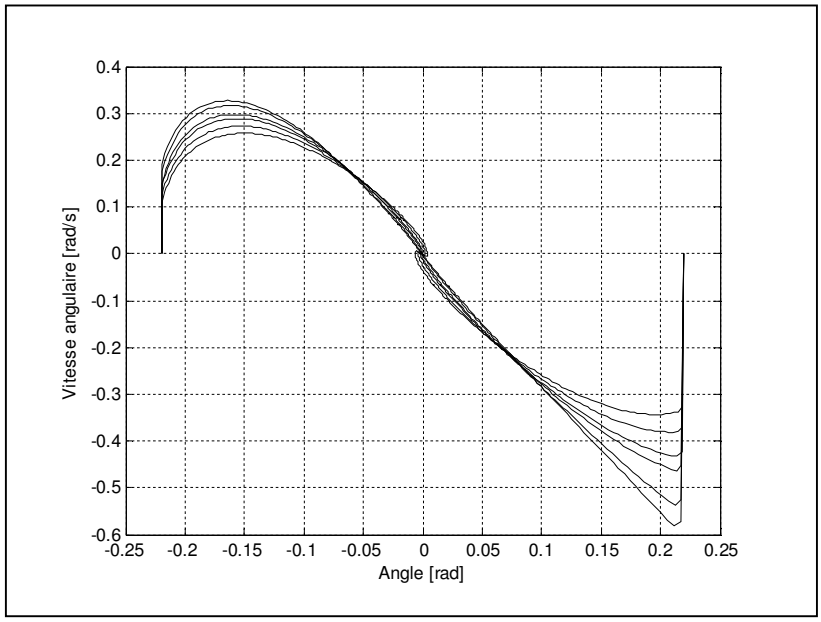


Fig. 3.4 : Réponses aux changements de poids du segment : Plan de phase

La robustesse du contrôleur est justifiée par sa capacité de bien réagir devant le changement des conditions initiales de fonctionnement ainsi que le changement des paramètres du pendule. Ce dernier est illustré par un changement du poids du pendule (voir la figure 3.4). Les valeurs considérées du poids sont (0.05, 0.1, 0.2, 0.25, 0.35, 0.45 [Kg]).

En analysant les figures 3.3 et 3.4, nous pouvons conclure que la stratégie proposée de conception accomplit, efficacement, les performances désirées.

Méthode	EQM	Temps de calcul [s]
AG à codage réel [57]	0.00196	4.60
RT [57], [58]	0.00051	3.88
AGE-RT	0.00040	9.1
OEP-RT	0.00038	3.16

Tab.3.3 : Comparaison des méthodes d'optimisation pour la commande du pendule inversé

D'après le tableau 3.3, on constate que l'algorithme OEP-RT garantie une optimisation du contrôleur flou pour la stabilisation du pendule inversé dans un minimum temps de calcul et une bonne précision.

3.3.1.2 Commande de la température d'un bain d'eau

Le but de cette section est de commander la température d'un bain d'eau. Le modèle récurrent (discret) du système est donné par l'expression (2.9). L'algorithme hybride OEP-RT est appliqué pour optimiser le contrôleur flou de type TS d'ordre zéro. Les valeurs spécifiques de l'algorithme d'optimisation OEP-RT du contrôleur flou ainsi que les contraintes imposées sont données par le tableau 3.4. Pour comparer l'approche proposée avec d'autre méthodes d'optimisation dans la littérature, on a choisi le même indice de performance Fit, qui est la somme des valeurs absolues des erreurs et le même nombre d'itérations.

Caractéristique	Valeur
Nombre de particules (N)	10
Nombre d'itérations ($\max\_iter$)	100
c_1, c_2	1.5
w	0.9
Facteurs d'échelle ($G_e, G_{\Delta e}, G_{\Delta u}$)	(38, 12, 1.1)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[0[v], 5[v]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	16
Nombre d'itérations de la RT (k)	1

Tab.3.4 : Valeurs spécifiques de l'algorithme hybride d'optimisation OEP-RT

La figure 3.5 montre l'évolution de l'indice de performance durant l'exécution de l'algorithme OEP-RT. La base de règles floues obtenue à la fin de l'exécution de l'algorithme OEP-RT est comme suit :

$R_1 : Si\ e\ est\ N(-1,-0.58,0.50)\ ET\ \Delta e\ est\ N(-1,-0.76,-0.08)\ Alors\ \Delta u = -0.71.$
$R_2 : Si\ e\ est\ Z(-0.58,0.50,0.95)\ ET\ \Delta e\ est\ Z(-0.76,-0.08,\ 0.62)\ Alors\ \Delta u = 0.31.$
$R_3 : Si\ e\ est\ P(\ 0.50,0.95,1)\ ET\ \Delta e\ est\ P(-0.08,\ 0.62,1)\ Alors\ \Delta u = 0.83.$

La commande du système sera calculée par la formule suivante:

$$u(k) = u(k-1) + G_{\Delta u}\,.\Delta u(k)$$

Où Δu est l'incrément de commande, $G_{\Delta u}$ est le facteur d'échelle de la variation de commande.

La figure 3.6 donne la réponse du système ainsi que la commande générée. L'algorithme de commande donne une bonne poursuite de trajectoire de référence et une bonne précision.

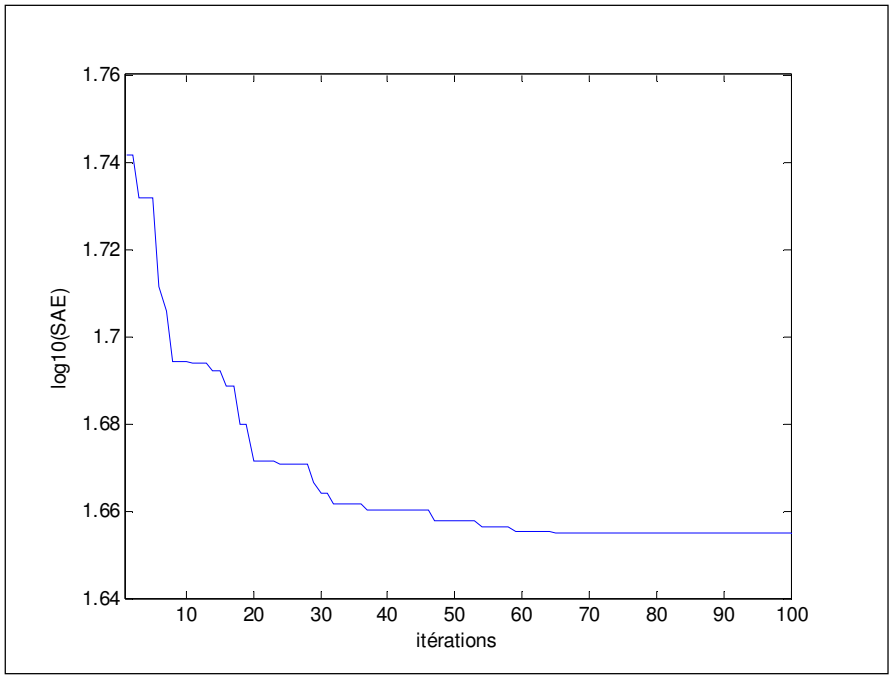


Fig. 3.5 : Evolution de la fonction coût

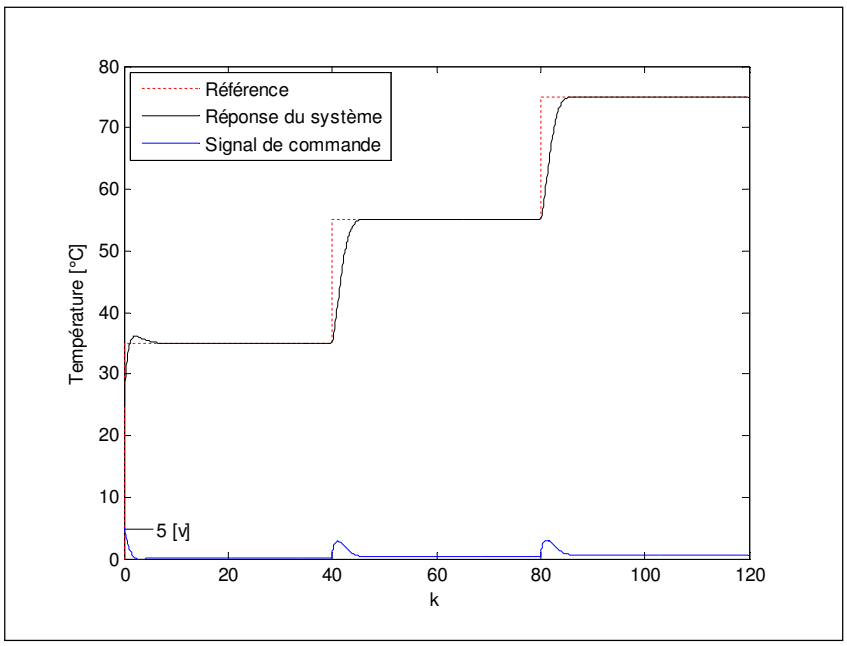


Fig. 3.6 : Réponse du système et commande générée.

La robustesse du contrôleur est justifiée par sa capacité de bien réagir lorsqu'on s'écarte des conditions de fonctionnement nominales. Dans ce cas, la robustesse est testée par le changement de la trajectoire de référence. Le nouveau milieu de fonctionnement sera défini par l'expression (2.12).

La figure 3.7 montre le test de robustesse en changeant la trajectoire de référence. L'algorithme de commande optimisée peut contrôler la température du bain d'eau quel que soit la trajectoire de référence désirée.

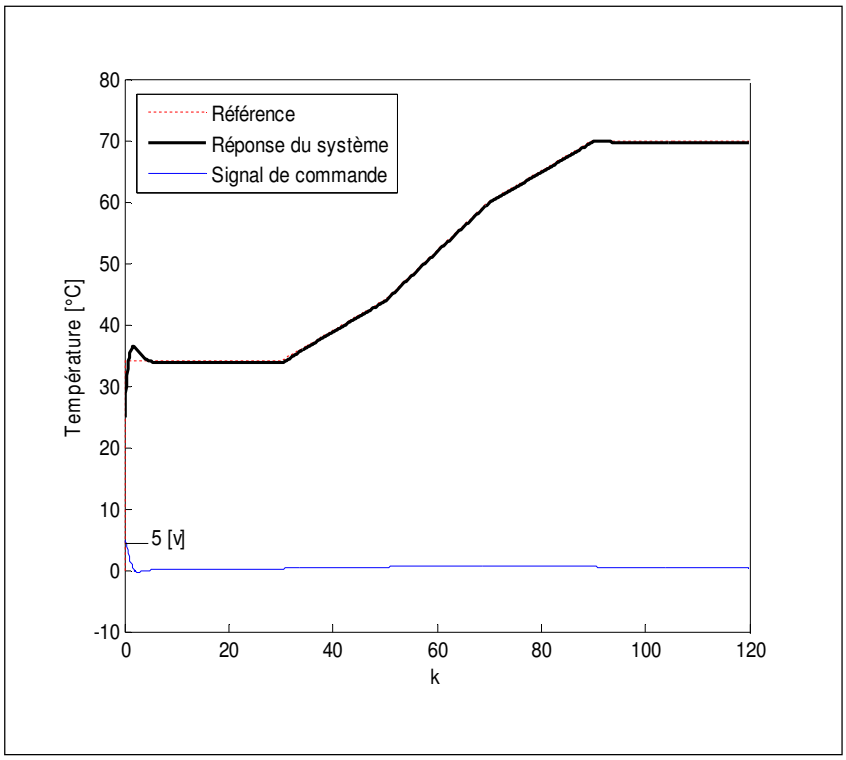


Fig. 3.7 : Test de robustesse en changeant le modèle de référence.

Le tableau 3.5 montre les résultats de différents algorithmes d'optimisation pour la commande de la température du bain d'eau. On note que l'algorithme hybride OEP-RT garantit de bonnes performances pour le système commandé.

Méthode	Nombre de règles floues	SAE
ACO [63]	9	103.2
EGA [60]	9	109.7
HGAPSO [64]	9	98.1
PSO-CREV[65]	9	108.7
RCACO [66]	9	63.6
AGE-RT	3	48.53
OEP-RT	3	45.15

Tab.3.5 : Comparaison de différentes méthodes d'optimisation pour le problème de contrôle de température de bain d'eau.

3.3.2 Commande floue des systèmes multivariables

3.3.2.1 Commande d'un simulateur d'hélicoptère

Le modèle d'hélicoptère est un système multivariables dynamique avec deux entrées manipulées u_1 et u_2 et deux sorties mesurées Φ et Ψ. Toutes les entrées et sorties sont couplées. Le système est essentiellement non linéaire et au moins du sixième ordre (voir la section 2.5.2.1). Le modèle mathématique du simulateur d'hélicoptère est donné par les équations différentielles (2.13). L'objectif est de stabiliser l'hélicoptère autour d'un point de référence (Ψ_r, Φ_r). Pour cela, deux techniques de commande d'un tel système MIMO seront considérées ; la première consiste à introduire deux régulateurs qui travaillent séparément (commande décentralisée): un contrôleur flou pour le sous-système d'élévation et l'autre pour le sous-système d'azimuth (voir la figure 2.22), tandis que la deuxième consiste à introduire un seule contrôleur flou multivariable (commande couplée) : un contrôleur flou pour le système global (voir la figure 2.25).

Pour illustrer les performances de la méthode proposée, nous considérons la commande en poursuite du système de l'hélicoptère dont la dynamique est donnée par l'expression (2.13).

Le signal de référence est un échelon unitaire : $(\Psi_r, \Phi_r) = (1\ [\text{rad}], 1\ [\text{rad}])$. Les conditions initiales sont choisies de sorte que : $(\Psi(0), \Phi(0)) = (0\ [\text{rad}], 0\ [\text{rad}])$.

3.3.2.1.a Commande décentralisée

L'indice de performance (la somme des erreurs quadratiques moyennes), Fit donnée par l'expression (2.14). Les valeurs spécifiques de l'algorithme d'optimisation OEP-RT du contrôleur flou ainsi que les contraintes imposées sont données par le tableau 3.6.

La figure 3.8 montre l'évolution l'indice de performance Fit durant l'exécution de l'algorithme OEP-RT.

Caractéristique	Valeur
Nombre de particules (N)	10
Nombre d'itérations ($\max\_iter$)	100
c_1, c_2	1.5
w	0.9
Facteurs d'échelle de l'angle d'élévation ($G_{e_1}, G_{\Delta e_1}, G_{u_1}$)	(0.1, 0.05, 1)
Facteurs d'échelle de l'angle d'azimut ($G_{e_2}, G_{\Delta e_2}, G_{u_2}$)	(0.4, 0.2, 0.09)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[-0.09 [v], 0.09 [v]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	4

Tab.3.6 : Valeurs spécifiques de l'algorithme hybride d'optimisation OEP-RT

La base de règles floues du CF1 obtenue après optimisation est comme suit :

$R_1 : Si\ e_1\ est\ N(-1,-0.69,-0.07)\ ET\ \Delta e_1 est\ N(-1,-0.60,-0.27)\ Alors\ u_1 = -0.85.$

$R_2 : Si\ e_1 est\ Z(-0.69,-0.07,0.62)\ ET\ \Delta e_1\ est\ Z(-0.60,-0.27,0.63)\ Alors\ u_1 = 0.18.$

$R_3 : Si\ e_1 est\ P(-0.07,0.62,1)\ ET\ \Delta e_1 est\ P(-0.27,0.63,1)\ Alors\ u_1 = 0.68.$

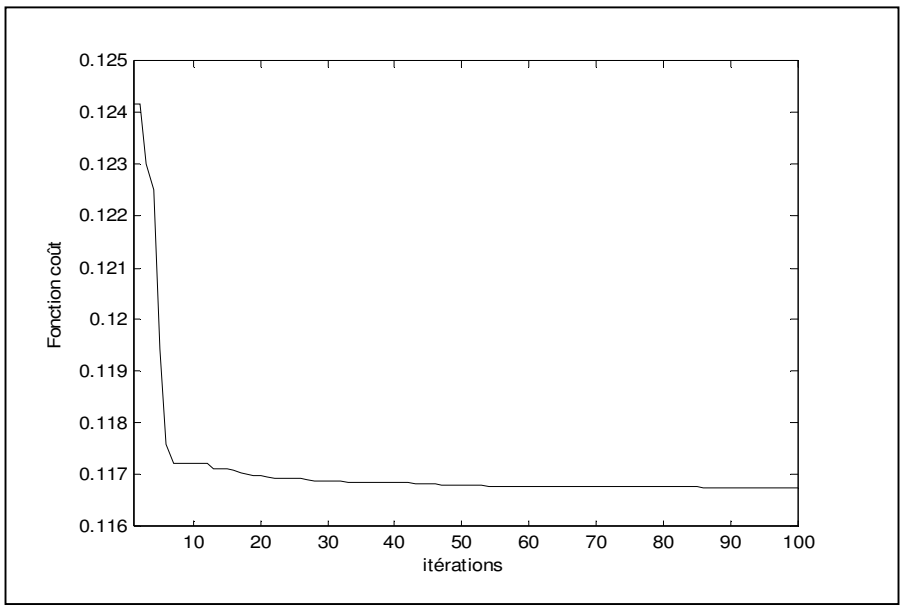


Fig. 3.8 : Evolution de la fonction coût.

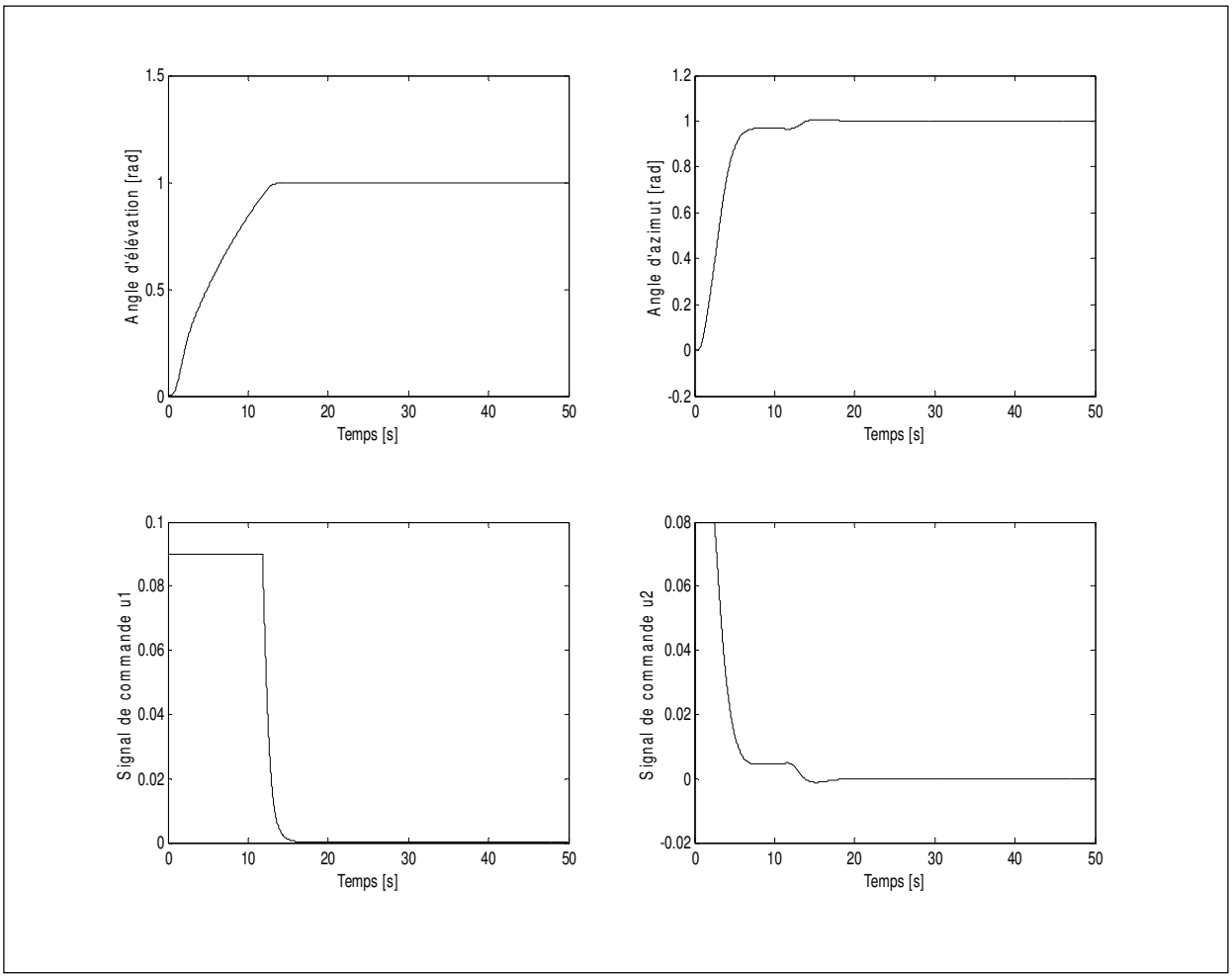


Fig. 3.9: Réponses du Simulateur d'hélicoptère après optimisation des contrôleurs flous.

La base de règles floues du CF2 obtenue après optimisation est comme suit :

$R_1 : Si\ e_2\ est\ N(-1,-0.60,-0.25)\ ET\ \Delta e_2\ est\ N(-1,-0.74,-0.07)\ Alors\ u_2 = -0.77.$

$R_2 : Si\ e_2\ est\ Z(-0.60,-0.25,0.56)\ ET\ \Delta e_2\ est\ Z(-0.74,-0.07,0.73)\ Alors\ u_2 = 0.22.$

$R_3 : Si\ e_2\ est\ P(-0.25,0.56,1)\ ET\ \Delta e_2\ est\ P(-0.07,0.73,1)\ Alors\ u_2 = 0.72.$

La figure 3.9 montre les réponses des angles d'élévation et d'azimut du système d'hélicoptère ainsi que leurs signaux de commande u_1 et u_2. L'algorithme proposé permet la conception des contrôleurs flous décentralisés pour un problème de poursuite de trajectoire du simulateur d'hélicoptère avec de bonnes performances.

3.3.2.1.b Commande couplée

La base de règles du contrôleur flou mutivariable est composée de neuf règles (voir la section 2.5.2.1.b), ce qui rend le vecteur de paramètres de la particule sera constitué de 15 paramètres. L'indice de performance (la somme des erreurs quadratiques moyennes), Fit est choisi comme la formule (2.14).

Les valeurs spécifiques de l'algorithme d'optimisation OEP-RT du contrôleur flou multivariable ainsi que les contraintes imposées sont données par le tableau 3.7.

Caractéristique	Valeur
Nombre de particules (N)	10
Nombre d'itérations ($\max\_iter$)	100
c_1, c_2	1.5
w	0.9
$[u_{min}, u_{max}]$	[-0.08[v], 0.08[v]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	1

Tab. 3.7 : Valeurs spécifiques de l'algorithme hybride d'optimisation OEP-RT

La figure 3.10 montre l'évolution l'indice de performance Fit durant l'exécution de l'algorithme OEP-RT. La base de règles floues obtenue du CF multivariable après la mise en échelles à l'entrée et à la sortie est comme suit :

$$R_1 : \textbf{Si}\ e_1 est\ N(-0.9,-0.55,-0.19)\ ET\ \Delta e_1\ est\ N\ (-0.06,-0.04,0.006)\ ET\ e_2\ est\ N\ (-1.1,-0.86,-0.34)\ ET\ \Delta e_2\ est\ N\ (-0.2,-0.17,-0.01)\ \textbf{Alors}\ u_1 = -1.46\ ET\ u_2 = -0.05$$

$$R_2 : \textbf{Si}\ e_1 est\ N(-0.9,-0.55,-0.19)\ ET\ \Delta e_1\ est\ Z\ (-0.04,0.006,0.03)\ ET\ e_2\ est\ N(-1.1,-0.86,-0.34)\ ET\ \Delta e_2\ est\ Z\ (-0.17,0.012,0.14)\ \textbf{Alors}\ u_1 = -1.46\ ET\ u_2 = -0.05.$$

$$R_3 : \textbf{Si}\ e_1\ est\ N(-0.9,-0.55,-0.19)\ ET\ \Delta e_1 est\ P(0.006,0.03,0.9)\ ET\ e_2\ est\ N(-1.1,-0.86,-0.34)\ ET\ \Delta e_2\ est\ P\ (0.012,0.14,0.2)\ \textbf{Alors}\ u_1 = -1.46\ ET\ u_2 = -0.05.$$

$$R_4 : \textbf{Si}\ e_1\ est\ Z(-0.55,-0.19,0.74)\ ET\ \Delta e_1\ est\ N(-0.06,-0.04,0.006)\ ET\ e_2\ est\ Z(-0.86,-0.34,0.79)\ ET\ \Delta e_2\ est\ N\ (-0.2,-0.17,-0.01)\ \textbf{Alors}\ u_1 = -1.46\ ET\ u_2 = -0.05.$$

$$R_5 : \textbf{Si}\ e_1\ est\ Z\ (-0.55,-0.19,0.74)\ ET\ \Delta e_1\ est\ Z(-0.04,0.006,0.03)\ ET\ e_2\ est\ Z(-0.86,-0.34,0.79)\ ET\ \Delta e_2\ est\ Z\ (-0.17,0.012,0.14)\ \textbf{Alors}\ u_1 = -0.17\ ET\ u_2 = -0.006.$$

$$R_6 : \textbf{Si}\ e_1\ est\ Z(-0.55,-0.19,0.74)\ ET\ \Delta e_1\ est\ P(0.006,0.03,0.06)\ ET\ e_2\ est\ Z(-0.86,-0.34,0.79)\ ET\ \Delta e_2\ est\ P\ (0.012,0.14,0.2)\ \textbf{Alors}\ u_1 = 1.55\ ET\ u_2 = 0.05.$$

$$R_7 : \textbf{Si}\ e_1 est\ P(-0.19,0.74,0.9)\ ET\ \Delta e_1\ est\ N(-0.06,-0.04,0.006)\ ET\ e_2\ est\ P\ (-0.34,0.79,1.1)\ ET\ \Delta e_2\ est\ N\ (-0.2,-0.17,-0.01)\ \textbf{Alors}\ u_1 = 1.55\ ET\ u_2 = 0.05.$$

$$R_8 : \textbf{Si}\ e_1\ est\ P(-0.19,0.74,0.9)\ ET\ \Delta e_1\ est\ Z(-0.04,0.006,0.03)\ ET\ e_2\ est\ P(-0.34,0.79,1.1)\ ET\ \Delta e_2\ est\ Z(-0.17,0.012,0.14)\ \textbf{Alors}\ u_1 = 1.55\ ET\ u_2 = 0.05.$$

$$R_9 : \textbf{Si}\ e_1\ est\ P(-0.19,0.74,0.9)\ ET\ \Delta e_1\ est\ P\ (0.006,0.03,0.9))\ ET\ e_2\ est\ P(-0.34,0.79,1.1)\ ET\ \Delta e_2\ est\ P\ (0.012,0.14,0.2)\ \textbf{Alors}\ u_1 = 1.55\ ET\ u_2 = 0.05.$$

La figure 3.11 montre les réponses des angles d'élévation et d'azimut du système d'hélicoptère ainsi que leurs signaux de commande optimisés u_1 et u_2. Il est clair que malgré la présence des interactions dans les systèmes multivariables, l'algorithme proposé a pu concevoir un contrôleur flou optimisé multivariable capable d'assurer une bonne poursuite de trajectoire d'un simulateur d'hélicoptère avec une bonne précision.

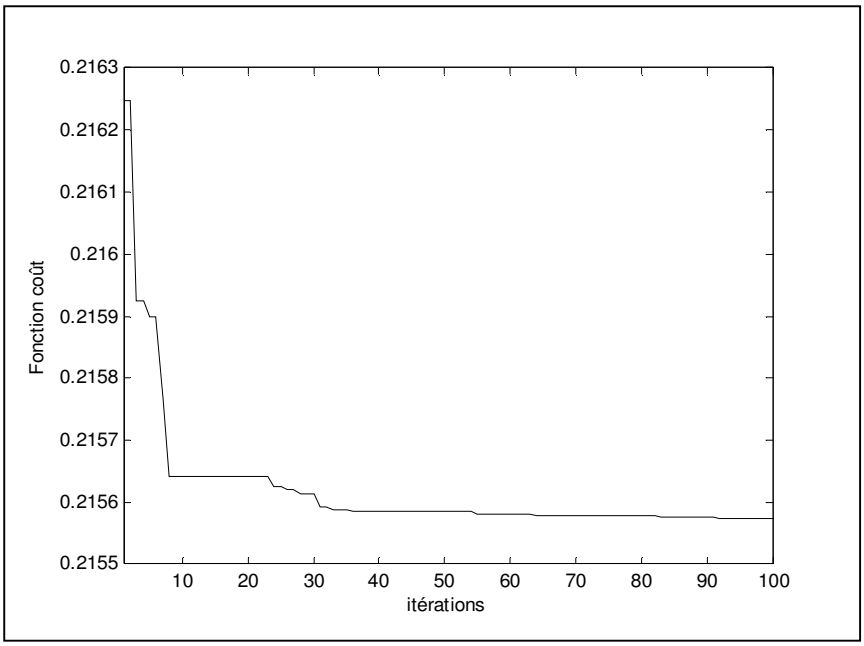


Fig. 3.10: Evolution de la fonction coût.

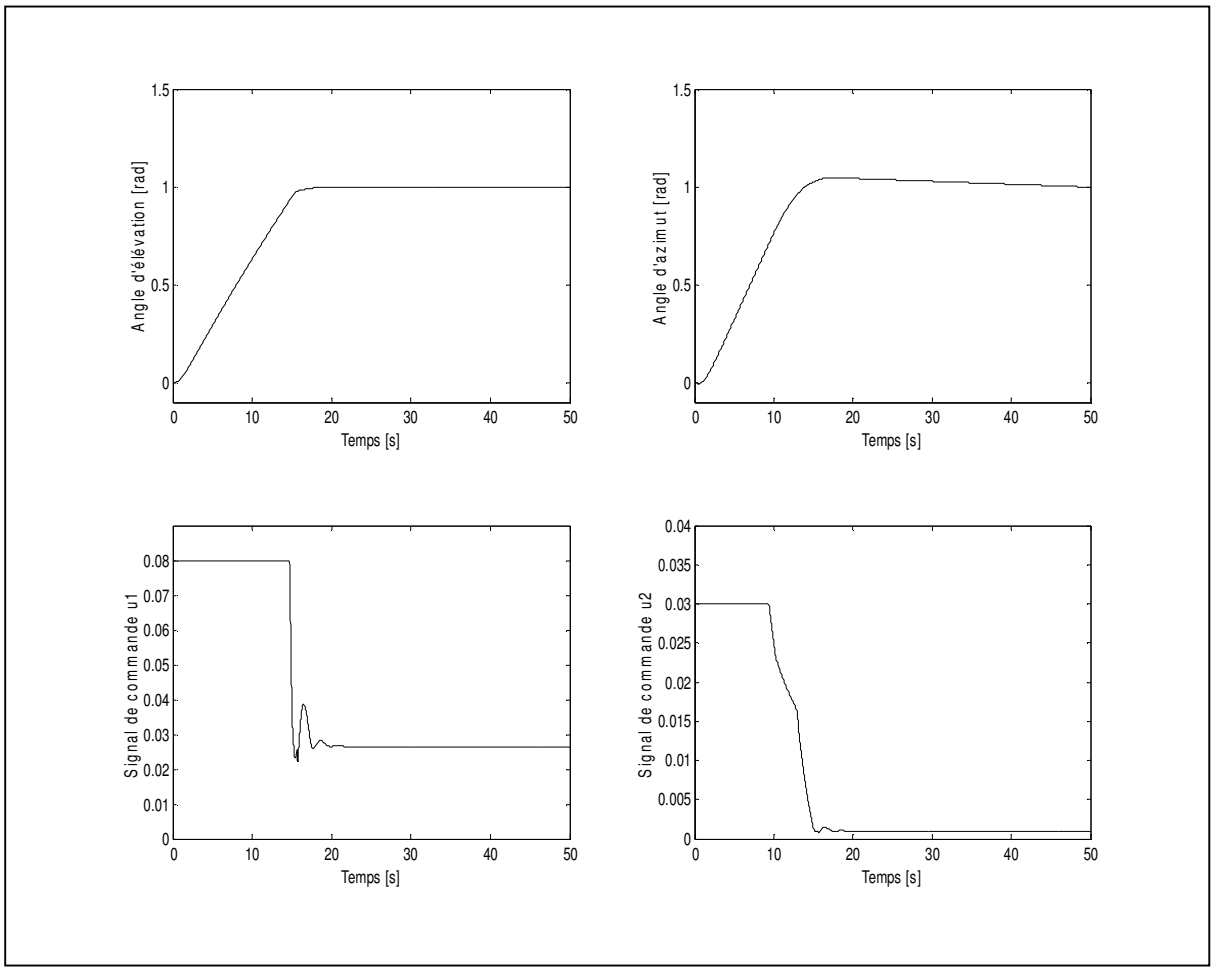


Fig. 3.11 : Réponses du système utilisant le contrôleur flou multivariable optimisé

3.3.2.2 Stabilisation du double pendule inversé

Comme il est décrit dans le chapitre précédent, le double pendule inversé est connecté par un ressort de rappel ; deux couples u_1 et u_2 sont appliqués aux entrées à travers des servomoteurs afin de garantir les positions perpendiculaires. La dynamique du système est représentée par l'expression (2.15). L'objectif est de maintenir le double pendule à l'état d'équilibre défini par $q_1 = 0$ [rad] et $q_2 = 0$ [rad] (commande en régulation). La structure de commande consiste à introduire deux contrôleurs flous qui travaillent séparément : un contrôleur flou pour l'angle q_1 et l'autre pour l'angle q_2. Les conditions initiales sont $(q_1(0), q_2(0)) =$ (0.5 [rad], -0.5 [rad]). Les valeurs spécifiques de l'algorithme hybride OEP-RT pour l'optimisation des contrôleurs flous du double pendule inversé ainsi que les contraintes imposées sont données par le tableau 3.8.

La figure 3.12 montre l'évolution de l'indice de performance Fit durant l'exécution de l'algorithme OEP-RT.

Caractéristique	Valeur
Nombre de particules (N)	10
Nombre d'itérations ($\max\_iter$)	100
c_1, c_2	1.5
w	0.9
Facteurs d'échelle de l'angle d'élévation ($G_{e_1}, G_{\Delta e_1}, G_{u_1}$)	(1, 0.5, 90)
Facteurs d'échelle de l'angle d'azimut ($G_{e_2}, G_{\Delta e_2}, G_{u_2}$)	(5, 2.6, 120)
Univers de discours	[-1, 1]
$[u_{min}, u_{max}]$	[-50 [N.m], 50 [N.m]]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	4

Tab.3.8 : Valeurs spécifiques de l'algorithme hybride d'optimisation OEP-RT.

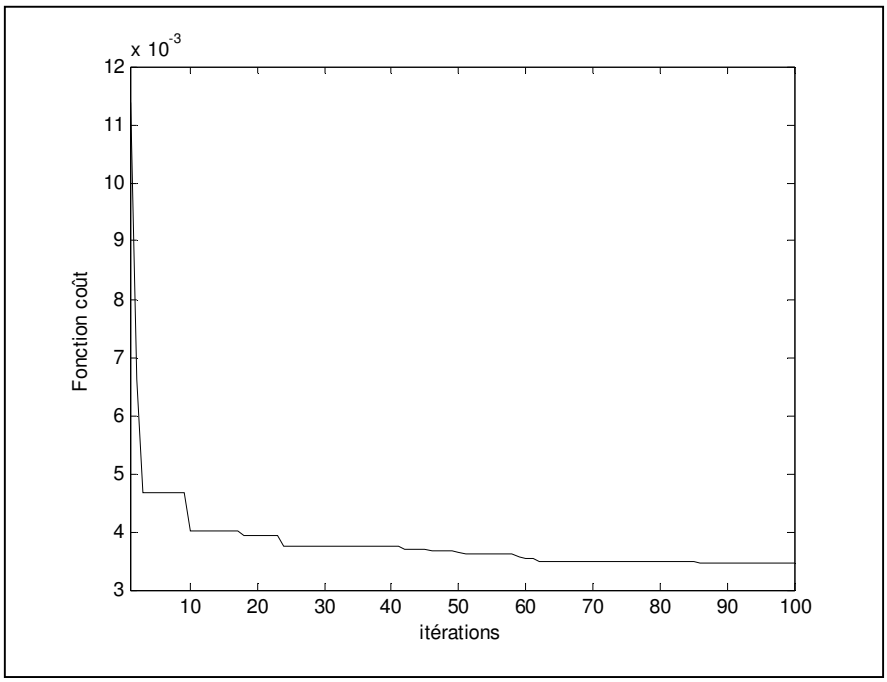


Fig. 3.12 : Evolution de la fonction coût.

La base de règles floues du CF1 obtenue après optimisation est comme suit :

$R_1 : Si\ e_1\ est\ N(-1,-0.80,0.46) ET\ \Delta e_1\ est\ N(-1,-0.79,0.00)\ Alors\ u_1 = -0.70.$

$R_2 : Si\ e_1\ est\ Z(-0.80,0.46,0.72) ET\ \Delta e_1\ est\ Z(-0.79,0.00,0.65)\ Alors\ u_1 = -0.43.$

$R_3 : Si\ e_1\ est\ P(0.46,0.72,1) ET\ \Delta e_1\ est\ P(0.00,0.65,1)\ Alors\ u_1 = 0.90.$

La base de règles floues du CF2 obtenue après optimisation est comme suit :

$R_1 : Sie_2\ est\ N(-1,-0.56,-0.5)\ ET\ \Delta e_2\ est\ N(-1,-0.69,0.25)\ Alors\ u_2 = -0.76.$

$R_2 : Si\ e_2\ est\ Z(-0.56,-0.5,0.89)\ ET\ \Delta e_2\ est\ Z(-0.69,0.25,0.57)\ Alors\ u_2 = 0.37.$

$R_3 : Si\ e_2\ est\ P(-0.5,0.89,1)\ ET\ \Delta e_2\ est\ P(0.25,0.57,1)\ Alors\ u_2 = 0.84.$

La figure 3.13 montre les réponses des positions angulaires q_1 et q_2, les vitesses angulaires $\dot{q}_1$ et $\dot{q}_2$ ainsi que leurs couples u_1 et u_2. Les résultats montrent la convergence des angles et des vitesses angulaires vers l'état d'équilibre.

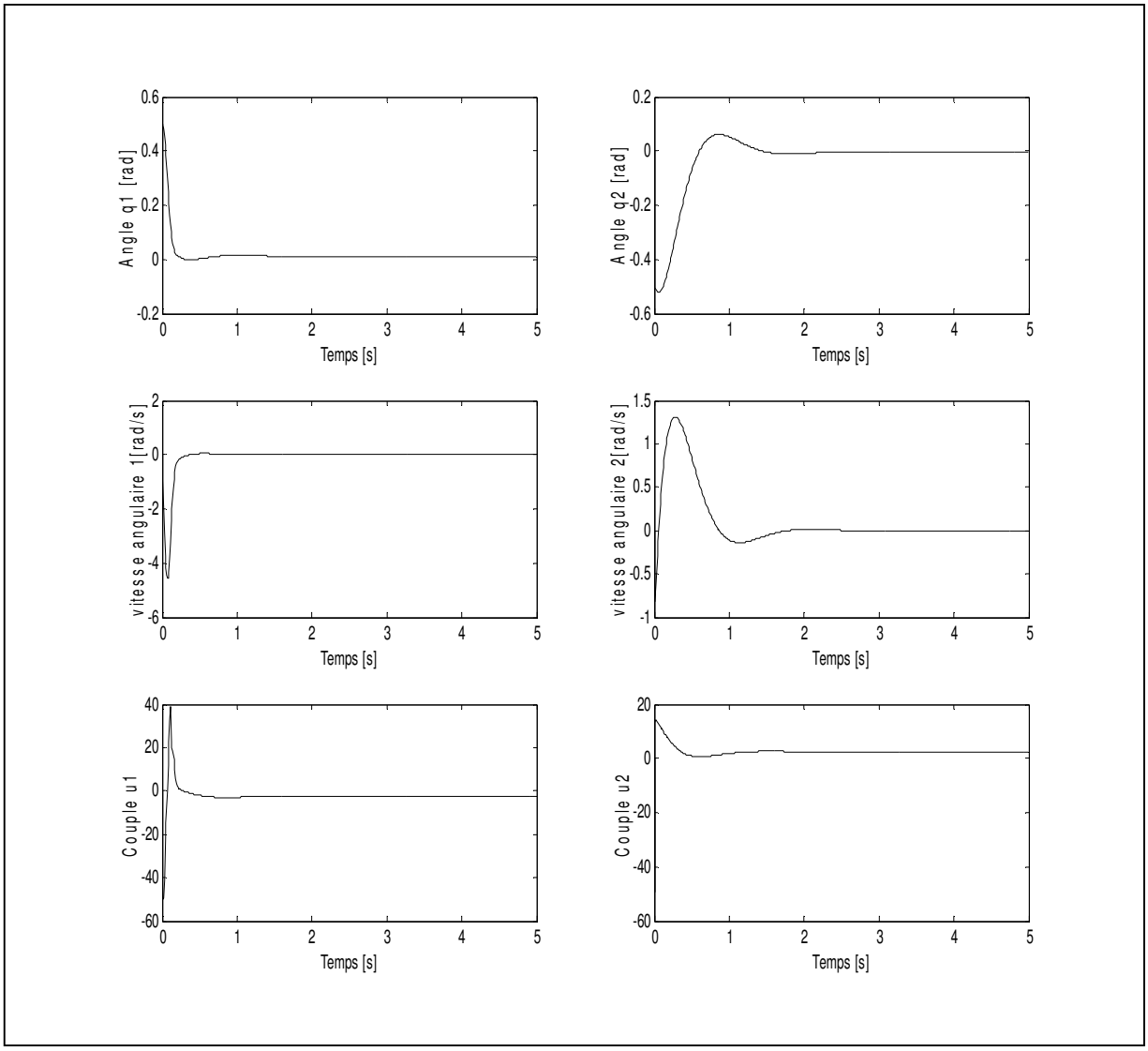


Fig. 3.13 : Les réponses du double pendule inversé soumis aux contrôleurs flous optimisés.

La robustesse du contrôleur est justifiée par sa capacité de bien réagir devant le changement des conditions initiales de fonctionnement ainsi que le changement des paramètres du double pendule inversé. Pour le premier cas, les conditions initiales testées sont les suivantes : $(q_1(0), q_2(0)) \in \{\pm 0.3\ , \pm 0.5, \pm 0.8, \pm 1.2\}$ [rad], avec $(\dot{q}_1(0), \dot{q}_2(0)) =$ (0 [rad/s],0 [rad/s]). La figure 3.14 illustre ces différentes situations.

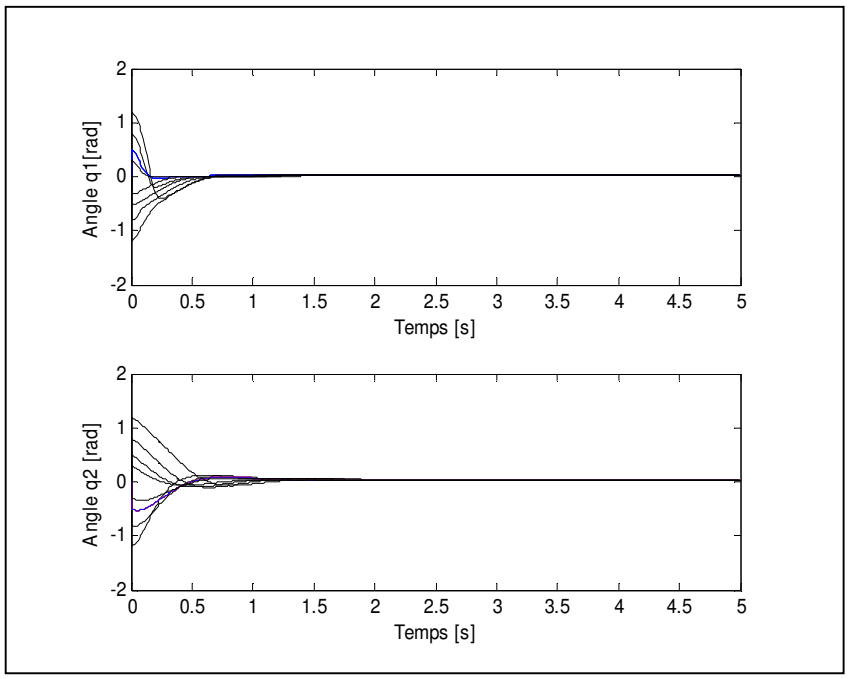


Fig. 3.14 : Réponses du système avec différentes conditions initiales

Dans le deuxième cas, la robustesse via le changement de paramètres est illustrée par une augmentation du paramètre $\delta = kr(l-b)$ [71]. La figure 3.15 montre l'évolution des positions angulaires pour 10δ. On remarque une convergence vers l'état d'équilibre avec une faible sensibilité au changement du paramètre δ. On conclut que l'algorithme de commande optimisée réalise de bonnes performances de régulation du double pendule inversé.

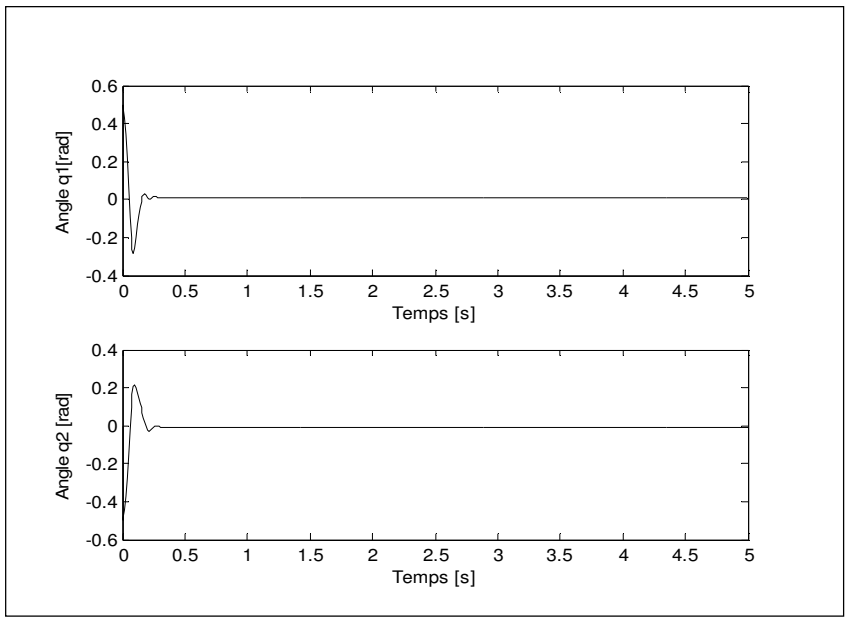


Fig. 3.15 : Test de robustesse via le changement du paramètre δ

3.4 Conclusions

Dans ce chapitre, nous avons proposé une méthodologie de synthèse d'une loi de commande simple, stable et robuste pour des systèmes non linéaires monovariables et multivariables. La stratégie de commande proposée, dans ce travail, repose sur l'hybridation d'un algorithme d'optimisation par essaim particulaire (OEP) et la recherche Tabou (RT). L'algorithme proposé permet l'optimisation paramétrique simultanée de prémisses et de conclusions de la base de règles floues d'un contrôleur flou de type Takagi-Sugeno d'ordre zéro. D'après les résultats de simulation des différents procédés non linéaires testés, nous avons pu constater que la stratégie de commande présente une robustesse en présence d'un changement dans les paramètres des systèmes non linéaires ainsi que le changement des conditions de fonctionnement.

La procédure de tests des paramètres de PSO représentent l'inconvénient majeur de l'algorithme. La lourde procédure d'optimisation peut être évitée par une bonne initialisation des paramètres fonctionnels de l'algorithme avec une représentation minimale.

Chapitre 4

Optimisation de la Base de Règles Floues pour la Modélisation des Systèmes Nonlinéaires par les Algorithmes Hybrides AGE-RT & OEP-RT

4.1 Introduction

L'avancement des recherches dans le domaine du flou a prouvé la capacité et la puissance des modèles flous dans l'identification floue des procédés non linéaires. Plusieurs chercheurs utilisent cette puissance d'optimisation pour représenter la dynamique des procédés non linéaires. La problématique de l'identification floue est basée sur la propriété d'approximation universelle des systèmes flous. En effet, ceux-ci sont capables d'approximer, avec un degré de précision arbitraire fixé, n'importe quelle dynamique non linéaire sur un ensemble compact [26] , [77] , [78]. Pour établir un modèle flou, nous devons fixer a priori quelques hypothèses qui permettent de trouver une représentation utile. Dans ce cas, des méthodes d'identification [79], [81], [82] peuvent être utilisées pour déterminer précisément la base de règles. Dans ce chapitre, nous étudions la construction de modèles flous de Takagi Sugeno d'ordre zéro pour les systèmes non linéaires par les algorithmes hybrides proposés AGE-RT et OEP-RT introduits aux chapitres précédents. Le modèle flou est à base de règles prédéterminée et fixe.

4.2 Identification d'un modèle flou

D'une manière générale, le problème d'identification floue consiste à choisir une forme de règles floues appropriée puis à concevoir des lois d'ajustement de paramètres ou de structure afin que, pour une même entrée, la sortie du modèle flou identifié approche suffisamment la sortie du procédé. La procédure d'identification des modèles flous de Takagi-Sugeno à conclusion constante peut être décomposée en une première phase d'identification de structure (nombre d'ensembles flous et leur répartition sur les univers de discours des entrées) suivie d'une seconde d'identification de paramètres (conclusions numériques dans les règles floues).

4.2.1 Identification de structure

L'objectif de l'identification de structure est de déterminer le nombre d'ensembles flous utilisés dans les prémisses de règles ainsi que leur répartition sur les univers de discours des entrées. Dans la littérature, il existe plusieurs techniques pour déterminer la structure d'un modèle flou de Takagi-Sugeno [82]. Dans cette thèse, on ne traitera pas cette tâche, c'est-à-dire que tout au long de ce travail, nous supposons que la structure du modèle flou (nombre d'ensembles flous, type de fonctions d'appartenance et le nombre de règles floues) est choisie à l'avance.

4.2.2 Identification de paramètres

Dans les méthodes d'identification, pour un procédé linéaire donné, la structure du modèle peut être choisie (modèle AR, ARMA, ARMAX, ...) dans la mesure où l'ordre du système est connu. Le problème d'identification est alors ramené à un problème d'estimation de paramètres. Cette philosophie peut être adoptée dans l'identification floue des procédés non linéaires. L'identification de paramètres suppose une structure de base de règles fixe. Dans ce cas, comme pour le cas linéaire, le problème d'identification est ramené à un problème d'estimation de paramètres.

Plusieurs chercheurs ont développé des techniques basées sur les métaheuristiques pour identifier les paramètres d'un modèle flou. Par exemple dans [83], la base de règles floues est construite en utilisant la recherche Tabou. Dans[66], l'optimisation par les colonies de fourmis est utilisée pour la conception de la base de règles floues de modèles flous,

l'hybridation des algorithmes génétiques et la recherche Tabou pour optimiser une base de règles floues pour un modèle flou de Takagi-Sugeno d'ordre zéro est proposée dans [56] tandis que dans [75] une approche hybride de l'optimisation par essaim particulaire et la recherche Tabou est utilisée. Ces deux dernières approches que nous avons proposées sont décrites dans ce qui suit.

4.3 Structure du modèle flou à identifier

4.3.1 Fuzzification et base de règles floues

On considère un modèle flou de type Takagi-Sugeno d'ordre zéro à deux entrées $x_1(t)$et $x_2(t)$, et une sortie $y_m(t)$.Les variables d'entrée sont caractérisés par des fonctions d'appartenance triangulaires, et la variable de sortie est caractérisée par des singletons flous. Une règle dans la base de règles floues a la forme générale suivante:

$$R_i: Si\ x_1(k)\ est\ A_1^i\ Et\ x_2(k)\ est\ A_2^i\ Alors\ y_m(k)\ est\ s^i \tag{4.1}$$

Où R_i est la $i^{ème}$ règle, x_j est la variable d'entrée, et y_m est la variable de sortie du modèle flou. A_j^iest une valeur linguistique floue définie par l'expression (4.2), et s^i sont des singletons flous.

$$A_j^i(x_j) = \max\left(\min\left[\frac{x_j - a_{j1}}{a_{j2} - a_{j1}}, \frac{a_{j3} - x_j}{a_{j3} - a_{j2}}\right], 0\right) \tag{4.2}$$

Avec a_{j1}, a_{j2} et a_{j3} sont les paramètres de la fonction d'appartenance triangulaire. Ces paramètres représentent les locations de point de départ, point de sommet et le point d'arrivée pour une fonction d'appartenance triangulaire, respectivement. La base de règles floues proposée se constitue seulement de trois règles actives extraites de l'analyse exprimées comme suit:

$R_1: Si\ x_1(k)\ est\ N(-1, a_{11}, a_{12})\ ET\ x_2(k)\ est\ N\ (-1, a_{21}, a_{22})\ Alors\ y_m(k) = s_1.$
$R_2: Si\ x_1(k)\ est\ Z\ (a_{11}, a_{12}, a_{13})\ ET\ x_2(k)\ est\ Z\ (a_{21}, a_{22}, a_{23})\ Alors\ y_m(k) = s_2.$
$R_3: Si\ x_1(k)\ est\ P\ (a_{12}, a_{13}, 1)\ ET\ x_2(k)\ est\ P\ (a_{22}, a_{23}, 1)\ Alors\ y_m(k) = s_3.$

Où N, Z et P sont des ensembles flous de la variable d'entrée ; leur signification est successivement Négative, Zéro et Positive.

4.3.2 Le moteur d'inférence et la défuzzification

Dans le mécanisme d'inférence, Le *ET* dans la règle floue est implémenté par le produit algébrique dans la théorie de la logique floue (selon Larsen). Ainsi, étant donné un ensemble de données d'entrée $\vec{x} = (x_1, x_2)$, le degrés d'activation $\beta_i(\vec{x})$ de la règle i est calculée par :

$$\beta_i(\vec{x}) = \prod_{j=1}^{2} A_j^i(x_j) \tag{4.3}$$

S'il y a r règles dans un CF, la sortie résultante de l'ensemble des règles est donnée par la moyenne des sorties individuelles pondérées par le degré d'activation des règles, soit :

$$y_m = \frac{\sum_{i=1}^{r} \beta_i(\vec{x}) s_i}{\sum_{i=1}^{r} \beta_i(\vec{x})} \tag{4.4}$$

Où s_i est la valeur de la conclusion de $i^{\text{ème}}$ règle.

L'optimisation d'un modèle flou comprend la détermination en ligne des paramètres de prémisses et de conclusion de chaque règle floue. La figure 4.1 montre la structure de modélisation d'un système nonlinéaire. $y(t+1)$ est la sortie du système à modéliser et $y_m(t+1)$ est la sortie du modèle flou optimisé.

Notre objectif est de définir les paramètres du modèle flou en minimisant un critère de performance. Dans ce chapitre, deux critères de performance permettant de mesurer la qualité du modèle flou sont utilisés:

- l'erreur quadratique moyenne définie par :

$$fit = EQM = \left(\frac{1}{nT}\right) \sum_{k=1}^{n} e^2(k) \tag{4.5}$$

Où, n est le nombre total d'échantillons, $e(k) = y(k) - y_m(k)$ et T est le tempsd'échantillonnage.

- la racine de l'erreur quadratique moyenne (REQM) définie par :

$$fit = \sqrt{EQM} = \sqrt{\left(\frac{1}{nT}\right) \sum_{k=1}^{n} e^2(k)} \tag{4.6}$$

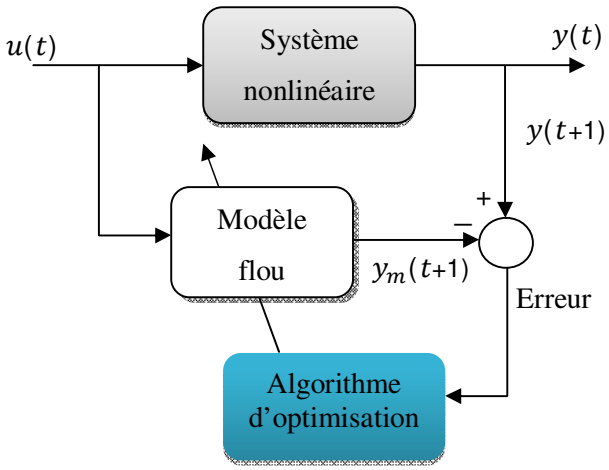


Fig.4.1 : Structure de modélisation

4.4 Simulations

Pour illustrer les performances des algorithmes hybrides d'optimisation AGE-RT et OEP-RT dans le domaine de conception de modèles flous, deux exemples sont utilisés : l'exemple de Narendra and Parthasarathy [84], et le système de Box and Jenkins [85].

4.4.1 Système nonlinéaire de Narendra et Parthasarathy

On considère un système non linéaire décrit par [84] :

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k) \qquad (4.7)$$

Où $u(k)$ et $y(k)$ sont les entrées et sorties du système, respectivement, à l'index de temps k (figure 4.2).

Les performances des algorithmes hybrides AGE-RT et OEP-RT sont évaluées en testant un ensemble de 250 points générés à partir d'un signal d'entrée sinusoïdal de la forme :

$$u(k) = \sin\left(\frac{\pi k}{50}\right) * \cos\left(\frac{\pi k}{30}\right), \qquad 0 \le k \le 250 \qquad (4.8)$$

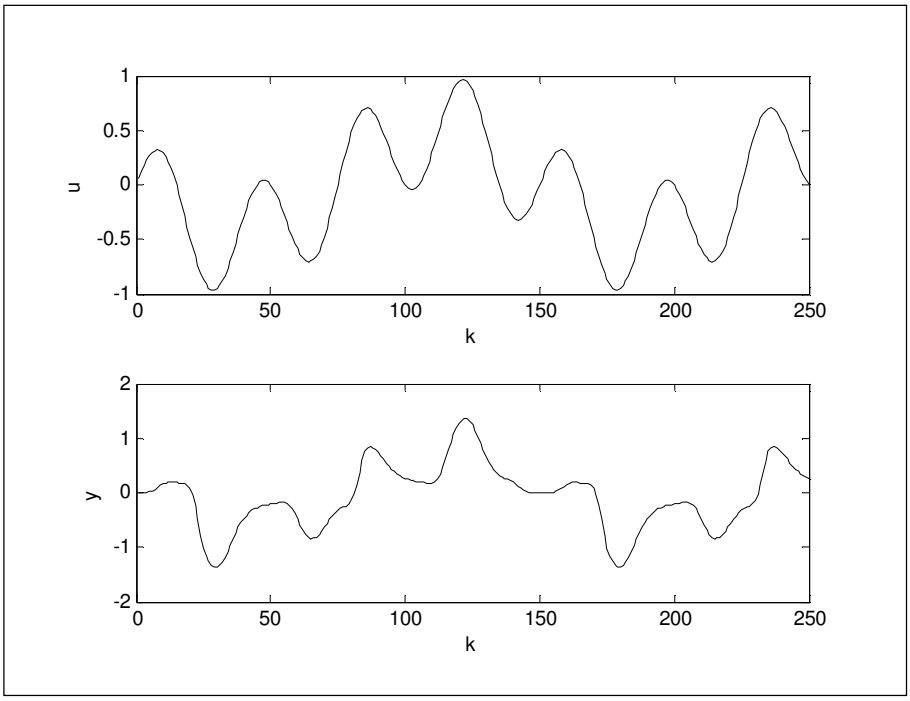


Fig. 4.2: Données d'entrée et de sortie du système de Narendra et Parthasarathy.

Les entrées du model flou sont $u(k)$ et $y(k)$ tandis que la sortie est $y_m(k+1)$. La fonction d'évaluation fit est donnée par :

$$fit = \sqrt{\frac{\sum_{k=0}^{249}(y_m(k+1) - y(k+1))^2}{250}} \tag{4.9}$$

4.4.2 Four à gaz de Box & Jenkins

Le système de Box & Jenkins est un processus dynamique nonlinéaire [85]. C'est un four à gaz, où l'air et le méthane sont combinés pour constituer un mélange de gaz contenant du CO2. L'entrée du four $u(t)$ correspond au débit du méthane, tandis que la sortie $y(t)$ correspond à la concentration en pourcentage de CO2. Le jeu des données d'entrée/sortie du four est constitué de 296 points (figure 4.3).

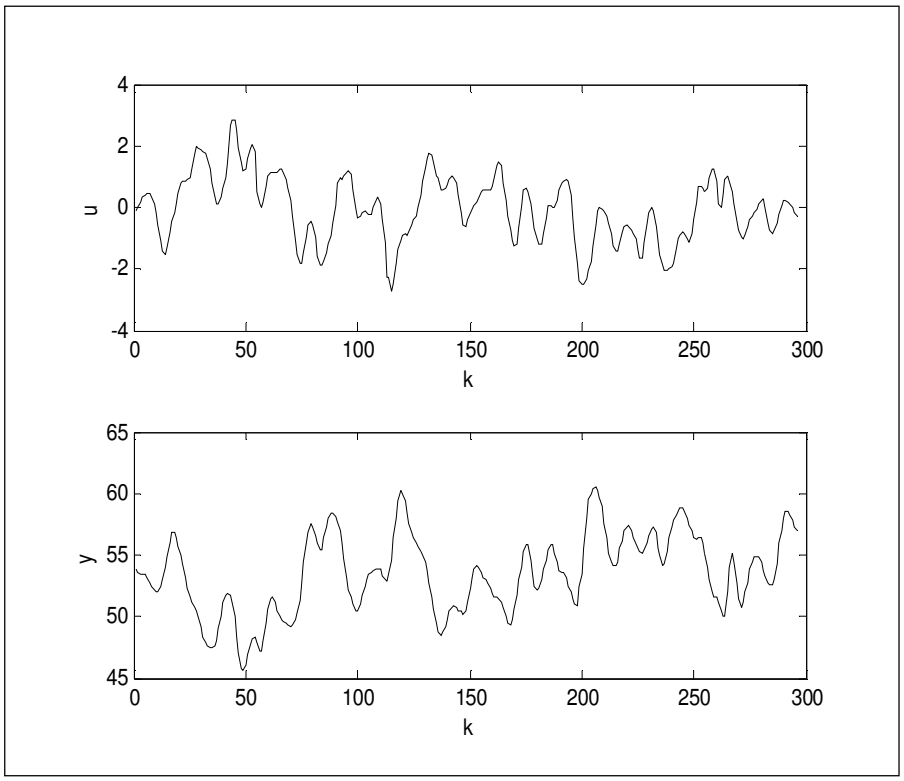


Fig. 4.3: Données d'entrée et de sortie du système de Box & Jenkins.

Les entrées du modèle flou sont $u(t-4)$, $y(t-1)$ respectivement, et la sortie du modèle est $y(t)$. Les intervalles de normalisation pour $u(t-4)$, $y(t-1)$ et $y(t)$ sont [-3,3], [44,62] et [44,62] respectivement. La fonction d'évaluation est choisi comme l'erreur quadratique moyenne donnée par :

$$fit = \frac{\sum_{k=0}^{295}(y_m(k) - y(k))^2}{296} \qquad (4.10)$$

4.4.3 Identification des modèles flous par l'approche AGE-RT

Avant de tester les performances de l'algorithme hybride AGE-RT pour l'identification des modèles flous, il faut préparer une plateforme pour l'algorithme principal des algorithmes génétiques et celui de la recherche Tabou. Pour cela les caractéristiques suivantes ont été considérées pour les AG :

- ☞ Le codage utilisé pour le chromosome est le codage réel.

- ☞ La population initiale sera générée aléatoirement pour couvrir tous l'espace de recherche.
- ☞ La méthode de la "loterie biaisée" est choisie pour l'opération de sélection.
- ☞ Le croisement en 2 points est utilisé.
- ☞ La mutation uniforme est choisie car elle est la plus simple et la plus adaptée au codage réel.
- ☞ La sélection finale pour la nouvelle génération sera par compétition, pour que les meilleurs individus (enfants et parents) participent dans la nouvelle génération.

Le chromosome est constitué de tous les paramètres de la base de règles floues, qui sont les paramètres des prémisses (valeurs modales), et les paramètres des conclusions des règles (dans notre cas des singletons).

4.4.3.1 Modèle flou de Narendra et Parthasarathy

Les valeurs spécifiques de chaque algorithme hybride AG-RT pour l'identification du modèle flou de Narendra et Parthasarathy sont données par le tableau 4.1. La figure 4.4 donne l'évolution de l'indice de performance durant l'exécution des 4 algorithmes AG-RT. Les 4 algorithmes donnent une bonne convergence vers la sortie désirée, mais la meilleure fonction d'évaluation est celle de l'AGE-RT.

Caractéristique	Valeur
Taille de la population (N)	10
Nombre de générations ($\max\_gen$)	100
Probabilité de croisement (p_c)	0.9
Probabilité de mutation (p_m)	0.08
Facteurs d'échelle ($G_u, G_y, G_{y\prime}$)	(8, 4.1, 4.1)
Univers de discours	[-1, 1]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	5

Tab.4.1 : Valeurs spécifiques de l'algorithme hybride d'optimisation AG-RT

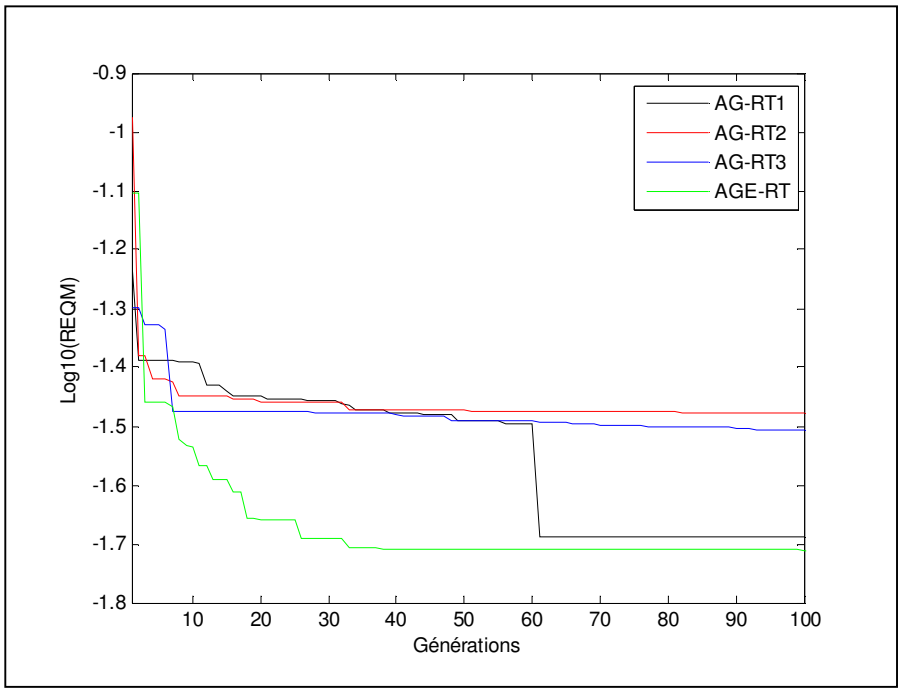


Fig. 4.4 : Evolution de la fonction d'évaluation.

La base de règles floues obtenue à la fin de l'exécution de l'algorithme AGE-RT est comme suit :

$R_1 : Si\ u\ est\ N(-1,\ -0.56, 0.43)\ ET\ y\ est\ N(-1, -0.59, -0.16)\ Alors\ y' = -0.59.$
$R_2 : Si\ u\ est\ Z(-0.56, 0.43, 0.57)\ ET\ y\ est\ Z(-0.59, -0.16,\ 0.70)\ Alors\ y' = 0.03.$
$R_3 : Si\ u\ est\ P(0.43, 0.57, 1)\ ET\ y\ est\ P(-0.16,\ 0.7, 1)\ Alors\ y' = 0.72.$

Le schéma de la figure 4.5 montre la sortie originale du système ainsi que la réponse du modèle flou optimisé. On remarque une convergence de la réponse du modèle flou vers la sortie désirée.

Le test de performance est réalisé en générant un nouveau signal d'entrée :

$$u(k) = 0.5 * \left(0.6 * u(k-1) + 0.6 * \sin\left(\frac{2\pi k}{25}\right) + 0.8 * \sin\left(\frac{\pi k}{32}\right)\right) \qquad (4.11)$$

La figure 4.6 montre le test de performance de l'algorithme hybride AGE-RT. Il est clair que l'algorithme d'optimisation AGE-RT est capable de concevoir un modèle flou optimisé avec une bonne approximation du système non linéaire.

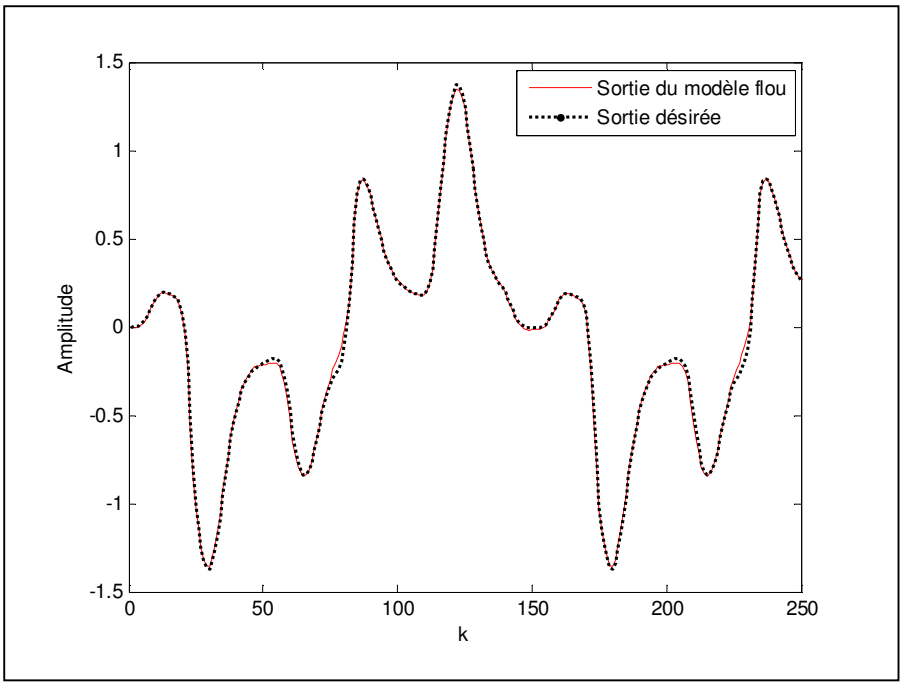


Fig. 4.5 : Résultats d'optimisation du modèle flou par AGE-RT

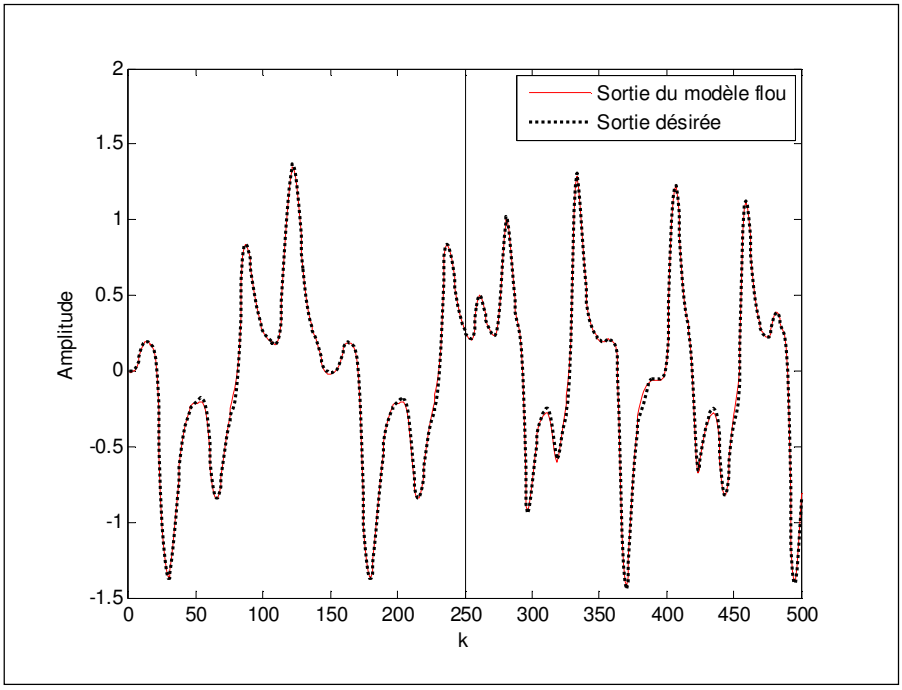


Fig. 4.6 : La validation du modèle flou de Narendra et Parthasarathy.

Les résultats d'identification du modèle flou de Narendra et Parthasarathy sont comparés avec ceux existés dans la littérature. Le tableau 4.2 montre les résultats d'optimisation par ACO, EGA, PSO, HGAPSO et RCACO pour le même problème d'identification [66]. En se

basant sur les résultats du tableau 4.2, on note que la fonction d'évaluation est minimale en comparant avec les autres méthodes.

Méthode d'optimisation	Nombre de règles floues	REQM
ACO [63]	5	0.130
EGA [62]	5	0.041
PSO [21]	5	0.045
HGAPSO [64]	5	0.040
RCACO [66]	5	0.026
AG-RT1 [56]	3	0.020
AG-RT2 [56]	3	0.027
AG-RT3 [56]	3	0.031
AGE-RT	3	0.019

Tab. 4.2 : Comparaison des performances de différentes méthodes d'identification des modèles flous de type TS d'ordre zéro pour l'exemple 1.

4.4.3.2 Modèle flou de Box & Jenkins

L'algorithme hybride AGE-RT est appliqué pour optimiser le modèle flou de type TS d'ordre zéro de Box & Jenkins. Les valeurs spécifiques de l'algorithme sont données par le tableau 4.3. La figure 4.7 montre l'évolution de l'indice de performance durant l'exécution de l'algorithme AGE-RT.

Caractéristique	**Valeur**
Taille de la population (N)	10
Nombre de générations ($\max\_gen$)	100
Probabilité de croisement (p_c)	0.4
Probabilité de mutation (p_m)	0.1
Facteurs d'échelle ($G_{u_{-4}}, G_{y_{-1}}, G_y$)	(6, 73, 73)
Univers de discours	[-1, 1]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	5

Tab.4.3 : Valeurs spécifiques de l'algorithme hybride d'optimisation AGE-RT

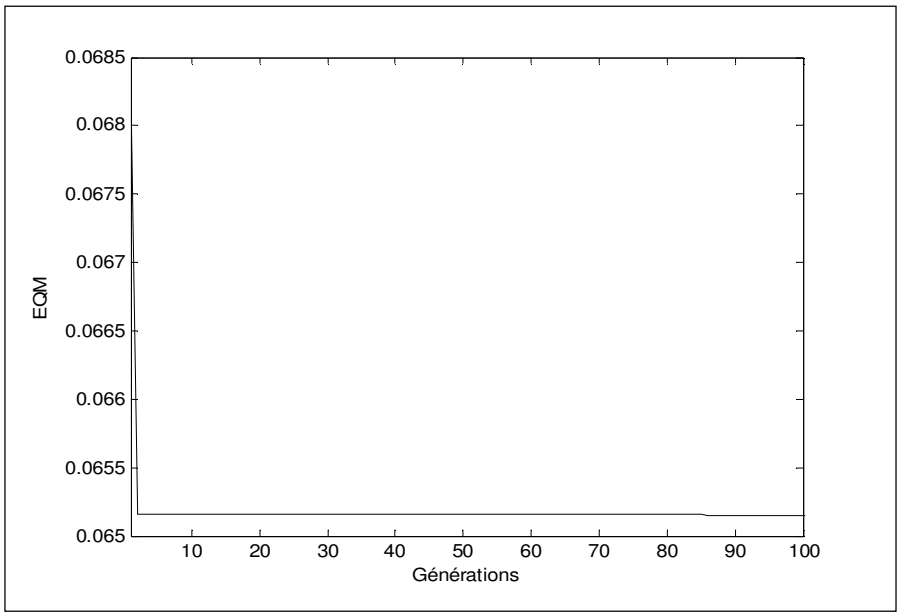


Fig. 4.7 : Evolution de la fonction d'évaluation.

La base de règles floues obtenue à la fin de l'exécution de l'algorithme AGE-RT est comme suit :

$R_1 : Si\ \ u_{-4}\ \ est\ N(-1,\ -0.63, -0.03)\ ET\ y_{-1}\ est\ N(-1, -0.80, -0.09)\ Alors\ y = -0.82.$

$R_2 : Si\ \ u_{-4}\ \ est\ Z(-0.63, -0.03, 0.54)\ ET\ y_{-1} est\ Z\ (-0.80, -0.09, 1.0)\ Alors\ y = 0.01.$

$R_3 : Si\ \ u_{-4}\ \ est\ P(-0.03, 0.54, 1)\ ET\ y_{-1}\ est\ P(-0.09, 1.0, 1)\ Alors\ \ y = 0.90.$

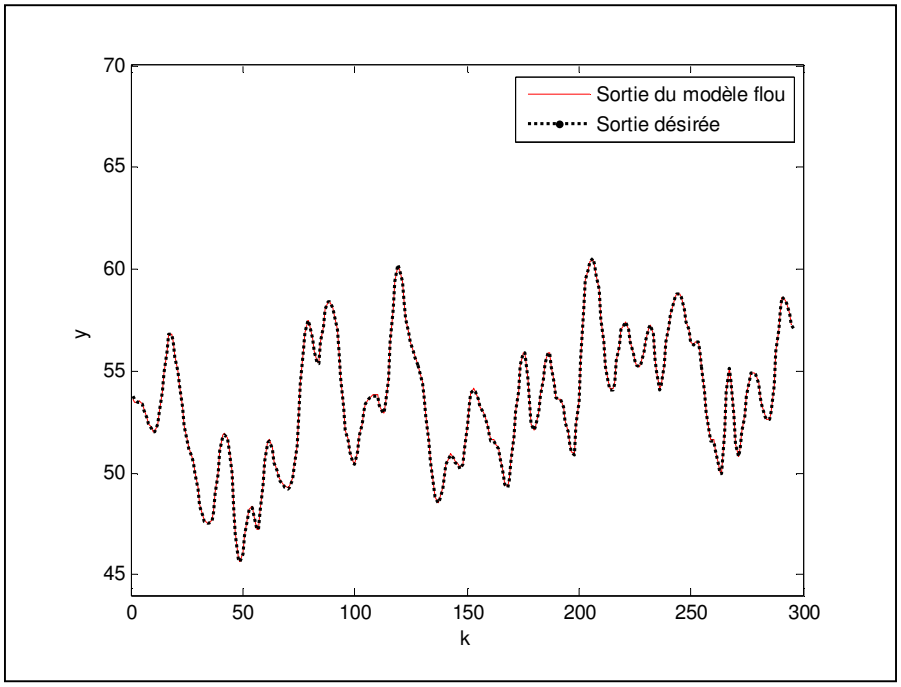


Fig. 4.8 : Résultats d'optimisation du modèle flou de Box & Jenkins.

Les résultats d'optimisation du modèle flou sont donnés par la figure 4.8. La figure montre une bonne approximation du système de Box & Jenkins.

Dans le tableau 4.4, le modèle optimisé est comparé avec d'autres modèles identifiés pour les mêmes données d'entrée/sortie du four de Box & Jenkins. En comparant la méthode proposée avec d'autres citées dans la littérature, il est clair qu'un nombre minimum de règles floues est suffisant pour garantir une convergence du modèle flou optimisé vers la sortie désirée.

Méthode d'optimisation	Nombre de règles floues	EQM
Box and Jenkins [85]	-	0.202
Tong [86]	19	0.469
Takagi and Sugeno [27]	6	0.190
Sugeno and Tanaka [87]	2	0.359
Wang and Langari [25]	5	0.158
Kang et *al* [88]	5	0.161
Bagis [89]	4	0.148
AGE-RT	3	0.065

Tab.4.4 : Comparaison de différentes méthodes d'optimisation pour le modèle flou de l'exemple 2.

4.4.4 Identification des modèles flous par l'approche OEP-RT

La particule est constituée des paramètres des prémisses (valeurs modales), et des paramètres des conclusions des règles (dans notre cas des singletons). Dans l'EOP, la population initiale est générée aléatoirement.

4.4.4.1 Modèle flou de Narendra et Parthasarathy

Les valeurs spécifiques de l'algorithme hybride OEP-RT pour l'identification du modèle flou de Narendra et Parthasarathy sont données par le tableau 4.5. L'évolution de la fonction coût au cours de l'exécution de l'algorithme hybride OEP-RT est illustrée par la figure 4.9.

Caractéristique	Valeur
Nombre de particules (N)	10
Nombre d'itérations (max $\_iter$)	100
c_1, c_2	1.5
w	0.9
Facteurs d'échelle ($G_e, G_{\Delta e}, G_u$)	(8, 3.5, 3.5)
Univers de discours	[-1, 1]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	5

Tab.4.5 : Valeurs spécifiques de l'algorithme hybride d'optimisation OEP-RT

La base de règles floues obtenue après optimisation est comme suit :

$R_1 : Si\ u\ est\ N(-1,\ -0.69, 0.17)\ ET\ y\ \ est\ N(-1, -0.73, -0.06)\ Alors\ \ y' = -0.79.$

$R_2 : Si\ u\ est\ Z(-0.69, 0.17, 0.62)\ ET\ y\ \ est\ Z(-0.73, -0.06,\ 0.64)\ Alors\ y' = -0.10.$

$R_3 : Si\ u\ est\ P(0.17, 0.62, 1)\ ET\ y\ est\ P(-0.06,\ 0.64\ 1)\ Alors\ \ y' = 0.88.$

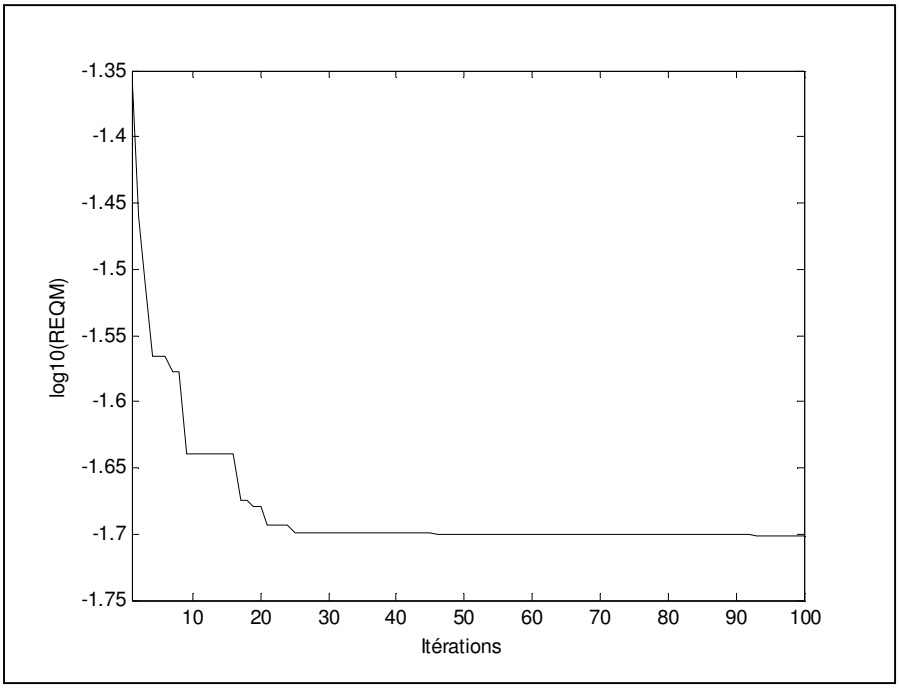


Fig. 4.9 : Evolution de la fonction d'évaluation.

La figure 4.10 montre la sortie originale du système ainsi que la réponse du modèle flou optimisé. On remarque une convergence de la réponse du modèle flou vers la sortie désirée.

Le test de performance est réalisé dans la même figure en générant un nouveau signal d'entrée :

$$u(k) = 0.5 * \left(0.6 * u(k-1) + 0.6 * \sin\left(\frac{2\pi k}{25}\right) + 0.8 * \sin\left(\frac{\pi k}{32}\right)\right) \qquad (4.11)$$

D'après la figure 4.10, il est clair que l'algorithme d'optimisation OEP-RT est capable de concevoir un modèle flou optimisé avec une bonne approximation du système à identifier.

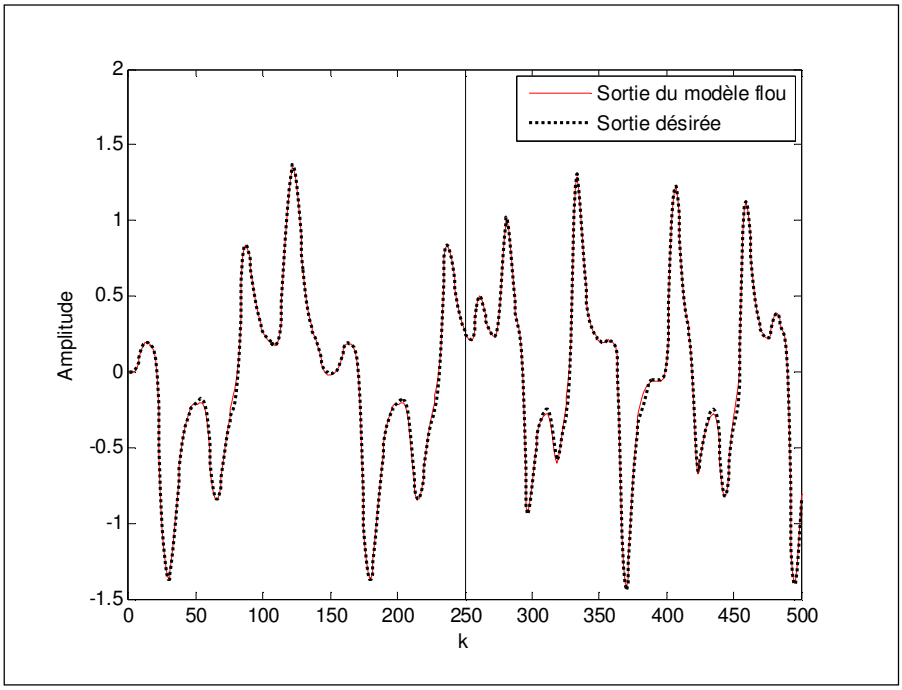


Fig. 4.10 : L'identification et la validation du modèle flou de Narendra et Parthasarathy par OEP-RT.

Les résultats d'optimisation du modèle flou de Narendra et Parthasarathy sont comparés avec ceux existés dans la littérature. Le tableau 4.6 montre les résultats d'optimisation par ACO, EGA, PSO, HGAPSO et RCACO pour le même problème d'identification [66]. En se basant sur les résultats du tableau 4.6, on note que la fonction d'évaluation est minimale en comparant avec les autres méthodes évolutionnaires.

Méthode d'optimisation	Nombre de règles floues	REQM
ACO [63]	5	0.130
EGA [62]	5	0.041
PSO [21]	5	0.045
HGAPSO [64]	5	0.040
RCACO [66]	5	0.026
AGE-RT	3	0.019
OEP-RT	3	0.019

Tab. 4.6 : Comparaison des performances de différentes méthodes d'identification des modèles flous de type TS d'ordre zéro pour l'exemple 1.

4.4.4.2 Modèle flou de Box & Jenkins

L'algorithme hybride OEP-RT est appliqué pour optimiser le modèle flou de type TS d'ordre zéro de Box & Jenkins. Les valeurs spécifiques de l'algorithme sont données par le tableau 4.7. La figure 4.11 montre l'évolution de l'indice de performance (erreur quadratique moyenne) durant l'exécution de l'algorithme.

Caractéristique	**Valeur**
Nombre de particules (N)	10
Nombre d'itérations ($\max\_iter$)	100
c_1, c_2	1.5
w	0.9
Facteurs d'échelle ($G_e, G_{\Delta e}, G_u$)	(6, 73, 73)
Univers de discours	[-1, 1]
Taille de liste tabou (T)	7
Nombre de solutions voisines (N_v)	8
Nombre d'itérations de la RT (k)	5

Tab.4.7 : Valeurs spécifiques de l'algorithme hybride d'optimisation OEP-RT

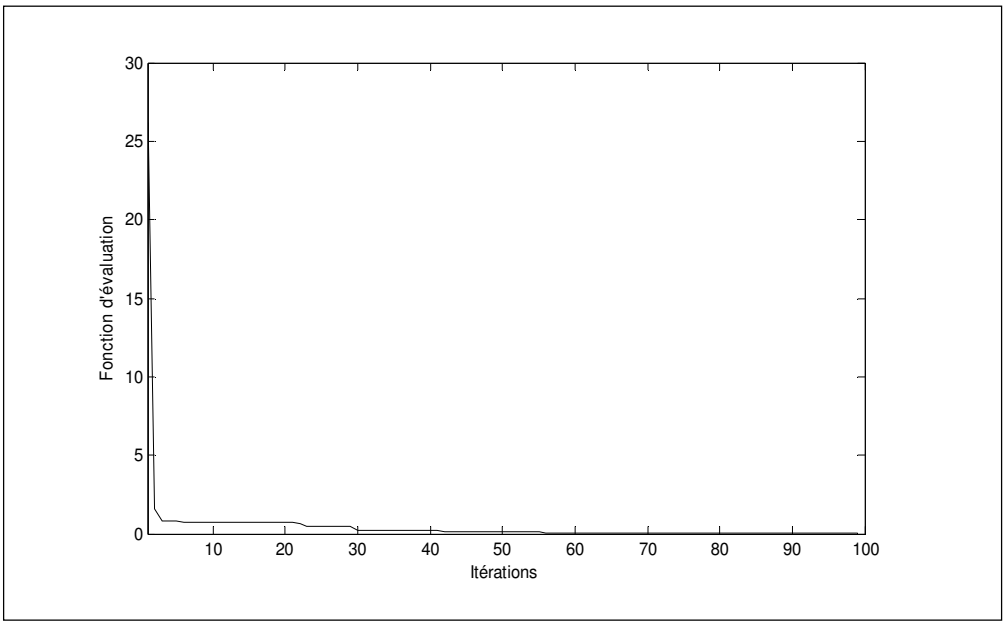


Fig. 4.11 : Evolution de la fonction d'évaluation.

La base de règles floues obtenue à la fin de l'exécution de l'algorithme OEP-RT est comme suit :

$R_1 : Si\ \ u_{-4}\ est\ N(-1,-0.7,-0.16)\ ET y_{-1}\ est\ N(-1,-0.73,-0.16)\ Alors\ y = -0.6.$

$R_2 : Si\ \ u_{-4}\ est\ Z(-0.7,-0.16,0.7)\ ET\ y_{-1}\ est\ Z(-0.73,-0.16,0.96)\ Alors\ \ y = -0.03.$

$R_3 : Si\ \ u_{-4}\ est\ P(-0.16,0.7,1)\ ET\ y_{-1}\ est\ P(-0.16,0.96,1)\ Alors\ y = 0.83.$

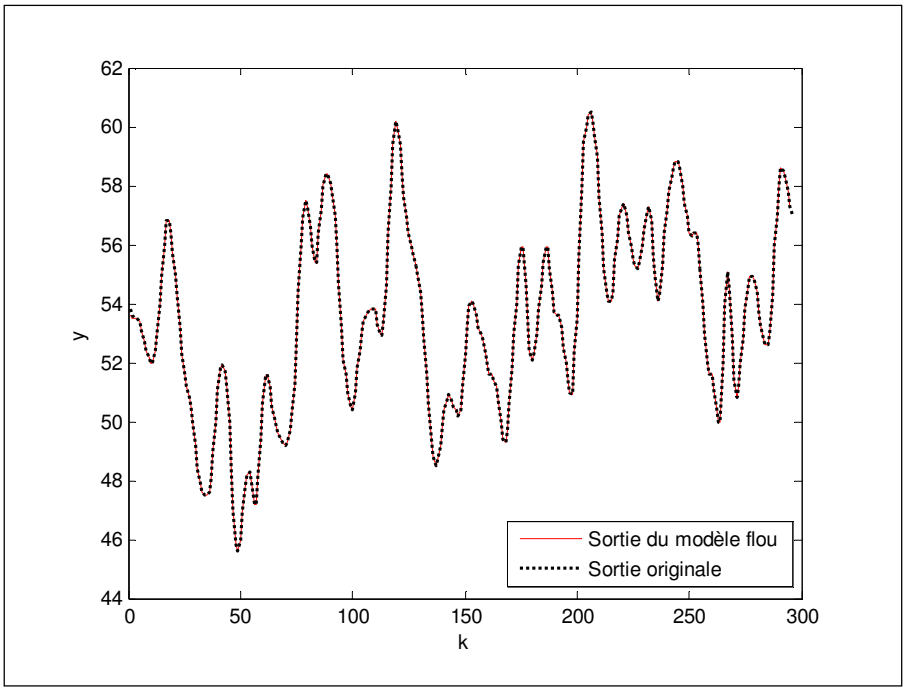


Fig. 4.12 : Résultats d'optimisation du modèle flou de Box & Jenkins.

Les résultats d'optimisation du modèle flou sont donnés par la figure 4.12. La figure montre une bonne approximation du système de Box & Jenkins.

Dans le tableau 4.8, le modèle optimisé est comparé avec d'autres modèles identifiés pour les mêmes données d'entrée/sortie du four de Box & Jenkins. En comparant la méthode proposée avec d'autres citées dans la littérature et avec AGE-RT, il est clair que la fonction d'évaluation dans les deux approches proposée est la même tous en garantissant une convergence du modèle flou optimisé vers la sortie désirée.

Méthode d'optimisation	Nombre de règles floues	EQM
Sugeno and Tanaka [86]	2	0.359
Wang and Langari [25]	5	0.158
Nakoula et *al* [87]	90	0.175
Kang et *al* [88]	5	0.161
Bagis [89]	4	0.148
AGE-RT	3	0.065
OEP-RT	3	0.065

Tab.4.8 : Comparaison de différentes méthodes d'optimisation pour le modèle flou de l'exemple 2.

4.5 Conclusions

Dans ce chapitre les algorithmes hybrides AGE-RT et OEP-RT ont été appliqués pour l'optimisation des prémisses et de conclusion de la base de règle floues pour une bonne approximation des systèmes non linéaires. Les résultats ont montré que les approches proposées pour l'identification des modèles flous accomplissent efficacement les performances désirées. En comparant les deux approches proposées avec d'autres existées dans la littérature, les deux algorithmes hybrides AGE-RT et OEP-RT ont de bonnes performances (grande précision), en utilisant une structure simple du modèle flou (nombre minimum de règles floues et d'ensembles flous).

Notons que l'inconvénient majeur des deux algorithmes est le choix des paramètres fonctionnels. La lourde procédure d'identification peut être évitée par une bonne initialisation des paramètres fonctionnels de l'algorithme avec une représentation minimale.

Conclusion Générale

Le travail présenté dans cette thèse concerne la commande et la modélisation floues des systèmes nonlinéaires en utilisant les techniques avancées de l'automatique telles que la logique floue et les outils d'optimisation tels que les algorithmes génétiques et l'algorithme par essaim particulaire. Les approches proposées reposent sur l'hybridation d'un algorithme évolutionnaire et un autre algorithme dit de recherche locale. Notre objectif est d'exploiter les avantages de chacun d'entre eux pour aboutir à un algorithme de recherche globale qui ne tombe jamais dans les optimums locaux. La simplicité d'implantation de la recherche tabou, son efficacité à cause de sa mémoire qui stocke toutes solutions visitées, la qualifie d'être la meilleure méthode de recherche locale pour l'hybridation avec les AG et l'OEP.

Les travaux présentés dans cette thèse s'articulent autour des principaux axes de la commande et la modélisation floues de type Takagi-Sugeno d'ordre zéro pour les systèmes nonlinéaires complexes. Ces travaux ont dépisté, d'une façon cohérente, deux grandes voies :

- La première a relaté la commande floue des systèmes nonlinéaires monovariables et multivariables.
- La deuxième s'est occupée de la modélisation floue des systèmes nonlinéaires à partir de données.

La première piste a finalisé la construction d'une loi de commande floue stable et robuste. En premier stade, l'hybridation des algorithmes génétiques et la recherche Tabou a pu optimiser en ligne les prémisses et les conclusions de la base de règles floues d'un contrôleur

flou de type TS d'ordre zéro. Selon la position de la méthode Tabou dans l'algorithme global des AG, on a pu distinguer quatre algorithmes d'optimisation. Ces algorithmes ont été testés pour le système du pendule inversé. D'après les résultats de simulation, on a choisi de continuer nos tests dans la commande et la modélisation floues par l'algorithme hybride des algorithmes génétiques avec élitisme et la recherche Tabou (AGE-RT). Après, une autre hybridation de l'algorithme par essaim particulaire et la recherche Tabou a été établie (OEP-RT). Le principe est que la recherche Tabou améliore le déplacement des particules de l'essaim en le dirigeant vers l'optimum global.

Les deux approches ont été testées et validées pour plusieurs applications, en simulation. Ces applications ont été faites sur divers procédés monovariables et multivariables, notamment, un pendule inversé, un système de contrôle de la température de bain d'eau, un simulateur d'hélicoptère et un système de double pendule inversé. La stratégie de commande optimisée par les deux approches AG-RT et OEP-RT, en régulation et en poursuite, a effectivement assuré la stabilité et la robustesse du système commandé.

Dans la deuxième partie de la thèse, Les différentes possibilités qu'offrent les modèles flous du type TS d'ordre zéro pour la modélisation des systèmes nonlinéaires en utilisant les algorithmes hybrides AGE-RT et OEP-RT. Les deux approches ont été appliquées pour l'identification de modèles flous, notamment, le modèle de Box & Jenkins et le modèle de Narendra & Parthasarathy. L'optimisation de modèles flous a donné une bonne approximation des systèmes nonlinéaires avec une grande précision.

Enfin, le succès des stratégies de commande et de modélisations floues provient de l'algorithme hybride proposé et l'ensemble structure-paramètres du système d'inférence floue d'un côté ; et du potentiel de transfert de connaissances de l'expertise sur le processus d'un autre côté. Car l'écriture de la base de règle floues et la définition de la structure du contrôleur ou du modèle flou nécessite un minimum d'informations (connaissances) sur le système à commander ou à modéliser.

Dans le cadre de développement futur de ce travail, il est intéressant d'élaborer une loi de commande optimisée par d'autres techniques d'optimisation (colonies de fourmis, colonies d'abeilles, etc) pour la commande couplée des systèmes multivariables en prenant en considération :

- ☞ Le couplage croisé dans le cas des systèmes multivariables,
- ☞ Les interconnexions entre les différents sous-systèmes, dans le cas des grands systèmes,
- ☞ Les perturbations externes qui peuvent affectées le système.

Bibliographie

[1] L. A. Zadeh, "Fuzzy sets", *Information and Control*, Vol. 8, 1965, pp.338-353.

[2] I. Hayashi, H. Nomura, and N. Wakami, "Acquisition of Inference Rules by Neural Network Driven Fuzzy Reasoning", *Japanese Journal of Fuzzy Theory and Systems* , 2(4), 1990, pp. 453-469.

[3] L.X. Wang, "Fuzzy Systems as Nonlinear Dynamic System identifiers Part I: Design", *Proceedings of the 31st Conference on Decision and Control*,Tucson, AZ, 1992, pp. 897-902.

[4] L.X. Wang and J.M. Mendel, "Back-Propagation Fuzzy System as Nonlinear Dynamic System Identifiers", *Proceedings of the first IEEE Conference on Fuzzy Systems* , 1992, pp.1409-1418.

[5] S. Horikawa, T. Furuhashi and Y. Uchikawa, "Composition Methods and Learning Algorithms of Fuzzy Neural Networks," Japanese Journal of Fuzzy Theory and Systems , 4(5), 1992, pp. 529-556.

[6] L.X. Wang, "Stable Adaptive Fuzzy Control of Nonlinear Systems", I*EEE, Trans. On Fuzzy Systems,* Vol. 1, No. 2, 1993, pp. 146-155.

[7] L.X. Wang, "Design and Analysis of Fuzzy Identifiers of Nonlinear Dynamic Systems", *IEEE Transactions on Automatic Control*, 40(1), 1995, pp. 11-23.

[8] H. Nomura, I. Hayashi and N. Wakami, "A Learning Method of Fuzzy Inference Rules by Descent Method," Proceedings of the first IEEE Conference on Fuzzy Systems , 1992, pp. 203-210.

[9] H. Nomura, I. Hayashi and N. Wakami, "Self-Tuning Fuzzy Reasoning By Delta Rule and Its Application to Obstacle Avoidance," Japanese Journal of Fuzzy Theory and Systems , 4(2), 1992, pp. 261-272.

[10] J.S.R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference Systems", *IEEE Transactions on Systems, Man, and Cybernetics,* 23(3), 1993, pp. 665-685.

[11] Y. Shi and M. Mizumoto, "An improvement of neuro-fuzzy learning algorithm for tuning fuzzy rules", *Fuzzy sets and Systems*, vol. 118, 2001, pp. 339-350.

[12] I. Rojas, H. Piomares, J. Ortega, A. Prieto, "Self-organized fuzzy system generation from training examples", *IEEE Trans. Fuzzy Systems*, vol. 1, 2000, pp. 23-36.

[13] N.K. Kasabov and Q. Song, "DENFIS: Dynamic Evolving Neural-Fuzzy Inference System and Its Application for Time-Series Prediction", *IEEE Trans. on Fuzzy Systems,* vol. 10, n°. 2, 2002, pp. 144-154.

[14] X. Yao, "Evolutionary computation─Theory and Applications", *World Scientific*, Singapore, 1999.

[15] P. Thrift, "Fuzzy logic synthesis with genetic algorithms",*In Proceedings of the Fourth International Conference on Genetic Algorithms*, 1991, pp. 509–513.

[16] C. L. Karr, "Applying genetics to fuzzy logic", *AI Expert*, 6(3), 1991,pp. 38-43.

[17] M. Mohammadian and R. stonier, "Generating fuzzy rules by genetic algorithms", *Proceeding of 3rd IEEE international Workshop on Robot and Human Communication,* Nagoya, 1994, pp. 362-367.

[18] F. Herrera,M. Lozano and J.L. Verdegay, "Tuning fuzzy logic controllers by genetic algorithms", *International J ournal of Approximate Reasoning,* vol. 12,1995,pp. 299–315.

[19] C.C. Chen and C.C. Wong, "Self-generating rule-mapping fuzzy controller design using a genetic algorithm", *In IEE Proceedings on Control Theory and Applications,* vol. 49, 2002, pp. 143–148.

[20] K. Belarbi, F. Titeli, W. Bourebia et K. Benmahammed,"Design of mamdani fuzzy logic controllers with rule base minimisation using genetic algorithm", *Engineering applications of artificial intelligence*, vol. 18, 2005, pp. 875–880.

[21] J. Kennedy et R.C. Eberhart, " Particle swarm optimization", *IEEE International Conference on Neural Networks,* Piscataway*,* 1995, pp. 1942-1948.

[22] C. F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, 2004, pp. 997–1006.

[23] A. Chatterjee, K. Pulasinghe, K. Watanabe, and K. Izumi, "A particleswarm-optimized fuzzy-neural network for voice-controlled robot systems," *IEEE Trans. Ind. Electron.*, vol. 52, no. 6, 2005, pp. 1478–1489.

[24] K. D. Sharma, A. Chatterjee, and A. Rakshit, "A hybrid approach for design of stable adaptive fuzzy controllers employing Lyapunov theory and particle swarm optimization," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 2, 2009, pp. 329–342.

[25] L. Wang, R. langari, "Complex system modeling via fuzy logic", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 26, n°1, 1996, pp. 100-106.

[26] B. Kosko, "Fuzzy systems as universal approximators", *IEEE Transactions on Computers*, vol.43, 1994, pp.1329-1333.

[27] T. Takagi, M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control", *IEEE Transactions on systems Man and cybernetics*, Vol. 15, n° 1, 1985, pp. 116-132.

[28] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. Part1", *IEEE Transactions on systems Man and Cybernetics*, Vol. 20, 1990, pp. 404-418.

[29] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. Part2", *IEEE Transactions on systems Man and Cybernetics*, Vol. 20, 1990, pp. 419-435.

[30] K. M. Passino, S. Yurkovich, "Fuzzy control", *Addison Wesley Longman, Inc*, 1998.

[31] E. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic systems", *Fuzzy Sets and Systems*, vol.26, 1977, pp.1182-1191.

[32] D. Corne, M. Dorigo and F. Glover, "New Ideas in Optimization", *McGraw-Hill*, 1999.

[33] F. Glover,"Tabu search - part I", *ORSA Journal on Computing*, 1(3), 1989, pp. 190–206.

[34] F. Glover, "Tabu search - part II", *ORSA Journal on Computing*, 2(1), 1990, pp. 4–32.

[35] F. Glover and M. Laguna, "Tabu search", *Kluwer Academic Publishers*, 1997.

[36] J.H. Holland, "Adaptation in Natural and Artificial Sytems", *2nd edition*, *MIT Press*,1992.

[37] D.E. Goldberg, "Genetic Algorithms in Search", *Optimization and Machine Learning*, *Addison-Wesley*, 1989.

[38] Z. Michalewicz,"Genetic Algorithms + Data Structures = Evolution Programs", *Springer-Verlag*, 1992.

[39] C.L. Bridges and D.E Goldberg, "An analysis of multipoint crossover", *In Proceedings of the Foundation Of Genetic Algorithms, FOGA*, 1991.

[40] J. Kennedy, R. C. Eberhart&Y. Shi, "Swarm intelligence", *New York: Morgan Kaufmann*, 2001.

[41] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '98)*, *IEEE Computer Society*, 1998, pp. 69–73.

[42] M. Clerc, and J. Kennedy, "The Particle Swarm : Explosion, Stability, and Convergence in a Multi-Dimensional Complex Space", *In Proceedings of the IEEE Transactions on Evolutionary Computation,* vol. 6, 2002, pages 58–73.

[43] H.Y. Fan, Y. Shi , "Study on Vmax of particle swarm optimization" , *Proceedings of the 2001 Workshop on Particle Swarm Optimization, University Indianapolis Press,* 2001.

[44] T.J. Procyk and E.H. Mamdani, "A Linguistic Self-Organising Process Controller", *Automatica*, Vol. 15, pp. 15-30, 1979.

[45] P.J. Mac Vicar-Whelan, "Fuzzy Sets for Man Machine Interaction", *Int. Journal of Man-Machine Studies*, No. 8, 1977, pp. 687-697.

[46] R. Palm, "Sliding Mode Fuzzy Control", *Proc. of the IEEE Conf. on Fuzzy Systems (Fuzz´IEEE 92)*, San Diego, USA, 1992, pp. 519-526.

[47] L.Foulloy, "Contrôle qualitative et contrôle flou: vers une méthode d'écriture des contrôleurs flous", *Actes des 12 ièmes journées internationales sur les systèmes expertset leurs applications",* Avignon, France, 1992.

[48] S. Galichet, M. Dussud and L. Foulloy, "Contrôleurs flous: Equivalences et études comparatives", *Actes des rencontres francophones sur la logique floue et ses applications (LFA'92)*, Nimes, France, 1992, pp. 229-236.

[49] H. Bersini, V. Gorrini, "FUNNY (Fuzzy or Neural Net) Methods for Adaptive Process Control", *in Proc. of the 1st European Congress on Fuzzy Intelligent Technologies EUFIT'93*, Aachen, Germany, 1993, pp. 55-61.

[50] F. Herrera, M. Lozano and J.L. Verdegay,“ A learning process for fuzzy control rules using genetic algorithms,” *Fuzzy Sets and Systems*, 100(1–3), 1998, pp.143–158.

[51] C.H. Chang, Y.C. Wu, “Genetic algorithm based tuning method for symmetric membership functions of fuzzy logic control system”, *Proc. IEEE/IAS Int. Conf. on Industrial Automation and Control Emerging Technologies*, 1995, pp. 421–428.

[52] A. Homaifar, E. McCormick, “Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms”, *IEEE Trans. Fuzzy Systems,*3 (2), 1995, pp. 129–139.

[53] K. Belarbi and F. Titel, “‘Genetic algorithm for the design of a class of fuzzy controllers : an alternative approach’, *IEEE Transaction on Fuzzy systems*, vol. 8 , N° 4, 2000, pp. 398-405.

[54] F. Hoffmann, “Evolutionary algorithms for fuzzy control system design,” *Proceedings of the IEEE*, 89(9), 2001, pp.1318–1333.

[55] F.H. Leug, L.K. Wong and P.K. Tam, “Fuzzy model controller for inverted pendulum”, *IEEE Trans. Computers,* vol. 40, no. 12, 1991, pp. 1320-1336.

[56] N. Talbi and K. Belarbi, “Automatic Generation of fuzzy rule base by a hybrid Approach: Application to Control and modeling”, *International Review of Automatic Control (I.RE.A.CO.)*, Vol. 5, N. 2, March 2012, pp. 284-291.

[57] N. Talbi and K. Belarbi, “A Self Organized Fuzzy PD Controller using Tabu Search”,*in Proc. IEEE*, *International Symposium onInnovations in Intelligent Systems and Applications (INISTA)*, Turkie, 2011, pp. 460-464.

[58] N. Talbi and K. Belarbi,, “Evolving Fuzzy Inference System by Tabu Search Algorithm and its application to control”, *in Proc. IEEE, International Conference on Multimedia Computing andSystems (ICMCS)*, Moroc, 2011, pp. 1-6.

[59] J. Tanomaru, S. Omatu, “process control by on-line trained Neural controllers”, *IEEE trans., Industrial Electronics*, vol. 39, N° 6, 1992, pp. 511-521.

[60] C.F. Juang, “A TSK-Type Recurrent Fuzzy Network for Dynamic Systems Processing by Neural Network and Genetic Algorithms”, *IEEE Transactions on Fuzzy Systems,* Vol. 10, No. 2, April 2002, pp. 155-170.

[61] Cheng-Jian Lin, ‘‘A GA-based neural fuzzy system for temperature control’’, *Fuzzy Sets and Systems,* Vol. 143, 2004, pp. 311-333.

[62] C. F. Juang, P. H. Chang, “Designing fuzzy-rule-based systems using continuous Ant colony optimization”, *IEEE trans., Fuzzy Syst.*, vol. 18, no. 1, 2010, pp. 138-149.

[63] C. F. Juang, C. M. Lu, C. Lo, and C. Y. Wang, “Ant colony optimization algorithm for fuzzy controller design and its FPGA implementation,”*IEEE Trans. Ind. Electron.*, vol. 55, no. 3, 2008, pp. 1453–1462.

[64] C. F. Juang, “A hybrid of genetic algorithm and particle swarm optimization for recurrent network design,” *IEEE Trans. Syst., Man, Cybern. B,Cybern.*, vol. 34, no. 2, 2004, pp. 997–1006.

[65] X. Chen and Y. Li, "A modified PSO structure resulting in high exploration ability with convergence guaranteed," *IEEE Trans. Syst., Man, Cybern.*, vol. 37, no. 5, 2007, pp. 1271–1289.

[66] C. F. Juang, P. H. Chang, "Designing fuzzy-rule-based systems using continuous Ant colony optimization", *IEEE trans., Fuzzy Syst.*, vol. 18, no. 1, 2010, pp. 138-149.

[67] CE150 Helicopter Model User's Manual, *Humusoft.*

[68] E.A. Wan. and A.A. Bogdanov, "Model predictive neural control with applications to a 6 DOF helicopter model," *in Proc. 2001 American Control Conference*, Arlington, Virginia, 2001, pp. 488-493.

[69] H. Boubertakh, M. Tadjine, "Tuning Fuzzy PD and PI controllers using reinforcement learning", *ISA trans.*,(49), 2010, pp. 543-551.

[70] J.T. Spooner and K.M. Passino, "Adaptive control of a class of decentralized nonlinear systems",*IEEE trans. Automatic and control*, Vol. 41, 1996, pp280-284.

[71] J.T. Spooner and K.M. Passino, "Decentralized adaptive Control of Nonlinear System Using Radial basic Neural Networks", *IEEE transactions on automatic control*, vol. 44, no. 11, 1999, pp 2050-2057.

[72] C.F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Trans. Syst., Man, Cybern. B, Cybern.,* vol. 34, no. 2, Apr. 2004, pp. 997–1006.

[73] Z.L. Gaing, "A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System", *IEEE Trans. On Energy Conversion*, vol. 19, no. 2, june 2004, pp. 384-391

[74] A. Chatterjee, K. Pulasinghe, K. Watanabe, and K. Izumi, "A particle-swarm-optimized fuzzy-neural network for voice-controlled robot systems," *IEEE Trans. Ind. Electron.*, vol. 52, no. 6, Dec.2005, pp. 1478–1489.

[75] N. Talbi and K. Belarbi, "Fuzzy Takagi Sugeno System Optimization using Hybrid Particle Swarm Optimization and Tabu Search Learning Algorithm", *International Journal of Tomography and Simulation,Ceser Publications*, vol.22, n°1, 2013, pp. 4-16.

[76] N. Talbi and K. Belarbi, "Designing Fuzzy Controllers for a Class of MIMO Systems using Hybrid Particle Swarm Optimization and Tabu Search", *International Journal of Hybrid Intelligent Systems, IOS Press,* vol. 10, n°1,2013,pp. 1–9.

[77] J. Buckley, "Universal Fuzzy Controllers", *Automatica*, Vol. 28, No. 6, 1992, pp. 1245-1248.

[78] J. Buckley, "Sugeno type Controllers are Universal Controllers", *Fuzzy sets and Systems 53,* pp. 299-303, 1993.

[79] X.J. Zengand M.G. Singh, "Fuzzy Bounded Least Squares Method for the Identification of Fuzzy Systems", *Proc. of the IEEE Conf. on Fuzzy Systems (Fuzz'IEEE 97)*, pp. 403-408, Bercelone, Espagne, 1997.

[80] Y. Yoshinari, W. Pedryczand K.Hirota, "Construction of Fuzzy Models Through Clustering Techniques", *Fuzzy Sets and Systems*, Vol. 54,1993, pp. 157-165.

[81] E. Kim, P.Park, S.Ji and M.Park, "A New Approch to Fuzzy Modeling", *IEEE Trans. on Fuzzy Systems*, Vol. 5, No. 3, August 1997, pp. 328-337.

[82] P.Y. Glorennec, "Algorithme d'apprentissage pour systèmes d'inférence floue", *Edition Hermès*, Paris, 1999.

[83] A. Bagis, "Fuzzy rule base design using tabu search algorithm for nonlinear system modeling", *ISA Transactions,* vol 47, 2008, pp. 32–44.

[84] K.S. Narendra, K. Parthasarathy, "Identification and control of dynamical systems using neural networks", *IEEE Trans Neural Network,*1990, 1(1), pp. 4–27.

[85] G. Box, GM. Jenkins, "Time series analysis, forecasting, and control", *San Francisco (CA): Holden Day*; 1970.

[86] RM. Tong, "Synthesis of fuzzy models for industrial processes-some recent results", *Int J General Syst*,1978;(4):143–62.

[87] M. Sugeno, K. Tanaka,"Successive identification of a fuzzy model and itsapplication to prediction of a complex system", *Fuzzy Sets and Systems*,1991;(42):315–34.

[88] S. Kang, C. Woo, H. Hwang, KB.Woo, "Evolutionary design of fuzzy rulebase for nonlinear system modeling and control", *IEEE Trans Fuzzy Syst*, 2000;8(1):37–45.

[89] A. Bagis, "Fuzzy rule base design using tabu search algorithm for nonlinear system modeling", *ISA Transactions,* vol 47, 2008, pp. 32–44.