N° Ordre……../Faculté/UMBB/2014

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE M'HAMED BOUGARA-BOUMERDES

Faculté des Hydrocarbures et de la Chimie

Mémoire de Magister

Présenté par

BOUDOUAOUI Yassine

Filière : Génie Electrique - Automatique
Option : Automatique et Informatique Industrielle

Thème

Contribution à la synthèse automatique des systèmes flous par approches méta-heuristiques

Devant le jury :

BOUDJEMA FARES	Professeur	ENP, Alger	Président
KIDOUCHE MADJID	Professeur	Université de Boumerdès	Examinateur
CHAIB AHMED	MC/A	Université de Boumerdès	Examinateur
HABBI HACENE	MC/A	Université de Boumerdès	Encadreur

Année Universitaire : 2014/2015

ملخص

الكلمات الدلالية: النمذجة, المنطق الغامض, مستعمرة النحل الاصطناعية.

نهدف من خلال هذه الأطروحة الاسهام في مجال النمذجة باستعمال المنطق الغامض بتصميم منهجية جديدة للحصول تلقائيا على نماذج من صنف المنطق الغامض بصفة مباشرة من البيانات الرقمية للنظام قيد الدراسة. أهمية هذه المنهجية تتمثل في الاستغلال المباشر للبيانات الرقمية دون الحاجة للالمام بقواعد الفيزياء أو التعمق في دراسة النظام مما يسهل عملية الحصول على النموذج التي غالبا ما تكون صعبة.

تم التأكد من نجاعة المنهجية المقترحة على عدة حالات نمذجة و تحكم. أنظمة Benchmark المدروسة مأخوذة من بحوث جد حديثة حتى تكون المقارنة موضوعية. النتائج المتحصل عليها لجميع الحالات تعتبر جد مرضية, و تمكنا من الحصول على نتائج أحسن من تلك الموجودة في المراجع السابقة.

قمنا أيضا بتطبيق المنهجية المقترحة على نظام جد معقد و الحصول على نموذجه يعتبر جد صعب. و تمكنا من الحصول على نتائج ممتازة من وجهة نظر عملية و اختصاصية. النتائج المتحصل عليها يمكن أن تتخذ كقاعدة لأبحاث كثيرة و في مجالات متعددة مستقبلا.

على العموم بالامكان أن نقول أن الأهداف المسطرة و المستهدفة من خلال هذه الأطروحة قد تم تحقيقها.

Résumé

Mot clés : clustering floue, métaheuristique, colonies d'abeilles artificielles.

La thématique abordée dans ce travail de mémoire concerne le problème de conception de systèmes à base de règles floues. Nous nous sommes fixés l'objectif d'apporter une contribution dans ce domaine de recherche particulier à travers l'élaboration d'une nouvelle méthodologie d'extraction automatique de modèle de représentation flous à partir de données numériques en se basant exclusivement sur le modèle d'optimisation par colonies d'abeilles artificielles. L'intérêt de l'approche d'identification floue proposée réside dans le fait qu'elle exploite directement la base de données numériques disponibles sur le système à identifier ou à commander pour caractériser son comportement dynamique. Son application est très simple et systématique ne nécessitant ni une modélisation physique, ni une connaissance profonde sur le processus à gouverner. Les différentes structures algorithmiques développées dans ce mémoire emploient le concept d'optimisation par colonies d'abeilles artificielles qui modélise le comportement des essaims d'abeilles lors du fourragement. Ainsi, le problème de conception de systèmes flous est traduit en un problème d'optimisation méta-heuristique.

La procédure de synthèse est validée sur plusieurs problèmes d'identification et de commande. Les problèmes benchmark considérés sont repris de travaux de recherche très récents. Une application à un échangeur de chaleur pilote exploitant des données expérimentales prélevées sur l'installation réelle est également envisagée. Les résultats obtenus pour l'ensemble des cas envisagés sont très satisfaisants, démontrant une nette amélioration des résultats déjà existants.

Abstract

Keywords: Fuzzy modeling, Data-based model, Artificial bee colony optimization,Swarm optimization.

In this work, artificial bee colony (ABC) optimization based methodology is proposed for automatically extracting Takagi-Sugeno (TS) fuzzy models with enhanced performance from data. The design procedure aims to find the structures and the parameters of the TS fuzzy models simultaneously without knowing the rule number as a priori. In the proposed method, a fuzzy model is encoded into a food source with appropriate string representation so that the TS model is entirely specified. The encoded premise and consequent parameters of the fuzzy model evolve together through artificial bee colony optimization strategy simulating the global foraging behavior of honey bee swarm so that good solutions can be achieved. Simulations on benchmark modeling and tracking control problems are performed and compared with other existing methods. The experimental results indicate that the proposed ABC optimization based fuzzy systems design algorithm can successfully find accurate fuzzy models with appropriate number of rules. Moreover, the proposed approach outperforms the compared methods and can provide considerable improvements in tackling complex modeling and control problems.

Sommaire

ملخص

Résumé

Abstract

Liste des figures 5

Liste des tableaux 16

Introduction générale 8

Chapitre I : Généralités sur les systèmes à base de règles floues 10

I.1 Introduction 10
I.2 Concepts de base d'un système flou 11
I.3 Principe d'un système à inférence floue 12
I.3.1 Fuzzification 13
I.3.2 Inférence des règles 13
I.3.3 Défuzzification 13
I.4 Classification des modèles de systèmes flous 13
I.4.1 Modèle flou linguistique 13
I.4.2 Modèle relationnel flou 14
I.4.3 Modèle flou de Takagi-Sugeno (TS) 16
I.5 Identification des systèmes flous 17
I.6 Méthodes d'identification 18
I.6.1 A partir de connaissances issues de l'expertise 18
I. 6.2 A partir de données 18
I.6.3 Par linéarisation dynamique 18
I.7 Identification des modèles TS à partir de données 20
I.7.1 Identification par classification floue 20
I.7.2 Identification par méthodes d'optimisation 21
I.8 Algorithmes de classification floue 22
I.8.1 L'algorithme Fuzzy C-means 22
I.8.2 L'algorithme de Gustafson-Kessel 23

I.8.3 Estimation des paramètres des conséquences 25
I.9 Conclusion 26
Chapitre II ; Les méta-heuristiques pour la synthèse des systèmes flous 27
II.1 Introduction 27
II.2 Problématique générale d'optimisation 28
II.3 Identification des systèmes flous par méthodes d'optimisation 29
II.3.1 Formulation du problème d'optimisation 29
II.3.2 Recherche de modèles optimaux 29
II.3.3 Mécanismes de codage 31
II.4 Méthodes d'optimisation méta-heuristiques 32
II.4.1 Optimisation par essaim particulaire 33
II.4.2 Les algorithmes génétiques 35
II.4.3 Algorithme d'optimisation à évolution différentielle 37
II.4.4 Algorithme de colonies de fourmis artificielles 39
II.5 Conclusion 41
Chapitre III Auto-génération des systèmes flous par colonies d'abeilles artificielles 42
III.1 Introduction 42
III.2 L'essaim d'abeilles 42
III.2.1 Composantes fondamentales de l'essaim 43
III.2.2 Modèle comportemental à essaim d'abeilles 44
III.3 L'algorithme de colonies d'abeilles artificielles 45
III.3.1 Structure algorithmique 46
III.3.2 Modèle de base de l'algorithme ABC 47
III.4 Algorithme ABC guidé par la meilleure solution globale 51
III.5 Une nouvelle méthodologie de conception des systèmes flous à base de l'algorithme ABC 51
III.5.1 Codage du modèle flou TS 52
III.5.2 Structure algorithmique de la méthodologie proposée 54
III.6 Hybridation de la méthodologie proposée avec la méthode des moindres carrés simple 54
III.7 Conclusion 58
Chapitre IV : Application à la conception de systèmes à base de règles floues 59
IV.1 Introduction 59

IV.2 Identification du système de four à gaz « Box–Jenkins » 59
IV.3 Identification d'un système non-linéaire non forcé 63
IV.4 Identification d'un système non linéaire statique 66
IV.5 Commande en poursuite d'un système non-linéaire 68
IV.6 Analyse de la robustesse de l'approche de modélisation floue proposée 71
IV.6.1 Par rapport à la taille de la population 71
IV.6.2 Par rapport au paramètre de contrôle C de l'algorithme g-ABC 74
IV.7 Application à un échangeur de chaleur 77
IV.7.1 Description du processus 77
IV.7.2 Identification floue de l'échangeur de chaleur 78
IV.7.3 Résultats d'identification 78
IV.8 Conclusion 85
Conclusions et perspectives 86
Bibliographie 88

Liste des figures

Figure I. 1: Système à inférence floue 12

Figure I. 2 : Exemple de variable linguistique dans un modèle flou de Mamdani 14

Figure I. 3 : Schéma descriptif d'un modèle relationnel flou 15

Figure I. 4 : Approximation linéaire d'une fonction y = f(x) par un modèle flou TS à trois règles 16

Figure I. 5 : Illustration du principe général du clustering flou 21

Figure II. 1: Exemples de schémas de codage pour la génération des systèmes flous 32

Figure II. 2 : Stratégie de déplacement dans un essaim particulaire 33

Figure II. 3 : Représentation d'un croisement en un point de deux chaînes 36

Figure II. 4 : Représentation d'une mutation de bits dans une chaîne 36

Figure III. 1 : Illustration d'une abeille en « Waggle dance » 44

Figure III. 2 : Le comportement d'une abeille lors du fourragement 45

Figure III. 3 : Organigramme du modèle de base de l'aglorithme ABC 50

Figure III. 4 : Codage d'un modèle flou TS dans la méthodologie proposée 52

Figure IV. 1 : Performance de l'approche proposée appliquée au système Box-Jenkins 63

Figure IV. 2 : Performance de l'approche proposée appliquée au système non linéaire non forcé 65

Figure IV. 3 : L'ensemble de données d'apprentissage et de test utilisé pour le système non linéaire statique 66

Figure IV. 4 : Performance de l'approche proposée appliquée au système non linéaire statique 67

Figure IV. 5 : Performance de l'approche proposée appliquée au problème de commande en poursuite d'un système non linéaire 69

Figure IV. 6 : Evolution des erreurs de poursuite moyennes (RMSE) évaluées pour les approches basées ABC et g-ABC 71

Figure IV. 7 : L'évolution de la moyenne des MSE pour le système Box-Jenkins obtenue par l'algorithme ABC pour différentes tailles de la population 72

Figure IV. 8 : L'évolution de la moyenne des MSE pour le système non linéaire non forcé obtenue par l'algorithme ABC pour différentes tailles de la population 72

Figure IV. 9 : L'évolution de la moyenne des MSE pour le système non linéaire statique obtenue par l'algorithme ABC pour différentes tailles de la population 72

Figure IV. 10 : L'installation d'échange thermique 77

Figure IV. 11 : La sortie réelle et la sortie du modèle de prédiction de T34 80

Figure IV. 12 : L'erreur d'approximation de la température d'eau T34 80

Figure IV. 13 : La sortie réelle et la sortie du modèle de prédiction de T16..... 80

Figure IV. 14 : L'erreur d'approximation de la température d'air T16 81

Figure IV. 15 : Les partitions floues associées aux entrées T34 et T16 du modèle de prédiction de la température T34 81

Figure IV. 16 : La sortie réelle et la sortie du modèle de prédiction de la température T34... 82

Figure IV. 17 : La sortie réelle et la sortie du modèle de prédiction de la température T16... 82

Figure IV. 18 : Résultats de prédiction en présence d'une fuite de 15% 83

Figure IV. 19 : Résultats de prédiction en présence d'une fuite de 25% 83

Figure IV. 20 : Résultats de prédiction en présence d'une fuite de 30% 84

Figure IV. 21 : Résultats de prédiction en présence d'une fuite de 40% 84

Liste des tableaux

Tableau I. 1 Algorithme des C-moyennes floues (FCM) 23

Tableau I. 2 L'algorithme de classification floue de Gustafson-Kessel 24

Tableau II. 1 Algorithme d'optimisation par Essaim Particulaire (PSO) 34

Tableau II. 2 Principe d'un algorithme génétique 36

Tableau II. 3 Algorithme d'optimisation à évolution différentielle 39

Tableau II. 4 Algorithme de colonie de fourmis artificielles (ACO) 40

Tableau III. 1 Structure de l'algorithme ABC 46

Tableau III. 2 L'algorithme de colonies d'abeilles artificielles (ABC) 49

Tableau III. 3 Structure algorithmique de la méthodologie de modélisation floue proposée .. 55

Tableau III. 4 Structure algorithmique de l'hybridation de la méthodologie proposée avec la méthode des moindres carrées simple 57

Tableau IV. 1 Résultats d'application de l'approche proposée au système Box-Jenkins 61

Tableau IV. 2 Résultats d'application de l'approche proposée au système non linéaire non forcé 64

Tableau IV. 3 Résultats d'application de l'approche proposée au système non linéaire statique 68

Tableau IV. 4 Résultats d'application de l'approche proposée au problème de commande en poursuite en considérant une population de taille 20 70

Tableau IV. 5 Résultats d'application de l'approche proposée au problème de commande en poursuite (pour les algorithmes comparés la population est de taille 100) 70

Tableau IV. 6 Résultats d'identification floue du système Box-Jenkins obtenus par la méthodologie proposée basée ABC pour différentes tailles de population 73

Tableau IV. 7 Résultats d'identification floue du système non linéaire non forcé obtenus par la méthodologie proposée basée ABC pour différentes tailles de population 73

Tableau IV. 8 Résultats d'identification floue du système statique non linéaire obtenus par la méthodologie proposée basée ABC-MCS pour différentes tailles de population 74

Tableau IV. 9 Résultats d'identification floue du système Box-Jenkins obtenus par la méthodologie proposée basée g-ABC et g-ABC-MCS pour différentes tailles de population et différentes valeurs du paramètre C 75

Tableau IV. 10 Résultats d'identification floue du système non linéaire non forcé obtenus par la méthodologie proposée basée g-ABC pour différentes tailles de population et différentes valeurs du paramètre C 75

Tableau IV. 11 Résultants d'identification floue du système non-linéaire statique obtenus par la méthodologie proposée basée g-ABC pour différentes tailles de population et différentes valeurs de C 75

Tableau IV. 12 Résultats du contrôle flou en poursuite obtenus par la méthodologie proposée basée g-ABC pour différentes valeurs du paramètre C 76

Tableau IV. 13 Résultats d'identification floue de l'échangeur de chaleur 79

Tableau IV. 14 Résultats de validation des modèles flous identifiés pour l'échangeur de chaleur 82

Tableau IV. 15 Evaluation des performances du modèle flou identifié en présence de variations paramétriques et structurelles 85

Introduction générale

L'évolution des systèmes de production industriels est actuellement synonyme de l'intégration de procédés très sophistiqués, comprenant des équipements très sensibles et des systèmes à dynamiques complexes. Ces systèmes sont en majorité non linéaires, multivariables et de grande dimension, présentant des couplages internes et souvent installés selon des configurations multiples. Certains procédés sont caractérisés par des modèles dynamiques qui sont difficiles à établir, nécessitant, pour certaines classes de systèmes, l'emploi d'outils mathématiques complexes pour leur développement. La complexité d'un système peut se traduire par plusieurs facteurs tels que le nombre d'éléments qui le composent, les relations entre ces éléments, leurs caractéristiques dynamiques, etc. Le contrôle de ces systèmes exige la disponibilité d'un modèle mathématique qui doit refléter le mieux possible le comportement du système, du moins pour certaines techniques de commande. Le développement de modèles de processus de forme mathématiquement exploitable est justement la problématique de toute approche d'identification ou de modélisation.

Généralement, pour le développement de modèles simplifiés, une hypothèse communément faite est la linéarité du système. Cependant, l'hypothèse de linéarité n'est vérifiée que dans une plage de fonctionnement restreinte autour d'un point d'équilibre donné . Alors, les performances du modèle se dégradent dès qu'on s'en éloigne et la recherche d'un modèle plus adapté et notamment non linéaire devient nécessaire. Pour surmonter cette situation, une alternative intéressante basée sur les multi-modèles a été apparu ces dernières années. Une multitude d'approches basées sur ce concept ont été rapidement développées et appliquées à des contextes variés de commande de processus industriels.

Les modèles de type Takagi-Sugeno (TS) manipulés dans le cadre de la logique floue sont assimilables à des multi-modèles avec une capacité d'approximation théoriquement prouvée. Une telle caractérisation leur a confié une importance majeure constatée à travers le nombre de travaux et d'applications rapportés dans la littérature.

L'identification des modèles flous continue à susciter l'intérêt de plusieurs chercheurs. Plusieurs approches ont été développées dans ce sens. Notre travail aborde cette thématique d'un point de vue optimisation, à travers la considération d'une certaine classe de méthodes d'optimisation dites « méta-heuristique ». Plus précisément, nous cherchions à apporter une contribution à la conception des systèmes flous par l'utilisation d'approches d'optimisation méta-heuristiques, en visant en particulier à développer une nouvelle méthodologie d'auto-génération des bases de règles floues en se basant sur l'algorithme de colonies d'abeilles artificielles (ABC).

Le présent travail de recherche revient justement sur les différents aspects et les différentes étapes de développement de cette méthodologie en passant tout d'abord par la position du problème d'identification floue à résoudre, suivi par la présentation de la structure algorithmique de l'approche d'identification floue proposée sous différentes versions (simple et hybride). Enfin, pour démontrer son utilité et ses performances, l'approche de synthèse de systèmes flous proposée est validée sur des problèmes d'identification et de commande de systèmes complexes.

Le mémoire est composé de quatre chapitres organisés comme suit.

Chapitre I : Ce chapitre présente les notions et les concepts de base manipulés dans la théorie des ensembles flous et de la logique floue. On y décrit également les systèmes d'inférence floue, leurs différents modèles de représentation ainsi que les principales méthodes utilisées pour leur obtention.

Chapitre II : Le chapitre II est dédié aux méthodes d'optimisation méta-heuristiques. Nous y présentons les différentes méthodes d'optimisation qui se fondent sur des essaims et sur la population, ainsi que leur utilisation pour la modélisation ou l'identification floue.

Chapitre III : Le chapitre III est intégralement consacré à la méthodologie d'identification floue proposée. Il s'agit plus précisément d'une approche d'auto-génération de bases de règles floues à partir de données en utilisant l'algorithme de colonies d'abeilles artificielles. Nous y présentons la structure algorithmique de cette approche sous différentes versions.

Chapitre IV : L'analyse de la performance et la validation de l'approche proposée constituent l'objet de ce chapitre. Pour ce faire, des problèmes d'identification et de commande y sont traités. Les résultats obtenus sont comparés à d'autres approches de conception de systèmes flous rapportées dans la littérature. Un autre cas étudié concerne la modélisation d'un échangeur de chaleur pilote par l'exploitation d'une base de données expérimentales prélevées sur une installation d'échange thermique réelle.

Chapitre I

Généralités sur les systèmes à base de règles floues

I.1 Introduction

Dans le contexte général de la manipulation de la notion de "système", il est fondamental de constater que bon nombre de disciplines scientifiques partagent ou utilisent cette notion pour désigner une entité ou un objet de nature quelconque au sein duquel se manifestent des phénomènes qui lui sont bien propres, faisant agir ou interagir les grandeurs ou les variables de ce système. Cette définition plutôt grossière se trouve bien adaptée pour la désignation de processus de natures diverses telles que les processus physiques, technologiques, biologiques ou encore mathématiques. La manipulation d'un système donné nécessite souvent la mise au point d'une forme de représentation qui lui est suffisamment caractéristique, ou du moins qui permet de caractériser une ou plusieurs composantes de ce système.

Par forme de représentation, on fait souvent allusion au concept de « modèle » de type statique ou dynamique, mathématique ou graphique, conceptuel ou à base de règles. Ainsi, un système à base de règles est un système décrit par un modèle basé sur des règles qui, comme tout autre type de modèles de représentation, peut être utilisé pour permettre l'appréhension, l'analyse et le traitement des phénomènes qui régissent le système étudié. Les systèmes flous n'échappent pas à cette règle générale, car ils sont tout simplement des systèmes décrits par des modèles à base de règles floues que l'on peut éventuellement exploiter pour résoudre différents problèmes d'identification et de commande de processus, de surveillance et de supervision, de prise de décision et bien d'autres. La synthèse de systèmes flous repose sur la manipulation d'outils et de concepts propres à la théorie des ensembles flous et de la logique floue. Ce chapitre passe en revue certains de ces concepts et revient sur les principales méthodes utilisées pour leur obtention.

I.2 Concepts de base d'un système flou

Les différentes notions et outils employés pour la synthèse, le traitement et la manipulation des systèmes à base de règles floues s'appuient sur le concept du raisonnement approché [1,2] défini dans la théorie de la logique floue. Ainsi, la modélisation ou l'identification dite "floue" est une pure démarche de caractérisation de la dynamique de processus qui vise à construire ou à développer des modèles de représentation autorisant la manipulation d'incertitudes, d'imprécisions et même d'incomplétudes. On introduit brièvement dans la suite certaines définitions et notions manipulées en logique floue.

Ensemble flou : Un ensemble flou est défini par une fonction d'appartenance qui peut prendre toutes les valeurs réelles comprises entre 0 et 1.

Terme linguistique : Terme associé à une fonction d'appartenance caractérisant une variable linguistique.

Variable linguistique : Variable numérique appliquée en entrée, pour fuzzification, ou en sortie, après défuzzification, d'un système à base de règles floues.

Degré d'appartenance : Un élément x appartient à un ensemble flou A avec un degré d'appartenance compris entre 0 et 1, donné par la fonction d'appartenance $\mu_A(x)$.

Fonction d'appartenance : Fonction $\mu_A(x)$ qui à toute valeur d'entrée x fait correspondre son degré d'appartenance à l'ensemble A. Cette valeur graduelle est comprise entre 0 et 1. Cette fonction peut être de forme triangulaire, gaussienne, trapézoïdale ou singleton.

Base de connaissances : Ensemble des fonctions d'appartenance et des règles d'un système flou contenant l'expertise, la connaissance de l'opérateur, de l'expert, etc.

Singleton : Fonction d'appartenance $\mu_A(x)$ « en bâton », c'est-à-dire nulle pour tout x, sauf en un point singulier x_0.

Prémisse : Appelée encore **condition**, la condition d'une règle est une proposition associant une variable linguistique et un terme linguistique écrite entre le IF et le THEN de la règle. Une prémisse peut être formée par la combinaison de plusieurs propositions floues.

Conclusion : Une conclusion de règle est une proposition associant une variable linguistique et un terme linguistique écrite après le THEN de la règle. Une conclusion peut être formée par la combinaison de plusieurs propositions floues.

Degré de vérité : Le degré de vérité, ou encore le **degré d'activation**, d'une règle prend une valeur comprise entre 0 et 1 déduite des degrés d'appartenance des prémisses de la règle. Il influe directement sur la valeur des conclusions de cette même règle.

Fuzzification : Transformation d'une valeur numérique en degré d'appartenance flou par évaluation d'une fonction d'appartenance.

Inférence : Cycle de calcul des degrés d'activation de toutes les règles de la base ainsi que de tous les ensembles flous des variables linguistiques se trouvant dans les conclusions de ces règles.

Défuzzification : Transformation, après inférence, d'un ensemble flou d'une variable linguistique de sortie en valeur numérique.

I.3 Principe d'un système à inférence floue

En liaison avec les concepts de base présentés ci-dessus, il est question dans ce paragraphe de montrer comment sont-elles manipulées ces notions dans le contexte d'un système à base de règles floues. Cette manipulation se fait généralement en trois étapes à travers trois modules distincts, comme le montre la Figure I.1 :

- Le module de fuzzification
- Le module d'inférence des règles
- Le module de défuzzification

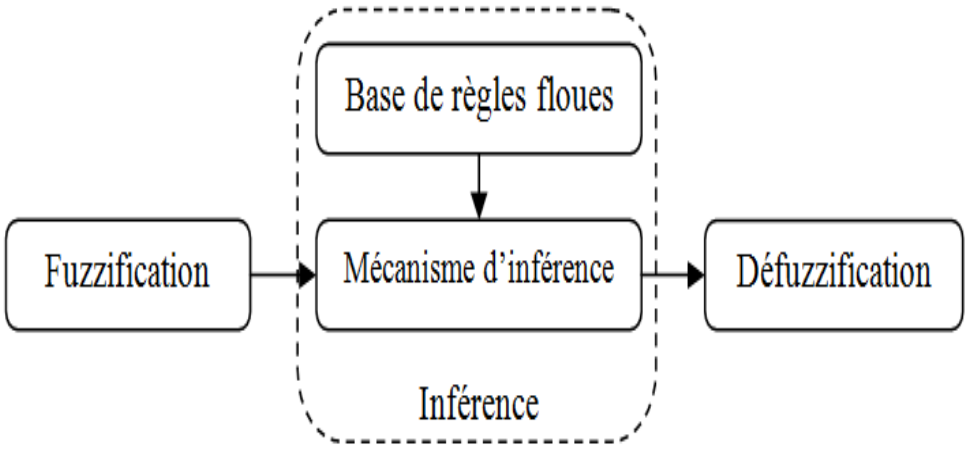


Figure I. 1: Système à inférence floue.

I.3.1 Fuzzification

La fuzzification consiste à définir des fonctions d'appartenance ou bien des partitions floues pour les différentes variables linguistiques. Ceci a pour but la conversion d'une grandeur physique en une grandeur floue.

I.3.2 Inférence des règles

L'inférence est formée de deux blocs :

- **La base de règles** : composée d'un ensemble de relations liant les variables linguistiques d'entrée aux variables linguistiques de sortie. Chaque relation désigne une règle floue avec une prémisse et une conclusion.
- **Le mécanisme d'inférence:** réalise le traitement numérique des règles d'inférence, décrites par des opérateurs flous, pour obtenir la sortie linguistique ou floue du système à base de règles. Cette opération est réalisée par différentes méthodes basées sur les opérateurs d'implications floues telles que la méthode Max-Min, la méthode Max-Dot, la méthode de Luckasiewicz, la méthode de Gödel [**1**].

I.3.3 Défuzzification

Il s'agit de la transformation d'une information floue en une information déterminée ou précise. Parmi les mécanismes de défuzzification, on cite à titre d'exemple la méthode du centre de gravité (COG), la méthode du maximum, la méthode de la moyenne des maximums (MOM), la méthode des surfaces et la méthode des hauteurs [**3**].

I.4 Classification des modèles de systèmes flous

En fonction de la forme de la conséquence de la règle conditionnelle IF-THEN, trois types de modèles flous peuvent être distingués : le modèle flou linguistique ou de Mamdani, le modèle flou relationnel et le modèle flou de Takagi-Sugeno (TS).

I.4.1 Modèle flou linguistique

Dans ce type de modèle, l'antécédent et la conséquence des règles sont des propositions floues simples ou composées. La forme générale de la règle IF-THEN s'exprime par :

$$\boldsymbol{R^i: IF x\ is\ A^i\ THEN\ y\ is\ B^i,} \qquad \boldsymbol{i = 1, \ldots m} \tag{I. 1}$$

où $x \in X \subset R^n$ est la variable antécédent représentant la variable d'entrée du système flou et $y \in Y \subset R^q$ est la variable conséquence représentant la sortie du système flou. Les termes A^i et B^i sont des termes linguistiques associés à la règle i et définis par les ensembles flous multidimensionnels $\mu_{A^i}(x): X \rightarrow [0\ 1]$ et $\mu_{B^i}(y): Y \rightarrow [0\ 1]$, respectivement.

Les variables de l'antécédent et de la conséquence sont appelées variables linguistiques parcequ'elles prennent des valeurs linguistiques. Comme il peut être constaté, les prémisses et les conclusions des règles sont formées de propositions linguistiques. Il est alors possible de les construire en se servant uniquement des connaissances issues de l'expertise. La Figure I.2 illustre un exemple de termes linguistiques associés à la variable linguistique "Température" dans un modèle flou de Mamdani. La qualité de ce modèle dépend des partitions floues, des opérateurs d'inférence et de la défuzzification.

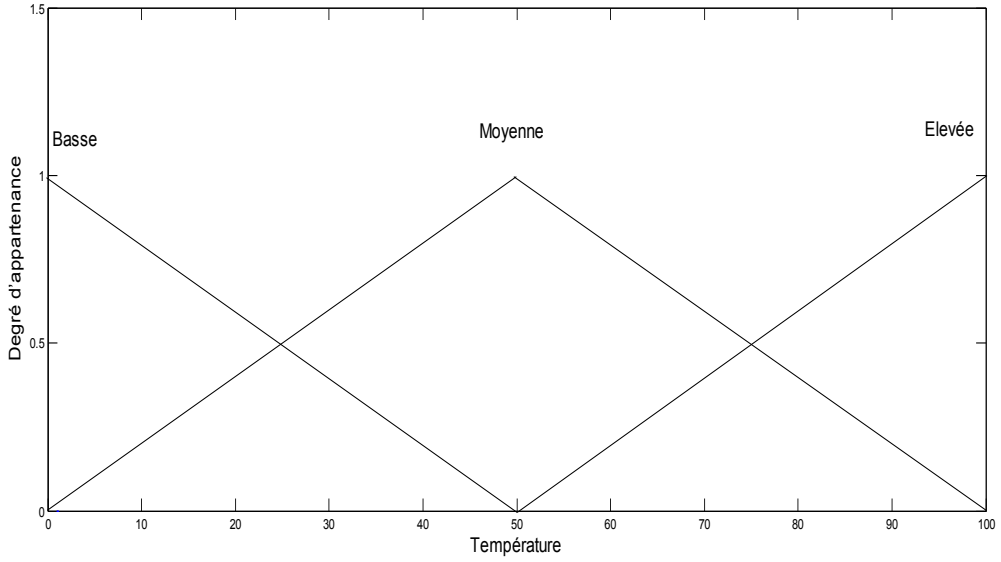


Figure I. 2 : Exemple de variable linguistique dans un modèle flou de Mamdani.

I.4.2 Modèle relationnel flou

Un modèle relationnel flou peut être considéré comme une généralisation du modèle linguistique, où un seul terme antécédent peut être associé à plusieurs termes de la conséquence par une relation floue.

Les éléments individuels de la relation représentent des poids d'association entre les ensembles flous. A titre d'exemple, soit un modèle statique avec une entrée $x \in X$ et une sortie $y \in Y$. Soit A une collection de M termes linguistiques (ensembles flous) définis sur le domaine X, et B une collection de N ensembles flous définis sur Y **:** $A=\{A_1,A_2,\ldots,A_M\}$ et $B=\{B_1,B_2,\ldots,B_N\}$.

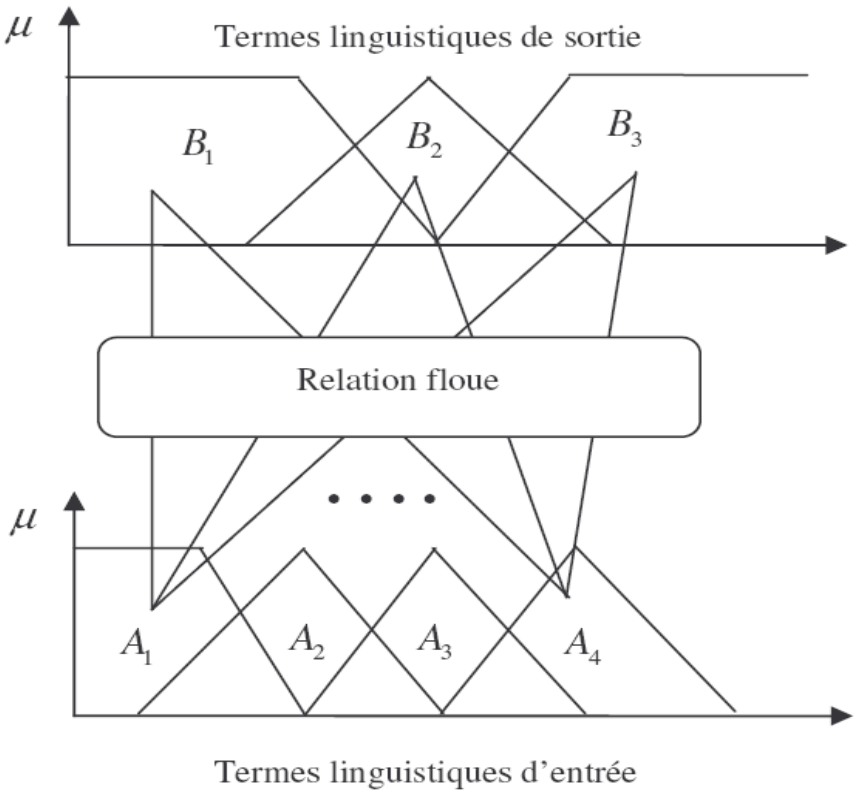


Figure I. 3 : Schéma descriptif d'un modèle relationnel flou.

Comme le montre la Figure I. 3, la relation floue $R = [r_{ij}]_{M\times N}$ définie une association $R: A \rightarrow B$ où chaque A_i est relié à chaque B_j avec un poids défini par l'élément r_{ij} de la relation. Par cette pondération nous pouvons ajuster les conséquences des règles sans changer les ensembles flous de référence (termes linguistiques). Donc, chaque conséquence peut prendre n'importe quelle valeur sur le domaine *Y*, contrairement aux modèles linguistiques qui possèdent des conséquences bien déterminées. Pour ce degré de liberté supplémentaire, nous aurons plus de paramètres libres, ce qui rajouterait d'autres contraintes pour leur identification [**1**].

I.4.3 Modèle flou de Takagi-Sugeno (TS)

Ce type de modèles flous a été introduit en 1985 par Takagi et Sugeno [**4**]. Dans ce modèle, les conséquences des règles sont des fonctions ordinaires des entrées du modèle :

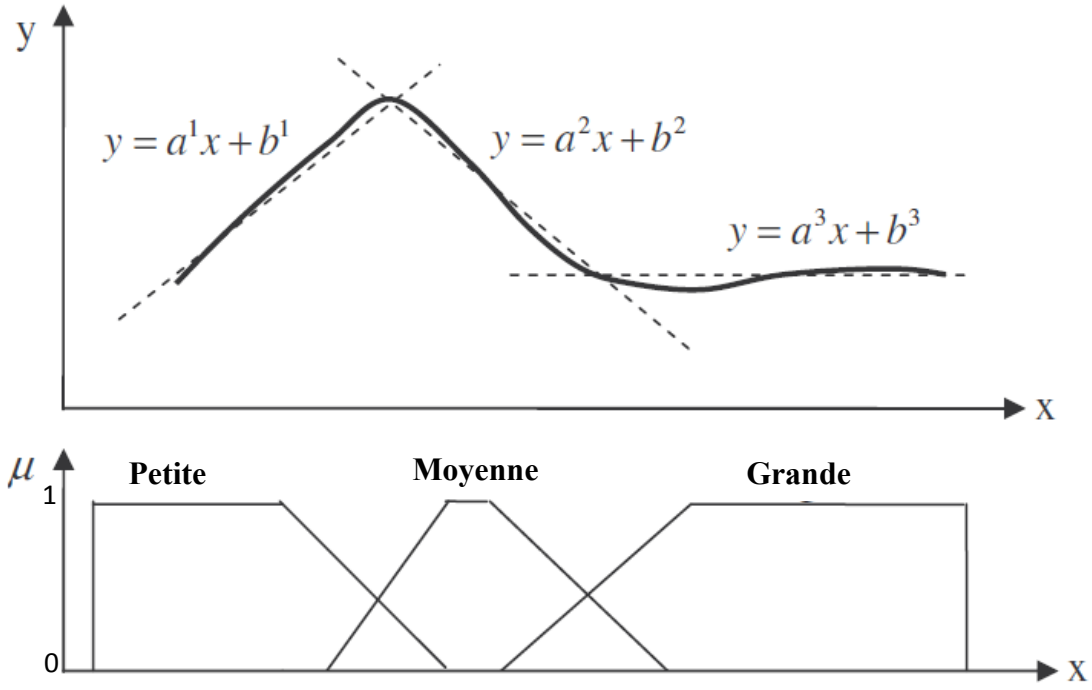


Figure I. 4 : Approximation linéaire d'une fonction y = f(x) par un modèle flou TS à trois règles.

$$\boldsymbol{R^i}: \boldsymbol{IF\ x\ is\ A^i\ THEN\ y^i = f^i(x)}, \qquad \boldsymbol{i = 1, \dots, m} \tag{I. 2}$$

où $x \in X$ est la variable d'entrée et $y \in Y$ est la variable de sortie. R^i désigne la ième règle et m le nombre des règles. A^i est l'ensemble flou antécédent de la ième règle défini par une fonction d'appartenance multidimensionnelle $\mu_{A^i}(x): X \rightarrow [0\ 1]$. Le modèle (I.2) peut être réécrit sous la forme :

$$\boldsymbol{R^i}: \boldsymbol{IF\ x_1 is\ A_1^i and\ x_2 is\ A_2^i and \dots and\ x_n is\ A_n^i\ THEN\ y^i = f^i(x)}, \quad \boldsymbol{i = 1, \dots, m} \tag{I. 3}$$

Les fonctions de conséquences f^i sont choisies typiquement de formes paramétriques et sont assimilées à des descripteurs locaux statiques ou dynamiques, le plus souvent de forme linéaire, c'est-à-dire :

$$\boldsymbol{y^i = a^{Ti}x + b^i} \tag{I. 4}$$

où a^i est le vecteur des paramètres et b^i un scalaire de compensation. La Figure I.4 représente un exemple de fonction scalaire représentée par un modèle TS à trois règles. L'antécédent de chaque règle défini une région de validité floue pour le modèle linéaire figurant dans la conséquence de la règle. Le modèle global est obtenu en concaténant l'ensemble des modèles locaux à l'aide d'un mécanisme d'inférence approprié. A titre d'illustration, la sortie résultante de l'ensemble des m règles est donnée par la moyenne des sorties individuelles pondérées par le degré d'activation des règles, soit :

$$\boldsymbol{y} = \frac{\sum_{i=1}^{m} \boldsymbol{w^i y^i}}{\boldsymbol{w^i}} \tag{I. 5}$$

avec w^i désigne le degré d'activation de chaque règle i et est calculé par l'équation suivante :

$$\boldsymbol{w^i} = \prod_{j=1}^{n} \mu(A_j^i) \tag{I. 6}$$

où $\mu(A_j^i)$ désigne la fonction d'appartenance associée à la variable de prémisse x_j .

Les modèles TS sont qualifiés d'approximateurs universels assimilables à des multimodèles. Ils sont reconnus par leur capacité d'approximation et de généralisation déjà démontrées dans la littérature [5,6]. Ils permettent entre autre d'approcher n'importe quelle fonction définie sur un espace compact donné à un certain degré de précision. Ils sont d'ailleurs bien adaptés aux problèmes de modélisation et d'identification des systèmes dynamiques complexes et à la représentation multimodèle. Leur utilisation pour la synthèse de stratégies de commande multimodèle suscite également un grand intérêt vu les résultats significatifs qui ont été reportés dans bon nombre d'applications.

I.5 Identification des systèmes flous

I.5.1 Problématique générale d'identification

Le problème de modélisation floue ou de construction de modèles flous peut être vu comme un processus d'identification de système. D'un point de vue conceptuel, l'identification d'un système flou est effectuée selon une méthodologie générale établie sur trois phases [6,7] :

- Détermination de la structure du modèle. : il s'agit de choisir le type du modèle flou, ses variables d'entrée et de sortie et les paramètres des prémisses et des conclusions des règles.

- Estimation des paramètres du modèle : les paramètres peuvent être estimés à partir d'un ensemble de données appelé ensemble d'apprentissage ou déterminés par voie de calcul ou d'optimisation.
- Validation du modèle résultant par rapport aux données d'apprentissage et de test.

I.6 Méthodes d'identification

I.6.1 A partir de connaissances issues de l'expertise

La conduite de processus est dans la plupart des cas guidée par les connaissances disponibles ou acquises par les experts. Le dimensionnement, la prise de décision ou encore le traitement de situations problématiques est souvent tributaire de ces connaissances. Dans le contexte particulier de l'identification de systèmes flous, l'idée sur laquelle se fond cette approche consiste à formuler ces connaissances suivant un mécanisme approprié de façon à permettre leur intégration dans le protocole d'identification du système appréhendé. Cette démarche vise principalement à construire des modèles flous linguistiques ou de Mamdani formés par un ensemble de règles IF-THEN, ce qui explique leur transparence et justifie leur interprétabilité. L'expert établit alors un certain nombre de termes linguistiques sur les espaces d'entrée et de sortie. Ces termes constituent le langage qu'il utilise pour décrire le système. Ils forment également les partitions des différents espaces. Les fonctions d'appartenance sont en général de forme connue, triangulaire, gaussienne, trapézoïdale, etc.

L'utilité de cette approche est néanmoins étroitement liée à la nature et la quantité de connaissances qualitatives disponibles. Il est parfois difficile, voire impossible, d'obtenir une base de connaissances suffisamment complète, surtout lorsqu'il s'agit de traiter des systèmes à dynamiques complexes [8,9].

I.6.2 A partir de données

Devant la non disponibilité ou la difficulté d'obtenir des connaissances sur le système à identifier, l'identification de systèmes flous peut être établie directement à partir de données. Ces données sont généralement issues d'un prélèvement de mesures sur l'entrée et la sortie du système à identifier. Une méthode de classification ou de coalescence floue peut être utilisée pour structurer et classifier les données, c'est-à-dire partitionner leur espace de représentation en plusieurs classes. La coalescence floue (fuzzy clustering) sert à déterminer

une partition de l'espace indépendamment pour chaque variable ou directement pour une collection de variables. Ceci s'effectue à l'aide d'algorithmes de classification floue dont l'avantage est de générer systématiquement et de manière simultanée l'ensemble ou une partie des paramètres du modèle flou à partir de la base de données. Certains de ces algorithmes sont déduits de la résolution de problèmes d'optimisation tels que l'algorithme des C-moyennes floues (FCM) [**10**], et l'algorithme de Gustafson-Kessel (GK) [**11**] , d'autres reposent sur des techniques utilisant des mécanismes de partage et de fusion de données telles que la méthode de "splitting and merging" [12,13].

I. 6.3 Par linéarisation dynamique

Un modèle non linéaire peut être linéarisé autour d'un certain nombre de points de fonctionnement choisis judicieusement. Les modèles locaux obtenus sont ensuite concaténés dans un modèle global à l'aide de fonctions d'appartenance.

Considérons le système non linéaire continu décrit dans l'espace d'état par le modèle :

$$\dot{\boldsymbol{x}}(\boldsymbol{t}) = \boldsymbol{F}\big(\boldsymbol{x}(\boldsymbol{t}), \boldsymbol{u}(\boldsymbol{t})\big) \qquad \text{(I. 7)}$$

avec $F \in C^1$ une fonction non linéaire, $x(t) \in R^n$ est le vecteur d'état et $u(t) \in R^r$ désigne le vecteur d'entrée. La linéarisation dynamique du système décrit par l'équation (I. 7) autour d'un point de fonctionnement variant (x^0, u^0) sur une trajectoire continue $(x^0(t), u^0(t)) \subset R^n \times R^r$ nous conduit à un système linéaire à temps variant (LTV) décrit par l'expression:

$$\dot{\boldsymbol{x}}(\boldsymbol{t}) = \boldsymbol{F}\Big(\boldsymbol{x^0}(\boldsymbol{t}), \boldsymbol{u^0}(\boldsymbol{t})\Big) + \frac{\boldsymbol{\partial F}}{\boldsymbol{\partial x}}\Big(\boldsymbol{x^0}(\boldsymbol{t}), \boldsymbol{u^0}(\boldsymbol{t})\Big)\Big(\boldsymbol{x}(\boldsymbol{t}) - \boldsymbol{x^0}(\boldsymbol{t})\Big) + \frac{\boldsymbol{\partial F}}{\boldsymbol{\partial u}}\Big(\boldsymbol{x^0}(\boldsymbol{t}), \boldsymbol{u^0}(\boldsymbol{t})\Big)\Big(\boldsymbol{u}(\boldsymbol{t}) - \boldsymbol{u^0}(\boldsymbol{t})\Big) \qquad \text{(I. 8)}$$

L'approximation du modèle (I. 8) par un modèle flou T-S composé de c modèles locaux issus de la linéarisation autour de c points de fonctionnement $\{(x^i, u^i), i = 1, \dots, c\} \subset R^n \times R^r$ du modèle non linéaire (I. 7) conduit à la représentation suivante :

$$\dot{\boldsymbol{x}}(\boldsymbol{t}) = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{c}} \Big[\boldsymbol{F}(\boldsymbol{x^i}, \boldsymbol{u^i}) + \frac{\boldsymbol{\partial F}}{\boldsymbol{\partial x}}(\boldsymbol{x^i}, \boldsymbol{u^i})(\boldsymbol{x}(\boldsymbol{t}) - \boldsymbol{x^i}) + \frac{\boldsymbol{\partial F}}{\boldsymbol{\partial u}}(\boldsymbol{x^i}, \boldsymbol{u^i})(\boldsymbol{u}(\boldsymbol{t}) - \boldsymbol{u^i})\Big] \boldsymbol{\Phi_i}\big(\boldsymbol{x}(\boldsymbol{t}), \boldsymbol{u}(\boldsymbol{t})\big) \text{ (I. 9)}$$

où $\Phi_i\big(x(t), u(t)\big)$ désigne l'ensemble des fonctions de base dites d'activation ou d'appartenance.

Notons que le modèle (I. 9) est obtenu à partir de la représentation d'état du système (I. 7). Ainsi, il serait envisageable d'utiliser les techniques d'analyse et de synthèse appliquées aux systèmes linéaires pour l'étude des systèmes non linéaires, ceci peut concerner la synthèse de lois de commande multimodèle, la conception de multi-observateurs, etc.

I.7 Identification des modèles TS à partir de données

Comme nous l'avons mentionné dans la section I.4.3, le modèle flou TS a plus de rapport avec les multimodèles qu'avec les modèles linguistiques de Mamdani. D'ailleurs, c'est pour cette raison qu'il est le plus convoité dans le domaine industriel. Dans la suite du chapitre, deux méthodes d'obtention de modèles flous TS sont examinées : l'identification par classification floue et l'identification par méthodes d'optimisation.

I.7.1 Identification par classification floue

Les méthodes de classification ont généralement pour but de créer des partitions de données qui soient capables de caractériser au mieux leur distribution. Le principe consiste à regrouper les éléments d'un ensemble $Z = \{z_1, z_2, \dots, z_n\}$ en un nombre optimal de classes ou de clusters selon leurs ressemblances.

Les méthodes d'identification basées sur la classification floue ou encore la coalescence ou le clustering flou sont des méthodes qui adoptent le concept d'appartenance partielle ou totale pour former ces partitions. Plus précisément, ce concept est manipulé à travers un degré dit d'appartenance avec lequel un certain objet, représenté comme un vecteur dans un espace de caractéristiques, est similaire à un certain objet prototype. Le degré de similitude est calculé en utilisant une mesure appropriée de distance.

L'idée de base du clustering flou est représentée dans la Figure I. 5. Dans cette même figure, les données sont regroupées dans deux groupes avec des prototypes v_1 et v_2, en utilisant une mesure Euclidienne de distance dans un espace caractéristique R^2. La partition des données est représentée dans la matrice de partition floue, $U = [u_{ik}]$, dont les éléments u_{ik} sont les degrés d'appartenance des points $z_k = [x_k, y_k]$ aux clusters flous avec prototypes v_i.

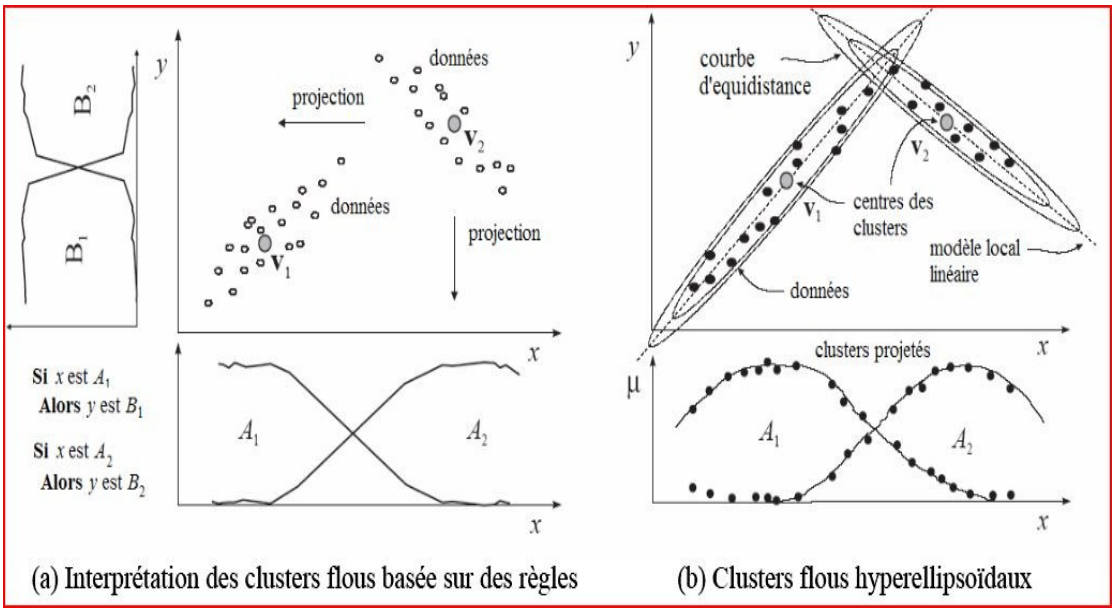


Figure I. 5 : Illustration du principe général du clustering flou [11].

Des règles floues de type IF-THEN peuvent être extraites en projetant les clusters sur les axes des coordonnées. La Figure I. 5 (a) montre un ensemble de données avec deux clusters apparents et deux règles floues associées. La similitude des données par rapport à un prototype permet d'envisager plusieurs formes pour la mesure de distance. Certains algorithmes utilisent la simple norme Euclidienne telle que l'algorithme des C-moyennes floues (FCM), d'autres algorithmes font appel à des normes adaptatives leur permettant la détection de clusters de différentes formes et différentes orientations telles que l'algorithme de Gustafson-Kessel (GK). La Figure I. 5 (b) illustre par exemple des clusters hyper-ellipsoïdaux. Dans la suite de ce chapitre, on revient avec plus de détails sur les algorithmes de clustering flou FCM et GK.

I.7.2 Identification par méthodes d'optimisation

La conception des modèles à base de règles floues peut être formulée comme un problème d'optimisation dans un espace multidimensionnel, où chaque point représente un modèle flou potentiel de structure et de paramètres appropriés. Ces dernières années, la tendance a été fortement axée sur les méthodes de recherche globale ou plus précisément les méthodes d'optimisation méta-heuristiques [**14**]. Citons à titre d'exemple, les algorithmes évolutionnaires (EA) tels que les algorithmes génétiques, les stratégies d'évolution et la programmation évolutionnaire, et les techniques d'intelligence en essaim telles que les

essaims de particules (PSO), les fourmis artificielles (ACO), les colonies d'abeilles artificielles (ABC, BCO, BA), etc.

L'identification de systèmes flous par optimisation est généralement basée sur un apprentissage qui permet de définir de façon itérative le meilleur jeu de paramètres pour une structure donnée du système flou. Les recherches sont concentrées principalement sur les approches suivantes :

- Optimisation des fonctions d'appartenance,
- Optimisation des règles floues,
- Optimisation simultanée des fonctions d'appartenance et des règles floues.

Le critère de performance que l'on associe à un tel problème d'optimisation exprime généralement la qualité du modèle flou à identifier. Il peut s'agir de la précision obtenue sur le modèle ou encore la transparence et l'interprétabilité du modèle. Ces caractéristiques sont déterminantes quant à l'utilité du modèle flou résultant. Le concepteur aura le plus souvent à réaliser un compromis entre précision et transparence en tenant compte du coût d'implémentation de l'algorithme d'identification floue.

I.8 Algorithmes de classification floue

I.8.1 L'algorithme Fuzzy C-means

L'algorithme des C-moyennes floues (Fuzzy C-means, FCM), issu des travaux de Dunn [**10**] et amélioré plus tard par Bezdek dans [**15**], est le plus utilisé parmi les différentes méthodes de coalescence floue reportées dans la littérature. Pour illustrer son principe, considérons un ensemble de données constitué de n vecteurs, chacun de dimension N : $X = \{x_1, \dots, x_n\}$; $x_n \in R^N$. On suppose que le nombre de classes est connu a priori, soit "c" ce nombre. La coalescence par la méthode des FCM produit:

- c vecteurs $[v_1, v_2, \dots, v_c]$ de dimension n, qui représentent les centres des classes.
- pour chaque vecteur $x_k \in X$, c degrés d'appartenance qui représentent les degrés de similarité de ce vecteur avec les centres des classes. On note μ_{ik} le degré d'appartenance du vecteur x_k à la classe v_i. Les degrés d'appartenance vérifient les relations :

$$\sum_{i=1}^{c} \boldsymbol{u_{ik}} = \mathbf{1}, \ \ \boldsymbol{k} = \mathbf{1} \dots \boldsymbol{N} \qquad \text{(I. 10)}$$

$$\mathbf{0} < \sum_{k=1}^{N} \boldsymbol{u_{ik}} \leq \boldsymbol{N}, \quad \boldsymbol{i} = \mathbf{1} \ldots \boldsymbol{c} \tag{I. 11}$$

La classification s'obtient par la minimisation de la fonction coût suivante:

$$\boldsymbol{J} = \sum_{i=1}^{c} \sum_{k=1}^{N} (\boldsymbol{u_{ik}})^{\delta} \boldsymbol{d_{ik}^2} \tag{I. 12}$$

où, $d_{ik} = \|x_k - v_i\|$ est la distance entre le vecteur x_k et le centre de la classe i. Le paramètre δ>1 contrôle le degré de flou « fuzziness index» des classes obtenues. Ce paramètre détermine en fait le recouvrement entre les classes. Plus la valeur de δ est grande, plus les classes se recouvrent. La valeur δ=2 est souvent utilisée.

La minimisation de la fonction J est effectuée par rapport aux variables d_{ik} et u_{ik}. La méthode de coalescence FCM peut être résumée par l'algorithme décrit dans la Tableau I. 1.

1. **Initialisation:** fixer δ et c et initialiser la matrice de partition floue $[u_{ik}]$
2. **Répéter**

   Calculer les centres des classes

   $$v_i = \frac{\left(\sum_{k=1}^{N} (u_{ik})^{\delta} . x_k\right)}{\left(\sum_{k=1}^{N} (u_{ik})^{\delta}\right)}; \quad i = 1 \ldots c$$

   Calculer $d_{ik} = \|\boldsymbol{x_k} - \boldsymbol{v_i}\|$; $i = 1 \ldots c$

   Calculer la matrice $[u_{ik}]$

   Si $d_{ik} \neq 0$ alors :

   $$u_{ik} = \frac{1/(d_{ik})^{2/(\delta-1)}}{\sum_{i=1}^{c} 1/(d_{ik})^{2/(\delta-1)}}$$

   Si $d_{ik} = 0$ alors: $u_{ik} = 1$ et $u_{jk} = 0 \; \forall \; i \neq j; \; (i, j = 1 \ldots c)$
3. **Jusqu'à** un certain seuil d'erreur sur J ou stabilité de la matrice$[u_{ik}]$

   $\left\|U^{(l)} - U^{(l-1)}\right\| < \varepsilon$

Tableau I.1 Algorithme des C-moyennes floues (FCM)

I.8.2 L'algorithme de Gustafson-Kessel

En 1979, Gustafson et Kessel ont développé une variante relativement différente du FCM qui permet de détecter des clusters hyper-ellipsoïdaux [**11**]. Les prototypes P_i sont entièrement définis par les centres des clusters et les matrices de covariance associées. L'algorithme de Gustafson-Kessel (GK) est reconnu d'être le plus adéquat pour l'identification des modèles flous vu la possibilité de détecter des clusters de différentes

formes et différentes orientations. Ceci est rendu possible grâce à l'utilisation d'une mesure de distance adaptative.

La distance entre un élément $x_k \in X$ et un cluster prototype P_i est définie par :

$$\boldsymbol{d_{ik}^2(x_k, v_i) = \|x_k - v_i\|_{M_i}^2 = (x_k - v_i)^T M_i (x_k - v_i)} \quad \text{(I. 13)}$$

où $V = [v_1, v_2, \dots, v_c]$ est le vecteur des centres des clusters à déterminer, et $M_i = \det(F_i)^{1/(n+1)} (F_i)^{-1}$ une matrice symétrique et définie positive qui dépend de la matrice de covariance F_i du ième prototype. L'algorithme de clustering flou de GK est donné dans la Tableau I.2.

1. **Initialisation** introduire la matrice de données X, fixer c $(1 < c < N)$, $\delta > 1$,$\varepsilon > 0$.
2. **Répéter**

   Calculer les centres des clusters :

$$v_i^{(l)} = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^{\delta} x_k}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^{\delta}}$$

   Calculer les matrices de covariance :

$$F_i = \frac{\sum_{k=1}^{N} \left(\mu_{ik}^{(l-1)}\right)^{\delta} (x_k - v_i^{(l)})(x_k - v_i^{(l)})^T}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^{\delta}}$$

   Evaluer les distances :

$$M_i = det(F_i)^{1/(n+1)} (F_i)^{-1}$$

$$d_{ik}^2 = (x_k - v_i)^T M_i (x_k - v_i)$$

   Actualiser la matrice de partition floue :

   Pour $1 \le i \le c, 1 \le k \le N$

   Si $d_{ik} > 0$

$$\mu_{ik}^{(l)} = \frac{1}{\sum_{j=1}^{c} [d_{ik}/d_{jk}]^{\frac{2}{\delta-1}}}$$

   Si $d_{ik} = 0$

$$\mu_{ik}^{(l)} = 1$$

3. **Jusqu'à** $\left\| U^{(l)} - U^{(l-1)} \right\| < \varepsilon$

Tableau I. 2 L'algorithme de classification floue de Gustafson-Kessel.

I.8.3 Estimation des paramètres des conséquences

Comme il vient d'être expliqué, les algorithmes de classification floue FCM et GK permettent la détection des centres de clusters et les matrices de partitions floues associées. Ainsi, ce sont les paramètres des conditions des règles formant le modèle flou TS qui sont alors générés. Les paramètres des conséquences des règles floues peuvent être estimés séparément à l'aide de la méthode des moindres carrés simple. Contrairement à l'estimation globale des paramètres, cette approche est intéressante car elle permet de surmonter le problème de compensation des paramètres, préservant ainsi la validité du modèle flou identifié [**16**]. Pour illustrer brièvement son principe, considérons la matrice de régression étendue $X_e = [\mathrm{X}\ 1]$, où X est la matrice des données. Le vecteur des paramètres à estimer est donné par :

$$\boldsymbol{\Theta_i} = \left[\boldsymbol{a_i^0}, \boldsymbol{a_i^1}, \ldots, \boldsymbol{a_i^{n_I}}\right]^{\boldsymbol{T}} \tag{I. 14}$$

On définit une matrice diagonale Ω_i ayant pour éléments les degrés d'activation . Cette matrice s'exprime par :

$$\boldsymbol{\Omega_i} = \begin{bmatrix} \boldsymbol{\mu_{i1}} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mu_{i2}} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \boldsymbol{\mu_{in_I}} \end{bmatrix} \tag{I. 15}$$

Comme formes paramétriques, on peut considérer des partitions de forme gaussiènne définies par leur centres $\boldsymbol{c_j^i}$ et leurs écart-types $\boldsymbol{\sigma_j^i}$:

$$\boldsymbol{\mu_{i1}} = \boldsymbol{exp}\left(-\left(\frac{x_j - c_j^i}{\sigma_j^i}\right)^{\mathbf{2}}\right) \tag{I. 16}$$

Les paramètres locaux $\boldsymbol{\Theta_i}$ sont estimés au sens des moindres carrés par l'expression suivante :

$$\boldsymbol{\Theta_i} = [[\boldsymbol{X_e}]^{\boldsymbol{T}} \times [\boldsymbol{\Omega_i}] \times [\boldsymbol{X_e}]]^{-\mathbf{1}} [\boldsymbol{X_e}]^{\boldsymbol{T}} \times [\boldsymbol{\Omega_i}] \times [\boldsymbol{y}] \tag{I. 17}$$

où y est le vecteur de sortie.

I.9 Conclusion

L'objectif de ce chapitre est de présenter une synthèse sur les systèmes flous en abordant les outils mathématiques nécessaires à leur manipulation ainsi que leurs méthodes d'obtention.

Après avoir introduit les concepts de base concernant la structure générale et les déférents types de modèles flous, nous nous sommes intéressés plus particulièrement aux modèles flous de type Takagi-Sugeno (TS). Leur intérêt réside principalement dans la possibilité d'associer à travers leur structure des connaissances à la fois qualitatives et quantitatives, ce qui leur confère la propriété d'approximation universelle. L'obtention de ces modèles à partir de données numériques constitue la principale orientation que l'on s'est fixée dans ce travail. A cet effet, nous avons illustré certaines des méthodes d'identification floue à partir de données en passant en revue celles qui se basent sur la classification floue et sur l'approche optimisation qui fera le principal objet du chapitre suivant.

Chapitre II

Les méta-heuristiques pour la synthèse des systèmes flous

II.1 Introduction

L'optimisation, en tant que théorie fondamentale à part entière, s'est graduellement dotée d'outils et de méthodes qui sont fréquemment utilisés dans bon nombre de disciplines. Plus généralement, un problème de synthèse de systèmes flous peut s'exprimer comme un problème d'optimisation dont la résolution se réduit à la recherche de solutions optimales (modèles flous) répondant à des critères de performance et des contraintes prédéfinis. La convergence vers une solution optimale "de qualité" dans un temps "raisonnable" n'est souvent pas une tâche aisée. Ceci constitue d'ailleurs un constat pour toutes les méthodes d'optimisation classiques ou avancées qui partagent le même souci à savoir la recherche d'un compromis entre la qualité des solutions obtenues et le temps de calcul utilisé.

C'est dans ce sens que sont introduites des méthodes d'optimisation dites méta-heuristiques qui se basent sur des mécanismes appropriés visant à assurer un compromis entre *diversification* (concentrer la recherche sur des zones difficiles à exploiter de l'espace de recherche) et *intensification* (rechercher les meilleures solutions dans la région de l'espace de recherche en cours d'analyse). Il existe un grand nombre de méta-heuristiques différentes, allant de la simple recherche locale à des algorithmes complexes de recherche globale. Elles sont souvent inspirées de processus naturels qui relèvent de la physique (l'algorithme du recuit simulé), de la biologie de l'évolution (les algorithmes génétiques) ou encore de l'intelligence en essaim (les algorithmes de colonies de fourmis, les essaims particulaires ou les colonies d'abeilles).

Ce chapitre décrit le problème général d'identification des systèmes flous par l'utilisation de méthodes d'optimisation. Une formulation l'exprimant comme un problème d'optimisation

est tout d'abord explicitée, suivie d'un rappel de certains algorithmes méta-heuristiques qui ont été utilisés pour le résoudre.

II.2 Problématique générale d'optimisation

Un problème d'optimisation au sens général est défini par un ensemble de variables x et/ou de paramètres Θ, une fonction objectif $J(x,\Theta)$ et un ensemble de contraintes d'égalité ou d'inégalité $\sigma(x,\Theta)$ que les variables doivent satisfaire. L'ensemble des solutions possibles du problème forme l'espace de recherche fini Γ, où chaque dimension correspond à une variable. Suivant le problème posé, nous cherchons à minimiser ou maximiser la fonction objectif $J(x,\Theta)$, c'est-à-dire résoudre le problème :

$$\mathbf{min/max}_{(\mathbf{x},\boldsymbol{\theta})\in\Gamma,}\boldsymbol{J}(\mathbf{x},\boldsymbol{\theta}) \quad \text{sous la contrainte } \sigma(x,\Theta) \qquad \text{(II. 1)}$$

Le problème d'optimisation (II.1) peut être statique ou dynamique (la fonction objectif change avec le temps), mono-objectif ou multi-objectif (plusieurs fonctions objectifs doivent être optimisées) et avec ou sans les contraintes $\sigma(x,\theta)$.

La recherche de solutions optimales pour un problème d'optimisation est étroitement liée à la méthode utilisée. Selon le cas étudié, on peut faire appel soit à des méthodes déterministes (ou exactes) telles que la programmation linéaire [**17**], quadratique [**18**] et/ou dynamique [**19**] , la méthode de Newton [**17**], la méthode du simplex [**20**] ou encore la méthode du gradient [**21**], lorsque la fonction objectif présente certaines caractéristiques de convexité, de continuité ou de dérivabilité ; soit à des méthodes de résolution approchées qui sont plutôt adaptées à des problématiques difficiles à variables discrètes ou à variables continues. Il convient de noter que les méta-heuristiques s'apprêtent bien aux problèmes difficiles qui présentent certaines caractéristiques complexes telles que la non-dérivabilité de la fonction objectif, la dimension élevée du problème d'optimisation, l'existence de nombreuses variables présentant des corrélations non identifiées et de nombreuses contraintes imposées ou encore la présence de bruits dans les mesures.

Le problème d'identification des systèmes flous à partir de données expérimentales peut se ramener à un problème d'optimisation à variables continues et sera traité alors conformément à cette formulation.

II.3 Identification des systèmes flous par méthodes d'optimisation

Comme nous l'avons vu dans le premier chapitre, les systèmes flous sont décrits par des modèles à base de règles floues de type IF-THEN. L'obtention de ces modèles peut être réalisée à l'aide de méthodes d'identification floue ou de modélisation floue appropriées. Certaines de ces méthodes se fondent partiellement ou totalement sur des approches d'optimisation déterministes ou approchées. Dans ce cas précis, la conception d'un système flou se ramène alors à la résolution d'un pur problème d'optimisation.

II.3.1 Formulation du problème d'optimisation

Soit un modèle flou de type Takagi-Sugeno (TS) composé de N_R règles de la forme:

$$\boldsymbol{R^i}: \boldsymbol{IF\ x_1 is\ A_1^i and \dots and\ x_{N_R} is\ A_{N_R}^i\ THEN\ y^i = a_0^i + a_1^i x_1 + \cdots + a_{N_R}^i x_{N_R}},$$

$$\mathbf{i = 1, \dots, N_R} \qquad \text{(II. 2)}$$

où $x_1, \dots, x_n$ sont les variables de prémisse et y la variable de sortie. La conception de ce modèle revient d'une part à trouver une structure appropriée par la détermination du nombre de règles N_R et les paramètres des prémisses ou les ensembles flous $A_j^i, i = 1, .., N_R, j = 1, \dots, n$, et d'autre part par l'estimation des paramètres de la conséquence des règles a_j^i .

La conception de ce modèle flou par voie d'optimisation peut obéir à deux orientations distinctes. Dans la première, la procédure d'optimisation manipule les paramètres et/ou la structure d'un seul modèle de représentation en vue d'un ajustement optimal par rapport à un critère de performance prédéfini. La deuxième orientation considère le problème d'identification floue comme une optimisation dans un espace multidimensionnel, où chaque point de l'espace est une solution potentielle de ce problème. La solution optimale ou quasi-optimale est obtenue via un processus d'optimisation basé sur la minimisation ou la maximisation d'un critère de performance donné, le plus souvent défini comme étant l'erreur quadratique moyenne entre les mesures et la sortie du modèle.

II.3.2 Recherche de modèles optimaux

Que l'on s'oriente vers l'une ou l'autre approche d'optimisation pour la conception de systèmes flous, on distingue généralement deux niveaux d'optimisation dans un système d'inférence floue : l'optimisation structurelle et l'optimisation paramétrique.

II.3.2.1 Optimisation structurelle

L'optimisation structurelle s'effectue en deux étapes : sélection des variables d'entrée et réduction de la base des règles. Dans de nombreuses applications, les bases de données disponibles sont importantes en volume et, par conséquent, la conception de systèmes d'inférence floue par ajustement d'une base de règles ou bien la génération automatique de la base des règles floues doit nécessairement intégrer une optimisation structurelle. Celle-ci est déterminante quant à l'utilité et la performance du système flou développé.

II.3.2.1.1 Sélection des variables

La sélection des variables consiste à choisir parmi les variables qui décrivent le système, celles qui sont susceptibles d'être utilisées comme entrée du système. Pour la sélection de ces variables, on peut tout simplement faire appel aux connaissances disponibles sur le système à identifier, la contribution de l'opérateur système est importante dans ce cas. D'autre part, cette sélection peut être formulée dans le processus d'optimisation adopté. Ainsi, elle sera réalisée globalement ou localement sur évaluation itérative de la fonction objectif représentant la qualité du modèle. Dans la sélection globale, la variable est exclue et aucune règle ne peut l'utiliser, alors que dans la sélection locale le choix des variables est réalisé au niveau des règles, ce qui peut éventuellement engendrer des règles incomplètes.

II.3.2.1.2 Optimisation de la base des règles

Dans certaines approches, l'optimisation de la base des règles consiste à éliminer les termes redondants afin de réduire le temps de calcul et l'exploitation du système flou de façon optimale. La technique du groupage flou [10,11] prend en compte le nombre de règles, en créant de nouveaux centres au fur et à mesure des besoins, et tend à minimiser le nombre de règles du système. Ce concept de synthèse dit "évolutif" a l'avantage de passer d'une règle à un nombre fini de règles, ce qui correspond à un processus d'identification floue évolutive à partir de données qui peut se baser partiellement ou totalement sur un algorithme d'optimisation. Une autre alternative consiste à utiliser des méthodes de réduction de règles dont l'utilité s'impose si l'on souhaite fusionner des règles d'une même base ou encore fusionner deux ou plusieurs bases de règles, par exemple l'une issue de l'expertise et l'autre issue des données. Dans ce cas, le nombre de règles floues est connu a priori.

L'optimisation de la base de règles peut également faire partie du problème d'optimisation globale sur lequel est fondé le processus d'identification du système flou. Le nombre de règles étant alors un des paramètres à optimiser qui n'est pas nécessairement connu a priori. La fonction objectif à optimiser dépend alors de ce paramètre. La solution du problème d'optimisation fournit par conséquent l'ensemble des paramètres du système et de manière simultanée le nombre de règles optimal.

II.3.2.2 Optimisation paramétrique

L'optimisation paramétrique concerne les paramètres établissant la forme des ensembles flous de la prémisse des règles et les conséquences des règles. Pour ce faire, des techniques classiques d'optimisation peuvent être utilisées à l'exemple des moindres carrés ou la descente du gradient [8,9]. Ces techniques s'appliquent plus particulièrement pour l'ajustement optimal d'un modèle donné. Une autre manière de procéder consiste à manipuler plusieurs solutions potentielles avec des paramètres de prémisses ou de conclusions propres à chaque solution. La procédure d'optimisation est alors définie par rapport à l'ensemble de ces paramètres qui évolueront simultanément pour converger vers une solution optimale. Les méta-heuristiques ont été largement employées pour résoudre cette dernière problématique [10, 11].

II.3.3 Mécanismes de codage

Pour appliquer les méta-heuristiques, le problème de la représentation de l'individu dans une population de solutions potentielles est souvent posé. Dans le contexte de l'identification floue, l'individu est censé d'être entièrement représentatif du modèle flou recherché. La représentation du modèle flou à identifier s'effectue à travers un mécanisme de codage approprié dont le choix dépend souvent du type de modèle et de la méthode d'optimisation méta-heuristique utilisée. On recense dans la littérature plusieurs schémas de codage qui sont définis en fonction des paramètres du modèle flou à identifier [22,23,24]. Ainsi, on pourra choisir de coder une seule partie du modèle (la conclusion des règles, par exemple) ou encore plusieurs parties du modèle de manière à ce que la structure et/ou les paramètres du système soient entièrement spécifiés. Les éléments du système flou que l'on peut intégrer dans un schéma de codage en vue d'une optimisation méta-heuristique sont les suivants :

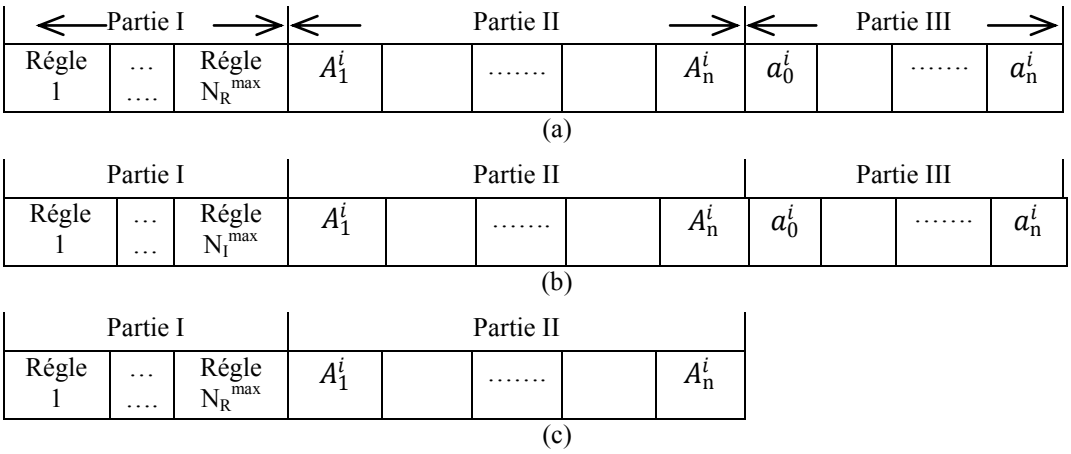


Figure II.1 : Exemples de schémas de codage pour la génération des systèmes flous

- Les ensembles flous associés aux variables de prémisse.
- Les ensembles flous associés aux variables de la conséquence des règles floues dans les modèles linguistiques.
- Les paramètres des fonctions associées aux conséquences des règles floues dans un modèle TS.
- Les variables de prémisse pour la sélection automatique des entrées du système flou.
- Les règles floues pour la génération automatique du nombre de règles.

La Figure II.1 illustre quelques mécanismes de codage utilisés pour la synthèse des systèmes flous. Comme il peut être constaté, on peut se limiter à deux parties associant le nombre de règles et les paramètres des prémisses (Fig. II.1(c)) ou encore trois parties associant les entrées à sélectionner, les prémisses et les conditions des règles (Fig. II.1(b)) ou le nombre de règles (Fig. II.1(a)).

II.4 Méthodes d'optimisation méta-heuristiques

Les méta-heuristiques (terme inventé par Glover en 1986 lors de la conception de la recherche tabou [**25**]) sont des algorithmes d'optimisation stochastiques destinés plus particulièrement à résoudre des problèmes d'optimisation difficile tels que le problème de synthèse de systèmes flous traité dans le cadre de ce mémoire. Ils se basent sur des mécanismes itératifs où une ou plusieurs solutions sont manipulées à la recherche de l'optimum. Plusieurs algorithmes méta-heuristiques simples, parallèlement combinés ou

encore hybrides ont été implémentés dans des études très récentes pour résoudre le problème d'extraction automatique de systèmes à base de règles floues. Certaines de ces méthodes sont décrites dans la suite de ce paragraphe.

II.4.1 Optimisation par essaim particulaire

L'optimisation par essaim particulaire « Particle Swarm Optimization (PSO) », est un algorithme évolutionnaire qui utilise une population de solutions candidates pour développer une solution optimale au problème. Cet algorithme a été proposé par Russel Eberhart et James Kennedy en 1995 [**26**]. Il s'inspire du comportement social des animaux évoluant en essaim, tels que les bancs de poissons et les vols groupés d'oiseaux.

L'essaim de particules correspond à une population d'individus appelés particules. Chaque particule (solution potentielle) possède une position et une vitesse. A chaque particule on associe une mémoire lui permettant de se souvenir de sa meilleure performance (en position et en vitesse) et de la meilleure performance atteinte par les autres particules voisines (informatrices).

Le déplacement d'une particule est illustré dans la Figure II.2. Il est influencé par trois composantes : une composante d'inertie, une composante cognitive et une composante sociale.

Une particule i de l'essaim est modélisée par son vecteur position $x_i = (x_{i1}, x_{i2}, \dots, x_{in})$ et son vecteur vitesse $v_i = (v_{i1}, v_{i2}, \dots, v_{in})$. Désignons par $p_i = (p_{i1}, p_{i2}, \dots, p_{in})$ la meilleure position atteinte par la particule i, et par $p_g = (p_{g1}, p_{g2}, \dots, p_{gn})$ la meilleure position atteinte par toutes les particules de l'essaim. La vitesse et la position de chacune

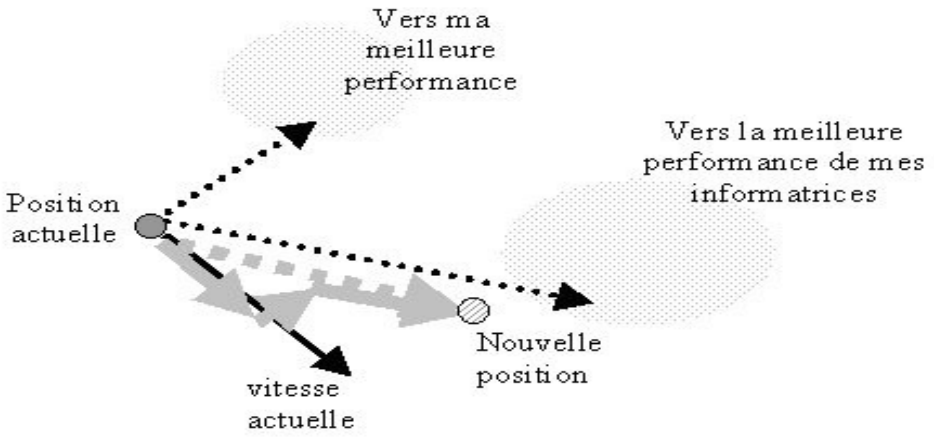


Figure II. 2 : Stratégie de déplacement dans un essaim particulaire [**26**]**.**

des particules évoluent itérativement selon les équations suivantes :

$$\boldsymbol{v_{ij}(t+1) = v_{ij}(t) + c_1 r_1\big(p_{ij}(t) - x_{ij}(t)\big) + c_2 r_2\big(p_{gj}(t) - x_{ij}(t)\big)} \qquad \text{(II. 3)}$$

$$\boldsymbol{x_i(t+1) = x_i(t) + v_i(t+1)} \qquad \boldsymbol{i = 1, \ldots N,\ j = 1, \ldots . N} \qquad \text{(II. 4)}$$

où c_1 et c_2 sont deux constantes, appelées coefficients d'accélération, r_1 et r_2 sont deux nombres aléatoires dans [0, 1] et N la taille de la population.

La structure algorithmique de la méthode PSO est illustrée dans la Tableau II. 1. Le critère d'arrêt peut être défini comme étant une "erreur d'approximation", comme il peut correspondre à un nombre maximum d'itérations ou un nombre maximum d'évaluations de la fonction objectif f.

1. **Initialiser** aléatoirement une population de N particules : position et vitesse ;
2. **Evaluer** les positions et les vitesses des particules selon les équations (II. 2) et (II. 3)
3. **Répéter**
4. **Pour** i = 1, ..., N

   Déplacer les particules selon (II. 2) et (II. 3)

   Si $f(x_i) < f(p_i)$

   $p_i = x_i$;

   Si $f(x_i) < f(p_g)$

   $p_g = x_i$;

5. **Jusqu'à atteindre le critère d'arrêt**

Tableau II. 1: Algorithme d'optimisation par Essaim Particulaire (PSO).

Les applications de l'algorithme PSO sont nombreuses et diversifiées. Dans le domaine de la classification de données, une approche de clustering flou utilisant l'algorithme PSO a été présentée dans [**27**]. Une approche d'optimisation hybride utilisant l'algorithme PSO et l'algorithme de colonies de fourmis (ACO) a été établie pour résoudre le problème de clustering flou dans [14]. Dans le contexte de l'identification de systèmes flous, des méthodes basées sur l'algorithme PSO ont été également développées dans [7,11,12]. Dans [**28**],

l'algorithme PSO est employé pour optimiser les paramètres de modèles flous dont la structure est déterminée à travers une approche de clustering implémentée en ligne. Dans ce même contexte, un algorithme parallèle à essaim de particules (CRPSO) a été développé dans [17] pour la génération automatique de bases de règles floues pour l'identification des modèles TS. D'autres applications de l'algorithme à des problèmes particuliers de conception de systèmes neuronaux et neuro-flous ont fait l'objet de plusieurs travaux de recherche [6,15,16].

II.4.2 Les algorithmes génétiques

Les algorithmes génétiques (GA) est une méthode d'optimisation itérative qui se base fondamentalement sur des mécanismes biologiques et sur le principe d'évolution de Darwin [**29**].

Le principe général d'un GA repose sur des mécanismes bien communs de sélection, de croisement et de mutation. Un GA standard nécessite, en premier, le codage de l'ensemble des paramètres du problème d'optimisation en une chaîne de longueur finie, appelée chromosome. Une population initiale d'individus (solutions) est tout d'abord générée de façon aléatoire. A chaque génération ou itération, une procédure de sélection des individus est appliquée en se basant sur une fonction d'adéquation prédéfinie. Ensuite, des opérateurs de croisement et de mutation sont opérés et une nouvelle population est alors créée.

II.4.2.1 Sélection

La procédure de sélection consiste à favoriser la propagation des meilleures solutions en identifiant les individus sur la base de leur fonction d'adéquation, les individus les mieux adaptés sont sélectionnés alors ceux qui le sont moins sont systématiquement écartés [**30**]. Plusieurs opérateurs de sélection ont été proposés dans la littérature, les plus connus étant la « roue de la fortune » et la « sélection par tournoi » [**29**].

II.4.2.2 Croisement

Cette étape crée de nouveaux individus en échangeant des parties des individus de la population (parents). Cet échange se fait en sélectionnant un point de découpage aléatoirement (avec une densité de probabilité uniforme), et en échangeant ensuite les deux sous chaînes de chacun des deux parents. La Figure II. illustre le principe de croisement.

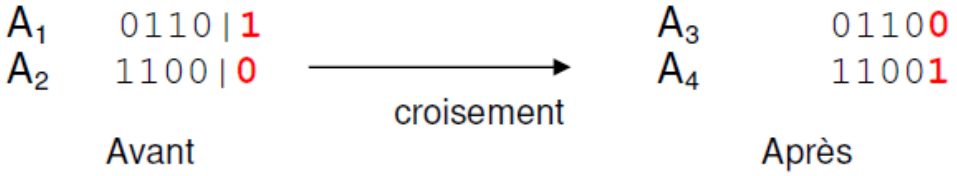


Figure II. 3 : Représentation d'un croisement en un point de deux chaînes.

II.4.2.3 Mutation

La mutation est exécutée seulement sur une seule chaîne. Dans un codage binaire, cela revient à changer un 1 en 0 et vice versa, comme montré en Figure II. . Cet opérateur contribue à la diversification des solutions recherchées.

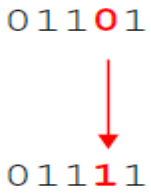


Figure II. 4 : Représentation d'une mutation de bits dans une chaîne.

La structure de base des algorithmes génétiques (GA) est fondée principalement sur ces trois opérateurs comme le montre la Tableau II.2.

1. **Initialisation** de la population d'individus $P(t)$
2. Evaluer chaque individu de $P(t)$
3. **Répéter**
   t = t +1
   Sélectionner P(t +1) à partir de P(t)
   Croisement P(t +1)
   Muter P(t +1)
   Evaluer P(t +1)
4. **Jusqu'à** ce que le critère d'arrêt est atteint

Tableau II. 2 Principe d'un algorithme génétique.

Dans la littérature, plusieurs références proposent l'utilisation des algorithmes génétiques dans le domaine de la classification et du clustering. Parmi les approches proposées, on distingue des algorithmes d'extraction ou de détection de classes basés entièrement sur les GA [20]. D'autres considèrent l'amélioration des performances de certains algorithmes de coalescence floue tels que l'algorithme des c-moyennes floues (FCM) par l'optimisation des prototypes de clusters et de la matrice de partition floue [19].

D'autres travaux se sont axés sur l'utilisation des algorithmes génétiques pour l'optimisation de la norme de distance définie dans l'algorithme FCM [**10**].

II.4.3 Algorithme d'optimisation à évolution différentielle

Proposé par Prince et Storn [**31**], l'évolution différentielle (Differential Evolution, DE) est une méta-heuristique stochastique d'optimisation qui a été inspirée par les algorithmes génétiques combinés avec une technique géométrique de recherche. Dans la méthode DE, la population initiale est générée par tirage aléatoire uniforme sur l'ensemble des valeurs possibles de chaque variable. Les bornes inférieures et supérieures des variables sont spécifiées par l'utilisateur selon la nature du problème. Après l'initialisation, l'algorithme DE effectue une série de transformations sur les individus, dans un processus appelé « évolution » [**31**].

Pour illustrer brièvement le principe de l'algorithme DE, considérons une population composée de **N** individus, représenté chacun par le vecteur

$$\mathbf{x_i} = (\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_N}) , \quad \boldsymbol{i} = \mathbf{1}, \mathbf{2}, \dots, \boldsymbol{N} \tag{II. 5}$$

A chaque génération, l'algorithme DE applique successivement les trois opérations de mutation, de croisement et de sélection sur chaque vecteur pour produire un vecteur d'essai « trial vector »:

$$\mathbf{u_i} = (\mathbf{u_1}, \mathbf{u_2}, \dots, \mathbf{u_N}), \quad \boldsymbol{i} = \mathbf{1}, \mathbf{2}, \dots, \boldsymbol{N} \tag{II. 6}$$

II.4.3.1 Mutation

Pour chaque vecteur courant x_i, on génère un vecteur mutant $\boldsymbol{v_i}$, qui peut être créé en utilisant une des stratégies de mutation suivantes [**32**]:

$$\boldsymbol{Rand/1}: \boldsymbol{v_i} = \boldsymbol{x_{r_1,}} + \boldsymbol{C}\left(\boldsymbol{x_{r_2}} - \boldsymbol{x_{r_3}}\right) \tag{II. 7}$$

$$\boldsymbol{Best/1}: \boldsymbol{v_i} = \boldsymbol{x_{best}} + \boldsymbol{C}\left(\boldsymbol{x_{r_1}} - \boldsymbol{x_{r_2}}\right) \tag{II. 8}$$

Current to best/1

$$\boldsymbol{v_i = x_i + C(x_{r_1} - x_{r_2}) + C(x_{best} - x_i)} \quad \text{(II. 9)}$$

$$\boldsymbol{Best/2 :\ v_i = x_{best} + C(x_{r_1} - x_{r_2}) + C(x_{r_3} - x_{r_4})} \quad \text{(II. 10)}$$

$$\boldsymbol{Rand/2 :\ v_i = x_{r_1} + C(x_{r_2} - x_{r_3}) + C(x_{r_4} - x_{r_5})} \quad \text{(II. 11)}$$

Les indices $r_1, r_2, r_3, r_4 et\ r_5 \in \{1,2,\dots,N\}$ sont des entiers aléatoires et tous différents. Ils sont également choisis différents de l'indice courant i. x_{best} est le meilleur individu. $C \in [0,2]$ est une valeur constante, appelée le poids différentiel « differential weight », qui contrôle l'amplification de la variation différentielle de $\left(x_{r_i} - x_{r_j}\right)$.

II.4.3.2 Croisement

Après la mutation, une opération de croisement binaire forme le vecteur d'essai final u_i, selon le vecteur x_i et le vecteur mutant correspondant $\boldsymbol{v_i}$. L'opération de croisement est introduite pour augmenter la diversité des vecteurs de paramètres perturbés. Le nouveau vecteur u_{jG} est donné par la formule suivante :

$$\boldsymbol{u_{jG} = \begin{cases} v_{1G} & si(randb(j) \le CR)\ ou\ j = randi \\ x_{jG} & si(randb(j) > CR)\ ou\ j \ne randi \end{cases} \quad pour\ tout\ j \in \{1, 2, \dots, N\}} \quad \text{(II. 12)}$$

où $randb(j)$ est la j[éme] valeur procurée d'un générateur de nombre aléatoire uniforme appartenant à l'intervalle [0,1]. CR est le coefficient de croisement qui appartient à l'intervalle [0,1] et est déterminé par l'utilisateur. $randi$ est un indice choisi aléatoirement dans l'ensemble {1,2,…,N}, et G c'est le nombre de génération.

II.4.3.3 Sélection

Pour décider quel vecteur, parmi u_i ou x_i, doit être choisi, on doit comparer les valeurs de la fonction objectif f de ces deux vecteurs. En effet, on garde le vecteur ayant la plus petite valeur en cas de minimisation. Le nouveau vecteur x_i est choisi selon l'expression suivante :

$$\boldsymbol{x_i = \begin{cases} u_i & si\ f(u_i) < f(x_i) \\ x_i & sinon \end{cases}} \quad \text{(II. 13)}$$

Ces quatre étapes sont réitérées jusqu'à ce qu'un nombre maximal d'itérations soit atteint ou jusqu'à ce qu'une solution soit trouvée.

L'algorithme DE est l'un des algorithmes évolutionnaires les plus utilisés, vu sa simplicité et ses capacités de recherche globale. Il a été largement employé pour résoudre les problèmes d'identification de systèmes flous. Citons à titre d'exemple, l'étude rapportée dans [23] qui examine les performances de l'algorithme DE sur la conception des modèles flous. Aussi, l'algorithme DE a été employé dans [**33**] pour l'estimation des paramètres des ensembles flous dans une procédure de construction de systèmes d'inférence floue (FIS) basée sur la méthode de coalescence soustractive.

La Tableau II. 3 résume les principales étapes de l'algorithme d'optimisation à évolution différentielle.

1. **Initialiser** la population aléatoirement
2. **Pour** tous les vecteurs x_i faire:
   Choisir r_1, r_2, r_3, r_4 et r_5 tels que $i \neq r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5$
   Calculer le vecteur mutant v_i
   Créer le vecteur d'essai u_j en utilisation l'équation (II.12)
   Si f(u_j) < f(x_i) remplacer x_i par u_i
3. **Fin pour**
4. génération = génération + 1
5. **Jusqu'à** ce que le critère d'arrêt est atteint

Tableau II. 3 Algorithme d'optimisation à évolution différentielle.

II.4.4 Algorithme de colonies de fourmis artificielles

L'algorithme d'optimisation par colonies de fourmis artificielles (Ant Colony Optimization, ACO) est un algorithme qui a été inspiré du comportement des fourmis. En 1990, Marco Dorigo a proposé un algorithme pour la recherche de chemins optimaux dans un graphe. Ce premier algorithme s'inspire du comportement des fourmis recherchant un chemin entre leur colonie et une source de nourriture [**34**].

L'idée originale provient de l'observation de l'exploitation des ressources alimentaires chez les fourmis. En effet, celles-ci, bien qu'ayant individuellement des capacités cognitives limitées, sont capables collectivement de trouver le chemin le plus court entre une source de nourriture et leur nid.

Le comportement des fourmis lors du fourragement peut être brièvement décrit comme suit [**35**]:

- Une première fourmi trouve la source de nourriture, via un chemin quelconque, puis revient au nid en laissant derrière elle une piste de phéromone.
- Les fourmis empruntent indifféremment les chemins possibles, mais le renforcement de la piste rend plus attractif le chemin le plus court.
- Les fourmis empruntent le chemin le plus court, les portions longues des autres chemins perdent leur piste de phéromones.

Dans l'algorithme ACO [**35**], une colonie d'une taille finie de fourmis artificielles, désignés en tant que P_s, est créée. Chaque fourmi est une solution. Pendant la construction de sa propre solution, chaque fourmi recueille de l'information sur la base des caractéristiques du problème d'optimisation à résoudre et ses propres performances. La mesure de la performance est basée sur une fonction de qualité (d'adéquation ou de performance) que l'on note f. L'algorithme ACO s'est appliqué initialement à des problèmes combinatoires discrets, où les solutions du problème d'optimisation peuvent être exprimées graphiquement par un chemin. Parmi tous les chemins possibles, l'algorithme ACO cherche à localiser l'un avec le minimum coût. Les informations recueillies par les fourmis pendant le processus de recherche sont stockées dans les traces de phéromone τ associées à la connexion de tous les bords. Pour déterminer la solution optimale, les fourmis partagent ces informations via les essais de phéromones. Les bords peuvent aussi avoir une valeur heuristique associée η. Une fois que toutes les fourmis ont terminé leurs visites (c'est-à-dire, à la fin de chaque itération), l'algorithme ACO actualise les traces de phéromone en utilisant toutes les solutions produites par la colonie de fourmis.

La structure algorithmique du modèle ACO de base est présentée dans la Tableau II. 4.

1. **Initialiser** les paramètres et les pistes de phéromones
2. **Tant que** (condition d'arrêt n'est pas satisfaite)
3. **Pour** i = 1 à P_s
   Générer des chemins de fourmis pour la construction de la solution
   Evaluer la fonction d'adéquation f
4. **Fin pour**
5. Mettre à jour les traces de phéromone
6. Sauvegarder la meilleure solution
7. **Fin tant que**

Tableau II. 4 Algorithme de colonie de fourmis artificielles (ACO)

Pour l'actualisation des traces de phéromone, de nombreux algorithmes ACO ont été proposées. Les exemples les plus courants sont l'algorithme ACS (Ant Colony System) [25] et l'algorithme MMAS (Max-Min Ant System) [27]. La conception de systèmes flous à l'aide de l'algorithme ACO a fait l'objet d'un certain nombre de travaux [28, 29]. Dans [**36**], l'algorithme ACO a été utilisé pour l'identification, de modèles flous à partir de données d'apprentissage collectées hors ligne. Dans ce même contexte, l'étude rapportée dans [29] présente une approche de clustering basée sur les colonies de fourmis (OCCA).

II.5 Conclusion

Ce chapitre traite plus particulièrement le problème de synthèse ou de conception de systèmes flous par méthodes d'optimisation. Comme il vient d'être souligné, l'identification floue des systèmes peut être formulée comme un problème d'optimisation structurelle et paramétrique. En présence de données d'apprentissage, un modèle de système flou peut être généré via une procédure d'optimisation appropriée qui permet de déterminer l'ensemble de ses paramètres ainsi que sa structure. Ainsi posé, le problème d'identification de systèmes flous peut être résolu à l'aide d'approches méta-heuristiques qui s'avèrent d'une utilité majeure vue la complexité du problème d'optimisation posé. A cet effet, certains algorithmes sont examinés dans ce chapitre, tout en illustrant brièvement quelques travaux rapportant leur utilisation pour le développement de modèles de représentation flous.

Chapitre III

Auto-génération des systèmes flous par colonies d'abeilles artificielles

III.1 Introduction

Comme il vient d'être montré dans le chapitre précédent, la synthèse des systèmes flous à partir de données peut être formulée comme un problème d'optimisation complexe nécessitant des outils bien adaptés pour le résoudre. Dans ce contexte, des approches méta-heuristiques bien diverses basées sur le calcul évolutionnaire ou encore l'intelligence en essaim ont été établies et appliquées à différents problèmes de modélisation floue. L'émergence des techniques d'optimisation intelligente, en particulier celles inspirées du comportement collectif d'agents (fourmis, abeilles, oiseaux, poissons,…) formant un essaim, a été d'un intérêt majeur pour une multitude de problèmes d'optimisation complexe. L'essaim d'abeilles est l'objet d'étude de ce chapitre où une nouvelle stratégie pour l'auto-génération de modèles flous à partir de données est présentée. Formulée comme un problème d'optimisation complexe, la démarche de synthèse que nous proposons exploite les performances tant prouvées de l'algorithme de colonies d'abeilles artificielles (ABC) en vue de concevoir une nouvelle plateforme algorithmique servant à l'extraction automatique et simultanée de la base des règles et des paramètres de systèmes flous.

III.2 L'essaim d'abeilles

Un essaim d'agents de nature quelconque désigne plus généralement toute collection d'espèces vivantes (insectes, oiseaux, cellules,..) en interaction. L'exemple d'un essaim d'abeilles est visiblement cette formation de groupes ou de colonies que ces insectes esquissent autour de leur ruche. Deux concepts fondamentaux sont à l'origine de la caractérisation d'un comportement intelligent en essaim : le concept d''auto-organisation et le concept de la répartition de tâches [**37**].Ces concepts sont bien évidemment déterminants

quant à la vie d'une colonie d'abeilles et constituent en fait la source de développement de toute sorte de mécanismes de résolution à caractère distribué.

L'auto-organisation dans un essaim d'abeilles peut être définie comme un ensemble de mécanismes dynamiques qui conduisent à construire des colonies (modèles, structures) à travers une série d'interactions entre ses différentes composantes. Ces mécanismes établissent des règles de base qui régissent le comportement collectif des abeilles de la même colonie. Les règles établies font en sorte que les interactions sont exécutées sur la base des informations recueillies localement (niveau bas), sans aucun rapport avec le modèle global de l'essaim (niveau haut). A l'intérieur d'un essaim, des agents (abeilles) "spécialisés" accomplissent des tâches différentes et de manière simultanée. La répartition des tâches au sein de la colonie trouve son efficacité dans le traitement distribué de ces tâches, qui est loin d'être nécessairement séquentiel. La performance collective de l'essaim d'abeilles est exclusivement basée sur ces deux concepts.

III.2.1 Composantes fondamentales de l'essaim

Le modèle minimal de la sélection du fourrage qui conduit à l'émergence de l'intelligence collective des essaims d'abeilles se compose de trois éléments essentiels: les sources de nourriture, les abeilles dites « Employed » et les abeilles dites « Onlooker ». Ce modèle caractérise deux principaux modes de comportement: le recrutement à une source de nourriture (ou de nectar) et l'abandon d'une source.

a) **Source de nourriture :** La qualité d'une source de nourriture dépend de plusieurs facteurs tels que la proximité du site, sa richesse en termes de teneur en nectar et la facilité d'extraction de cette nourriture. Par souci de simplicité, le modèle comportemental peut utiliser un descripteur unique (fonction d'adéquation) pour évaluer la rentabilité d'une source de nourriture.
b) **Abeilles « Employed » :** Elles sont associées à une source de nourriture particulière en qualité d'exploitants effectifs du site ou bien employées pour l'exploitation. Elles ont également pour tâche de véhiculer et partager des informations sur cette source particulière à savoir son emplacement par rapport à la ruche (distance, direction,…), sa rentabilité, etc.
c) **Abeilles « Onlooker » :** Elles sont constamment à la recherche d'une source de nourriture à exploiter. Il existe deux types d'abeilles Onlooker: les scouts qui prospectent l'environnement avoisinant la ruche à la recherche de nouvelles sources de nourriture, et

les assistantes qui restent en attente dans la ruche en vue de recevoir les informations recueillies par les abeilles Employed. Le nombre moyen de scouts est d'environ 5-10% sur la population globale [**37**].

III.2.2 Modèle comportemental à essaim d'abeilles

L'échange d'informations entre les différentes catégories d'abeilles décrites ci-dessus constitue la phase la plus importante dans la formation de la connaissance collective. Cet échange qui vise à partager des éléments clefs relatifs à la source de nourriture à exploiter s'effectue à travers des mécanismes naturels propres aux abeilles. Parmi ces mécanismes, on distingue un phénomène très particulier identifié sous le nom de « *Waggle Dance* ». La Figure III. 1 : Illustration d'une abeille en **« Waggle** dance » illustre schématiquement ce phénomène.

Figure III. 1 : Illustration d'une abeille en « Waggle dance » [**37**].

Dès que les informations sur toutes les sources sont disponibles aux abeilles Onlooker à travers l'analyse des différentes danses, la décision concernant l'exploitation du site le plus profitable est rapidement prise. Afin de mieux appréhender le comportement de base des abeilles, on propose d'examiner la Figure III. 2. Considérons les deux sites de nourriture A et B. Initialement, une abeille de la colonie dépourvue de tâches spécifiques peut être appelée à accomplir l'un des rôles suivants :

- Elle peut être une scout et commence à prospecter spontanément le voisinage de la ruche à la recherche d'une éventuelle source de nourriture. Ce comportement peut être à l'origine de certaines motivations internes ou externes (représentées par des flèches orientées "S").
- Elle peut être appelée pour exploiter une source de nourriture "R", et ce après avoir reçue les informations correspondantes.

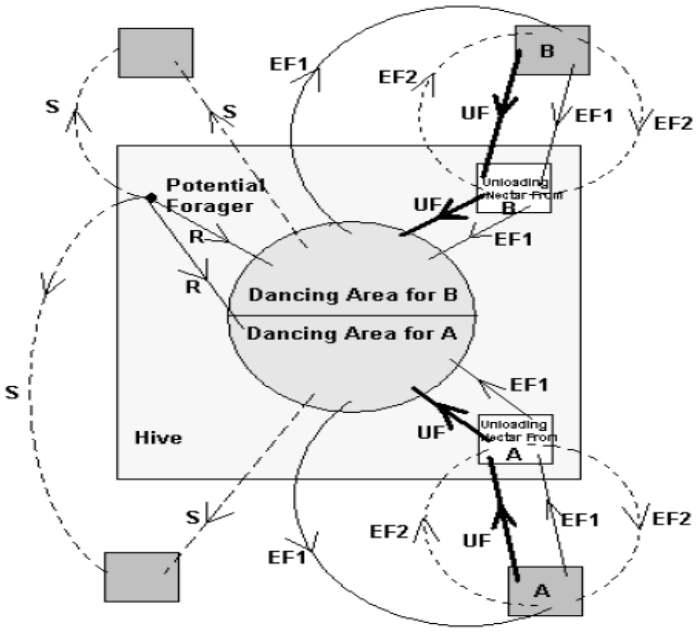


Figure III. 2 : Le comportement d'une abeille lors du fourragement [**37**]

Après avoir localisé la source de nourriture, l'abeille utilise sa propre capacité d'apprentissage pour mémoriser son emplacement, ensuite passe immédiatement à la phase d'exploitation. Ainsi, l'abeille en question deviendra une abeille "Employed" et pourra se manifester selon l'un des cas de figures décrits ci-après :

- Elle n'aura aucune tâche spécifique après l'abandon de la source de nourriture (UF).
- Elle transmettra les informations sur la source découverte aux autres abeilles de la colonie et repartira pour l'exploitation de la même source (EF1).
- Elle continue à exploiter la source de nourriture qu'elle a découverte sans revenir à la ruche (EF2).

Il est important de noter que les expériences ont confirmé que seulement une partie des abeilles de la colonie s'engagent dans la prospection de nourriture et non toutes les abeilles simultanément [**37**].

III.3 L'algorithme de colonies d'abeilles artificielles

En 2005, Karaboga et al [**37**] ont développé un nouvel algorithme d'optimisation dit « l'algorithme ABC (Artificial Bee Colony Algorithm) ». L'algorithme, qui a été initialement appliqué à des problèmes d'optimisation de fonctions numériques, s'inspire du modèle comportemental des abeilles lors du fourragement.

III.3.1 Structure algorithmique

Dans l'algorithme de colonies d'abeilles artificielles (ABC), la population d'individus ou d'agents est assimilée à une colonie d'abeilles au sein de laquelle on distingue trois catégories d'abeilles : les abeilles "Employed", les "Onlooker" et les "Scouts". La colonie est équitablement répartie entre abeilles Employed et abeilles Onlooker. Pour chaque source de nourriture, une seule abeille est affectée. En d'autres termes, le nombre d'abeilles Employed est égal au nombre de sources de nourriture autour de la ruche. L'abeille Employed dont la source de nourriture est épuisée devient une abeille scoute. La Tableau III.1 illustre sommairement les différentes phases de l'algorithme ABC.

1. Initialisation
2. **REPETER**
   - Phase Employed : Envoyer les abeilles Employed aux sources de nourriture et actualiser chaque solution
   - Phase Onlooker : Réaliser une sélection basée sur la fonction d'adéquation puis actualiser chaque solution.
   - Phase Scout : Repérer la solution la plus inactive (épuisement de site), et la remplacer par une nouvelle solution générée aléatoirement.
3. . **JUSQU'A** (satisfaire les conditions d'arrêt)

Tableau III.1 Structure de l'algorithme ABC.

Comme représenté en Tableau III.1, chaque cycle de recherche se fond sur trois étapes: la phase Employed, la phase Onlooker et la phase Scout. Dans cette structure, une position donnée de la source de nourriture représente une solution potentielle au problème d'optimisation. La quantité de nectar d'une source de nourriture correspond à la qualité de la solution représentée par cette source de nourriture. Les Onlookers sont affectées aux sources de nourriture en se basant sur un processus de sélection dite "gourmande" qui utilise un mécanisme probabiliste. La teneur en nectar d'une source est élevée d'autant que sa probabilité de sélection par les Onlookers augmente. Chaque colonie possède des scouts qui sont les explorateurs de la colonie. Les scouts n'ont pas d'orientation particulière, elles se contentent de prospecter le voisinage de la ruche à la recherche d'éventuelles sources.

Dans l'algorithme ABC une abeille "Employed" est remplacée par une scout lorsque la solution représentée par cette abeille n'est pas améliorée. Cette situation fait référence à l'épuisement de la source de nourriture qui est contrôlée par un paramètre de contrôle appelé "limit". En d'autres termes, au cas où une solution représentant une source de nourriture donnée n'est pas améliorée après un nombre prédéterminé d'essais ou d'itérations, cette source de nourriture sera alors abandonnée par l'abeille Employed qui deviendra par la suite une scout. Le nombre d'essais nécessaire à l'abandon d'une source de nourriture est égal à la valeur de seuil "limit" qui est un paramètre de contrôle important de l'algorithme ABC.

III.3.2 Modèle de base de l'algorithme ABC

Dans l'algorithme ABC, la population est constituée d'un ensemble de solutions possibles x_i représentées par les positions des sources de nourriture dont la teneur en nectar correspond à la qualité de la solution associée. Soit $x_i = \{x_{i1}, x_{i2}, \dots, x_{iD},\}$, *i*=1,2,…, *SN,* la source de nourriture *i* dans la population, où *SN* désigne la taille de la population et D la dimension du problème d'optimisation. La population initiale de *SN* solutions est générée aléatoirement à partir de l'équation suivante :

$$\boldsymbol{x_{ij} = x_{min}^{j} + rand(0,1)\left(x_{max}^{j} - x_{min}^{j}\right)} \quad \text{(III. 1)}$$

où x_{min}^{j} et x_{max}^{j} sont les limites inférieures et supérieures du j^{éme} paramètres de la solution i.

Une abeille "Emplyed" effectue par la suite une modification de la position de la source de nourriture qu'elle mémorise et trouve une source avoisinante selon l'équation (III.2) :

$$\boldsymbol{v_{ij} = x_{ij} + \emptyset_{ij}\left(x_{ij} - x_{kj}\right)} \quad \text{(III. 2)}$$

où $\emptyset_{ij}$ est un nombre aléatoire compris entre [-1, 1], v_i est une nouvelle position d'une source de nourriture potentielle, x_i est la position de la source de nourriture actuelle, x_k est une position de source de nourriture voisine avec $k = 1,2, \dots, SN\ et\ k \neq i$ pour tout j∈{1,2,...,D}. Une fois la position v_i évaluée, elle est comparée à x_i. Un processus de sélection dite "gourmande" est appliqué : lorsque la teneur en nectar de la nouvelle position est plus élevée que celle de la précédente, l'abeille mémorise la nouvelle position et abandonne l'ancienne. Dans le cas contraire, elle préserve la position de la précédente source. Après que toutes les abeilles Employed complètent le processus de recherche, elles reviennent à la ruche et partagent toutes ces informations (position et teneur en nectar) avec les abeilles Onlooker en

manifestant une danse particulière dite " Waggle dance". Le choix d'une source de nourriture est réalisé de manière probabiliste par l'évaluation de la probabilité p_i, qui dépend de la teneur en nectar de la source de nourriture *i* :

$$\boldsymbol{p_i = \frac{fit_i}{\sum_{l=1}^{SN} fit_l}} \quad \text{(III. 3)}$$

où fit_i est la valeur de la fonction d'adéquation de la solution *i* qui est proportionnelle à la teneur en nectar de la source de nourriture *i.* Elle peut être calculée par :

$$\boldsymbol{fit_i = \frac{1}{1+f_i}} \quad \text{(III. 4)}$$

où f_i est la valeur de la fonction objectif de la solution associée *i*.

Dans l'algorithme ABC de base, le processus de sélection est généralement basé sur le mécanisme de « roulette and wheel » [**37**]. D'autres mécanismes peuvent être cependant utilisés tels que la sélection par rang, la sélection steady-state, la sélection par tournoi, la sélection par élitisme, et la sélection uniforme [**32**].

Lorsque le nectar d'une source de nourriture est épuisé par les abeilles Employed et Onlookers, l'abeille Employed de cette même source devient une abeille scout et détermine une nouvelle source de nourriture aléatoirement par l'équation (III.1). Afin de décider si une source de nourriture est abandonnée ou pas, le paramètre de contrôle "limit" est utilisé. A la fin des itérations, des valeurs limites sont comparées avec le nombre d'essais non améliorés de chaque solution. Il y'a un compteur pour chaque solution qui est incrémenté par la valeur "1" après chaque échec ou est mis à zéro à la suite d'un essai mené avec succès, soit par une abeille Employed ou une abeille Onlooker. Les principales étapes de l'algorithme ABC de base sont explicitées en Tableau III 2, l'organigramme correspondant est donné en Figure III.

.

1. Générer la population initiale $X_i, i = 1, \dots, SN$ par l'équation (III. 1)
2. Évaluer la population initiale par l'équation (III. 4)
3. Cycle = 1
4. **Répéter**
5. Pour chaque abeille Employed{
    - Produire une nouvelle solution v_i par l'équation (III. 2)
    - Calculer la valeur de la fonction d'adéquation fit_i par l'équation (III. 4)
    - Appliquer la sélection gourmande}
6. Calculer la probabilité p_i associée à la solution x_i par l'équation (III. 3)
7. Pour chaque abeille Onlooker{
    - Sélectionner une solution x_i en fonction de p_i
    - Produire une nouvelle solution v_i
    - Calculer la valeur de la fonction d'adéquation fit_i
    - Appliquer la sélection gourmande}
8. **S**'il y'a une solution abandonnée
    - **Alors** remplacer la avec une nouvelle solution qui sera produite aléatoirement par l'équation (III. 1)
9. Mémoriser la meilleure solution trouvée
10. $cycle = cycle + 1$
11. **Jusqu'à** $cycle = cycle_{max}$

Tableau III. 2 L'algorithme de colonies d'abeilles artificielles (ABC)

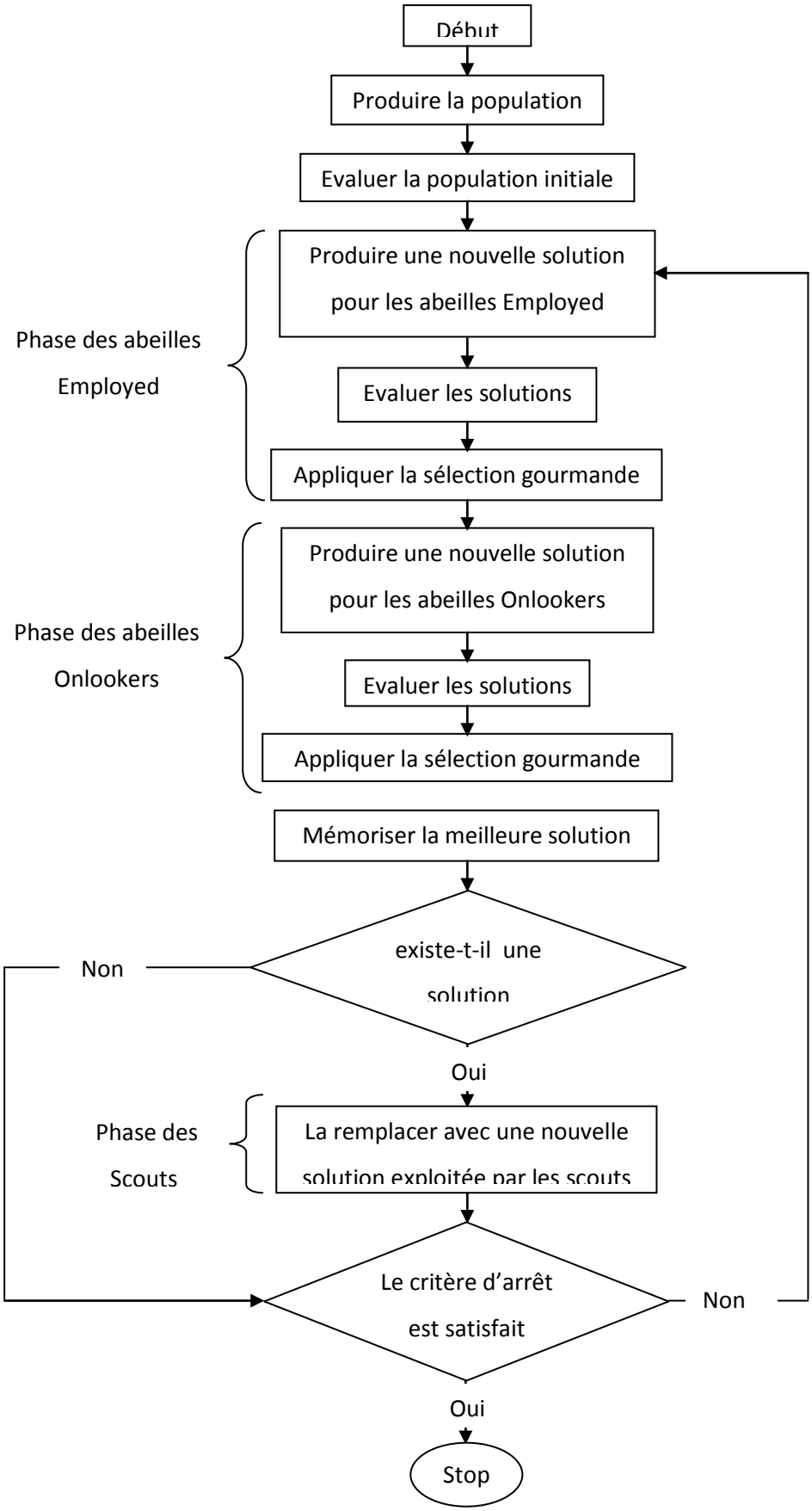


Figure III. 3 : Organigramme du modèle de base de l'aglorithme ABC

III.4 Algorithme ABC guidé par la meilleure solution globale

Les capacités d'exploration et d'exploitation sont les caractéristiques clefs des algorithmes méta-heuristiques. L'exploration se réfère à la capacité de l'algorithme à prospecter les différentes régions inconnues dans l'espace de recherche, alors que l'exploitation définit la capacité d'utiliser les meilleures solutions trouvées pour déterminer les solutions optimales [**38**]. Ces deux caractéristiques fondamentales résultent d'un compromis à résoudre pour une meilleure approche d'optimisation.

Inspiré par l'algorithme PSO, l'algorithme ABC guidé par la solution globale (g-ABC) exploite la meilleure solution globale (Gbest) pour guider la recherche vers d'autres solutions potentielles. Ainsi, l'équation de recherche de solutions décrite par (III. 2) a été modifiée dans [**39**] pour établir une nouvelle expression donnée par :

$$\boldsymbol{v_{ij}} = \boldsymbol{x_{ij}} + \emptyset_{\boldsymbol{ij}}\left(\boldsymbol{x_{ij}} - \boldsymbol{x_{kj}}\right) + \boldsymbol{\psi_{ij}}(\mathbf{y_j} - \boldsymbol{x_{kj}}) \qquad \text{(III. 5)}$$

où y_j est l'élément j de la meilleure solution globale, ψ_{ij} est un nombre aléatoire uniforme dans [0 C], où C est une constante positive ou nulle. Selon l'équation (III. 5), le terme Gbest représenté par y_j sert à conduire la nouvelle solution vers la meilleure solution globale, ce qui a pour effet d'améliorer l'exploitation dans l'algorithme ABC. A noter que le paramètre C dans l'équation (III. 5) est déterminant, sa valeur influe considérablement sur la qualité des solutions.

III.5 Une nouvelle méthodologie de conception des systèmes flous à base de l'algorithme ABC

La conception des systèmes flous peut être formulée comme un problème d'identification que l'on peut envisager à résoudre en fonction de trois critères distincts : la précision de la modélisation, la compacité du modèle et l'interprétabilité du modèle. La question de précision est bien entendu fondamentale et essentielle, car l'objectif principal de la modélisation floue consiste à approximer une fonction non linéaire d'entrée-sortie quelconque par la détermination d'une structure de modèle optimale de telle sorte qu'une correspondance acceptable entre la sortie du modèle et la sortie du processus est réalisée. La précision du modèle identifié doit être évaluée par rapport à des données d'apprentissage et de test, et dans la plupart des cas, l'indice de performance à optimiser est exprimé en termes de la précision du modèle. . Bien que la compacité et d'interprétabilité du modèle soient aussi importantes

que la qualité d'approximation obtenue, le concepteur aura à formuler le problème de modélisation floue à résoudre en fonction de l'application étudiée.

III.5.1 Codage du modèle flou TS

Tout d'abord, rappelons la forme générale d'une règle floue *i* dans une représentation par modèle TS :

$$\boldsymbol{R^i}: \boldsymbol{IF\ x_1 is\ A_1^i and\ x_2\ is\ A_2^i and \dots and\ x_n\ is\ A_n^i\ THEN\ y^i = a_0^i + a_1^i x_1 + \cdots + a_n^i x_n}$$

(III. 6)

où a^i est le vecteur des paramètres associé à la règle *i*, avec $= 1 \dots N_R$, N_R est le nombre de règles, $x = (x_1, x_2, \dots, x_n)$ est le vecteur d'entrée , y^i désigne la sortie du modèle et $A_j^i, j = 1, \dots, n,$ sont les ensembles flous de prémisse.

Afin de pouvoir exploiter l'algorithme ABC pour le développement de la méthodologie d'identification des modèles flous TS, il est essentiel de définir un schéma de codage approprié servant à la représentation des modèles TS dans la population d'agents. Il existe cependant plusieurs schémas de codage qui ont été proposés dans la littérature, chaque mécanisme est en fait lié à la stratégie d'optimisation utilisée.

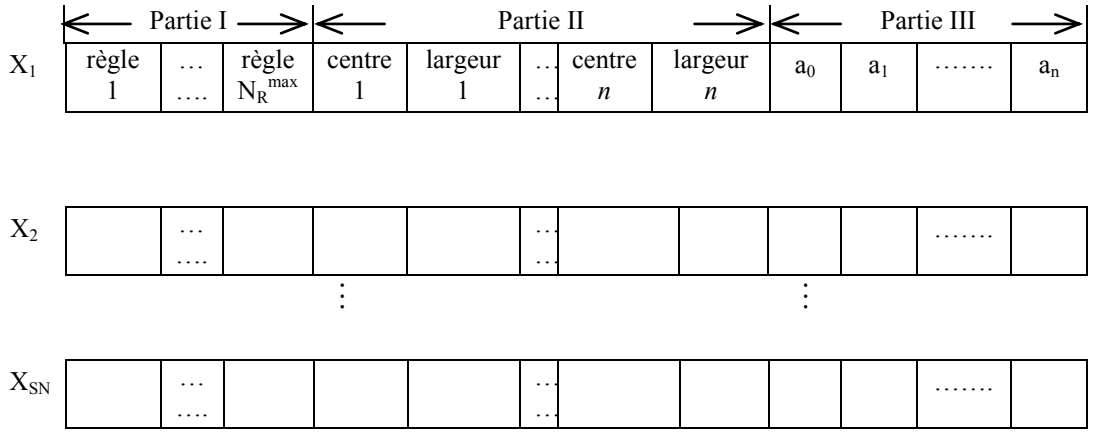


Figure III. 4 : Codage d'un modèle flou TS dans la méthodologie proposée.

Notre méthodologie est basée sur l'algorithme d'optimisation ABC, un modèle flou TS solution potentielle du problème d'optimisation posé représente alors « une source de nourriture », qui sera visitée par une abeille lors du fourragement. Le mécanisme de codage du modèle flou est représenté en Figure III.4, ce dernier est établi de telle sorte que l'ensemble des paramètres du modèle codé soient entièrement spécifiés. Un modèle flou est

dit entièrement spécifié si l'ensemble des paramètres des prémisses et des conclusions des règles, ainsi que le nombre de règles sont déterminés. Le schéma de codage de la Figure III. emploie une « source de nourriture » qui se compose de trois parties.

La première partie correspond à la sélection et l'optimisation du nombre de règles floues, la deuxième partie est consacrée à l'optimisation des ensembles flous de l'antécédent des règles et la troisième partie concerne l'optimisation des paramètres de la conséquence des règles. Comme on peut le constater, l'ensemble des paramètres du modèle TS sont codés réalisant une évolution simultanée guidée par le modèle d'optimisation ABC.

Plus précisément, une source de nourriture représente un modèle flou potentiel qui est exploité par une seule abeille Employed. Dans la première partie du schéma de codage (Partie I), un nombre réel est affecté pour chaque position de la source de nourriture. L'évolution de cette partie indiquera la création d'une nouvelle règle ou la suppression d'une règle existante. Ainsi, pour établir le mécanisme d'inférence floue décrit par (III. 10) les règles floues sont construites en fonction de la valeur affectée à la position de la source de nourriture.

La partie II du schéma de codage regroupe les paramètres des ensembles flous de la condition des règles, à savoir les centres c_j^i et les écart-types σ_j^i. Les paramètres des conséquences sont codés dans la Partie III. Ainsi établi, la structure et les paramètres du modèle flou TS seront identifiés simultanément.

Selon le schéma de codage adopté, la longueur d'une source de nourriture donnée est fixe, mais le nombre de règles floues induites n'est pas connu a priori. Seul le nombre maximal de règles $N_R^{\max}$ est fixé, ce qui signifie que la longueur de la première partie (Partie I) est $L_R = N_R^{\max}$. De même, puisque nous avons au total n variables linguistiques d'entrée, le nombre de positions de nourriture représentant les paramètres des prémisses des règles est alors : $L_P = 2 \times n$, et le nombre de positions de nourriture de la troisième partie est $L_C = n + 1$. Par conséquent, la longueur totale d'une source de nourriture représentant un modèle flou TS devient : $L_F = (3n+2) \times N_R^{\max}$, que l'on convient de représenter selon la forme suivante:

$$\begin{aligned} F_S = [&r_1,\ldots,r_{N_R^{\max}}, c_1^1,\sigma_1^1,\ldots,c_1^n,\sigma_1^n,a_1^0,a_1^1,\ldots,a_1^n,\ldots, \\ &c_{N_R^{\max}}^1,\sigma_{N_R^{\max}}^1,\ldots,c_{N_R^{\max}}^n,\sigma_{N_R^{\max}}^n,a_{N_R^{\max}}^0,a_{N_R^{\max}}^1,\ldots,a_{N_R^{\max}}^n] \end{aligned} \quad \text{(III. 7)}$$

Une abeille Employed produit dans sa mémoire une modification dans les paramètres du modèle flou qu'elle visitera, trouvera ensuite une source de nourriture voisine, puis évaluera la qualité du modèle flou déterminé en termes d'une mesure d'adéquation exprimée par l'erreur quadratique moyenne (MSE) :

$$MSE = \frac{1}{N}\sum_{k=1}^{N}[y_d(k) - \hat{y}(k)]^2 \qquad \text{(III. 8)}$$

avec N est la taille de la base de données d'apprentissages, $y_d(k)$ est la sortie désirée, et $\hat{y}(k)$ la sortie du modèle.

La modification de la source de nourriture opérée par une abeille Employed altère la structure du modèle flou visité. A ce niveau, les formes des fonctions d'appartenance de la prémisse des règles sont ajustées, et par conséquent une nouvelle sortie du modèle est induite. Ces modifications seront partagées avec les abeilles Onlookers qui, à leur tour, auront à choisir un modèle flou avec une probabilité exprimée en termes de la qualité du modèle évalué. Une abeille Onlooker produit une modification de la structure et les paramètres du modèle flou sélectionné, puis effectue une sélection « gourmande » entre l'ancienne et la nouvelle solution. Cette phase est suivie par l'étape scout qui contrôlera les positions de nourriture à abandonner après épuisement. La procédure d'optimisation qu'on vient de décrire est répétée pour un nombre prédéfini d'évaluations.

III.5.2 Structure algorithmique de la méthodologie proposée

Comme il vient d'être souligné, la méthodologie de modélisation floue proposée dans ce travail vise à concevoir des modèles flous TS par la génération automatique de la base des règles floues à partir de données numériques. Ses principales étapes sont résumées dans l'algorithme de la Tableau III.3.

III.6 Hybridation de la méthodologie proposée avec la méthode des moindres carrés simple

L'hybridation de l'approche d'identification floue présentée ci-dessus avec la méthode des moindres carrés vise à réduire le nombre de paramètres à optimiser. Cette version que l'on désigne par « approche ABC-MCS » est fondée sur les mêmes étapes que l'approche de base, sauf que ce sont les paramètres des conclusions des règles qui seront estimés par la méthode MCS.

1. Coder tous les paramètres du modèle flou à l'aide du schéma de codage proposé. Le nombre maximal de règles et les entrées sont spécifiés.
2. Générer aléatoirement la population initiale : un modèle flou (solution potentielle) représente une source de nourriture.
3. Evaluer la qualité de chaque modèle flou potentiel {
   - Réaliser la sélection des règles floues (Partie I du schéma de codage)
   - Appliquer le mécanisme d'inférence floue en utilisant l'équation (III. 10)
   - Calculer l'erreur d'approximation (MSE) en utilisant l'équation (III. 8)

   }
4. **Répéter**
   - Produire une nouvelle solution pour les abeilles Employed.
   - Calculer la sortie du modèle flou.
   - Calculer la fonction objectif (MSE).
   - Appliquer la sélection gourmande.
   - Calculer la valeur de probabilité p_i pour la solution courante.
   - Produire une nouvelle solution pour les abeilles Onlookers
   - Calculer la sortie du modèle flou.
   - Calculer la fonction objectif (MSE).
   - **S**'il y'a une solution abandonnée **Alors** la remplacer avec une nouvelle solution qui sera produite aléatoirement avec une scout.
   - Mémoriser la meilleure solution trouvée
   - $cycle = cycle + 1$
5. **Jusqu'à** $cycle = cycle_{max}$

Tableau III. 3 Structure algorithmique de la méthodologie de modélisation floue proposée

Considérons le modèle T-S suivant :

$$\boldsymbol{if\ x_1\ est\ A_i^1\ et\ \dots et\ x_n\ est\ A_i^n\ then\ y_i = a_i^0 + a_i^1 x_1 + \cdots + a_i^n x_n} \qquad \text{(III. 9)}$$

où $a_i^0, a_i^1, \dots, a_i^n$ sont les paramètres de la conséquence, $= 1 \dots N_R$, N_R est le nombre des règles, $x = (x_1, \dots, x_n)$ est le vecteur d'entrée, y^i est la sortie, A_j^i, $j = 1, \dots, n$ sont les ensembles flous de prémisse.

La sortie résultante de l'ensemble des règles est donnée par la moyenne des sorties individuelles pondérées par le degré d'activation des règle, soit :

$$\boldsymbol{y} = \frac{\sum_{i=1}^{N_R} \boldsymbol{w^i y_i}}{\sum_{i=1}^{N_R} \boldsymbol{w^i}} \quad \text{(III. 10)}$$

Le degré d'activation $\boldsymbol{w^i}$ de chaque règle i est calculé par l'équation :

$$\boldsymbol{w^i} = \prod_{\mathbf{j=1}}^{\boldsymbol{n}} \boldsymbol{\mu}\,(\boldsymbol{A_j^i}) \quad \text{(III. 11)}$$

Posons :

$$[\boldsymbol{\Omega_i}] = \begin{bmatrix} \boldsymbol{\mu_{i1}} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mu_{i2}} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{\mu_{in}} \end{bmatrix}, \text{ avec a } \boldsymbol{\mu_{ij}} = \boldsymbol{exp}\left(-\left(\frac{x_j - c_j^i}{\sigma_j^i}\right)^2\right) \quad \text{(III. 12)}$$

Avec n le nombre d'entrées.

Après la reformulation de la conséquence de l'équation (III. 9) et (III. 10) sous la forme matricielle on obtient :

$$[\boldsymbol{y_i}] = [\boldsymbol{X_e}] * [\boldsymbol{\Phi_i}] \quad \text{(III. 13)}$$

avec $\boldsymbol{\Phi_i} = [a_i^0, a_i^1, \dots, a_i^n]^T$ est le vecteur qui représente les paramètres da la conséquence, et $[X_e]$ est définit comme suit : $[X_e] = [1\ x_1\ x_2\ \dots\ x_n]$.

Les paramètres des modèles locaux sont estimés séparément, en utilisant un ensemble de critères d'estimation locale [**4**], soit :

$$\min_{\boldsymbol{\Phi_i}} \|\boldsymbol{y} - \boldsymbol{y_i}\|^2 \boldsymbol{\Omega_i} \quad \text{(III. 14)}$$

Si on remplace y_i par son expression dans (III. 14), on obtient :

$$\min_{\boldsymbol{\Phi_i}} \|\boldsymbol{y} - \boldsymbol{X_e \Phi_i}\|^2 \boldsymbol{\Omega_i}$$

$$\min_{\boldsymbol{\Phi_i}} \left\|\sqrt{\boldsymbol{\Omega_i}}\boldsymbol{y} - \sqrt{\boldsymbol{\Omega_i}}\boldsymbol{X_e \Phi_i}\right\|^2 \quad \text{(III. 15)}$$

L'estimation des paramètres des conséquences des règles par la méthode des moindres carrés pondérée est alors donnée par:

$$\sqrt{\boldsymbol{\Omega_i}}\boldsymbol{y} = \sqrt{\boldsymbol{\Omega_i}}\boldsymbol{X_e \Phi_i} \quad \text{(III. 16)}$$

1. Coder tous les paramètres à l'aide du schéma de codage proposé. Le nombre maximal de règles et les entrées sont spécifiés.
2. Générer aléatoirement la population initiale : un modèle flou (solution potentielle) représente une source de nourriture.
3. Estimer les paramètres des conséquences des règles par la méthode des moindres carrées en utilisant l'Equation (III.18).
4. Calculer la sortie du modèle flou en utilisant l'Equation (III.10).
5. Calculer la fonction objectif MSE en utilisant l'équation (III.8). .
6. Enregistrer chaque fonction objectif, et associé à fonction objectif la valeur d'adéquation appropriée.
7. **Répéter**
   - Produire une nouvelle solution pour les abeilles Employed.
   - Estimer les paramètres des conséquences des règles à l'aide de l'Equation (III.18)
   - Calculer la sortie du modèle flou à l'aide de l'équation (III.10).
   - Calculer la fonction objectif (MSE).
   - Appliquer la sélection gourmande.
   - Calculer la valeur de probabilité p_i pour la solution x_i.
   - Produire une nouvelle solution pour les abeilles Onlookers depuis la solution x_i sélectionnée en se basant sur la probabilité p_i.
   - Estimer les paramètres des conséquences des règles à l'aide de l'équation (III.18)
   - Calculer la sortie du modèle flou à l'aide de l'équation (III.18).
   - Calculer la fonction objectif (MSE).
   - **S**'il y'a une solution abandonnée **Alors** la remplacer avec une nouvelle solution qui sera produite aléatoirement avec une abeille scout.
   - Mémoriser la meilleure solution trouvée
   - $cycle = cycle + 1$
8. **Jusqu'à** $cycle = cycle_{max}$

Tableau III. 4 Structure algorithmique de l'hybridation de la méthodologie proposée avec la méthode des moindres carrées simple.

Dans cette dernière équation, tous les vecteurs sont donnés (pour $[y]$ on utilise la sortie réelle). L'objectif de ces transformations est de calculer les paramètres de la conséquence a_i^j

qui forment la matrice $[\Phi_i]$. Par conséquent, la détermination de la matrice $[\Phi_i]$ conduit à la détermination des paramètres a_i^j des conséquences de règles floues.

Dans l'équation (III. 17), la matrice $[X_e]$ n'est pas une matrice carrée. En multipliant les deux partie de l'égalité par $[X_e]^T\sqrt{\Omega_i}$, on obtient :

$$[\boldsymbol{X_e}]^{\boldsymbol{T}} * [\boldsymbol{\Omega_i}] * [\boldsymbol{X_e}] * [\boldsymbol{\Phi_i}] = [\boldsymbol{X_e}]^{\boldsymbol{T}} * [\boldsymbol{\Omega_i}] * [\boldsymbol{y}] \quad \text{(III. 17)}$$

L'estimateur au sens des moindres carrés du vecteur des paramètres $[\boldsymbol{\Phi_i}]$ s'exprime alors par l'équation :

$$[\boldsymbol{\Phi_i}] = [[\boldsymbol{X_e}]^{\boldsymbol{T}} * [\boldsymbol{\Omega_i}] * [\boldsymbol{X_e}]]^{-\mathbf{1}}[\boldsymbol{X_e}]^{\boldsymbol{T}} * [\boldsymbol{\Omega_i}] * [\boldsymbol{y}] \quad \text{(III. 18)}$$

La structure algorithmique de l'approche d'identification floue basée sur l'hybridation de l'algorithme ABC et la méthode des moindres carrés est donnée en Tableau III. 4.

III.7 Conclusion

Ce chapitre est intégralement consacré à la méthodologie d'identification floue proposée. Il s'agit plus précisément d'une approche d'auto-génération de bases de règles floues à partir de données en utilisant l'algorithme de colonies d'abeilles artificielles. Afin de réduire la dimension du problème d'optimisation, une version hybride est présentée. Elle consiste à utiliser la méthode des moindres carrés simple pour l'estimation des paramètres des conséquences des règles. Ceci permet de réduire le nombre de paramètres à optimiser, ce qui contribue à l'amélioration du temps de calcul. La validation de cette nouvelle approche d'identification de systèmes flous fera l'objet du chapitre suivant.

Chapitre IV

Application à la conception de systèmes à base de règles floues

IV.1 Introduction

Ce chapitre est consacré à l'analyse de la performance et la validation de l'approche d'auto-génération de modèles de systèmes à base de règles floues faisant l'objet de ce mémoire. Pour ce faire, une étude en simulation traitant des problèmes d'identification de systèmes et de synthèse de lois de commande est menée. Plus précisément, on y traite des problèmes d'identification de modèles flous pour les systèmes suivants : un four à gaz dénommé « système Box-Jenkins », un système non linéaire non forcé et un système non linéaire statique. La méthodologie proposée est également validée sur un problème de commande en poursuite d'un système non linéaire, ainsi qu'un problème d'identification floue d'un échangeur de chaleur à courants parallèles. Pour chaque cas d'étude, une description générale du problème à résoudre est tout d'abord fournie. Ensuite, des tests en simulation sont effectués et les résultats obtenus sont comparés à d'autres résultats rapportés dans d'autres études récentes. Pour l'évaluation de la performance de notre approche, nous avions considéré l'algorithme d'optimisation ABC avec ces deux variantes : la variante de base et la variante g-ABC décrites dans le chapitre III. L'approche hybride ABC- MCS employant les deux versions du modèle ABC (ABC de base et g-ABC) décrite dans le paragraphe III.6 est également testée et validée sur quelques cas étudiés.

IV.2 Identification du système de four à gaz « Box–Jenkins »

La base de données d'entre/sortie du four à gaz dit « Box-Jenkins » est fréquemment utilisée dans les travaux de recherche proposant de nouvelles méthodes de modélisation. Le processus est un four à gaz avec une seule entrée u(k) (débit de gaz) et une seule sortie y(k) (concentration de CO_2). L'ensemble de données contient 296 échantillons. Le problème d'identification consiste en la détermination d'une relation mathématique entrée/sortie

représentée par une fonction non linéaire f(.) quelconque, pouvant dépendre des variables u(k), y(k) et de leurs valeurs retardées telles que : {u(k-1), u(k-2), u(k-3), u(k-4), u(k - 5), u(k - 6) , y(k - 1), y(k - 2), y(k - 3), y(k - 4)}. Dans notre cas, on considère les variables y(k), y(k - 1) et u(k - 4), et on propose d'identifier un modèle flou décrivant la relation entrée/sortie suivante [**40**] :

$$y(k) = f(y(k-1), u(k-4)) \qquad \text{(IV. 1)}$$

L'objectif étant de construire un modèle flou de Takagi-Sugeno caractérisant au mieux la dynamique du four à gaz représentée par l'équation (VI. 1). La méthodologie de conception de systèmes flous proposée est appliquée aux données du système Box-Jenkins et les résultats obtenus sont comparés à ceux obtenus par approches de modélisation floue basées sur les algorithmes d'optimisation méta-heuristiques CRPSO, PSO, GA, DE, ainsi que d'autres approches combinant les techniques de clustering et les méta-heuristiques telles que l'algorithme ABC-FCM variable (VABC-FCM) et l'algorithme GA-FCM variable (VGA-FCM).

Pour que l'on puisse établir une comparaison objective avec les approches existantes (PSO, GA et DE), nous avions considéré dans la simulation les mêmes paramètres de ces algorithmes. Ainsi, le nombre d'itérations maximal est fixé à 2000 et la taille de la population représentant le nombre des sources de nourriture ou celui des abeilles Employed est 30. Les coefficients de la conséquence des règles sont limités dans l'intervalle [-10, 10], les écart-types des ensembles flous gaussiens des prémisses sont pris dans [0,5], et leurs centres entre [0,5]. Pour les besoins de test de robustesse, chaque algorithme est exécuté 50 fois.

Les résultats obtenus à l'aide de l'approche proposée avec ses différentes versions (basée ABC, basée g-ABC, basée ABC-MCS et basée g-ABC- MCS) sont indiqués dans le Tableau IV.1 en termes de moyennes et d'écart-types évalués pour les erreurs d'approximation (MSE d'apprentissage) et de prédiction (MSE de test). Comme il peut être clairement constaté, les résultats obtenus à l'aide de l'approche proposée sont plus performants que ceux obtenus par l'utilisation des algorithmes PSO, CRPSO, GA, DE, VABC-FCM, VGA-FCM, etc.

Méthode	Nombre de règles (Moyenne)	MSE (Moyenne)	MSE (Var.)
PSO [41,42]	3.64	0.1550	0.0427
CRPSO [41,42]	3.96	0.1428	0.0138
GA [41**]**	4.32	0.1474	0.0109
DE [41,42]	4.96	0.1768	0.0602
VGA-FCM [42**]**	3.74	0.1515	0.0312
VABC-FCM [42**]**	4	0.1256	0.7755e-5
Xu and Yong [43**]**	25	0.328	-
Chen et al. [43**]**	3	0.2678	-
Li et al. [44**]**	4	0.426	-
Notre approche (basée ABC)	4.08 .	0.1160 .	0.0312 .
Notre approche (basée g-ABC)	4.54 .	0.1095 .	0.0248 .
Notre approche (basée ABC-MCS)	4.22 .	0.0734 .	4.51e-4 .
Notre approche (basée g-ABC-MCS)	4.16	0.0730	3e-4

Tableau IV. 1 Résultats d'application de l'approche proposée au système Box-Jenkins.

Parmi l'ensemble des modèles obtenus, le modèle flou TS optimal identifié pour le système Box-Jenkins est déterminé par l'approche hybride g-ABC-MCS avec C = 1.5. Il se compose de cinq règles floues de la forme :

R_1 : If y(k-1) is in A_1^1 with (0.84 0.18) and u(k - 4) is in A_2^1 with (1.16 1.00),
Then $y^1(k) = 0.88 + 0.16\, y(k-1) - 1.04u(k-4)$

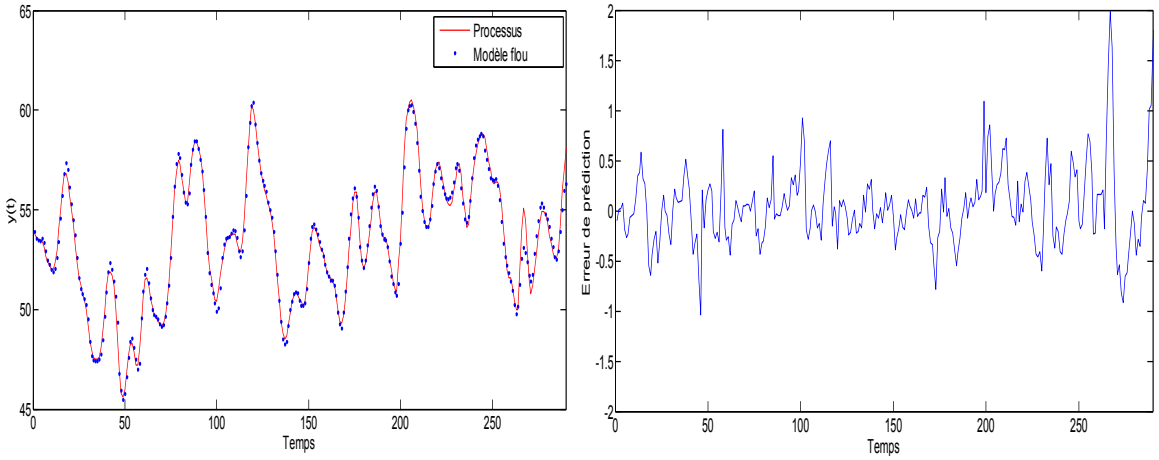
R_2 : If y(k-1) is in A_1^2 with (0.03 0.16) and u(k - 4) is in A_2^2 with (0.62 0.17),
Then $y^2(k) = 9.51 + 2.39y(k-1) + 0.35u(k-4)$

R_3 : If y(k-1) is in A_1^3 with (0.90 0.71) and u(k - 4) is in A_2^3 with (0.12 0.67),
Then $y^3(k) = 0.48 + 0.55y(k-1) - 0.93u(k-4)$

R_4 : If y(k-1) is in A_1^4 with (0.4 0.11) and u(k - 4) is in A_2^4 with (0.62 0.14),
Then $y^4(k) = -0.89 + 1.83y(k-1) - 5.12u(k-4)$

R_5 : If y(k-1) is in A_1^5 with (0.82 0.22) and u(k - 4) is in A_2^5 with (0.11 0.11),
Then $y^5(k) = 0.11 + 1.41y(k-1) - 1.08u(k-4)$

Pour ce modèle flou, l'erreur d'approximation moyenne est 0,0730. Ce modèle a presque la même qualité (précision) que celle obtenue en utilisant l'algorithme hybride ABC-MCS. Cependant, la base de règles induites par le modèle ABC-MCS est composée de six règles floues. En plus, le modèle trouvé par notre approche basée ABC donne une erreur MSE de 0.1160 alors que l'erreur MSE résultant pour le modèle obtenu par l'approche basée g-ABC est 0.1095. En outre, il convient de remarquer que le meilleur modèle flou identifié par l'algorithme VABC-FCM possède quatre règles avec une erreur d'approximation $MSE = 0.125547$, tandis que le modèle déterminé à l'aide de l'algorithme CRPSO est formé de trois règles avec une erreur $MSE = 0.1275$. La performance de notre approche d'auto-génération de modèles flous appliquée au problème d'identification du système Box-Jenkins peut être déduite à partir de la Figure IV.1. La Figure IV.1(a) compare la sortie du processus et celle du modèle identifié, tandis que l'erreur de prédiction est représentée dans la Figure IV.1 (b).



(a) (b)

Figure IV. 1 Performance de l'approche proposée appliquée au système Box-Jenkins : a) La sortie réelle et la sortie du modèle flou, b) L'erreur de prédiction.

La très bonne correspondance entre les deux courbes pour les données d'apprentissage et de test que montre cette figure justifie la capacité d'approximation et de généralisation que réalise le modèle TS flou identifié.

IV.3 Identification d'un système non-linéaire non forcé

Le système non linéaire non forcé est décrit par les équations [**42**]:

$$\boldsymbol{y(k) = g(y(k-1), y(k-2)) + u(k)} \qquad \text{(IV. 2)}$$

$$\text{avec } \boldsymbol{g(y(k-1), y(k-2)) = \frac{y(k-1)y(k-2)(y(k-1)-0.5)}{1+y(k-1)^2+y(k-2)^2}} \qquad \text{(IV. 3)}$$

Le système possède deux entrées et une seule sortie. L'objectif consiste à déterminer une approximation de la composante non linéaire $g(y(k-1), y(k-2))$, et ce à partir d'un ensemble de données d'entrée/sortie. Pour ce faire, une base regroupant 400 données a été construite à partir du modèle (VI. 2) et (VI. 3). La base d'apprentissage se compose de 200 échantillons calculés en utilisant comme entrée un signal aléatoire u(k) réparti uniformément dans [-1.5, 1.5]. Pour les données de test, 200 autres échantillons sont générés en utilisant un signal sinusoïdal de la forme $\boldsymbol{u(k) = sin(2\pi K/25)}$.

Pour l'identification floue de ce système, les paramètres des algorithmes ABC et g-ABC sont fixés aux mêmes valeurs que celles utilisées dans le paragraphe IV.2: le nombre d'itérations maximal est 2000, la taille de la population est 30 et 50 expériences sont menées en simulation pour chacun des algorithmes.

Le Tableau IV.2 montre les résultats d'application de l'approche proposée au problème d'identification floue du système non linéaire non forcé.

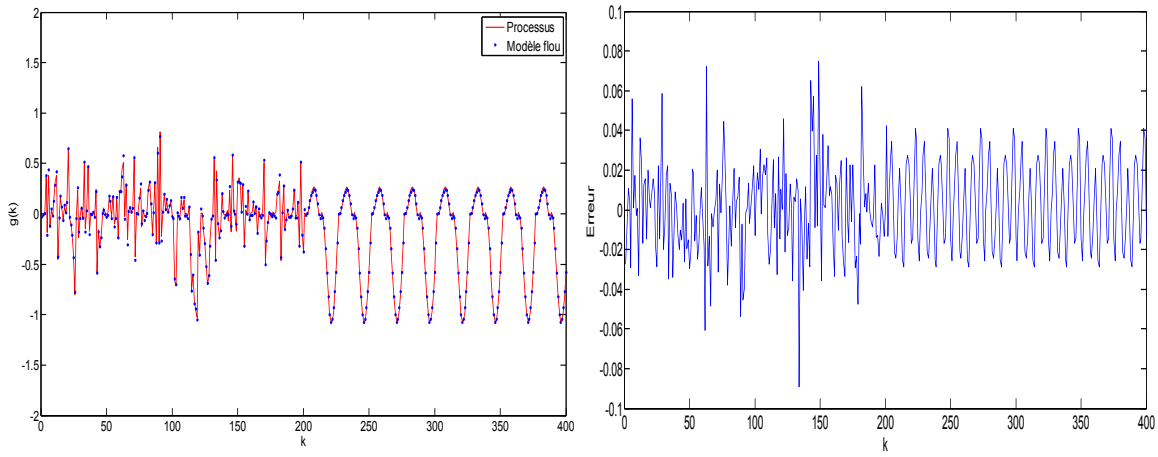
Méthode	Nombre de règles (Moyenne)	MSE (apprentissage)		MSE (test)	
		Moyenne	Var.	Moyenne	Var.
PSO [44]	5.64	0.0055	0.0055	0.0063	0.0072
CRPSO [44]	5.60	0.0020	0.0016	0.0035	0.0044
GA [44]	3.62	0.0038	0.0053	0.0045	0.0051
DE [44]	5.32	0.0081	0.0041	0.0086	0.0062
Notre approche (basée ABC)	4.98	0.0014	0.0005	0.0019	0.0012
Notre approche (basée g-ABC)	5.16	**0.0012**	**0.0004**	**0.0016**	**0.0007**
Notre approche (basée ABC-MCS)	4.88	**4.27e-4**	**1.17e-4**	**3.17e-4**	**8.77e-5**
Notre approche (basée g-ABC-MCS)	4.86	**4.19e-4**	**1.29e-4**	**2.74e-4**	**7.49e-5**

Tableau IV. 2 Résultats d'application de l'approche proposée au système non linéaire non forcé.

Comme nous pouvons le constater, l'approche proposée réalise de meilleures performances tant dans la phase d'apprentissage (approximation) que dans la phase de test (prédiction). Sur 50 exécutions, l'erreur moyenne d'approximation obtenue sur la base du modèle ABC est 0.0014 conduisant à un modèle flou de six règles, alors que l'erreur moyenne est évaluée à 0.0012 pour la version g-ABC (C = 1) qui donne un modèle flou composé de cinq règles. De même, l'approche d'identification basée ABC-MCS fournit une erreur moyenne de 4.27e-4 avec un modèle de six règles, et celle trouvée pour l'approche basée g-ABC-MCS (C = 1) est de l'ordre de 4.19e-4 avec un modèle flou de cinq règles. En somme, le meilleur modèle flou identifié est composé de six règles avec des erreurs moyennes d'apprentissage et de test de 1.42e-4 et 1.92e-4, respectivement.

Ce modèle s'exprime linguistiquement comme suit :

$$R_1 : \text{If } y(k\text{-}1) \text{ is } A_1^1 \text{ with } (0.89\ 0.1) \text{ and } y(k\text{-}2) \text{ is } A_2^1 \text{ with } (0.21\ 0.19),$$
$$\text{Then } y^1 = 0.29 + 0.054y(k\text{-}1) + 0.45y(k\text{-}2)$$
$$R_2 : \text{If } y(k\text{-}1) \text{ is } A_1^2 \text{ with } (0.11\ 0.17) \text{ and } y(k\text{-}2) \text{ is } A_2^2 \text{ with } (0.56\ 0.1),$$
$$\text{Then } y^2 = 2.00 + 0.24y(k\text{-}1) + 0.40y(k\text{-}2)$$
$$R_3 : \text{If } y(k\text{-}1) \text{ is } A_1^3 \text{ with } (0.98\ 0.15) \text{ and } y(k\text{-}2) \text{ is } A_2^3 \text{ with } (0.10\ 0.20),$$
$$\text{Then } y^3 = 0.42 + 0.02y(k\text{-}1) + 0.24y(k\text{-}2)$$
$$R_4 : \text{If } y(k\text{-}1) \text{ is } A_1^4 \text{ with } (0.66\ 1.42) \text{ and } y(k\text{-}2) \text{ is } A_2^4 \text{ with } (0.75\ 1.77),$$
$$\text{Then } y^4 = 0.46 - 0.17y(k\text{-}1) + 0.47y(k\text{-}2)$$
$$R_5 : \text{If } y(k\text{-}1) \text{ is } A_1^5 \text{ with } (0.14\ 0.1) \text{ and } y(k\text{-}2) \text{ is } A_2^5 \text{ with } (0.4.78\ 0.11),$$
$$\text{Then } y^5 = -0.16 - 0.02y(k\text{-}1) + 1.37y(k\text{-}2)$$
$$R_6 : \text{If } y(k\text{-}1) \text{ is } A_1^6 \text{ with } (0.44\ 0.1) \text{ and } y(k\text{-}2) \text{ is } A_2^6 \text{ with } (0.14\ 0.11),$$
$$\text{Then } y^6 = -0.79 + 0.64y(k\text{-}1) + 0.27y(k\text{-}2)$$



(a) (b)

Figure IV. 2 Performance de l'approche proposée appliquée au système non linéaire non forcé, a) La sortie réelle et la sortie du modèle flou, b) L'erreur de prédiction.

La Figure IV.2 illustre clairement la bonne approximation que réalise le modèle flou pour les deux bases de données d'apprentissage et de test.

IV.4 Identification d'un système non linéaire statique

Le système non linéaire statique est décrit par l'équation suivante [**41**]:

$$y = \left(1 + x_1^{-2} + x_2^{-1.5}\right)^2, \quad 1 \leq x_1, x_2 \leq 5 \qquad \text{(IV. 4)}$$

Les entrées du système sont $x_1 \, et \, x_2$, sa sortie est y. Comme dans [**45**], une base de données composée de 100 échantillons d'entrée-sortie est utilisée. Les premiers 50 échantillons sont dédiés à l'apprentissage et les 50 restants sont réservés à la phase de test. L'ensemble de ces données sont illustrées en Figure IV. 3.

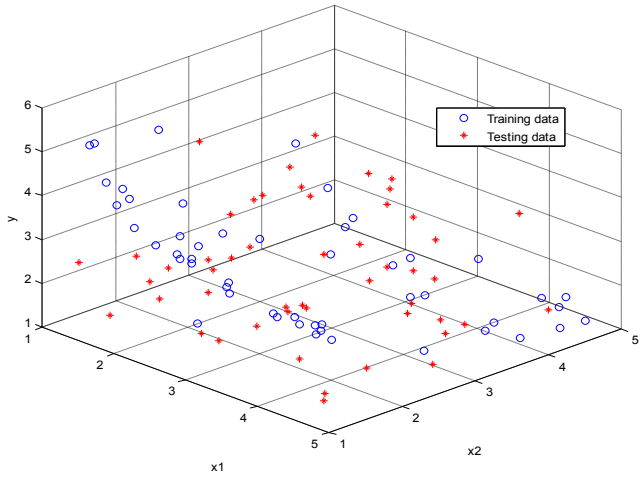


Figure IV. 3 L'ensemble de données d'apprentissage et de test utilisé pour le système non linéaire statique.

Le problème d'identification à résoudre est celui de la recherche d'un modèle flou TS où x_1, x_2 figurent comme variables de prémisses et y comme variable de conséquence. On utilise les mêmes paramètres de conception employés pour les deux cas d'étude précédents. Les expériences en simulation sont répétées 50 fois afin d'évaluer les erreurs moyennes d'approximation et de prédiction.

Dans ce cas d'étude, la meilleure solution a été atteinte par l'approche basée g-ABC-MCS avec C fixé à 0.5. Le modèle comporte six règles floues et peut être décrit linguistiquement comme suit :

R_1 : if x_1 is A_1^1 with (1.05, 0.10) and x_2 is A_2^1 with (1.06, 0.10), Then

$y^1 = 4.11 - 0.45x_1 - 0.48x_2$

R_2 : if x_1 is A_1^2 with (1.14, 0.10) and x_2 is A_2^2 with (1.06, 0.10), Then

$y^2 = 5.7580 - 6.3091x_1 - 3.5379x_2$

R_3 : if x_1 is A_1^3 with (2.12, 0.10) and x_2 is A_2^3 with (4.93, 0.31), Then

$y^3 = 4.44 - 0.03x_1 - 5.43x_2$

R_4 : if x_1 is A_1^4 with (1.05, 0.10) and x_2 is A_2^4 with (1.06, 0.10), Then

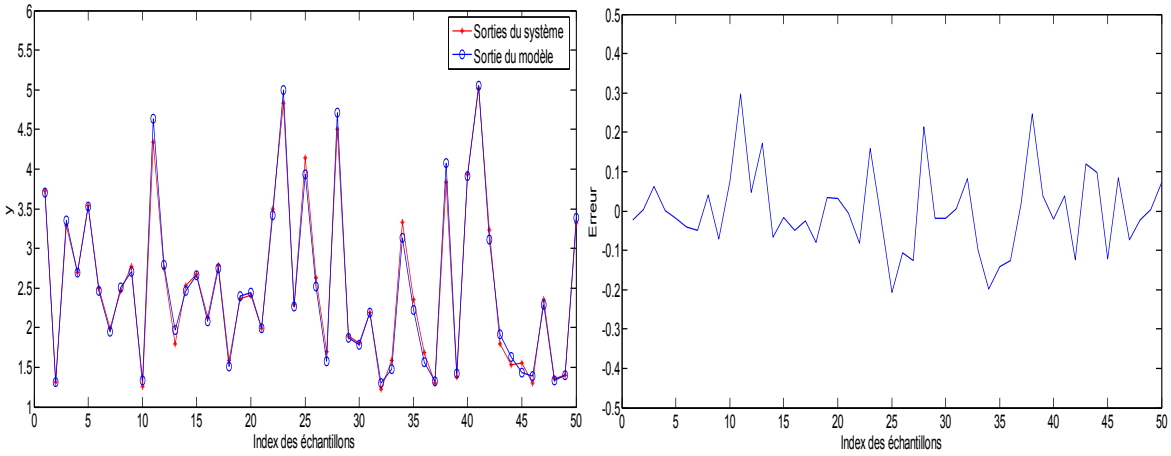
$y^4 = 3.92 - 2.89x_1 + 0.44x_2$

R_5 : if x_1 is A_1^5 with (3.4917, 0.14) and x_2 is A_2^5 with (4.94, 0.31), Then

$y^5 = 2.33 + 0.62x_1 + 0.85x_2$

R_6 : if x_1 is A_1^6 with (1.05, 0.10) and x_2 is A_2^6 with (1.06, 0.10), Then

$y^6 = 3.58 - 0.97x_1 - 0.07x_2$



(a) (b)

Figure IV. 4 Performance de l'approche proposée appliquée au système non linéaire statique : (a) Les sorties du système et du modèle flou identifié, (b) L'erreur de prédiction

La Figure IV. 4 présente les résultats d'identification floue du système non linéaire statique résultant de l'approche proposée basée g-ABC-MCS (C = 0,5). La sortie du modèle flou et l'erreur de prédiction correspondant aux 50 échantillons de test sont représentées graphiquement dans les Figure IV.4 (a) et IV.4(b), respectivement.

Le Tableau IV.3 compare les résultats d'identification du système non linéaire statique. Il apparait clairement que l'approche proposée avec ses différentes variantes domine toutes les

autres approches de modélisation comparées. La meilleure erreur de prédiction moyenne obtenue sur 50 exécutions est 0.0136 avec un écart-type de 0.0038. Cette dernière correspond

Méthode	Nombre de règles (moyenne)	MSE de test (moyenne)	MSE de test (var.)
Sugeno et al. [34]	6	0.079	-
Kim et al. [24]	3	0.0197	-
Li et al. [28]	4/5	0.114/0.102	-
PSO [4]	6	0.0972	-
PSO + MEP [4]	6	0.0831	-
VGA-FCM [33]	4	0.069	0.012
VABC-FCM [33]	3	0.0556	1.8136e-5
Notre approche (basée ABC-MCS)	4.62	**0.0159**	0.0058
Notre approche (basée g-ABC-MCS)	4.72	**0.0136**	0.0038

Tableau IV. 3 Résultats d'application de l'approche proposée au système non linéaire statique.

à l'approche basée g-ABC-MCS avec C = 0,5. Dans ce cas, l'erreur d'apprentissage moyenne et l'erreur de test moyenne sont les mêmes. Les erreurs d'apprentissage et de test du meilleur modèle flou identifié sont 0,0091 et 0,0085, respectivement. Ces résultats sont à comparer avec ceux obtenus par l'algorithme VABC-FCM qui a déterminé un modèle flou composé de trois règles, avec des erreurs d'apprentissage et de test de 0,0249 et 0,0556, respectivement.

IV.5 Commande en poursuite d'un système non-linéaire

Cette étude de cas présente un exemple de synthèse d'une loi de commande floue pour un système non linéaire en utilisant l'approche d'identification floue proposée dans ce mémoire. Le système non linéaire considéré est décrit par [**46**]:

$$\boldsymbol{y(k+1)} = \frac{\boldsymbol{y(k)}}{\boldsymbol{1+y^2(k)}} + \boldsymbol{u^3(k)} \qquad \text{(IV. 5)}$$

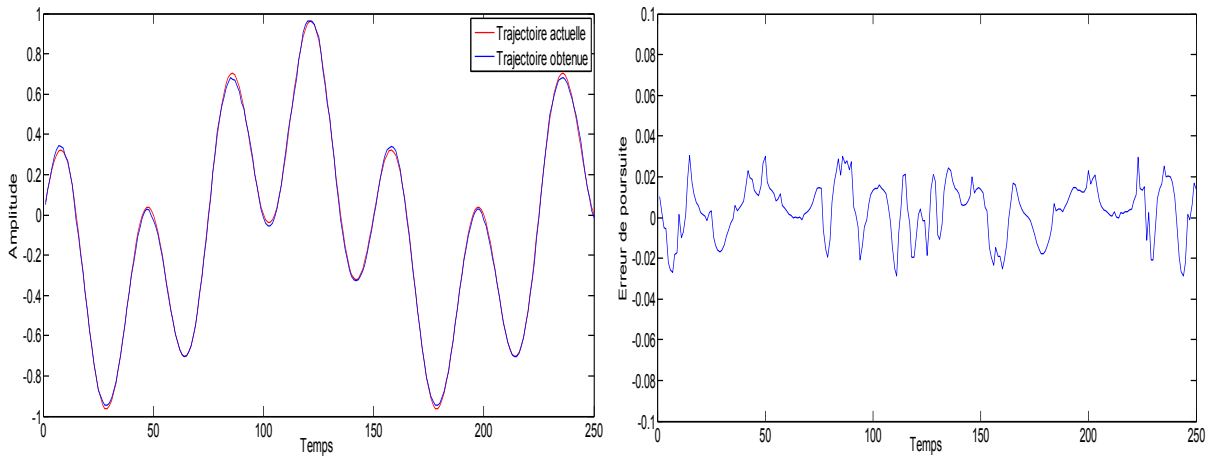
où $u(k)$ est la variable de commande et $y(k)$ la sortie du système, avec $-2 \leq y(k) \leq 2$, $y(0) = 0,\ et\ -1 \leq u(k) \leq 1$. Le problème de commande consiste à concevoir un

contrôleur flou de poursuite ayant pour entrée les variables $y_d(k+1)$ et $y(k)$ de telle sorte que la sortie $y(k)$ évolue selon la trajectoire donnée par :

$$\boldsymbol{y_d(k) = \sin\left(\frac{\pi k}{50}\right)\cos\left(\frac{\pi k}{30}\right), \qquad 1 \le k \le 250} \qquad \text{(IV. 6)}$$

Pour résoudre ce problème, nous utilisons le principe de la commande par modèle inverse [**46**]. L'apprentissage dans ce cas est réalisé en temps réel (online), c'est-à-dire les données d'apprentissage sont générées instantanément à partir de la boucle de commande. Le problème de synthèse du contrôleur flou de poursuite est formulé comme un problème d'optimisation où la structure et les paramètres du système flou sont déterminés en utilisant l'approche d'identification floue proposée basée sur les algorithmes ABC et g-ABC. La fonction objectif à minimiser est donnée par :

$$\boldsymbol{RMSE = \sqrt{\frac{\sum_{k=0}^{249}(y_d(k+1)-y(k+1))^2}{250}}}. \qquad \text{(IV. 7)}$$



(a) (b)

Figure IV. 5 Performance de l'approche proposée appliquée au problème de commande en poursuite d'un système non linéaire: (a) Résultats de la commande, (b) Erreur de poursuite.

Les paramètres de conception sont donnés comme suit : pour l'approche basée ABC et l'approche basée g-ABC, la taille de la population est fixée à 20, le nombre total de cycles est de 500 et la valeur limite est 200. Les résultats sont comparés à ceux rapportés dans [7] et obtenus par les algorithmes TPSIA, ACO, PSO-TVAC, HPSO-TVAC, HGAPSO et PSO-CREV. Toutes les expériences sont répétées 50 fois.

Méthode	PSO	CACO	PSO-TVAC	HPSO-TVAC	HGAPSO	PSO-CRE	TPSIA	ABC	g-ABC
RMSE (Moy.)	0.045	0.13	0.042	0.039	0.040	0.041	0.033	**0.0265**	**0.0246**
Var.	0.017	0.03	0.0134	0.014	0.008	0.012	0.012	**0.0066**	**0.0048**

Tableau IV. 4 Résultats d'application de l'approche proposée au problème de commande en poursuite en considérant une population de taille 20.

Méthode	TGA [46]	EGA [46]	MGAPSO [46]	ABC	g-ABC
RMSE	0.1170	0.0589	0.0440	**0.0265**	**0.0246**
Var.	0.0438	0.0208	0.0139	**0.0066**	**0.0048**
Nombre de règles	4	4	4	3.77	4.30

Tableau IV. 5 Résultats d'application de l'approche proposée au problème de commande en poursuite (pour les algorithmes comparés la population est de taille 100).

Le Tableau IV. 4 compare la méthodologie proposée avec d'autres algorithmes. Il est clair que l'approche proposée basée g-ABC (C = 1) permet d'obtenir la plus petite erreur quadratique moyenne de poursuite qui est de l'ordre de 0,024 avec un écart-type de 0,0048. Le contrôleur flou est composé de cinq règles floues et réalise une erreur de poursuite de RMSE=0.0152. Les résultats de l'approche proposée basée ABC sont meilleurs par rapport à ceux obtenus par l'algorithme TPSIA qui propose de résoudre le problème de commande en deux phases utilisant les techniques d'optimisation ACO et PSO. La Figure IV. 5 illustre le comportement du système commandé par le contrôleur flou de poursuite conçu par l'approche basée ABC. L'erreur de poursuite globale est évaluée à 0,0140. L'évolution du processus d'optimisation des erreurs de poursuite moyennes est illustrée en Figure IV. 6. Nous pouvons alors constater que les algorithmes de contrôle basés ABC et g-ABC (C = 1) ont atteint quasiment les mêmes performances.

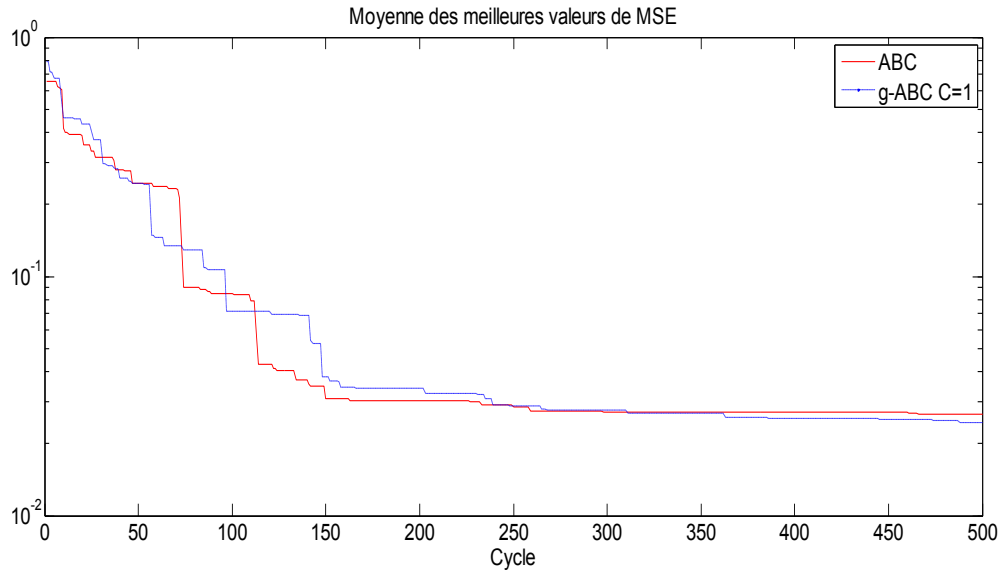


Figure IV. 6 Evolution des erreurs de poursuite moyennes (RMSE) évaluées pour les approches basées ABC et g-ABC.

D'autres méthodes de conception d'un contrôleur flou de poursuite pour le même problème de commande ont été considérées dans littérature. Le Tableau IV. 5 présente quelques résultats obtenus par les algorithmes MGAPSO, TGA et EGA rapportés dans [**46**]. Il convient de noter que la taille de la population considérée est 100 et le nombre total d'itérations est 300. Il est clair que notre approche réalise les meilleures perfromances avec un nombre d'évaluations nettement inférieur.

IV.6 Analyse de la robustesse de l'approche de modélisation floue proposée

Dans cette section, on propose d'évaluer la robustesse de l'approche de modélisation floue proposée vis-à-vis de la variation de la taille de la population et du paramètre de contrôle C utilisé dans l'algorithme g-ABC.

IV.6.1 Par rapport à la taille de la population

Comme il peut être constaté dans les exemples précédents, la taille de la population correspondant au nombre des abeilles Employed ou Onlooker a été fixée à 30 pour les problèmes d'identification floue décrits ci-dessus, et ce en vue de permettre une comparaison objective avec les approches de modélisation rapportées dans la littérature. Pour étudier les aspects de robustesse, ces mêmes problèmes sont considérés avec différentes tailles de la population. Les Tableaux IV.6-IV.12 indiquent les résultats obtenus pour les tailles de

population 30, 50 et 100. Les évolutions de la moyenne des erreurs d'approximation pour les cas étudiés sont présentées dans les Figures IV. 7, IV. 8 et IV. 9.

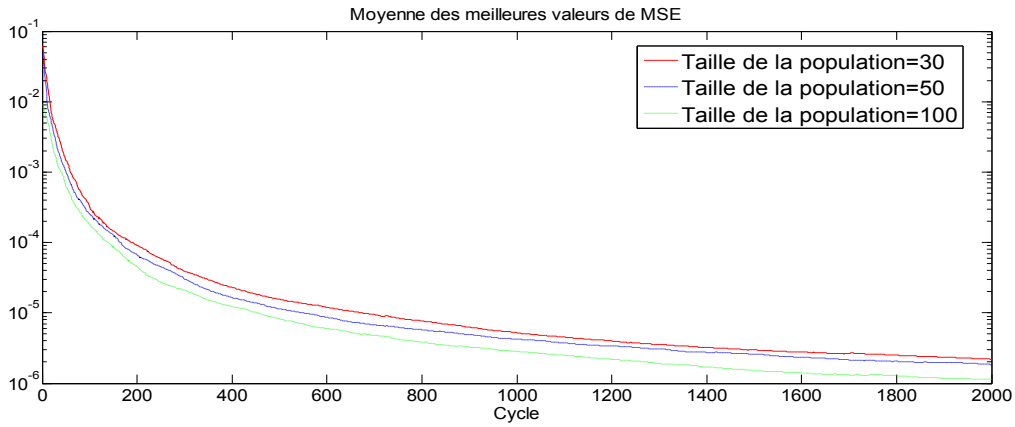


Figure IV. 7 L'évolution de la moyenne des MSE pour le système Box-Jenkins obtenue par l'algorithme ABC pour différentes tailles de la population.

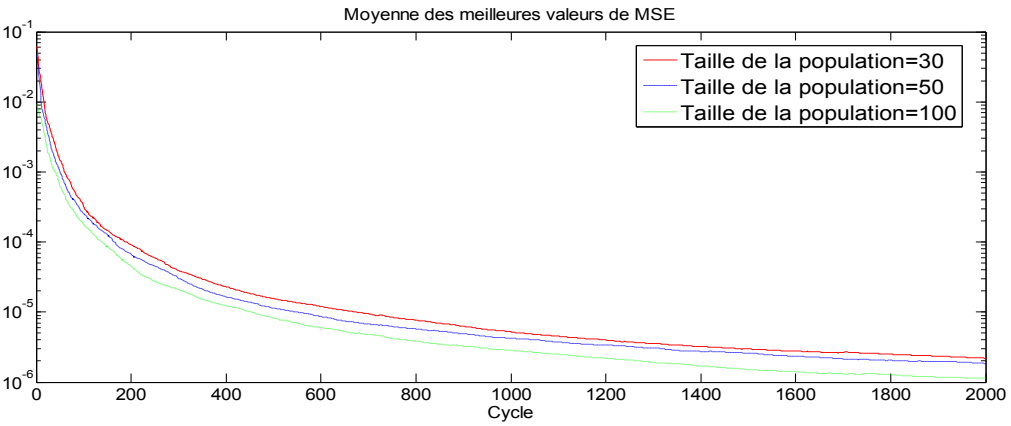


Figure IV. 8 L'évolution de la moyenne des MSE pour le système non linéaire non forcé obtenue par l'algorithme ABC pour différentes tailles de la population.

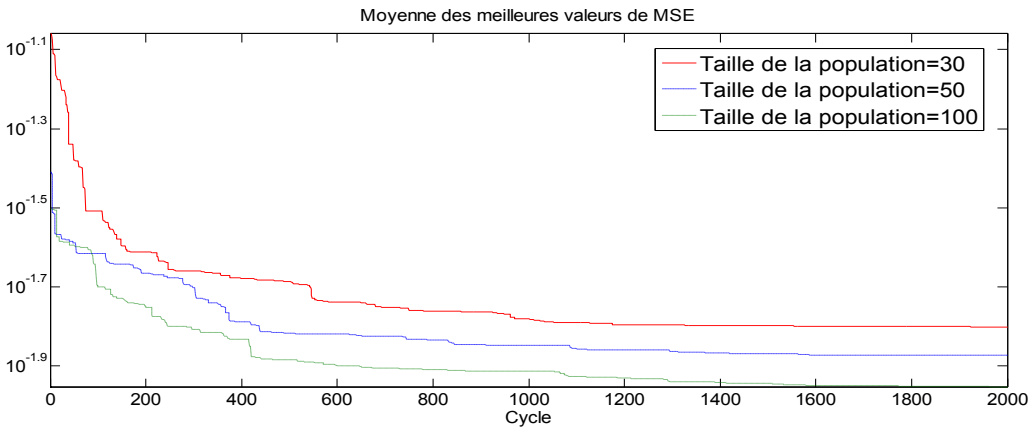


Figure IV. 9 L'évolution de la moyenne des MSE pour le système non linéaire statique obtenue par l'algorithme ABC pour différentes tailles de la population.

D'après ces Tableaux et figures, nous pouvons conclure que l'augmentation de la taille de la population provoque une amélioration remarquable des résultats d'identification. Ces remarques sont valables pour l'algorithme ABC et l'algorithme g-ABC et même pour les algorithmes hybrides ABC-MCS et g-ABC-MCS. Pour une taille de population 100, une nette amélioration des erreurs d'approximation moyennes est constatée sur les résultats d'identification du système Box-Jenkins obtenus par l'approche proposée basée ABC et g-ABC, comme le montrent les Tableaux IV.6 et IV.9. Cette remarque est aussi valable pour le cas du système non linéaire statique où la moyenne MSE sur 50 exécutions est environ 0.011 pour toutes les solutions déterminées par les algorithmes d'identification floue basés ABC et g-ABC, tel que montré dans les Tableaux IV.8 et IV.11. Pour le système non linéaire non forcé, les Tableaux IV.7 et IV.10 montrent que sur 50 exécutions, l'erreur d'apprentissage moyenne trouvée par l'approche basée ABC est 9.8572e-4, tandis que l'approche basée g-ABC donne une meilleure valeur de 8.3706e-4. L'approche hybride améliore nettement ces résultats avec une erreur moyenne de 3.56e-4 pour l'approche ABC-MCS et 1.87e-4 pour g-ABC-MCS. En conclusion, on peut affirmer que l'augmentation de la taille de la population de 30 à 100 a un influence positive sur la qualité d'approximation et de prédiction des modèles flous identifiés.

Taille de la population	**MSE (ABC)**	**MSE (ABC-MCS)**	**MSE (Optimale)**
30	0.1160	0.0734	**0.0722**
50	0.1052	0.0733	0.0724
100	**0.0952**	**0.0731**	0.0725

Tableau IV. 6 Résultats d'identification floue du système Box-Jenkins obtenus par la méthodologie proposée basée ABC pour différentes tailles de population.

Taille de la population	**MSE d'Apprentissage**		**MSE de test**		**MSE Optimale**
	Moyenne (ABC)	**Moyenne (ABC-MCS)**	**Moyenne (ABC)**	**Moyenne (ABC-MCS)**	
30	0.0014	4.27e-4	0.0019	5.61e-4	1.92e-4
50	0.0011	3.74e-4	**0.0013**	4.84e-4	1.90e-4
100	**9.85e-4**	**3.56e-4**	0.0013	**4.79e-4**	**1.89e-4**

Tableau IV. 7 Résultats d'identification floue du système non linéaire non forcé obtenus par la méthodologie proposée basée ABC pour différentes tailles de population.

Taille de la population	MSE d'Apprentissage		MSE de test		MSE Optimale
	Moyenne (ABC-MCS)	Var.	Moyenne (ABC-MCS)	Var.	
30	0.0158	0.0039	0.0159	0.0058	0.0110
50	0.0134	0.0039	0.0143	0.0056	0.0093
100	**0.0111**	**0.0026**	**0.0129**	**0.0040**	**0.0081**

Tableau IV. 8 Résultats d'identification floue du système statique non linéaire obtenus par la méthodologie proposée basée ABC-MCS pour différentes tailles de population.

IV.6.2 Par rapport au paramètre de contrôle C de l'algorithme g-ABC

Comme il est motionné dans la section III.6 du chapitre III, le paramètre de contrôle C de l'algorithme g-ABC joue un rôle important pour assurer l'équilibre entre exploitation et exploration lors de la recherche d'éventuelles solutions au problème d'optimisation. Pour analyser l'effet de ce paramètre sur l'approche d'identification floue proposée, plusieurs valeurs du paramètre C sont implémentées pour différentes tailles de la population. Les résultats de cette analyse sont cités dans les Tableaux IV.9-IV.11.

Pour le problème du Box-Jenkins, le Tableau IV.9 montre que pour toutes les valeurs de C considérées, l'erreur moyenne MSE sur 50 exécutions diminue avec la croissance de la taille de la population. Cependant, il convient de remarquer que pour une taille de 100 de la population, augmenter le paramètre C de 0.5 à 1.5 implique une augmentation de l'erreur d'approximation moyenne sauf pour la taille 30 où l'erreur MSE moyenne a diminué de 0.1106 à 0.1095 lorsque C est passé de 1 à 1.5. En plus, l'erreur MSE du meilleur modèle identifié s'améliore avec l'augmentation du paramètre C, elle passe en fait de 0.0762 à 0.0806 avec l'augmentation de la valeur du paramètre C de 1 à 1.5 pour la taille de la population 50. Aussi, le Tableau IV. 10 illustre l'effet du paramètre C sur les performances de la méthodologie proposée pour le problème d'identification floue du système non linéaire non forcé. Nous pouvons observer que pour la taille de 30 et 50, augmenter la valeur du paramètre C provoque une baisse de l'erreur MSE moyenne, ce qui signifie une amélioration de la solution. La même observation peut être faite pour la taille 100, lorsque C augmente de 0.5 à 1. Cependant, pour C=1.5 l'erreur d'apprentissage moyenne augmente légèrement à 6.4236e-4 qui est quand même une bonne erreur d'approximation.

La taille de la population	MSE (C = 0.5)			MSE (C=1)			MSE (C = 1.5)		
	g-ABC	g-ABC-MCS	MSE optim.	g-ABC	g-ABC-MCS	MSE optim.	g-ABC	g-ABC-MCS	MSE optim.
30	0.1074	0.0732	0.0723	0.1106	0.0730	0.0724	0.1095	0.0730	0.0723
50	0.0961	0.0730	0.0722	0.0997	0.0730	0.0721	0.1020	0.0729	**0.0719**
100	**0.0926**	**0.0729**	**0.0719**	**0.0927**	**0.0727**	**0.0718**	**0.0936**	**0.0727**	**0.0719**

Tableau IV. 9 Résultats d'identification floue du système Box-Jenkins obtenus par la méthodologie proposée basée g-ABC et g-ABC-MCS pour différentes tailles de population et différentes valeurs du paramètre C.

La taille de la population	MSE (C = 0.5)			MSE (C=1)			MSE (C = 1.5)		
	MSE g-ABC	MSE g-ABC MCS	Optim.	MSE g-ABC	MSE g-ABC MCS	Optim.	MSE g-ABC	MSE g-ABC MCS	Optim.
30	0.0013	4.15e-4	2.00e-4	0.0012	4.27e-4	1.99e-4	0.0014	4.19e-4	1.87e-4
50	9.99e-4	3.83e-4	1.86e-4	0.0010	3.64e-4	**1.92e-4**	0.0012	3.62e-4	1.86e-4
00	**8.37e-4**	**3.07e-4**	**1.75e-4**	**8.88e-4**	**3.11e-4**	1.94e-4	**0.0011**	**2.96e-4**	**1.74e-4**

Tableau IV. 10 Résultats d'identification floue du système non linéaire non forcé obtenus par la méthodologie proposée basée g-ABC pour différentes tailles de population et différentes valeurs du paramètre C.

Pour la taille 50 une amélioration de l'erreur moyenne MSE a été remarquée avec le changement de C de 0.5 vers 1.5. Dans presque tous les cas les performances du meilleur modèle flou identifié s'améliore avec l'augmentation de la valeur du paramètre C.

Taille de la population	MSE (C = 0.5)			MSE (C=1)			MSE (C = 1.5)		
	Appre. (Moy.)	Test (Moy.)	Optim.	Appre. (Moy.)	Test (Moy.)	Opti.	Appre. (Moy.)	Test (Moy.)	Optim.
30	0.0136	0.0136	0.0091	0.0141	0.0143	0.0095	0.0137	0.0160	0.0095
50	0.0126	**0.0126**	0.0093	0.0126	0.0148	0.0095	0.0116	0.0130	**0.0078**
100	**0.0116**	0.0130	**0.0082**	**0.0109**	**0.0137**	**0.0087**	**0.0110**	**0.0126**	0.0090

Tableau IV. 11 Résultants d'identification floue du système non-linéaire statique obtenus par la méthodologie proposée basée g-ABC pour différentes tailles de population et différentes valeurs de C.

C	Nombre de règles (moyenne)	RMSE (moyenne)	Var.	RMSE optimal
0.5	3.67	0.0253	0.0049	0.0158
1	4.30	**0.0246**	**0.0048**	**0.0152**
1.5	3.80	0.0251	0.0058	0.0161

Tableau IV. 12 Résultats du contrôle flou en poursuite obtenus par la méthodologie proposée basée g-ABC pour différentes valeurs du paramètre C.

Pour le système non linéaire statique, nous avions observé une augmentation de l'erreur moyenne lorsque le paramètre C est diminué de 1.5 à 0.5 pour les tailles de population 30 et 100, et lorsque C est passé de 1 à 0.5 pour la taille de 50, comme le montre le Tableau IV. 11. D'autre part, le Tableau IV. 12 montre que la variation de C induit un effet visible sur les résultats du contrôle en poursuite du système non linéaire traité dans le paragraphe IV.5. En effet, pour un nombre d'abeilles Employed fixé à 20, la performance en poursuite des contrôleurs flous synthétisés a basculé entre 0.0158 et 0.0161 avec une erreur moyenne entre 0.0251 et 0.0253. Le meilleur résultat est obtenu pour C=1 avec un contrôleur flou composé de cinq règles.

Enfin, il est important de noter que le choix d'une taille élevée de population ne fournit pas nécessairement les meilleures solutions. Par exemple, le meilleur modèle identifié pour le système Box-Jenkins est déterminé pour une taille de 30 à l'aide de l'approche basée ABC. L'approche basée g-ABC améliore légèrement ce résultat avec une taille de population égale à 100. En outre, nous constatons que, parmi toutes les simulations effectuées pour le système statique non linéaire, la meilleure erreur d'approximation (MSE = 0,0078) a été obtenue avec une taille de population égale à 50, et ce à l'aide de l'approche basée g-ABC (C = 1,5). Le MSE de test est 0.008, ce qui démontre une bonne capacité d'approximation et de prédiction de la méthode d'auto-génération des modèles flous proposée. Ainsi, la méthodologie de modélisation floue proposée est capable de déterminer de bons modèles flous même avec l'emploi d'un nombre réduit d'abeilles Employed ou Onlooker.

Cette analyse de robustesse détaillée justifie la validité mais aussi l'utilité de notre méthodologie d'auto-génération de systèmes flous. Elle sera appliquée dans la suite au problème d'identification floue d'un échangeur de chaleur à courants parallèles en utilisant des données expérimentales collectées sur une installation pilote.

IV.7 Application à un échangeur de chaleur

IV.7.1 Description du processus

Le processus d'échange thermique étudié est la partie principale de l'installation thermique pilote schématisée à la Figure IV. 10. Il se compose de trois sous-systèmes: le dispositif de chauffage, le circuit d'air et le circuit d'eau. De façon plus détaillée, le système est composé d'un dispositif de chauffage électrique de l'air (E), les conduites d'air et d'eau, un échangeur à courants parallèles (HE), deux électrovannes (Vr) et (Ve) pour contrôler le recyclage et l'évacuation d'air et une pompe à vitesse variable (SP) pour contrôler le débit d'eau Qw. L'eau entrant dans l'échangeur de chaleur avec la température T33 est chauffé jusqu'à la température T34 à l'aide de l'air chaud. La quantité d'air provenant du dispositif de chauffage électrique avec une température T14 pénètre dans l'échangeur de chaleur avec la température T16 après avoir parcourue la conduite d'air. Sa température à la sortie de l'échangeur de chaleur est T15. Un recyclage total ou partiel de l'air peut être considéré en fonction de la position des deux vannes motorisées Vr et Ve. Toutes ces variables de processus peuvent être mesurées grâce à des capteurs installés.

Les variables manipulées sont la puissance de chauffage, la position de la vanne de recyclage d'air et la position de la vanne d'évacuation d'air.

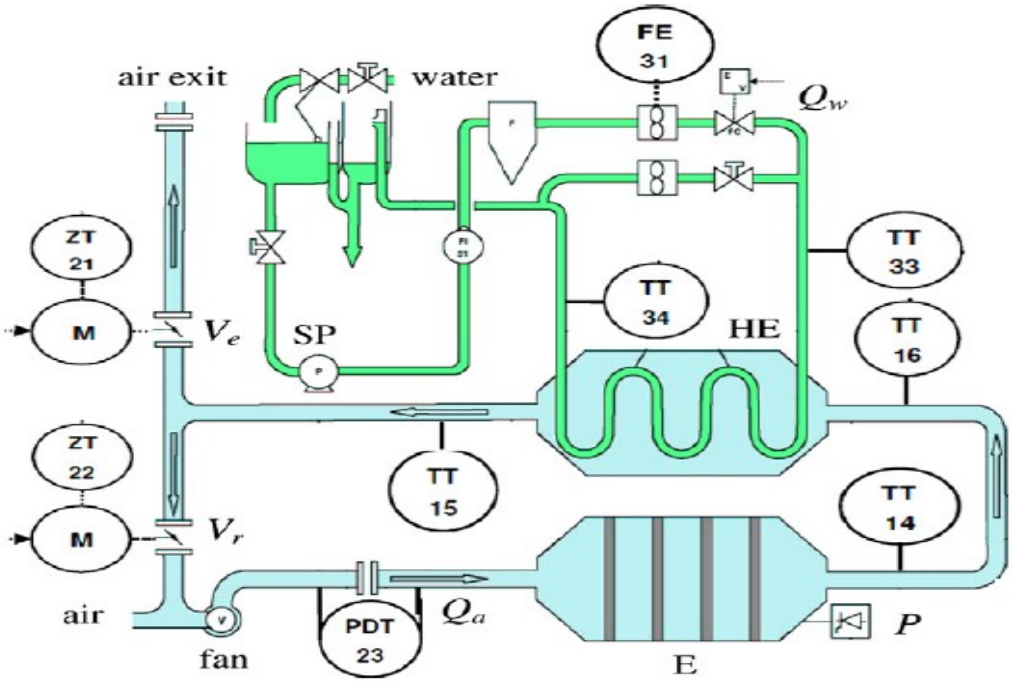


Figure IV. 10 L'installation d'échange thermique [**47**]

IV.7.2 Identification floue de l'échangeur de chaleur

Comme déjà rapporté dans [**47**], le comportement d'entrée-sortie du circuit d'eau et du circuit d'air de l'installation thermique dépend principalement de la puissance de chauffage P, de la position de la vanne de recyclage d'air Vr , de la position de la vanne d'évacuation d'air Ve, de la température d'air T14 et T16, et de la température d'eau à la sortie de l'échangeur de chaleur T34.

Compte tenu de ces considérations, un modèle de la forme suivante a été proposé dans [**47**]:

$$\boldsymbol{T_{34}(k+1) = f_1\big(T_{34}(k), T_{16}(k), P(k), V_r(k)\big)} \quad \text{(IV. 8)}$$

$$\boldsymbol{T_{16}(k+1) = f_2\big(T_{16}(k), T_{14}(k), P(k), V_r(k)\big)} \quad \text{(IV. 9)}$$

avec $f_1$et f_2 sont des fonctions non-linéaires.

L'objectif est de concevoir un modèle flou TS pour chacun des circuits en utilisant l'approche d'auto-génération de systèmes flous proposée dans ce mémoire. A cet effet, les données réelles de l'échangeur de chaleur pilote sont exploitées.

La structure du modèle flou de Takagi-Sugeno à générer à partir de ces données servira à la prédiction de l'évolution des températures du procédé. Il est décrit par un ensemble de règles IF-THEN floues de la forme:

$$\boldsymbol{IF\ T_{34}\ is\ A_{11}^i\ and\ T_{16}\ is\ A_{12}^i\ and\ P(k) is\ A_{13}^i\ and\ V_r(k)\ is\ A_{14}^i}$$
$$\boldsymbol{THEN\ T_{34}(k+1) = b_1^i + a_{11}^i T_{34}(k) + a_{12}^i T_{16}(k) + a_{13}^i P(k) + a_{14}^i V_r(k)} \quad \text{(IV. 10)}$$

$$\boldsymbol{IF\ T_{16}\ is\ A_{21}^i\ and\ T_{14}\ is\ A_{22}^i\ and\ P(k) is\ A_{23}^i\ and\ V_r(k)\ is\ A_{24}^i}$$
$$\boldsymbol{THEN\ T_{16}(k+1) = b_2^i + a_{21}^i T_{16}(k) + a_{22}^i T_{14}(k) + a_{23}^i P(k) + a_{24}^i V_r(k)}$$

Pour déterminer les paramètres des prémisses et des conclusions des règles, ainsi que le nombre de règles du modèles (IV. 10), un ensemble de données d'apprentissage contenant 2000 mesures est exploité.

IV.7.3 Résultats d'identification

La méthodologie de modélisation floue basée sur les algorithmes hybrides ABC-MCS et g-ABC-MCS est appliquée à ces données. Une population de taille 30 est choisie avec un nombre maximal d'itérations de 2000. Les paramètres des algorithmes sont les suivants :

- Les centres des ensembles flous des prémisses sont pris entre [0,1], car les données ont été d'abord normalisées.
- Les écart-types des ensembles des prémisses sont pris entre [0,5].
- La valeur limite égale à 200.
- Pour l'algorithme g-ABC, le paramètre de contrôle C=1.5.

Les résultats d'identification sont indiqués dans le Tableau IV. 13. Le nombre de règles déterminées pour chaque température de sortie est 6. Le modèle flou TS identifié pour le circuit d'eau (température T34) est décrit par l'ensemble des règles floues suivantes:

R_1 : If T_{34} is A_1^1 with (1 1.26) and T_{16} is A_2^1 with (0 4.36) and p is A_3^1 with (0 1.30) and Ve is A_4^1 with (0.18 2.21), Then $y^1 = 1.68e-3 + 0.098 \times T_{34} + 5.05e-4 \times T_{16} + 1.32e-3 \times P - 2.96e-4 \times Ve$

R_2 : If T_{34} is A_1^2 with (1 1.74) and T_{16} is A_2^2 with (1 5) and P is A_3^2 with (0.64 0.62) and Ve is A_4^2 with (1 1.87), Then $y^2 = 1.11e-3 + 9.71e-2 \times T_{34} + 1.56e-3 \times T_{16} + 4.76e-5 \times P - 1.33e-4 \times Ve$

R_3 : If T_{34} is A_1^3 with (1 3.10) and T_{16} is A_2^3 with (1 0.98) and P is A_3^3 with (1 3.47) and Ve is A_4^3 with (1 3.47), Then $y^3 = 4.39e-4 + 9.83e-2 \times T_{34} + 9.74e-4 \times T_{16} + 7.07e-4 \times P - 1.61e-4 \times Ve$

R_4 : If T_{34} is A_1^4 with (0 4.68) and T_{16} is A_2^4 with (1 2.86) and P is A_3^4 with (0 1.53) and Ve is A_4^4 with (0 0.96) Then $y^4 = -4.38e-3 + 9.86e-2 \times T_{34} + 8.35e-4 \times T_{16} + 1.52e-3 \times P - 5.67e-4 \times Ve$

R_5 : If T_{34} is A_1^5 with (0 4.03) and T_{16} is A_2^5 with (0 1.65) and P is A_3^5 with (0 5) and Ve is A_4^5 with (0 1.18), Then $y^5 = -8.71e-3 + 9.78e-2 \times T_{34} + 2.44e-3 \times T_{16} + 9.29e-4 \times P - 7.21e-4 \times Ve$

R_6 : If T_{34} is A_1^6 with (0 2.88) and T_{16} is A_2^6 with (0 2.52) and P is A_3^6 with (0.67 0.1) and Ve is A_4^6 with (1 3.58), Then $y^6 = 4.99e-3 + 9.22e-2 \times T_{34} + 4.05e-3 \times T_{16} - 1.04e-2 \times P + 1.20e-3 \times Ve$

Les sorties des modèles flous identifiés sont comparées aux mesures réelles des températures d'air et d'eau dans les Figures IV.11 et IV.13.

	La sortie	Nombre de règles	Données d'apprentissage	
			MSE Parallèle	MSE Série
ABC-MCS	T_{34}	6	0.0420	9.1084e-5
	T_{16}	6	0.0090	9.5338e-4
g-ABC-MCS	T_{34}	6	0.0404	8.1057e-5
	T_{16}	6	0.0087	9.5176e-4
GK [47]	T_{34}	3	0,0421	0,0001
	T_{16}	3	0,0097	0,0013

Tableau IV. 13 Résultats d'identification floue de l'échangeur de chaleur.

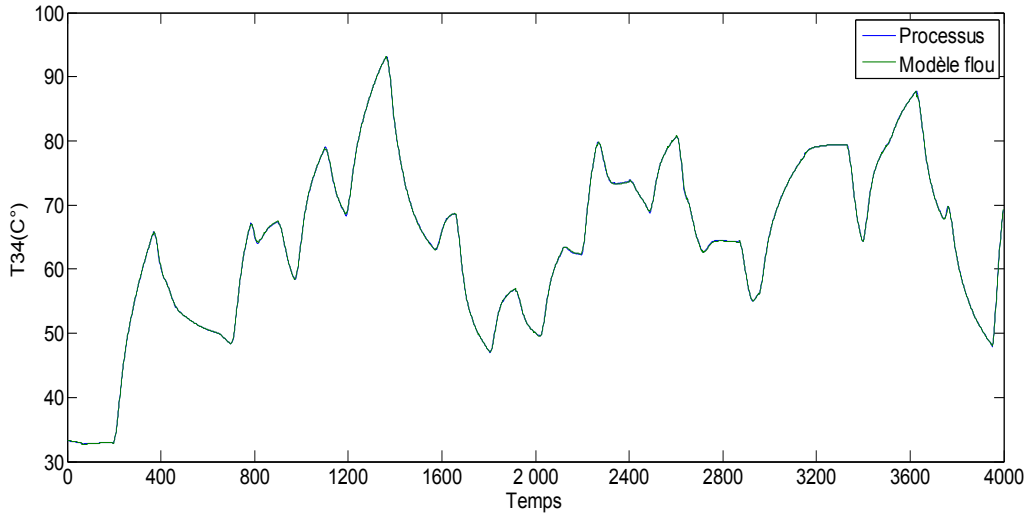


Figure IV. 11 La sortie réelle et la sortie du modèle de prédiction de T34

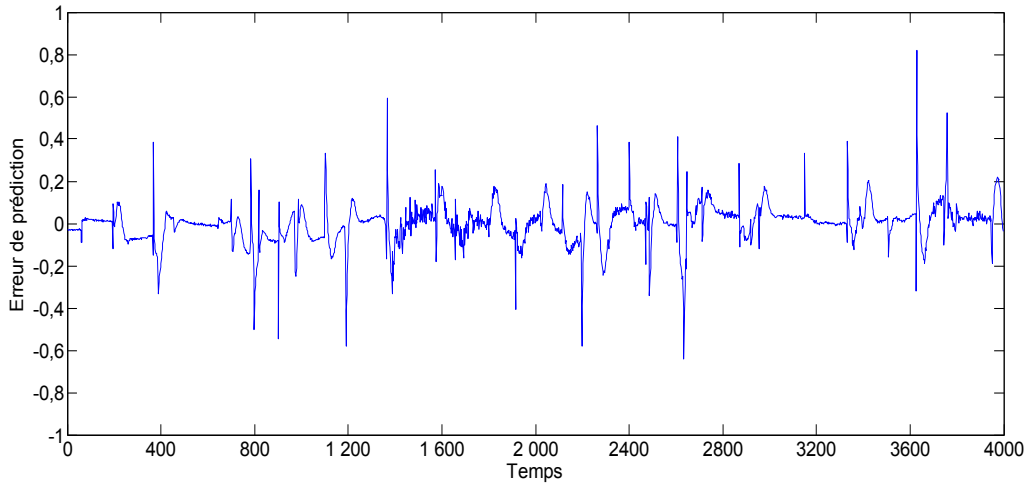


Figure IV. 12 L'erreur d'approximation de la température d'eau T34

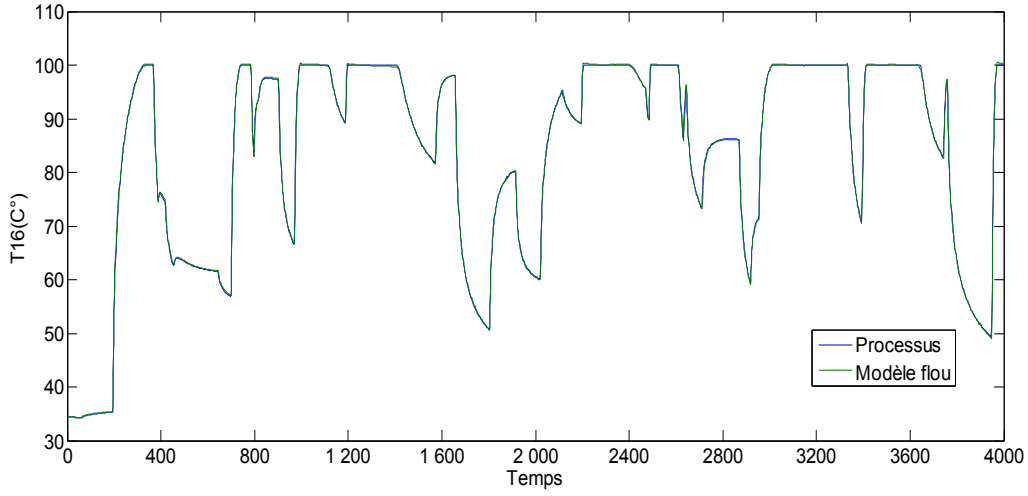


Figure IV. 13 La sortie réelle et la sortie du modèle de prédiction de T16.

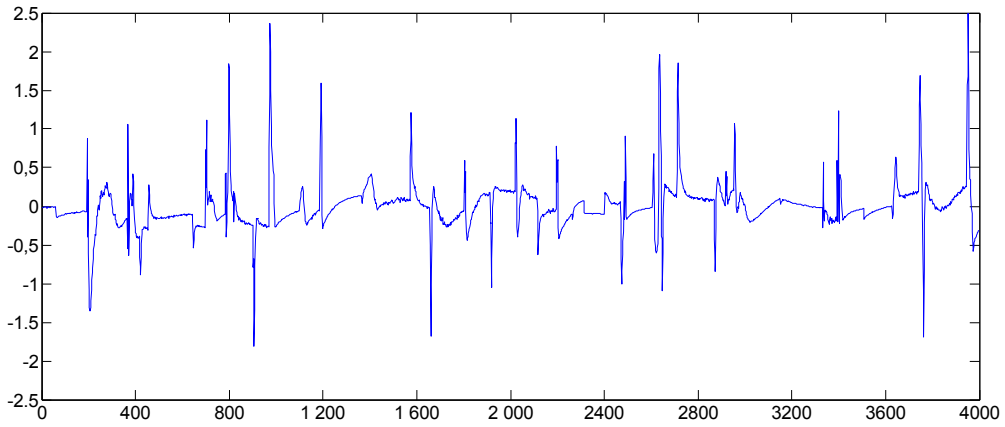


Figure IV. 14 L'erreur d'approximation de la température d'air T16.

A partir de ces tracés, nous pouvons constater que les modèles flous obtenus réalisent une bonne prédiction de la dynamique des températures d'air et d'eau de l'échangeur de chaleur.

Les partitions floues résultantes sont illustrées dans la Figure IV. 15. Le chevauchement constaté sur ces distributions rend difficile l'interprétation des modèles obtenus, bien que leur qualité est nettement appréciable. Afin de rendre plus transparentes ces partitions, on peut éventuellement faire appel à des techniques de réduction du nombre de règles floues. Or, celles-ci peuvent altérer la qualité d'approximation ou de prédiction des modèles initiaux.

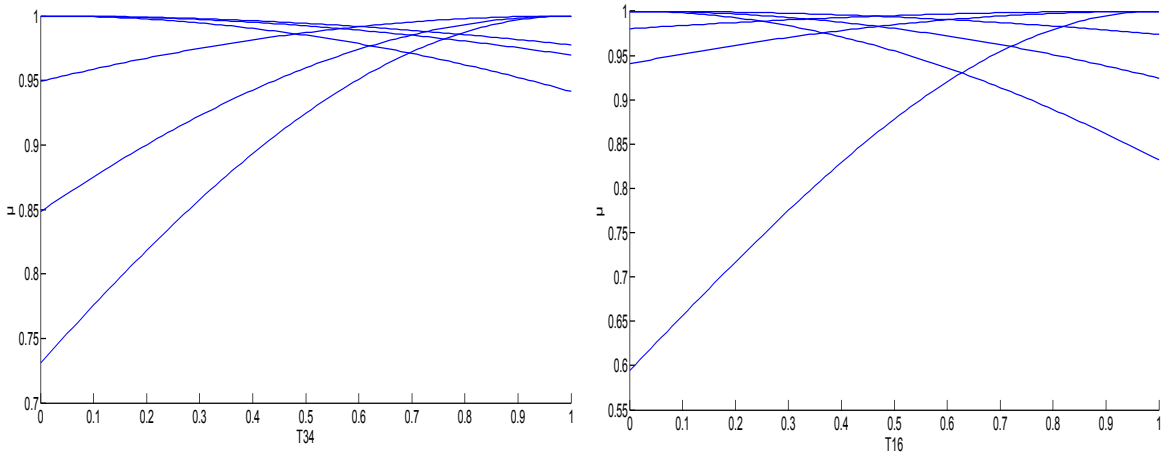


Figure IV. 15 Les partitions floues associées aux entrées T34 et T16 du modèle de prédiction de la température T34.

Pour la validation du modèle flou identifié pour chaque circuit, un ensemble de données de validation différent de l'ensemble de données d'apprentissage est utilisé. La performance du modèle identifié floue sur les données de validation est représentée dans les Figure IV. 16 et IV. 17. Le Tableau IV. 14 résume ces résultats en termes d'erreurs de prédiction moyennes évaluées pour les deux températures d'air et d'eau.

	La sortie	Données de validation	
		MSE Parallèle	MSE Série
ABC-MCS	T_{34}	0.0278	4.5455e-5
	T_{16}	0.0079	4.8911e-4
g-ABC-MCS	T_{34}	0.0262	4.5455e-5
	T_{16}	0.0071	4.6547e-4
GK [47]	T_{34}	0,0292	8,2810e-5
	T_{16}	0,0087	9,4483e-4

Tableau IV. 14 Résultats de validation des modèles flous identifiés pour l'échangeur de chaleur.

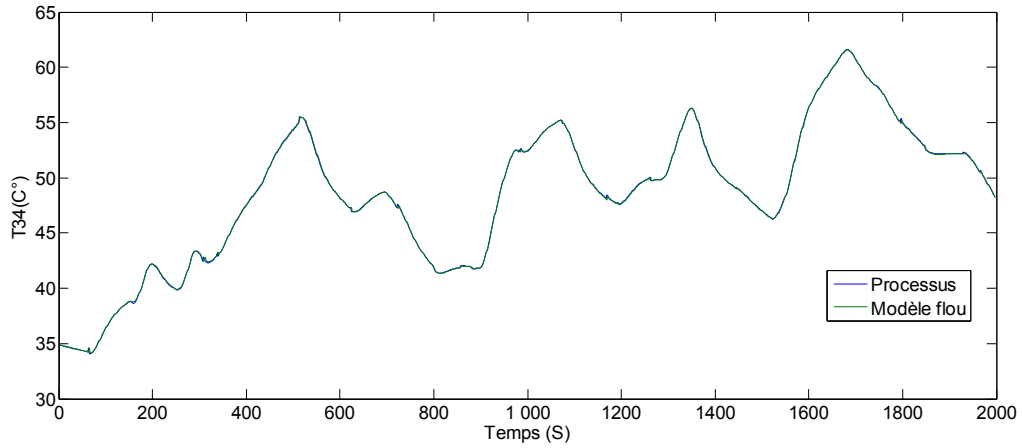


Figure IV. 16 La sortie réelle et la sortie du modèle de prédiction de la température T34.

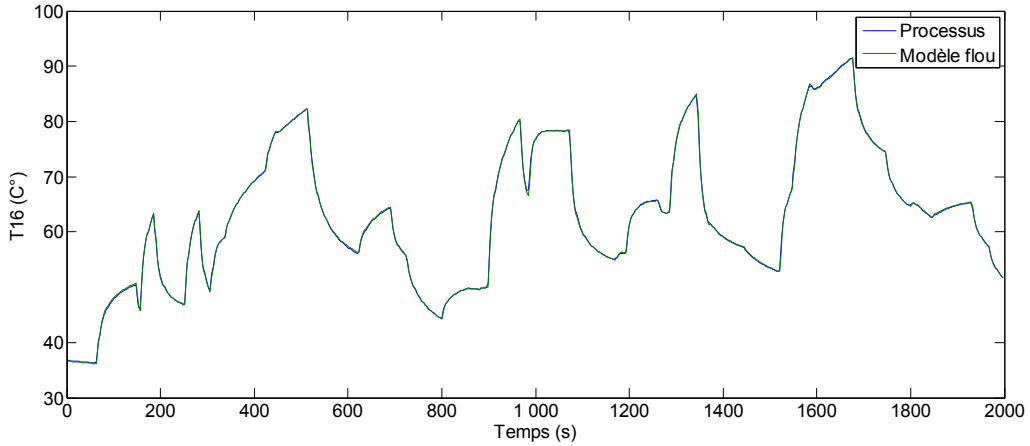


Figure IV. 17 La sortie réelle et la sortie du modèle de prédiction de la température T16.

Il est clair que les erreurs quadratiques moyennes de prédiction sont très proches de zéro, ce qui justifie la capacité de généralisation des modèles flous identifiés par la méthodologie d'identification basée ABC-MCS et g-ABC-MCS.

D'autres tests de validation sont effectués. Il s'agit de données reflétant des dynamiques variables de l'échangeur thermique. Plus précisément, elles correspondent à des mesures prélevées en présence de fuites dans la conduite de circulation d'eau [**47**].

Les cas de test suivants ont été considérés :

- Test 1 : Fuite de 15% introduite à t=924s avec 0≤P≤5 KW.
- Test 2 : Fuite de 25% introduite à t=0s avec 0≤P≤10 KW.
- Test 3 : Fuite de 30% introduite à t=1220s avec 0≤P≤10 KW.
- Test 4 : Fuite de 40% introduite à t=580s avec 0≤P≤7 KW.

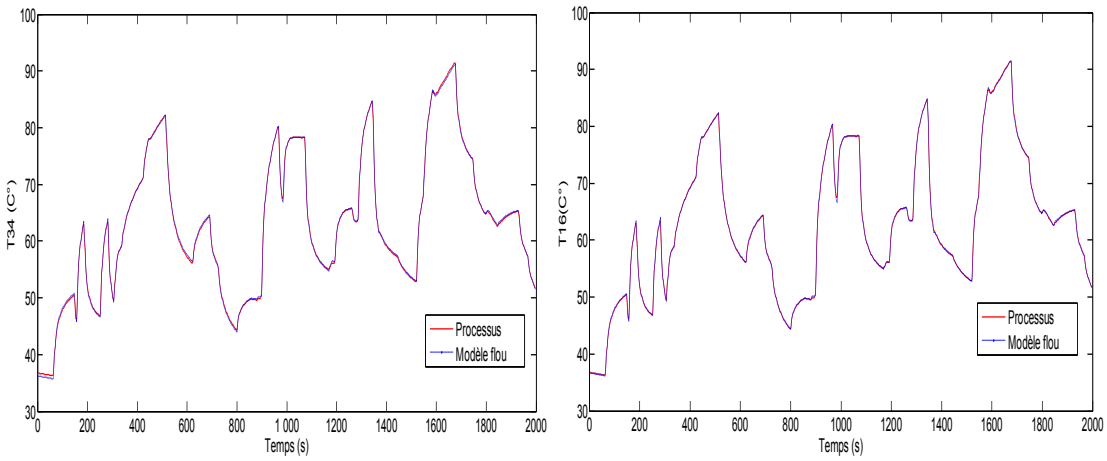


Figure IV. 18 Résultats de prédiction en présence d'une fuite de 15%.

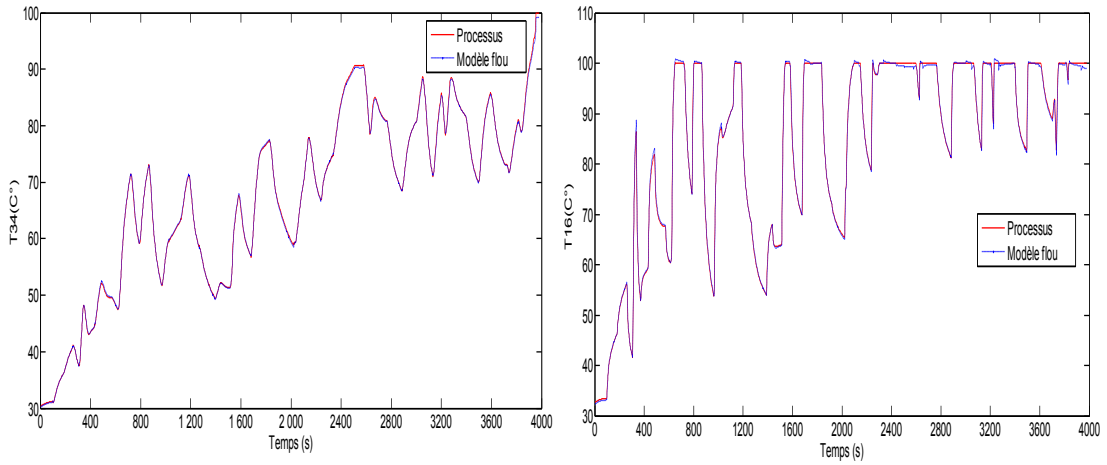


Figure IV. 19 Résultats de prédiction en présence d'une fuite de 25%.

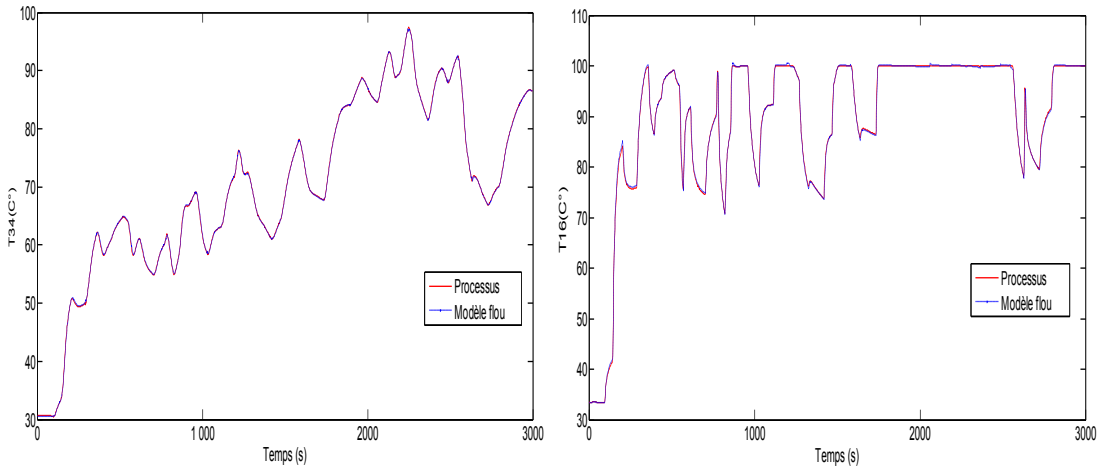


Figure IV. 20 Résultats de prédiction en présence d'une fuite de 30%.

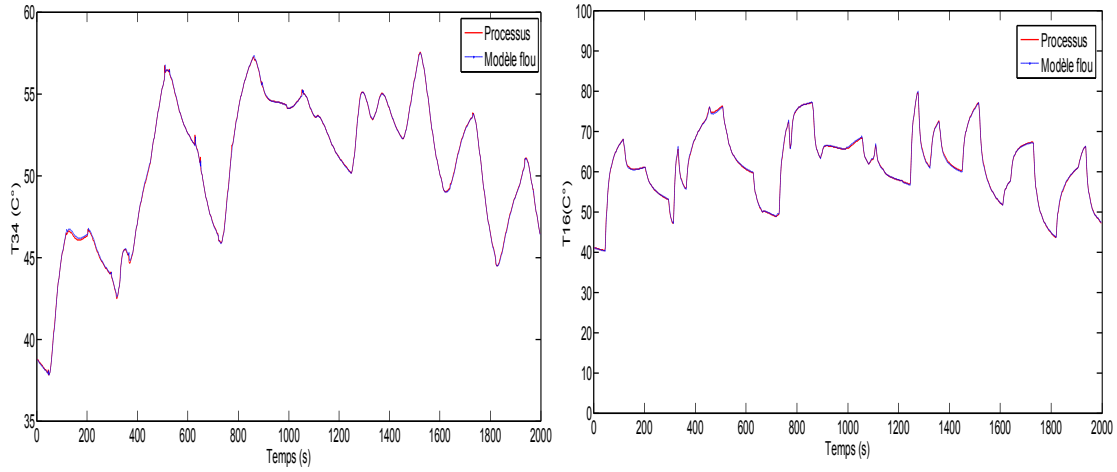


Figure IV. 21 Résultats de prédiction en présence d'une fuite de 40%.

Les sorties du processus réel et celles du modèle flou sont montrées dans les Figure IV. 18 Résultats de prédiction en présence d'une fuite de 15%.IV.19 et IV.20. La bonne capacité de prédiction du modèle flou développé est clairement visible. En effet, il est clair que le modèle flou réalise une très bonne prédiction de la température d'eau T34 et de la température d'air T16 en présence de fuites de différentes amplitudes. Ces fuites traduisent des variations paramétriques et structurelles dans la dynamique de l'échangeur que le modèle identifié arrive bien à les reproduire. Ces performances sont traduites en termes d'erreurs de prédiction moyennes puis indiquées dans le Tableau IV. 15.

Amplitude de la fuite	MSE T34	MSE T16
15%	0.00091	0.00048
25%	0.00068	0.0033
30%	0.00024	0.0020
40%	0.000071	0.00055

Tableau IV. 15 Evaluation des performances du modèle flou identifié en présence de variations paramétriques et structurelles.

IV.8 Conclusion

Dans ce chapitre, nous avons analysé la performance de l'approche d'auto-génération de modèles de systèmes à base de règles floues proposée. L'approche a été validée sur plusieurs problèmes d'identification et de commande. Les résultats obtenus pour l'ensemble des cas étudiés sont très concluants, montrant une amélioration significative des résultats déjà existants.

La méthodologie développée a été également appliquée aux données expérimentales d'un échangeur de chaleur à courants parallèles. C'est un système qui fait partie de la classe des systèmes à paramètres distribués, dont la dynamique est difficile à représenter sous une forme exploitable sur une large zone de fonctionnement. Les résultats d'identification floue obtenus montrent clairement la bonne adéquation entre les mesures (température de l'air et de l'eau) et les sorties des modèles flous identifiés, et ce même dans des situations de variations paramétriques.

Globalement, on peut affirmer que les résultats montrés dans ce chapitre sont très prometteurs, et peuvent former une base pour nos recherches futures.

Conclusions et perspectives

Nous avons présenté dans ce mémoire une nouvelle méthodologie pour l'identification des systèmes à base de règles floues. Il s'agit d'une approche d'auto-génération de bases de règles floues à partir de données numériques (mesures entrée/sortie d'un procédé, par exemple), utilisant l'algorithme de colonies d'abeilles artificielles (ABC). Cette méthodologie trouve son intérêt majeur dans la résolution de problèmes d'identification de systèmes complexes ou de synthèse de lois de commande floue.

Comme il peut être constaté, la méthodologie d'identification floue proposée exploite directement la base de données numérique disponible sur le système à identifier ou à commander. Son application est plutôt systématique ne nécessitant ni la maîtrise des lois physiques, ni la connaissance profonde du processus appréhendé. Il suffit juste d'avoir les données et les adapter conformément à la structure algorithmique de la méthodologie présentée.

La structure algorithmique de l'approche proposée est fondée sur un codage approprié de la structure et des paramètres des modèles flous. Le codage du modèle flou est une étape essentielle car il constitue la plate-forme sur laquelle évoluent l'ensemble ou une partie des paramètres à optimiser. Ces paramètres auront à évoluer simultanément selon le concept du modèle ABC où l'optimalité est synonyme de sites riches en nourriture (nectar), ou du moins ainsi perçu par les abeilles qui les ont prospectés.

Notre contribution a été validée sur plusieurs problèmes d'identification et de commande. Les problèmes benchmarck considérés sont repris de travaux de recherche très récents afin de permettre une comparaison objective. Les résultats obtenus pour l'ensemble des cas envisagés sont très satisfaisants, démontrant une nette amélioration des résultats déjà existants.

L'autre cas d'étude considéré est celui du problème d'identification d'un échangeur de chaleur à courants parallèles. C'est un système qui fait partie de la classe des systèmes à paramètres distribués, dont la dynamique est difficile à représenter sous une forme exploitable

sur une large zone de fonctionnement. Les résultats obtenus sont très convaincants. Nous envisagerons d'utiliser les modèles flous développés dans les futurs travaux pour la conception de systèmes de détection et du diagnostic de défauts.

Malgré les performances démontrées, il est important de noter que le modèle ABC est parfois pénalisant en termes de temps de calcul, surtout pour les problèmes d'optimisation de grande dimension. On peut éventuellement recenser un certain nombre de versions améliorées, néanmoins, l'aspect temps de calcul demeure la contrainte de toute approche d'optimisation basée sur une population, d'où la nécessité d'aborder parallèlement cette problématique surtout pour les besoins d'implémentation en temps réel (en commande, par exemple). De toutes les manières, ce travail ouvre la voie sur plusieurs pistes de recherche qui d'une part le compléteraient et d'autre part, pourraient en tirer profit.

Bibliographie

[1] G.J. Klir and B. Yuan, "Fuzzy sets and fuzzy logic: theory and applications", Prentice Hall, 2008.

[2] Lotfi A. Zadeh, "Fuzzy logic, neural networks, and soft computing," *Fuzzy logic, neural networks, and soft computing*, vol. 37, no. 3, pp. 77-84 , 1994.

[3] HJ Zimmermann,"Fuzzy Set Theory—and Its Applications"*, Kluwer Academic Publishers*, 2001.

[4] R. Babuska, "Fuzzy Modeling for Control," *Kluwer Academic Publishers*, 1998.

[5] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 15, no. 1, pp. 116 - 132, 1985.

[6] P. Eykhoff, "Process parameter and state estimation," *Automatica*, vol. 4, no. 4, pp. 205–224, 1967.

[7] GEP Box, GM Jenkins and GC Reinsel." Time series analysis: forecasting and control", Wiley, 2013.

[8] Lotfi A Zadeh, "Outline of a New Approach to the Analysis of Complex Systems and Decision Processes," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 3, no. 1, pp. 28 - 44, 1973.

[9] H. Mamdani, "An Experiment in Linguistic Synthesis with a Fuzzy," *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1-13, 1975.

[10] J. C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," *Journal of Cybernetics*, vol. 3, no. 3, pp. 32-57, 1973.

[11] D.E. Gustafson and W.C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," *Decision and Control including the 17th Symposium on Adaptive Processes*, 1978.

[12] P. REY, "Les structures de donnéees", Books on Demand, 2010.

[13] Xiaomeng Wang and Christian Borgelt," Representations for Learning, Reasoning and Data Mining (2nd Edition) " , Wiley, (2009).

[14] Yuhui Shi, Meng-Hiot Lim and Bijaya Ketan Panigrahi, *Handbook of Swarm Intelligence : Adaptation, Learning, and Optimization,*, Springer, 2010.

[15] James C. Bezdek, "*Pattern Recognition with Fuzzy Objective Function Algorithms: Recognition with Fuzzy Objective Function Algorithms"* Springer, 1981.

[16] H. Habbi, M. Kidouche, and M. Zelmat, "Data-driven fuzzy models for nonlinear identification of a complex," *Applied Mathematical Modelling*, vol. 35, no. 3, pp. 1470-1482, 2010.

[17] A. Schrijver, "*Theory of Linear and Integer Programming*" Wiley, 1998.

[18] J. Nocedal and S J. Wright, "*Numerical Optimization Second Edition*." Springer, 2006.

[19] D P. Bertsekas, *Dynamic "Programming and Optimal Control, 3rd Edition*." Athena Scientific, 2011.

[20] R Mead JA Nelder, "A Simplex Method for Function Minimization," *The computer journal*, vol. 7, no. 4, pp. 308-313, 1965.

[21] S. Mokhtar and MS. Bazaraa, "*Nonlinear programming: theory and algorithms, 3rd"* Wiley 2013.

[22] Cheng-Jian Lin, "An efficient immune-based symbiotic particle swarm optimization learning algorithm for TSK-type neuro-fuzzy networks design," *Fuzzy Sets and Systems*, vol. 159, no. 21, pp. 2890-2909, 2008.

[23] Chia-Chong Chen, "A PSO-based method for extracting fuzzy rules directly from numerical," *Cybernetics and Systems: An International Journal,*. vol. 37, no.7 , pp. 707-723, 2006.

[24] Chang-Hyun Kim and Min-Soeng Kim, "Evolving Compact and Interpretable Takagi;Sugeno Fuzzy Models With a New Encoding Scheme," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 36, pp. 1006 - 1023, 2006.

[25] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Computers & Operations Research*, vol. 13, no. 5, pp. 533–549, 1986.

[26] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," *Proceedings of the Sixth International Symposium on Micro Machine and Human Science,*, pp. 39 - 43, 1995.

[27] A. Khosla, S. Kumar, and K.K. Aggarwal, "A Framework for Identification of Fuzzy Models through Particle Swarm Optimization Algorithm," *INDICON, 2005 Annual IEEE*, pp. 388 - 391 , 2005.

[28] I-Fang Chungb,Chao-Hsin Hsua and Chia-Feng Juanga, "Automatic construction of feedforward/recurrent fuzzy systems by clustering-aided simplex particle swarm optimization," *Fuzzy Sets and Systems*, vol. 158, no. 18, pp. 1979–1996, 2007.

[29] O. Cordon and F. Herrera, "Ten years of genetic fuzzy systems: current framework and new trends," *IFSA World Congress and 20th NAFIPS International Conference*, vol. 3, pp. 1241 - 1246, 2001.

[30] D.B Fogel, Z Michalewicz and Thomas Baeck, " Evolutionary computation 1: Basic algorithms and operators", Taylor & Francis Group, (2000).

[31] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341-359, 1997.

[32] R. Poli and W. B. Langdon, *Foundations of Genetic Programming*.: Springer , 2002.

[33] Stephen L. Chiu, "Fuzzy model identification based on cluster estimation" *Journal of Intelligent and Fuzzy Systems*, vol. 2, no. 3, pp. 267-278, 1994.

[34] M. Dorigo and L.M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Trans. on Evolutionary Computation*, vol. 1, no. 1, pp. 53 - 66, 1997.

[35] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Trans. on Computational Intelligence Magazine*, vol. 1, no. 1, pp. 28 - 39, 2006.

[36] O. Cordón, I. F. J. Casillas, "Learning cooperative linguistic rules using the best–worst ant system algorithm," *International Journal of Intelligent Systems*, vol. 20, no. 4, pp. 433–452, 2005.

[37] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Technical Report, Kayseri/Türkiye , 2005.

[38] Ioan Cristian Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information Processing Letters*, vol. 85, no. 6, pp. 317–325, 2003.

[39] S. Kwong and G. Zhu, "Gbest-guided artificial bee colony algorithm for numerical," *Applied Mathematics and Computation*, vol. 217, no. 7, pp. 3166–3173, 2010.

[40] J. Espinosa, J. Vandewalle, and V. Wertz, "*Fuzzy Logic, Identification and Predictive Control"*, Springer : Advances in Industrial Control, 2005.

[41] Pei-hong Wang, Jiong Shen, Yu-fei Zhang, Lu Chen Zhi-gang Su∗, "Convenient T–S fuzzy model with enhanced performance using a novel swarm," *Journal of Process Control*, vol. 22, no. 1, pp. 108–124, 2012.

[42] Feng Qiana, Yupu Yangb, Yong Zengb, Haijun Sub Liang Zhaoa, "Automatically extracting T–S fuzzy models using cooperative random learning," *Applied Soft Computing*, vol. 10, no. 3, pp. 938-944, 2010.

[43] Chen-Wei Xu and Yong-Zai Lu, "Fuzzy Model Identification and Self-Learning for Dynamic Systems," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 17, no. 4, pp. 683 - 689 , 1987.

[44] Jian-Qin Chen and Yu-Geng Xi, "A clustering algorithm for fuzzy model identification," *Fuzzy Sets and Systems*, vol. 98, no. 3, pp. 319–329 , 1998.

[45] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling," *Fuzzy Systems*, vol. 1, no. 1, p. 518, 1993.

[46] Chiang Lo Chia-Feng Juang, "Zero-order TSK-type fuzzy system learning using a two-phase swarm intelligence algorithm" *Fuzzy Sets and Systems, vol. 21, no. 1, pp. 2910–2926*, 2008.

[47] Mr Habbi Hacene, "Identification et Surveillance de Processus Dynamiques Complexes par Logique Floue.," Thèse de doctorat UMBB, 2007.