

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique



Université de Jijel
Faculté des Sciences et de la Technologie
Département d'Electronique

MM703

Projet de fin d'études pour l'obtention du Diplôme de
Master II en Electronique

Option : Electronique et Analyse Des Systèmes

Thème :

Conception de Modèles Flous en utilisant
l'Algorithme d'Optimisation par Essaim
Particulaire

Présenté par :

M^r: Hezza Houssem

Encadré par :

M^{me}: Talbi Nesrine

Promotion : Juin 2014.

Remerciements :

Je voudrais remercier tout d'abord, Dieu, pour être mon meilleur confident et pour me permettre de réaliser mes rêves.

La rédaction de ce mémoire est l'occasion pour moi de faire un bilan des deux années passées depuis l'obtention de la licence. Ces années ont été riches de rencontres et je mesure aujourd'hui la chance que j'ai eu d'étudier avec des personnes qui m'ont beaucoup appris tout en partageant avec eux de très bons moments.

Je tiens vivement à remercier mon encadreur M^{me} : Talbi Nesrine je tiens à lui exprimer ma profonde gratitude pour l'aide nécessaire ainsi que pour les conseils bénéfique en vue du bon déroulement du travail durant la réalisation de ce projet.

Je remercie évidemment les enseignements avec qui j'ai eu plaisir à étudier durant ces dernières années: C.Boubakir, K. Kemih, A. Soukko, S.Dib, H. Mordjana, A. Boukabou, ...

Je remercie l'ensemble des membres du jury qui m'ont fait l'honneur d'accepter de lire et de juger ce mémoire.

Mes remerciements sont adressés également au chef du département d'électronique Mr : R. Remmouche.

Je ne saurais terminer sans remercier l'ensemble des collègues et amis qui m'ont aidé et supporté durant ces dernières années. Un grand merci à vous tous.

À mes proches d'ici et d'ailleurs.

Dédicaces

Je dédie ce travail à mes chers parents pour leurs encouragements et leur soutien moral et matériel durant toutes mes années d'études que Dieu le tout puissant me les gardes.

Mon cher père et ma chère mère

A mes sœur : Imane, Ilhame,

Et toute ma famille,

A toutes mes amies,

A tous ceux qui, de près ou de loin n'ont cessé de m'apporter leur soutien durant mes études,

Prénom

Houssem

Table des matières

| | |
|--------------------------------------|-------------|
| Remerciement : | i |
| Dédicace : | ii |
| Table des matières : | iii |
| Liste des figures : | Vi |
| Liste des tableaux : | Vii |
| Liste des abréviations: | Viii |
| Introduction générale : | 1 |

Chapitre 1:Modélisation floue

| | |
|--|------------------------------------|
| 1. Introduction : | 4 |
| 2. Théorie des sous ensembles flous : | 5 |
| 2.1. Historique : | 5 |
| 2.2. Ensembles nets et ensembles flous :..... | 5 |
| 2.3. Types de fonctions d'appartenance : | 6 |
| 2.4. Variable linguistique : | 7 |
| 2.5. Opérations sur les ensembles flous :..... | 7 |
| 2.6. Caractéristiques des ensembles flous : | 8 |
| 2.7. Opérations de base sur les sous ensembles floues :..... | 9 |
| 3.Structure d'un système d'inférence flou : | 10 |
| 3.1.Base de règles :..... | 11 |
| 3.2. Fuzzification :..... | 11 |
| 3.3. Moteur d'inférence : | Erreur ! Signet non défini. |
| 3.4. Défuzzification : | 12 |

| | |
|--|-----------|
| 4. Modélisation floue des systèmes : | 13 |
| 4.1. Modèles flous : | 13 |
| 4.2. Structure générale et différents types de modèles flous : | 13 |
| 5. Avantages et limitations de la logique floue : | 16 |
| 5.1. Avantages : | 16 |
| 5.2. Limitations : | 16 |
| 6. Conclusion : | 17 |

Chapitre 2: L'optimisation par essaim particulaire

| | |
|---|-----------|
| 1. Introduction : | 18 |
| 1.2. Métaheuristique : | 19 |
| 1.2.1. Généralités : | 19 |
| 1.2.2. Évolutif ou non : | 20 |
| 1.2.3. Applications : | 20 |
| 1.2.4. Avantages et inconvénients : | 21 |
| 1.2.5. Variantes : | 22 |
| 1.3. Optimisation par Essaim Particulaire : | 23 |
| 1.3.1. Principe général : | 23 |
| 1.3.2. Description informelle : | 24 |
| 1.3.3. Principales caractéristiques : | 25 |
| 1.3.4. Formalisation : | 26 |
| 1.3.5. Etat et comportement d'une particule : | 27 |
| 1.3.6. Voisinage: | 31 |
| 1.4. Avantages et inconvénients de l'algorithme de base de l'OEP : | 32 |
| 2. Conclusion : | 33 |

Chapitre 3: L'identification des modèles flous par l'OEP

1. Introduction :..... 34

2. Identification d'un modèle flou :..... 34

 2.1. Identification de structure :..... 35

 2.2. Identification de paramètres :..... 35

3. Structure du modèle flou à identifier :..... 36

 3.1. Fuzzification et base de règles floues :..... 36

 3.2. Le moteur d'inférence et la défuzzification :..... 37

4. Simulations :..... 38

 4.1. Système non linéaire de Narendra et Parthasarathy 1:..... 38

 4.2. Identification du Modèle flou :..... 39

 4.3. Système non linéaire de Narendra et Parthasarathy 2:..... 43

5. Conclusions :..... 48

Conclusion générale :..... 50

Bibliographie :..... 51

Liste des figures

| | |
|---|----|
| <i>Figure 1.1 : Présentation d'un ensemble classique (A) et d'un ensemble flou (B).</i> | 6 |
| <i>Figure 1.2 : Exemples de fonctions d'appartenance triangulaire (a), trapézoïdale (b) et gaussienne (c)</i> | 6 |
| <i>Figure 1.3 : Exemple des sous ensembles flous.</i> | 7 |
| <i>Figure 1.4 : Caractéristiques d'un sous ensemble flou A.</i> | 9 |
| <i>Figure 1.5 : Schéma général d'un Système d'inférence flou.</i> | 11 |
| <i>Figure 2.1 : Principe général des métaheuristiques.</i> | 20 |
| <i>Figure 2.2 : Déplacement d'une particule.</i> | 25 |
| <i>Figure 2.3 : un voisinage social.</i> | 29 |
| <i>Figure 2.4 : Topologie en étoile.</i> | 32 |
| <i>Figure 2.5 : Topologie en anneau.</i> | 32 |
| <i>Figure 2.6 : Topologie en rayon.</i> | 33 |
| <i>Figure.3.1 : Structure de modélisation.</i> | 39 |
| <i>Figure. 3.2: Données d'entrée et de sortie du système de Narendra et Parthasarathy.</i> | 40 |
| <i>Figure. 3.3 : Evolution de la fonction d'évaluation.</i> | 41 |
| <i>Figure 3.4 : Disposition et formes des fonctions d'appartenance après optimisation.</i> | 42 |
| <i>Figure 3.5 : Résultats d'identification du modèle flou par OEP.</i> | 43 |
| <i>Figure 3.6 : La validation du modèle flou de Narendra et Parthasarathy.</i> | 44 |
| <i>Figure 3.7 : Signal d'entrée aléatoire $u(t)$.</i> | 45 |
| <i>Figure 3.8 : Evolution de la fonction d'évaluation.</i> | 45 |
| <i>Figure 3.9 : Disposition et formes des fonctions d'appartenance après optimization.</i> | 46 |
| <i>Figure 3.10 : Résultats d'identification du modèle flou par OEP.</i> | 47 |

Figure 3.11 : Signal d'entrée sinusoïdal $u(t)$ 47

Figure 3.12: Test de performance pour une entrée sinusoïdale..... 48

List des Tableaux

| | |
|---|-----------|
| <i>Tableau 3.1: Base de règles du modèle flou.</i> | <i>37</i> |
| <i>Tableau 3.2 : Valeurs spécifiques de l'algorithme d'optimisation OEP.</i> | <i>41</i> |
| <i>Tableau 3.3 : paramètres des prémisses optimaux obtenus par l'OEP.</i> | <i>42</i> |
| <i>Tableau 3.4 : Comparaison des performances de différentes méthodes d'identification des modèles flous de type TS d'ordre zéro pour l'exemple 1.</i> | <i>44</i> |
| <i>Tableau 3.5 : paramètres des prémisses optimaux obtenus par l'OEP.</i> | <i>46</i> |
| <i>Tableau 3.6: Comparaison des performances de différentes méthodes d'identification des modèles flous de type TS d'ordre zéro pour l'exemple 2.</i> | <i>48</i> |

Liste des abbreviations

- COG Méthode de centre de gravité.
- MM Méthode moyenne des maximums.
- TS Takagi – Sugeno.
- OEP Optimisation par Essaim Particulaire.
- PSO Particle Swarm Optimisation.
- REQM Racine de l'erreur Quadratique Moyenne.
- EQM Erreur Quadratique Moyenne.
- ACO Optimisation par Algorithme de Colonie de fourmi.
- GA Algorithme Génétique.

Introduction générale

Les systèmes modernes comme dans l'industrie, l'économie, le marketing et l'économie reposent de plus en plus, souvent, sur des systèmes informatiques, qui sont chargés à aider les utilisateurs au traitement des problèmes et à la prise de décision. La complexité de systèmes informatiques devant être mis en place amène beaucoup de chercheurs à utiliser une modélisation pour la réalisation de ces systèmes. Cette approche permet en effet de tenir compte de la complexité de ces systèmes en extrayant les entités d'actions qui doivent être présentés dans la modélisation du système.

La représentation d'un procédé par un modèle mathématique est caractérisée par des équations différentielles linéaires ou non linéaires reliant les entrées, les états et les sorties. Ces équations différentielles sont parfois très difficiles à trouver, notamment pour les procédés non linéaires complexes qui font apparaître beaucoup de paramètres (moteur thermique, procédé chimique,...). Une solution à ce problème peut être envisagée à partir d'une représentation du comportement du procédé non linéaire par un modèle flou où les relations entre les entrées, les sorties et les états sont exprimées par des règles floues.

L'avancement des recherches dans le domaine du flou a prouvé la capacité et la puissance des modèles flous dans l'identification floue des procédés non linéaires. Plusieurs chercheurs utilisent cette puissance pour représenter la dynamique des procédés non linéaires. La problématique de l'identification floue est basée sur la propriété d'approximation universelle des systèmes flous. En effet, ceux-ci sont capables d'approximer, avec un degré de précision arbitraire fixé, n'importe quelle dynamique non linéaire sur un ensemble compact.

Les systèmes flous sont capables de modéliser des systèmes d'aide à des solutions simples et qui sont : non-linéaires et quelques fois impossible de décrire par un formalisme mathématique. Cependant, le choix optimal de structure et paramètres du classifieur flou (les fonctions d'appartenance, les règles flous, etc.. .) représente une tâche très difficile. Cette tâche requiert du temps, de l'expérience et de l'habileté de la part de l'expérimentateur. En principe, il n'y a pas de méthode générale pour

l'ajustement d'un classifieur flou même si des méthodes métaheuristiques et itératives ont été proposées dans la littérature pour ajuster les fonctions d'appartenance.

L'étude de phénomènes réels est une source nouvelle d'inspiration en ingénierie d'informatique et en biomédicale, où l'étude et la modélisation des systèmes complexes sont très présentes. Ces systèmes sont notamment étudiés en robotique, en classification ou encore en optimisation.

Les métaheuristiques constituent une famille d'algorithmes inspirés de la nature. Ces algorithmes sont particulièrement utiles pour résoudre des problèmes où les algorithmes d'optimisation classiques sont incapables de produire des résultats satisfaisants.

Ce travail est consacré à l'identification des modèles flous de type TS d'ordre zéro pour l'approximation des systèmes non linéaires. L'optimisation de paramètres de fonctions d'appartenances des entrées et les conclusions de règles floues est réalisé par l'algorithme d'optimisation par essaim particulaire, qui est une procédure de recherche basée sur une population d'individus, appelés particules, qui changent leur position (état) avec le temps selon des règles prédéfinies, en vue d'atteindre une solution globale souhaitée.

Le manuscrit de ce mémoire est structuré en trois chapitres :

Le premier chapitre a décrit les principes de base de la logique floue. Nous introduisons la théorie des éléments de bases de la logique floue ainsi que les diverses opérations nécessaires à la modélisation floue. Ensuite, nous donnons la structure d'un système flou (fuzzification, inférence et la défuzzification), ainsi que leurs caractéristiques. Puis nous présentons les deux types de modèles flous les plus utilisés (Mamdani et Takagi-Sugeno), et nous terminons le chapitre par les avantages et les limitations de la logique floue.

Le deuxième chapitre présente les métaheuristiques d'optimisation aussi bien pour les problèmes d'optimisation. Un intérêt particulier est apporté à la méthode d'optimisation par essaim particulaire (OEP), qui constitue le sujet principale de ce travail. De plus, nous présentons une analyse sur les avantages et les inconvénients de

l'algorithme d'optimisation par essai particulaire (OEP), La situation actuelle de recherches de l'OEP.

Dans le troisième chapitre , nous considérons l'application de la théorie des sous-ensembles flous dans le domaine de l'identification des systèmes non linéaires. La phase identification consistera à construire un modèle flou ; cette phase est réalisée en utilisant l'algorithme OEP pour l'identification de paramètres des fonctions d'appartenances et de règles floues du modèle flou de type Takagi-Sugeno d'ordre zéro. Nous nous intéressons aux valeurs modales des fonctions d'appartenance des entrées du contrôleur flou ainsi aux conclusions des règles. L'optimisation consistera donc à minimiser un critère d'erreur entre la consigne et la sortie du système. Pour illustrer les performances de l'algorithme d'optimisation OEP dans le domaine de conception de modèles flous, deux exemples de systèmes non linéaires sont illustrés.

Nous clôturons ce mémoire par une conclusion générale.

Chapitre 1

La modélisation floue

1. Introduction :

L'Intelligence Artificielle (IA) a été l'objectif de plusieurs travaux de recherche émanant de plusieurs communautés scientifiques: philosophes, psychologues, informaticiens, mathématiciens, etc. Son objectif est de développer des systèmes intelligents capables d'imiter certaines capacités des humains reconnues par tous comme étant des caractéristiques fondamentales de l'intelligence.

Cependant, les approches de l'Intelligence artificielle, classifiées principalement dans deux catégories: Symbolisme et Connexionnisme, ont développé séparément des stratégies et des techniques qui ont été plus au moins utilisées avec succès dans le développement de ces systèmes intelligents. Récemment, le domaine de l'intelligence artificielle a connu des progressions remarquables suite au développement de plusieurs théories solidement fondées telles que la Logique Floue , les Algorithmes Génétiques (AG) et la théorie du Chaos.

La logique floue, développée par Zadeh en 1965, est celle qui a contribué, peut être, de plus dans le développement de l'Intelligence Artificielle moderne [1].

2. Théorie des sous ensembles flous :

2.1. Historique [2] :

- 1965 : Introduction de la Théorie des ensembles flous par L.A. Zadeh .
- En 1973, le Pr. Zadeh publie un article (dans l'IEEE Transactions on Systems, Man and Cybernetics) qui mentionne pour la première fois le terme de variables linguistiques (dont la valeur est un mot et non un nombre).
- En 1974, première application industrielle : Régulation floue d'une chaudière à vapeur réalisée par Mamdani.
- En 1980, F.L. Smidth & Co. A/S (au Danemark) met en application la théorie de la logique floue dans le contrôle de fours à ciment. C'est la première mise en œuvre pratique de cette nouvelle théorie.
- Dans les années 80, plusieurs applications commencent à immerger (notamment au Japon).
- 1990: Généralisation de l'utilisation de cette technique :
 - ✓ Appareils électroménagers (lave-linges, aspirateurs, autocuiseurs,...etc).
 - ✓ Systèmes audio-visuels (appareils de photos autofocus, caméscopes à stabilisateur d'images, Photocopieurs,...).
 - ✓ Systèmes automobiles embarqués (BVA, ABS, suspension, climatisation,...etc.).
 - ✓ Systèmes autonomes mobiles.
 - ✓ Systèmes de contrôle/commande dans la plupart des domaines industriels de production.

2.2. Ensembles nets et ensembles flous:

La théorie mathématique des sous ensembles flous est une extension de la théorie des ensembles classiques pour la prise en compte des sous ensembles définis de façon imprécise. A l'inverse de la logique booléenne, la logique floue permet à une condition d'être un autre état que vrai ou faux. La logique floue tient compte de l'imprécision de la forme des connaissances et propose un formalisme rigoureux afin d'inférer de nouvelles connaissances, Ainsi, la notion d'un sous ensemble flou permet de considérer des classes d'objets, dont les

frontières ne sont pas clairement définis, par l'introduction d'une fonction caractéristique prenant des valeurs entre 0 et 1. La théorie des ensembles flous repose sur la notion d'appartenance partielle de chaque élément appartient partiellement ou graduellement aux ensembles flous qui ont été définis.

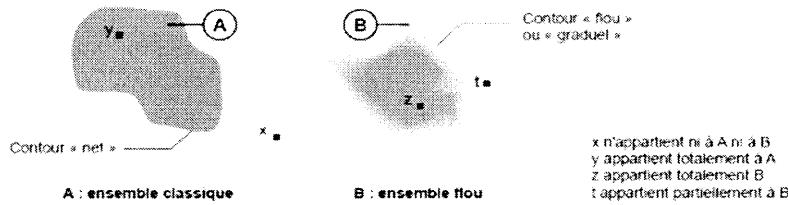


Figure 1.1: Présentation d'un ensemble classique (A) et d'un ensemble floue (B).

La notion d'ensemble flou provient du constat que « très souvent, les classes d'objets rencontrés dans le monde physique ne possèdent pas de critères d'appartenance bien définis ». Mathématiquement, un ensemble flou A est défini sur un univers de discours X, par une fonction d'appartenance $\mu_A(x)$ qui prend cette valeur dans l'intervalle $[0,1]$. Cette fonction donne le degré d'appartenance de chaque élément x à A. En fait, un élément de l'univers de discours peut être entièrement ou partiellement membre de l'ensemble A.

2.3. Types de fonctions d'appartenance :

Chaque ensemble flou peut être représenté par sa fonction d'appartenance. En général, la forme de fonction d'appartenance dépend de l'application. Les fonctions d'appartenance peuvent avoir différentes formes : Triangulaire (figure (1.2.a)), Trapézoïdales (figure (1.2.b)) et en forme de cloche (Gaussiennes) (figure (1.2.c)).

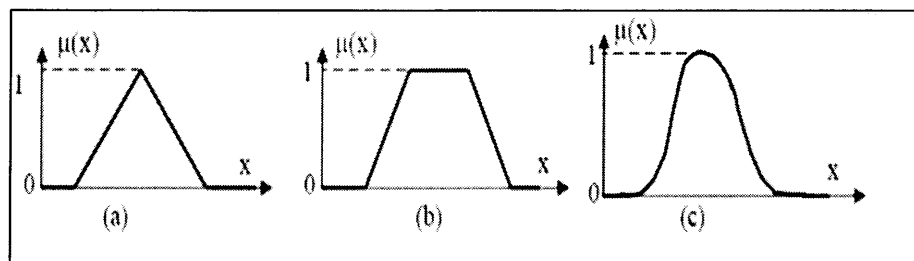


Figure 1.2 : Exemples de fonctions d'appartenance triangulaire (a), trapézoïdale (b) et gaussienne (c).

2.4. Variable linguistique :

C'est une variable dont les valeurs ne sont pas des nombres, mais des mots ou phrases exprimés en langage naturel. Une variable linguistique x est généralement caractérisée par un triplet $x, T(x), X$ où x désigne le nom de la variable, X son univers de discours, et $T(x)$ son ensemble de termes ou valeurs linguistiques. Dans notre cas, l'univers de discours X est toujours un sous-ensemble des réels. Par exemple, si l'erreur est considérée comme variable linguistique défini dans le domaine $X [-10 \ 10]$, ses valeurs linguistiques peuvent être définies comme suit : $T(\text{erreurs}) = \{\text{Négative Grande (NG), Négative Petite (NP), Environ Zéro (EZ), Positive Petite (PP), Positive Grande (PG)}\}$. Ces valeurs linguistiques peuvent être considérées comme des ensembles flous dont les fonctions d'appartenance sont montrées dans la figure 1.3.

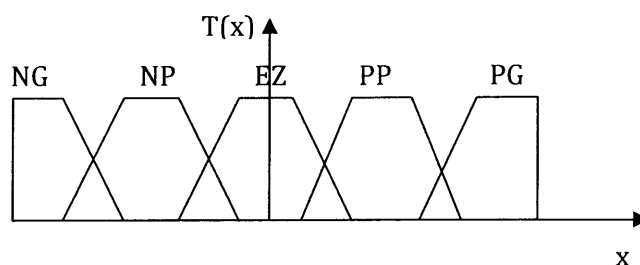


Figure 1.3: Exemple des sous ensembles flous.

2.5. Opérations sur les ensembles flous :

Les opérations les plus importantes dans la théorie des ensembles classiques sont l'union, l'intersection et la complémentation. Ces opérations sont également définies en logique floue grâce aux fonctions d'appartenance. Les formulations utilisées pour l'intersection et l'union sont respectivement des normes triangulaires (T-norme) et des conormes (T-conorme).

- Un ensemble flou est vide si et seulement si sa fonction d'appartenance est nulle sur tout X .
- Deux ensembles flous A et B sont égaux ($A=B$), si et seulement si ;

$$\mu_A(x) = \mu_B(x) \quad \text{pour tout } x \in X \quad (1.1)$$

➤ **Le complément d'un ensemble flou :**

Le complément d'un ensemble flou A noté A' est défini par :

$$\mu_{A'}(x) = 1 - \mu_A(x) \quad (1.2)$$

➤ **L'union floue (T-conorme) :**

L'union de deux ensembles flous A et B est un ensemble flou C ($C = A \cup B$), dont la fonction d'appartenance est liée à celles de A et B par :

$$\mu_C(x) = \max[\mu_A(x), \mu_B(x)] \quad (1.3)$$

➤ **L'intersection floue (T-norme) :**

L'intersection de deux ensembles flous A et B est un ensemble flou C ($C = A \cap B$), tel que sa fonction d'appartenance est donnée par :

$$\mu_C(x) = \min[\mu_A(x), \mu_B(x)] \quad (1.4)$$

2.6. Caractéristiques des ensembles flous :

Il existe 4 caractéristiques essentielles caractérisant l'ensemble flou de l'ensemble booléen : Le type, le noyau, la hauteur et le support.

✚ **Le type :** ils peuvent être présentés par des fonctions d'appartenance triangulaire, gaussienne, trapézoïdale, sigmoïdale ou de type Pic 'singleton', ...etc.

✚ **Le noyau:** Le noyau est défini par :

$$\eta(A) = \{x \in X / \mu_A(x) = 1\} \quad (1.5)$$

C'est l'ensemble booléen de tous les éléments appartenant de façon absolue à l'ensemble flou A. Quand $\eta(A) = v \in]0,1]$, celui-ci est appelée 'valeur modale' de A. Pour la fonction d'appartenance triangulaire, la valeur modale correspond à la valeur du sommet.

- ↓ **La hauteur:** C'est la valeur maximale de la fonction d'appartenance, généralement, égale à 1. Elle est définie par :

$$h(A) = \max_{x \in X} \{\mu_A(x)\} \quad (1.6)$$

- ↓ **Le support:** C'est l'ensemble des éléments de X qui appartiennent au moins un peu à A. Il est défini par:

$$s(A) = \{x \in X / \mu_A(x) > 0\} \quad (1.7)$$

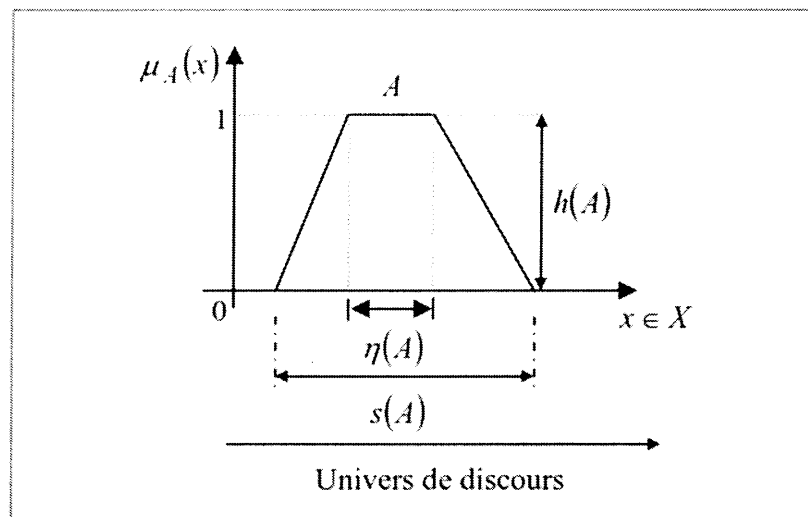


Figure 1.4 : Caractéristiques d'un sous ensemble flou A.

2.7. Opérations de base sur les sous ensembles flous :

Supposons que A et B sont deux sous-ensembles flous définis dans un univers du discours X par les fonctions d'appartenance μ_A et μ_B . On peut définir des opérations ensemblistes telles que l'inclusion, l'intersection, l'union et le complément grâce à des opérations sur les fonctions d'appartenance.

a) Inclusion :

A est dit inclus dans B, propriété que l'on note $A \subseteq B$, si tout élément x de X qui appartient à A appartient aussi à B avec un degré au moins aussi grand :

$$\forall x \in X: \mu_A(x) \leq \mu_B(x) \quad (1.8)$$

b) Intersection :

L'intersection de A et B, que l'on note $A \cap B$, est le sous-ensemble floue constitué des éléments de X affectés du plus petit des deux degrés d'appartenance μ_A et μ_B :

$$\forall x \in X: \mu_{A \cap B} = \min(\mu_A(x), \mu_B(x)) \quad (1.9)$$

c) Union :

L'union de A et B, que l'on note $A \cup B$, est le sous-ensemble flou constitué des éléments de X affectés du plus grand des deux degrés d'appartenance μ_A et μ_B :

$$\forall x \in X: \mu(A \cup B) = \max(\mu_A(x), \mu_B(x)) \quad (1.10)$$

d) Complément :

Le complément de A, que l'on note A^c , est le sous-ensemble floue de X constitué des éléments x lui appartenant d'autant plus qu'ils appartiennent peu à A :

$$\forall x \in X: \mu_{A^c}(x) = 1 - \mu_A(x) \quad (1.11)$$

3. Structure d'un système d'inférence flou :

Un système flou peut être interprété selon deux points de vue : mathématique ou logique. D'un point de vue mathématique, un système flou est une fonction non linéaire reliant un vecteur de données d'entrée à un vecteur de sortie et, de point de vue logique, un système flou est un système à base de connaissance particulière composé de quatre modules principaux, à savoir : la base de règles, la fuzzification, le moteur d'inférence et la défuzzification .

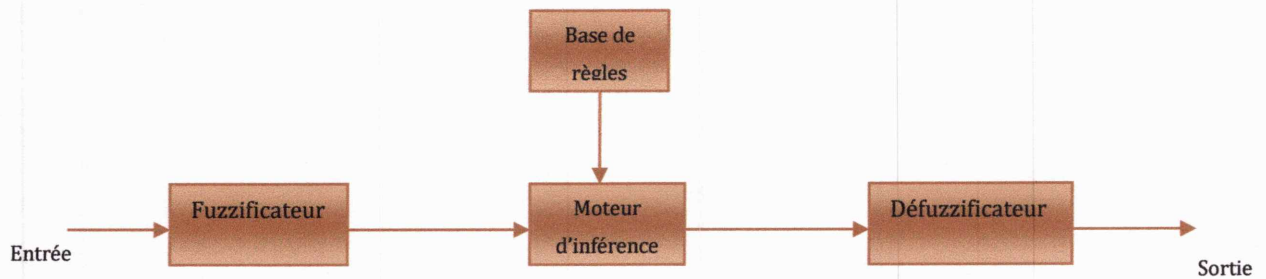


Figure 1.5: Schéma général d'un Système d'inférence floue.

3.1. Base de règles :

La base de règles floues, ou base de connaissances, contient les règles floues décrivant le comportement du système ; elle est le cœur du système entier dans le sens où tous les autres composants sont utilisés pour interpréter et combiner ces règles pour former le système final. Ces règles peuvent être fournies par un expert ou peuvent être extraites de données numériques. Dans les deux cas, les règles prennent la forme « **Si** prémisse **Alors** conclusion ». D'une manière générale, la base de règles d'un système flou doit respecter les conditions de complétude et de consistance afin d'assurer le bon fonctionnement de ce dernier.

Les systèmes à logique floue utilisent une expertise exprimée sous forme d'une base de règles du type: **Si...Alors**

3.2. Fuzzification [3]:

La fuzzification transforme la grandeur physique d'entrée en une grandeur floue. On l'utilise pour activer les règles qui sont exprimées à l'aide de variables linguistiques associées à des ensembles flous. Dans la littérature de la commande floue, deux approches de fuzzification sont généralement utilisées, à savoir : la fuzzification singleton et la fuzzification non-singleton.

La fuzzification des variables est une phase délicate du processus mis en œuvre par la logique floue. Elle est souvent réalisée de manière itérative et requiert de l'expérience. Mais Comment fuzzifier ?

Pour fuzzifier, il faut donner :

- L'univers du discours : plage de variations possibles de l'entrée considérée.
- Une partition en classe floue de cet univers.
- Les fonctions d'appartenances de chacune de ces classes.
- Il faut fuzzifier les entrées et les sorties du processus flou.

3.3. Moteur d'inférence :

Le moteur d'inférence floue transforme, à l'aide des techniques de raisonnement flou, la partie floue issue de la fuzzification en une nouvelle partie floue. En fait, le moteur d'inférence combine les règles floues, en utilisant les principes de la logique floue, pour effectuer une transformation à partir des ensembles flous dans l'espace d'entrée vers des ensembles flous dans l'espace de sortie.

3.4. Défuzzification :

La défuzzification transforme la grandeur floue issue de l'inférence en une grandeur physique. Cependant, il n'existe pas une procédure systématique pour choisir la stratégie de défuzzification. Un critère de choix des méthodes de défuzzification en commande floue est la simplicité des calculs.

Deux principales méthodes de défuzzification sont utilisées:

a. Méthode de centre de gravité (COG) :

La sortie réelle est l'abscisse du centre de gravité de la surface sous la courbe résultante.

$$\text{Sortie} = \frac{\int^u (y \cdot \mu(y) \cdot dy)}{\int^u (\mu(y) \cdot dy)} \quad (1.12)$$

En commande floue, la défuzzification COG est presque toujours utilisée. Elle prend en compte l'influence de l'ensemble des valeurs proposées par la solution floue.

b. Méthode moyenne des maximums (MM) :

La sortie réelle est la moyenne des valeurs de sorties les plus vraisemblables.

$$\text{Sortie} = \frac{\int^s y \cdot dy}{\int^s dy} \quad (1.13)$$

$$\text{Où } s = \{y_0 \in U \mid \mu(y_0) = \text{SUP}(\mu(y))\}$$

La défuzzification MM est plutôt utilisée lorsqu'il s'agit de discriminer une valeur de sortie (Ex: reconnaissance de formes).

4. Modélisation floue des systèmes [4] :

4.1. Modèles flous :

Un modèle flou d'un système est une représentation de son comportement par les concepts de la théorie des sous-ensembles flous. Cette représentation caractérise la relation entre les variables d'entrées et de sorties du système.

Les modèles flous peuvent être considérés comme des modèles logiques qui utilisent des règles du type « **Si...Alors...** » pour établir des relations qualitatives entre les variables du modèle. Ils reposent sur une dualité linguistique/numérique dans laquelle les ensembles flous servent d'interfaces entre les variables qualitatives (linguistiques) impliquées dans les règles et les valeurs numériques présentées par des données entrées/sorties du modèle. La nature des modèles flous, basée sur des règles, permet l'utilisation d'information sous la forme des expressions du langage naturel, ce qui facilite la formalisation de la connaissance des experts ainsi que la transparence et l'interprétabilité du modèle. De plus, les modèles flous ont une structure mathématique flexible capable d'approximer un grand nombre de systèmes

non linéaires complexes avec une bonne précision. Les algorithmes d'apprentissage sont aussi utilisés pour la modélisation floue, ils peuvent même être combinés avec des techniques conventionnelles de régression.

4.2. Structure générale et différents types de modèles flous :

En général les systèmes flous s'appuient sur une représentation de la connaissance sous forme de règles « Si-Alors » qui permettent de représenter les relations entre les variables d'entrée et de sortie dont l'expression générique est de la forme :

Si antécédent **Alors** conséquent

Dans un premier temps et afin de faciliter l'interprétation, on peut considérer l'antécédent (prémisse) comme une description linguistique qui indique les conditions de validité du phénomène représenté. D'une autre part, le conséquent (conclusion) représente le comportement associé aux conditions de validité décrites par l'antécédent. Considérons à titre illustratif la règle suivante :

Si concentration de polluant est élevée **Alors** temps de dégradation est long

Les règles floues établissent des relations logiques entre les variables du système en associant valeurs qualitatives d'une variable (la concentration de polluant est élevée) avec celles d'une autre variable (le temps de dégradation est long). Les valeurs qualitatives ont typiquement une interprétation linguistique, elles sont nommées termes linguistiques. La signification des termes linguistiques par rapport aux variables d'entrée/sortie numériques (concentration de polluant, temps de dégradation) est définie par des ensembles flous appropriés. Dans ce sens, les ensembles flous, ou plus précisément leurs fonctions d'appartenance, fournissent une interface entre les variables numériques d'entrée/sortie et les valeurs linguistiques qualitatives dans les règles.

Selon la structure particulière de la proposition conséquente, on peut distinguer deux types de modèles flous basés sur des règles :

- ✦ Modèle flou linguistique (ou modèle Mamdani), dans lequel l'antécédent et le conséquent sont tout les deux des propositions floues qui utilisent des variables linguistiques avec une base de règles de la forme :

$$\text{Si } x_1 \text{ est } A_1 \text{ et } x_2 \text{ est } A_2 \text{ et } \dots \text{ et } x_n \text{ est } A_n \text{ Alors } y_1 \text{ est } B_1 \text{ et } y_2 \text{ est } B_2 \text{ et } \dots \text{ et } y_m \text{ est } B_m \quad (1.14)$$

Où x_i et y_j sont respectivement les variables d'entrées et de sorties linguistiques et A_i et B_j sont des ensembles linguistiques flous. Ces modèles nécessitent un grand nombre de règles pour approcher le comportement global d'un système : ils possèdent une capacité de représentation limitée.

Les modèles linguistiques ne permettent pas d'intégrer directement des connaissances objectives sur le système qui ne sont pas exprimées par des sous-ensembles flous. Ainsi des connaissances exprimées sous forme analytiques décrivant la structure physique du système ne peuvent pas être pris en compte directement. Sugeno et ses collaborateurs ont proposé une classe de modèles appelés modèles (TS) qui permet de représenter ce type d'information.

- ✦ Modèle flou Takagi-Sugeno (TS), dans lequel le conséquent utilise des variables numériques plutôt que des variables linguistiques, sous la forme d'une constante, d'un polynôme ou de manière plus générale d'une fonction ou d'une équation différentielle dépendant des variables associées à la proposition antécédent.

Soit un système à r entrées et une sortie, le modèle TS est constitué d'une base de règles de la forme :

$$\text{Si } x_1 \text{ est } A_1^{i_1} \text{ et } x_2 \text{ est } A_2^{i_2} \text{ et } \dots \text{ et } x_r \text{ est } A_r^{i_r} \text{ Alors } y = f_i(x_1, x_2, \dots, x_r) \quad (1.15)$$

Où x_1, x_2, \dots, x_r sont les variables numériques d'entrée du modèle flou et f_i est une fonction numérique de l'univers d'entrée $X_1 * X_2 * \dots * X_r$ dans Y .

Chaque règle représente un modèle local sur une région floue d'entrée, ou sur un sous-espace d'entrée. Dans chaque région, le modèle flou est défini par la fonction f_i qui relie les

entrées à la sortie numérique. Le modèle global est constitué par interpolation des modèles locaux.

✚ Dans la littérature, les f_i sont souvent des fonctions linéaires, soit:

$$f_i(x_1, x_2, \dots, x_r) = b_{i0} + b_{i1}x_1 + b_{i2}x_2 + \dots + b_{ir}x_r \quad (1.16)$$

Où $b_{i0}, b_{i1}, b_{i2}, \dots, b_{ir}$ sont des coefficients constants.

Chaque règle peut alors être vue comme un modèle local, linéaire selon les variables d'entrée x_1, x_2, \dots, x_r .

Ces modèles permettent d'approcher le comportement d'un système complexe par une collection de modèles locaux. Ils ont une capacité de représentation importante. En effet, le nombre de règles nécessaires pour approcher un système à un certain degré de précision est généralement réduit [5].

Le problème de modélisation floue ou de construction de modèles flous peut être vu comme un processus d'identification de système. Ce processus est composé de trois phases :

☞ **Phase d'identification de la structure du modèle :**

Cette phase se base sur la détermination de la structure de la base des règles, le nombre des sous ensembles flous et la structure des conclusions des règles.

☞ **Phase d'identification des paramètres du modèle :**

L'identification des paramètres d'un modèle se fait à partir d'un ensemble de données appelé ensemble d'apprentissage. Cette phase permet l'identification des paramètres d'entrée, des paramètres de sortie et de la base des règles.

☞ **Phase de validation du modèle :**

Cette phase sert à tester la validité du modèle d'abord par rapport à l'ensemble d'apprentissage, puis sur de nouvelles données de test.

5. Avantages et limitations de la logique floue :

5.1. Avantages :

- Facilité de construction et d'interprétation des règles : les règles sont formulées de manière naturelle par les experts, comme en symbolique.
- Interpolation numérique entre les règles : lorsque deux règles sont actives car leurs prémisses sont en partie vérifiées, alors la conclusion du système peut prendre une valeur intermédiaire entre les conclusions proposées par celle-ci.
- Intégration de la connaissance à priori.
- Possibilité de combiner règles floues et règles logiques.

5.2. Limitations :

- Construction manuelle des règles suivant l'intuition de l'opérateur ou les connaissances de l'expert.
- Optimisation manuelle des fonctions d'appartenances : nombre de termes linguistiques, choix de la forme des fonctions d'appartenance, positionnement du centre, largeur...
- Optimisation de l'importance de chaque règle : le concepteur peut facilement ajouter de l'importance à une règle en lui affectant un poids, mais ce poids est choisi de manière intuitive comme les paramètres des fonctions d'appartenances.

Dans ce mémoire, on essaye d'éliminer ces limitations avec les méthodes qui seront décrites dans le chapitre suivant.

6. Conclusion :

La théorie des ensembles flous offre un excellent outil pour représenter l'incertitude associée à la tâche de décomposition, en fournissant des transitions douces entre les sous-modèles linéaires et afin d'intégrer divers types de connaissance dans un même cadre. La logique floue et la théorie des ensembles flous servent à la modélisation graduelle et nuancée de la connaissance experte, en proposant un mode de raisonnement approché et analogique, tout en permettant la définition de catégories aux bornes mal définies.

Dans ce chapitre, après avoir introduit les concepts de base concernant la structure générale d'un système d'inférence floue, nous avons présenté les différents types de modèles flous, et différentes phases d'identification des systèmes non linéaires. En effet, au cours des dernières années la discipline a évolué d'une façon graduelle mais décisive, vers une utilisation pratiquement exclusive des systèmes flous dans lesquels le conséquent des règles utilise des variables numériques sous la forme des fonctions (modèle de type TS) plutôt que des variables linguistiques (modèle de type Mamdani). Ces modèles conservent les caractéristiques de transparence et d'interprétabilité linguistique qui distinguent les modèles flous d'autres approches similaires de type boîte noire.

Dans le cas de l'identification floue des systèmes, le formalisme Takagi Sugeno est mieux adapté à une démarche plus systématique pour la construction de modèles non linéaires complexes, grâce à leur bonne capacité d'interpolation numérique et d'"apprentissage" à partir de données, et, dans cette raison, on va utiliser dans ce projet ce type de modèle.

Chapitre 2

L'Optimisation par Essaim Particulaire

1. Introduction :

L'un des principes les plus fondamentaux de notre monde, qui occupe aussi une place importante dans le monde informatique et industriel, est la recherche d'un état optimal. En effet, de nombreux problèmes scientifiques, sociaux, économiques et techniques ont des paramètres qui peuvent être ajustés pour produire un résultat plus souhaitable. Ceux-ci peuvent être considérés comme des problèmes d'optimisation et leur résolution est un sujet central en recherche opérationnelle. Des techniques ont été conçues pour résoudre ces problèmes, notamment les problèmes « difficiles » (par exemple ceux qui présentent de nombreux extrema locaux pauvres), en déterminant des solutions qui ne sont pas rigoureusement optimales, mais qui s'en approchent. Ces méthodes, appelées heuristiques et métaheuristiques, se basent généralement sur des phénomènes physiques, biologiques, socio-psychologiques, et peuvent faire appel au hasard.

Ce chapitre est structuré comme suit : dans la section 1.2, nous présentons les métaheuristiques d'optimisation ; dans la section 1.3, nous détaillons l'algorithme d'Optimisation par Essaim Particulaire (OEP, ou *PSO en anglais*), qui constitue le sujet principal de ce travail de mémoire.

1.2. Métaheuristique :

Une **métaheuristique** est un algorithme d'optimisation visant à résoudre des problèmes d'optimisation difficile (souvent issus des domaines de la recherche opérationnelle, de l'ingénierie ou de l'intelligence artificielle) pour lesquels on ne connaît pas de méthode classique plus efficace.

Les métaheuristices sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, c'est-à-dire l'extremum global d'une fonction, par échantillonnage d'une fonction *objectif*. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème afin d'en trouver une approximation de la meilleure solution.

Il existe un grand nombre de métaheuristices différentes, partant d'une recherche locale simple à des algorithmes de recherche globale complexes. Ces méthodes utilisent cependant un haut niveau d'abstraction, leur permettant d'être adaptées à une large gamme de problèmes différents.

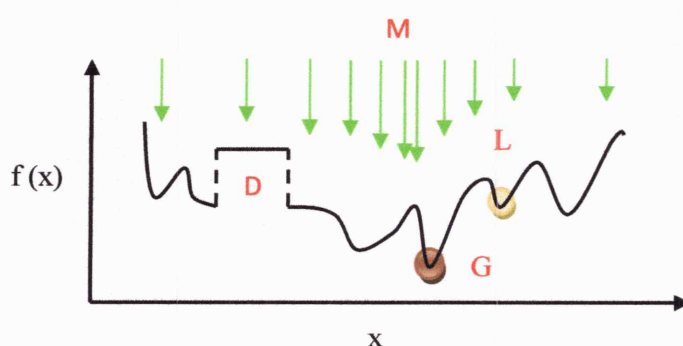


Figure 2.1: Principe général des métaheuristices.

- Les métaheuristices (M) sont souvent des algorithmes utilisant un échantillonnage probabiliste. Elles tentent de trouver l'optimum global (G) d'un problème d'optimisation difficile (avec des discontinuités — D —, par exemple), sans être piégé par les optima locaux (L).

1.2.1. Généralités :

Les **méta-heuristices** sont une forme d'algorithmes d'optimisation stochastique, hybridés avec une recherche locale. Le terme méta est donc pris au sens où les algorithmes peuvent regrouper plusieurs heuristiques. On rencontre cette définition essentiellement dans la

littérature concernant les algorithmes évolutionnaires, où elle est utilisée pour désigner une spécialisation.

Les métaheuristiques sont souvent inspirées par des systèmes naturels, qu'ils soient pris en physique (cas du recuit simulé), en biologie de l'évolution (cas des algorithmes génétiques) ou encore en éthologie (cas des algorithmes de colonies de fourmis ou de l'optimisation par essaims particuliers).

Le but d'une métaheuristique est de résoudre un problème d'optimisation donné : elle cherche un objet mathématique (une permutation, un vecteur, etc.) minimisant (ou maximisant) une fonction *objectif*, qui décrit la qualité d'une solution au problème.

L'ensemble des solutions possibles forme l'espace de recherche. L'espace de recherche est au minimum borné, mais peut être également limité par un ensemble de contraintes.

1.2.2. Évolutionnaire ou non :

On trouve parfois une classification présentant les algorithmes d'optimisations stochastiques comme étant « évolutionnaires » (ou « évolutionnistes ») ou non. L'algorithme sera considéré comme faisant partie de la classe des algorithmes évolutionnaires s'il manipule une population via des opérateurs, selon un algorithme général donné.

Cette façon de présenter les métaheuristiques dispose d'une nomenclature adaptée : on parlera d'opérateurs pour toute action modifiant l'état d'une ou plusieurs solutions. Un opérateur construisant une nouvelle solution sera dénommé générateur, alors qu'un opérateur modifiant une solution existante sera appelé mutateur.

1.2.3. Applications :

Les métaheuristiques sont souvent employées pour leur facilité de programmation et de manipulation. Elles sont en effet facilement adaptables à tout type de problème d'optimisation. Toutefois, elles sont le plus judicieusement employées sur des problèmes d'optimisation difficile, où des méthodes d'optimisation plus classiques (méthodes déterministes, notamment) montrent leurs limites.

Pour tester une métaheuristique, une première étape consiste à utiliser des fonctions mathématiques spécialement conçues [6]. Les algorithmes sont évalués sur la base d'un ensemble de fonctions, plus ou moins difficiles, puis comparés entre eux.

Les métaheuristicues étant généralement stochastiques, les tests doivent en principe être répétés un grand nombre de fois, puis exploités via des méthodes statistiques. Cependant, cette pratique reste relativement peu répandue dans la littérature spécialisée.

1.2.4. Avantages et inconvénients :

- Les métaheuristicues étant très généralistes, elles peuvent être adaptées à tout type de problème d'optimisation pouvant se réduire à une « boîte noire ». Elles sont souvent moins puissantes que des méthodes exactes sur certains types de problèmes. Elles ne garantissent pas non plus la découverte de l'optimum global en un temps fini. Cependant, un grand nombre de problèmes réels n'est pas optimisable efficacement par des approches purement mathématiques, les métaheuristicues peuvent alors être utilisées avec profit.
- La notion d'efficacité se rapporte généralement à deux objectifs contradictoires : la vitesse et la précision. La vitesse est souvent mesurée en nombre d'évaluations de la fonction *objectif*, qui est la plupart du temps la partie la plus gourmande en temps de calcul. La précision se rapporte à la distance entre l'optimum trouvé par la métaheuristique et l'optimum réel, soit du point de vue de la solution, soit de celui de la valeur. Bien souvent, un algorithme rapide est peu précis, et inversement.
- Généralement, un choix doit être fait quant au critère d'arrêt adéquat. Un nombre d'évaluation ou un temps imparti est souvent utilisé, mais on peut également choisir d'atteindre une valeur donnée de la fonction objectif (le but étant alors de trouver une solution associée). Il est également possible de choisir des critères dépendants du comportement de l'algorithme, comme une dispersion minimale de la population de points ou un paramètre interne approprié. En tout état de cause, le choix du critère d'arrêt influencera la qualité de l'optimisation.
- L'utilisation de métaheuristicues peut paraître relativement simple, en première approche, mais il est souvent nécessaire d'adapter l'algorithme au problème optimisé. Tout d'abord, principalement dans le cadre de l'optimisation

combinatoire, le choix de la représentation des solutions manipulées peut être crucial. Ensuite, la plupart des métaheuristiques disposent de paramètres dont le réglage n'est pas nécessairement trivial. Enfin, obtenir de bonnes performances passe généralement par une étape d'adaptation des diverses étapes de l'algorithme (initialisation, notamment). En pratique, seul le savoir-faire et l'expérience de l'utilisateur permet de gérer ces problèmes.

- Il est admis que, d'un point de vue très général, aucune métaheuristique n'est réellement meilleure qu'une autre. En effet, une métaheuristique ne peut prétendre être plus efficace sur tous les problèmes, bien que certaines instances (c'est-à-dire l'algorithme lui-même, mais aussi un choix de paramètres et une implémentation donnée) puissent être plus adaptées que d'autres sur certaines classes de problèmes.
- En dernière analyse, il est parfois possible que le choix de la représentation des solutions, ou plus généralement des méthodes associées à la métaheuristique, ait plus d'influence sur les performances que le type d'algorithme lui-même. En pratique, cependant, les métaheuristiques se montrent plus puissantes que les méthodes de parcours exhaustif ou de recherche purement aléatoire.

1.2.5. Variantes :

1.2.5.1. Liste de métaheuristiques :

Les métaheuristiques les plus connues sont :

- Les algorithmes évolutionnistes, parmi lesquels :
 - les stratégies d'évolution.
 - les algorithmes génétiques.
 - les algorithmes à évolution différentielle.
 - les algorithmes à estimation de distribution.
 - les systèmes immunitaires artificiels.
 - la recombinaison de chemin.
- le recuit simulé.
- les algorithmes de colonies de fourmis.
- L'algorithme d'optimisation par essaims particuliers [8] [7].
- la recherche avec tabous.

- la méthode GRASP.

La recherche dans le domaine étant très active, il est impossible de produire une liste exhaustive des différentes métaheuristiques d'optimisation. La littérature spécialisée montre un grand nombre de variantes et d'hybridations entre méthodes, particulièrement dans le cas des algorithmes évolutionnaires.

1.3. Optimisation par Essaim Particulaire [9] :

1.3.1. Principe général :

L'optimisation par essaim particulaire (OEP), ou Particle Swarm Optimization (PSO) en anglais, 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 (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995 [7].

Il s'inspire de l'origine du monde du vivant, plus précisément du comportement social des animaux évoluant en essaim, tels que les bancs de poissons et les vols groupés d'oiseaux. En effet, on peut observer chez ces animaux des dynamiques de déplacement relativement complexes, alors qu'individuellement chaque individu a une « intelligence » limitée, et ne dispose que d'une connaissance locale de sa situation dans l'essaim.

L'information locale et la mémoire de chaque individu sont utilisées pour décider de son déplacement. Des règles simples, telles que « rester proche des autres individus », « aller dans une même direction » ou « aller à la même vitesse », suffisent pour maintenir la cohésion de l'essaim, et permettent la mise en œuvre de comportements collectifs complexes et adaptatifs.

L'essaim de particules correspond à une population d'agents simples, appelés particules. Chaque particule est considérée comme une solution du problème, où elle possède une position (le vecteur solution) et une vitesse. De plus, chaque particule possède une mémoire lui permettant de se souvenir de sa meilleure performance (en position et en valeur) et de la meilleure performance atteinte par les particules « voisines » (informatrices) : chaque particule dispose en effet d'un groupe d'informatrices, historiquement appelé son voisinage. Un essaim de particules, qui sont des solutions potentielles au problème d'optimisation,

« Survole » l'espace de recherche, à la recherche de l'optimum global. Le déplacement d'une particule est influencé par les trois composantes suivantes :

1. **Une composante d'inertie** : la particule tend à suivre sa direction courante de déplacement.
2. **Une composante cognitive** : la particule tend à se diriger vers le meilleur site par lequel elle est déjà passée.
3. **Une composante sociale** : la particule tend à se fier à l'expérience de ses congénères et, ainsi, à se diriger vers le meilleur site déjà atteint par ses voisins.

La stratégie de déplacement d'une particule est illustrée dans la figure 2.2 :

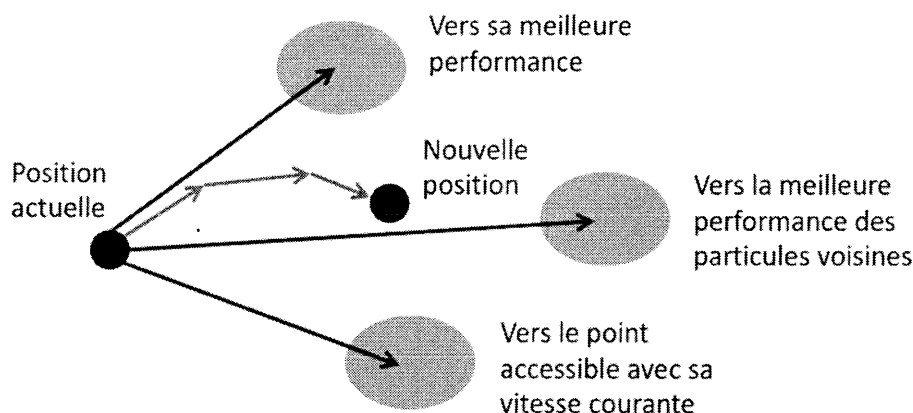


Figure 2.2: Déplacement d'une particule.

1.3.2. Description informelle :

La version historique peut facilement être décrite en se plaçant du point de vue d'une particule. Au départ de l'algorithme, un essaim est réparti au hasard dans l'espace de recherche, chaque particule ayant également une vitesse aléatoire. Ensuite, à chaque pas de temps :

- chaque particule est capable d'évaluer la qualité de sa position et de garder en mémoire sa meilleure performance, c'est-à-dire la meilleure position qu'elle a atteinte jusqu'ici (qui peut en fait être parfois la position courante) et sa qualité (la valeur en cette position de la fonction à optimiser).

- chaque particule est capable d'interroger un certain nombre de ses congénères (ses informatrices, dont elle-même) et d'obtenir de chacune d'entre elles sa propre meilleure performance (et la qualité afférente).
- À chaque pas de temps, chaque particule choisit la meilleure des meilleures performances dont elle a connaissance, modifie sa vitesse en fonction de cette information et de ses propres données et se déplace en conséquence.

Le premier point se comprend facilement, mais les deux autres nécessitent quelques précisions. Les informatrices sont définies une fois pour toutes de la manière suivante :

On suppose toutes les particules disposées (symboliquement) en cercle et, pour la particule étudiée, on inclut progressivement dans ses informatrices, d'abord elle-même, puis les plus proches à sa droite et à sa gauche, de façon à atteindre le total requis.

Il y a bien sûr de nombreuses variantes, y compris celle consistant à choisir les informatrices au hasard, mais celle-ci est à la fois simple et efficace.

Une fois la meilleure informatrice détectée, la modification de la vitesse est une simple combinaison linéaire de trois tendances, à l'aide de coefficients de confiance :

- la tendance « aventureuse », consistant à continuer selon la vitesse actuelle,
- la tendance « conservatrice », ramenant plus ou moins vers la meilleure position déjà trouvé.
- la tendance « panurgienne », orientant approximativement vers la meilleure informatrice.

1.3.3. Principales caractéristiques :

Ce modèle présente quelques propriétés intéressantes, qui font un bon outil pour de nombreux problèmes d'optimisation, particulièrement les problèmes fortement non linéaires, continus ou mixtes (certaines variables étant réelles et d'autres entières) :

- Il est facile à programmer, quelques lignes de code suffisent dans n'importe quel langage évolué.
- Il est robuste (de mauvais choix de paramètres dégradent les performances, mais n'empêchent pas d'obtenir une solution).

1.3.4. Formalisation :

Dans un espace de recherche de dimension D , la particule i de l'essaim est modélisée par son vecteur position : $x_i = (x_{i1}, x_{i2}, x_{i3}, \dots, x_{iD})$, et par son vecteur vitesse : $v_i = (v_{i1}, v_{i2}, v_{i3}, \dots, v_{iD})$.

La qualité de sa position est déterminée par la valeur de la fonction objectif en ce point. Cette particule garde en mémoire la meilleure position par laquelle elle est déjà passée, que l'on note :

$$Pbest_i = (pbest_{i1}, pbest_{i2}, pbest_{i3}, \dots, pbest_{iD}).$$

La meilleure position atteinte par les particules de l'essaim est notée :

$$Gbest_i = (gbest_{i1}, gbest_{i2}, gbest_{i3}, \dots, gbest_{iD}).$$

Nous nous référons à la version globale de l'OEP, où toutes les particules de l'essaim sont considérées comme voisines de la particule i , d'où la notation $Gbest$ (Global best).

✦ Remarque :

Le terme de « vitesse » est ici abusif, car les vecteurs v_i ne sont pas homogènes à une vitesse. Il serait plus approprié de parler de « direction de déplacement ».

Cependant, pour respecter l'analogie avec le monde animal, les auteurs ont préféré utiliser le terme de « vitesse ».

Au départ de l'algorithme, les particules de l'essaim sont initialisées de manière aléatoire dans l'espace de recherche du problème. Ensuite, à chaque itération, chaque particule se déplace, en combinant linéairement les trois composantes citées ci-dessus. En effet, à l'itération $t + 1$, le vecteur vitesse et le vecteur position sont calculés à partir de l'équation (2.1) et de l'équation (2.2), respectivement :

$$v_{i,j}^{t+1} = wv_{i,j}^t + c_1r_{1,i,j}^t [pbest_{i,j}^t - x_{i,j}^t] + c_2r_{2,i,j}^t [gbest_{i,j}^t - x_{i,j}^t], j \in \{1, 2, \dots, D\} \quad (2.1)$$

$$x_{i,j}^{t+1} = x_{i,j}^t + v_{i,j}^{t+1}, j \in \{1, 2, \dots, D\} \quad (2.2)$$

Où w est une constante, appelée coefficient d'inertie ; c_1 et c_2 sont deux constantes, appelées coefficients d'accélération ; r_1 et r_2 sont deux nombres aléatoires tirés uniformément dans $[0, 1]$, à chaque itération t et pour chaque dimension j .

Les trois composantes mentionnées ci-dessus (c-à-d d'inertie, cognitive et sociale) sont représentées dans l'équation (2.1) par les termes suivants :

1. $w \cdot v_{i,j}^t$ correspond à la composante d'inertie du déplacement, où le paramètre w contrôle l'influence de la direction de déplacement sur le déplacement futur.
2. $c_1 r_{1,i,j}^t [pbest_{i,j}^t - x_{i,j}^t]$ correspond à la composante cognitive du déplacement, où le paramètre c_1 contrôle le comportement cognitif de la particule.
3. $c_2 r_{2,i,j}^t [gbest_{i,j}^t - x_{i,j}^t]$ correspond à la composante sociale du déplacement, où le paramètre c_2 contrôle l'aptitude sociale de la particule.

Une fois le déplacement des particules effectué, les nouvelles positions sont évaluées et les deux vecteurs $pbest$ et $gbest$ sont mis à jour, à l'itération $t + 1$, suivant les deux équations (2.3) (dans le cas d'une minimisation) et (2.4) (dans une version globale de l'OEP), respectivement.

Cette procédure est présentée dans l'Algorithme de l'OEP, où N est le nombre de particules de l'essaim :

$$pbest_i(t+1) = \begin{cases} pbest_i(t), & \text{si } f(x_i(t+1)) \geq pbest_i(t) \\ x_i(t+1), & \text{si non} \end{cases} \quad (2.3)$$

$$gbest(t+1) = \arg \min_{pbest_i} f(pbest_i(t+1)), 1 \leq i \leq N. \quad (2.4)$$

1.3.5. Etat et comportement d'une particule :

1.3.5.1. Information utilisée par une particule pour déterminer sa prochaine position :

- ✚ Sa vitesse actuelle.
- ✚ Sa meilleure performance.
- ✚ La meilleure performance de ses voisins.
- ✚ D'où trois types de comportement :
 - Égoïste : suivre sa propre voie.
 - Conservateur : revenir en arrière.
 - Panurgienne : suivre le meilleur de tous.

1.3.5.2. Initialisation :

On pose : une particule = une solution du problème d'optimisation Pour chaque particule :

1. fixer sa position au hasard dans l'espace de recherche.
2. fixer sa vitesse au hasard.
3. définir son voisinage :
 - Géographique (distance), et donc dynamique.
 - Social, et donc fixe.

➤ Comment définir son voisinage ?

✚ Géographique :

- nécessite une notion de distance.
- doit être mis à jour à chaque déplacement.

✚ Dynamique.

✚ Coût computationnel.

✚ Social :

- Fixé une fois pour toute à l'initialisation (Comment?).

➤ Comment définir un voisinage social ?

- Fixé une fois pour toute à l'initialisation sur un cercle Virtuel.
- En fait converge vers un voisinage géographique.

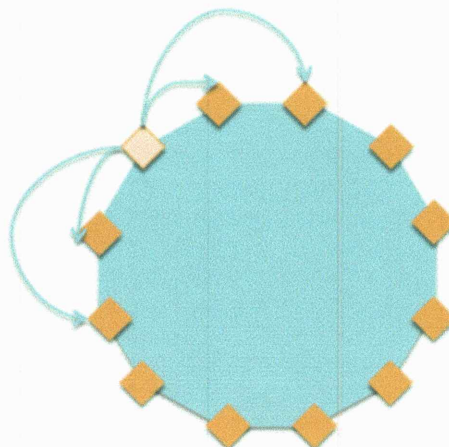


Figure 2.3: un voisinage social.

1.3.5.3. Confinement d'intervalle :

⚡ Problème :

- éviter qu'une particule lors d'un déplacement « sorte » de l'espace de recherche.
- Possible erreurs d'exécution du programme.

⚡ Solution :

- la ramener au point le plus proche dans l'espace.
- modifier sa vitesse.

1.3.5.4. Algorithme OEP standard:

```

1. Initialiser aléatoirement N particules : position et vitesse.
2. Pour chaque particule  $i$ ,  $pbest_i = x_i$ 
3. Calculer  $gbest$  selon (2.4).
4. tant que le critère d'arrêt n'est pas satisfait faire
    Pour  $i=1$  à N faire
        Déplacement de la particule  $i$  à l'aide de (2.1) et (2.2).
        Évaluer les nouvelles positions des particules.
        Si  $f(x_i) < f(pbest_i)$  Alors
             $pbest_i = x_i$ 
        Fin Si
        Si  $f(pbest_i) < f(gbest_i)$  Alors
             $gbest_i = pbest_i$ 
        Fin Si
    Fin pour
Fin tant que

```

Plus une particule est éloignée de la meilleure solution globale et de sa meilleure solution, plus la variation de sa vitesse est importante, afin de faire bouger la particule vers les meilleures solutions.

L'algorithme s'exécute tant qu'un critère de convergence n'a pas été atteint. Cela peut être :

- Un nombre fixe d'itérations.
- En fonction de la fitness.

– Lorsque la variation de vitesse est proche de 0.

1.3.5.5. Algorithme OEP modifié:

Pour garantir une bonne convergence de l'algorithme, plusieurs améliorations ont été proposées. Parmi ces améliorations, on cite celles de la mise sous contraintes de la vitesse de l'essaim et la modification automatique du facteur d'inertie w .

a) Vitesse maximale [10]:

Pour éviter que les particules se déplacent trop rapidement d'une région à une autre dans l'espace de recherche, on fixe une vitesse maximale V_{max} pour $v_{i,j}(t)$ sachant que cette dernière est la vitesse de la particule i au temps t dans la dimension j :

❖ Algorithme de fixation de la vitesse maximale :

Si $v_{i,j}(t) > V_{max}$ **Alors**

$$v_{i,j}(t) = V_{max}$$

Fin Si

Si $v_{i,j}(t) < -V_{max}$ **Alors**

$$v_{i,j}(t) = -V_{max}$$

Fin Si

V_{max} est généralement dépendante de l'échelle du problème.

b) Facteur d'inertie :

Le facteur d'inertie w qui décroît en fonction du temps, est introduit pour contrôler l'influence de la vitesse obtenue au pas précédent. Un grand facteur d'inertie provoque une grande exploration de l'espace de recherche alors qu'un petit facteur d'inertie concentre la recherche sur un petit espace. La convergence est lorsque les valeurs de w est généré selon la contrainte suivante :

$$\frac{1}{2} * (c_1 + c_2) - 1 < w \leq 1 \quad (2.5)$$

1.3.6. Voisinage:

Le voisinage constitue la structure du réseau social. Les particules à l'intérieur d'un voisinage communiquent entre-elles [11]. Différents voisinages ont été étudiés :

- **Topologie en étoile (figure 2.4) :** le réseau social est complet, chaque particule est attirée vers la meilleure particule notée *gbest* et communique avec les autres.

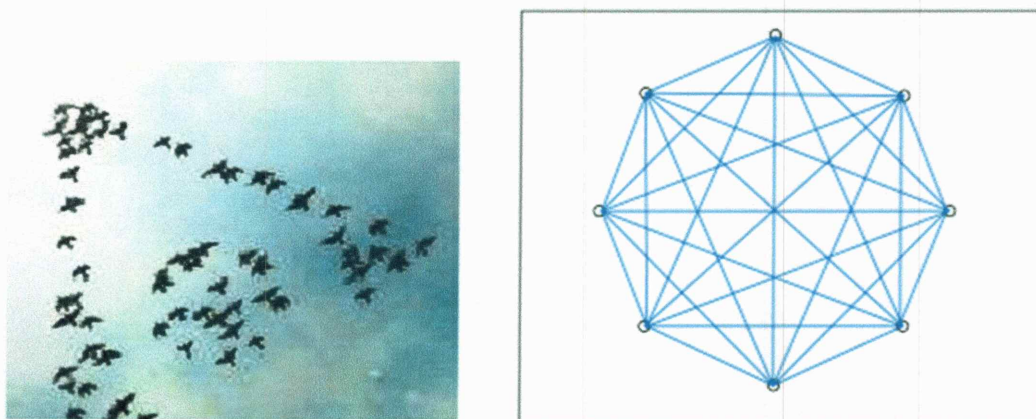


Figure 2.4 : Topologie en étoile.

- **Topologie en anneau (figure 2.5) :** chaque particule communique avec *n* voisines immédiates. Chaque particule tend à se déplacer vers la meilleure dans son voisinage local notée *lbest*.

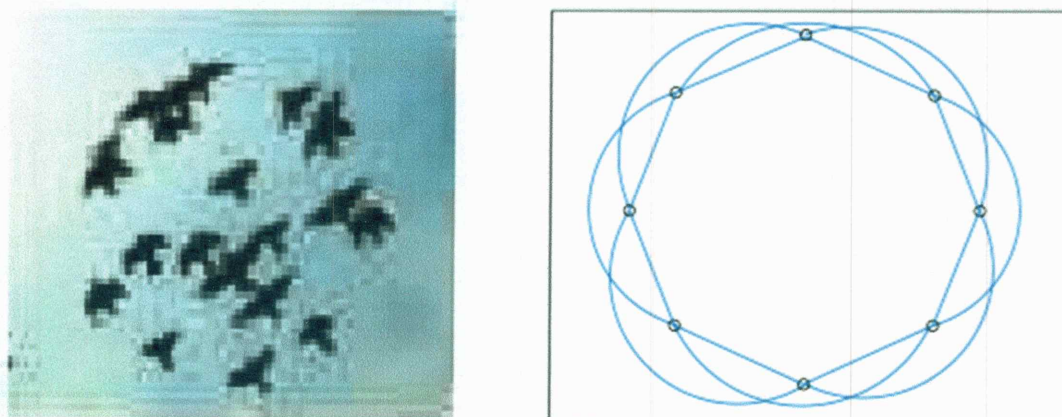


Figure 2.5 : Topologie en anneau.

- **Topologie en rayon (figure 2.6)** : une particule « centrale » est connectée à toutes les autres. Seule cette particule centrale ajuste sa position vers la meilleure, si cela provoque une amélioration l'information est propagée aux autres.

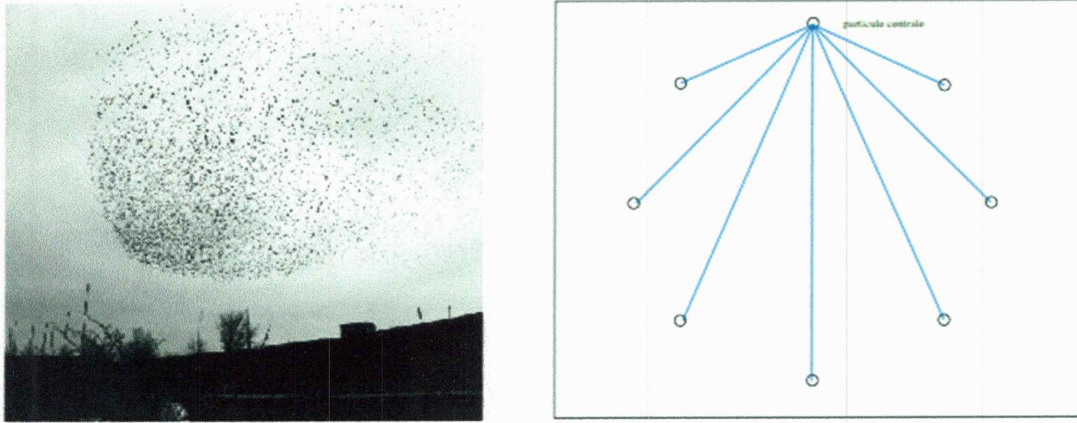


Figure 2.6 : Topologie en rayon.

Le choix de la topologie de voisinage a un grand effet sur la propagation de la meilleure solution trouvée par l'essaim. En utilisant le modèle gbest la propagation est très rapide (c.-à-d. toutes les particules dans l'essaim seront affectées par la meilleure solution trouvée dans l'itération t , immédiatement dans l'itération $t+1$). Cependant, en utilisant les topologies anneaux et rayon ralentiront la convergence parce que la meilleure solution trouvée doit propager par plusieurs Voisinage avant d'affecter toutes particules dans l'essaim. Cette propagation lente convergence.

1.4. Avantages et inconvénients de l'algorithme de base d'optimisation par essaim de particules :

L'OEP n'ont aucun calcul de recouvrement et de mutation. La recherche peut être effectuée par la vitesse de la particule. Pendant le développement de plusieurs générations, seulement la plupart de particule d'opportuniste peut transmettre l'information sur les autres particules, et la vitesse de recherche est très rapide. Par contre, L'OEP a deux principaux inconvénients:

- Le premier inconvénient est que l'essaim peut prématurément converger quand des individus attirent l'essaim vers un optimum local ou lors d'une mauvaise initialisation de la population.

- Le deuxième inconvénient est que les approches stochastiques ont un problème de dépendance. Cette dépendance résulte habituellement des arrangements de paramètres de chaque algorithme. Ainsi, en utilisant de différents arrangements de paramètres pour un seul algorithme de recherche stochastique on obtient des résultats de haute performance. Mais en général, l'arrangement d'un paramètre ne peut être appliqué à tous les problèmes. Ce problème est souvent rencontré dans l'OEP où la modification d'un de ses paramètres peut avoir comme conséquence un grand effet proportionnel. Par exemple, augmenter la valeur du moment d'inertie, w , augmentera la vitesse des particules et provoquera une grande exploration de l'espace de recherche (recherche globale), mais une petite exploitation (recherche locale).

2. Conclusion :

Parmi les métaheuristiques présentées pour résoudre Les problèmes d'optimisation, un intérêt particulier a été porté à la méthode d'optimisation par essaim particulaire (OEP). Cette méthode, qui est inspirée du monde du vivant, a rencontré un succès remarquable depuis sa création, grâce à sa simplicité. Elle présente l'avantage d'être efficace sur une vaste gamme de problèmes, sans pour autant que l'utilisateur ait à modifier la structure de base de l'algorithme. Cependant, l'OEP présente un problème majeur, qui rebute encore certains utilisateurs: le problème de la convergence prématurée, qui peut conduire les algorithmes de ce type à stagner dans un optimum local. De nombreux travaux ont été proposés pour améliorer les performances de l'OEP, tout en essayant de remédier à ce problème. Parmi ceux-ci, nous avons décrit les travaux basés sur l'hybridation avec d'autres méthodes, le mode d'initialisation des particules, la topologie de voisinage, etc...

Dans ce mémoire, l'algorithme d'optimisation par essaim particulaire est utilisé pour l'identification des modèles flous.

Chapitre 3

L'Identification des modèles flous par l'OEP

1. Introduction :

L'avancement des recherches dans le domaine du flou a prouvé la capacité et la puissance des modèles flous dans l'identification floue des procédés non linéaires. Plusieurs chercheurs utilisent cette puissance d'optimisation pour représenter la dynamique des procédés non linéaires. La problématique de l'identification floue est basée sur la propriété d'approximation universelle des systèmes flous. En effet, ceux-ci sont capables d'approximer, avec un degré de précision arbitraire fixé, n'importe quelle dynamique non linéaire sur un ensemble compact [12], [13], [14]. Pour établir un modèle flou, nous devons fixer a priori quelques hypothèses qui permettent de trouver une représentation utile. Dans ce cas, des méthodes d'identification [15], [16], [17] peuvent être utilisées pour déterminer précisément la base de règles. Dans ce chapitre, nous étudions la construction de modèles flous de Takagi Sugeno d'ordre zéro pour les systèmes non linéaires par l'algorithme d'optimisation OEP introduit aux chapitre précédent. Le modèle flou est à base de règles prédéterminée et fixe.

2. Identification d'un modèle flou :

D'une manière générale, le problème d'identification floue consiste à choisir une forme de règles floues appropriée puis à concevoir des lois d'ajustement de paramètres ou de

structure afin que, pour une même entrée, la sortie du modèle flou identifié approche suffisamment la sortie du procédé. La procédure d'identification des modèles flous de Takagi-Sugeno à conclusion constante peut être décomposée en une première phase d'identification de structure (nombre d'ensembles flous et leur répartition sur les univers de discours des entrées) suivie d'une seconde d'identification de paramètres (conclusions numériques dans les règles floues).

2.1. Identification de structure :

L'objectif de l'identification de structure est de déterminer le nombre d'ensembles flous utilisés dans les prémisses de règles ainsi que leur répartition sur les univers de discours des entrées. Dans la littérature, il existe plusieurs techniques pour déterminer la structure d'un modèle flou de Takagi-Sugeno [17]. Dans ce mémoire, on ne traitera pas cette tâche, c'est-à-dire que tout au long de ce travail, nous supposons que la structure du modèle flou (nombre d'ensembles flous, type de fonctions d'appartenance et le nombre de règles floues) est choisie à l'avance.

2.2. Identification de paramètres :

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

Dans ce cas, comme pour le cas linéaire, le problème d'identification est ramené à un problème d'estimation de paramètres. Plusieurs chercheurs ont développé des techniques basées sur les métaheuristiques pour identifier les paramètres d'un modèle flou. Par exemple, la base de règles floues est construite en utilisant la recherche Tabou [18], L'optimisation par les colonies de fourmis est utilisée pour la conception de la base de règles floues de modèles flous [19], l'hybridation des algorithmes génétiques et la recherche Tabou pour optimiser une base de règles floues pour un modèle flou de Takagi-Sugeno d'ordre zéro [20]. Dans ce mémoire, l'algorithme de l'OEP est appliqué pour l'identification de la base de règles floues d'un modèle flou des processus non linéaires.

3. Structure du modèle flou à identifier :

3.1. Fuzzification et base de règles floues :

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

Une règle dans la base de règles floues a la forme générale suivante:

$$R_i: \text{Si } x_1(k) \text{ est } A_1^i \text{ ET } x_2(k) \text{ est } A_2^i \text{ Alors } y_m(t) \text{ est } s^i \quad (3.1)$$

Ou R_i est la $i^{\text{ème}}$ règle, x_j est la variable d'entrée, et y_m est la variable de sortie du modèle flou. A_j^i est une valeur linguistique floue définie par l'expression (3.2), et s^i sont des singletons flous.

$$A_j^i(x_j) = \max\left(\min\left[\frac{x_j - a_{j1}}{a_{j2} - a_{j1}}, \frac{a_{j3} - x_j}{a_{j3} - a_{j2}}\right], 0\right) \quad (3.2)$$

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

| | | | |
|----------------------|---|---|---|
| $X_1 \backslash X_2$ | N | Z | P |
| N | N | N | Z |
| Z | N | Z | P |
| P | Z | P | P |

Tableau 3.1: Base de règles du modèle flou.

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

3.2 Le moteur d'inférence et la défuzzification :

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

$$B_i(\vec{x}) = \prod_{j=1}^2 A_j^i(x_j) \quad (3.3)$$

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

$$y_m = \frac{\sum_{i=1}^r B_i(\vec{x})s_i}{\sum_{i=1}^r B_i(\vec{x})} \quad (3.4)$$

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

L'optimisation d'un modèle flou comprend la détermination en ligne des paramètres de prémisses et de conclusion de chaque règle floue. La figure 3.1 montre la structure de modélisation d'un système non linéaire. $y(t+1)$ est la sortie du système à modéliser et $y_m(t+1)$ est la sortie du modèle flou optimisé.

Notre objectif est de définir les paramètres du modèle flou en minimisant un critère de performance. Deux critères de performance permettant de mesurer la qualité du modèle flou sont:

- l'erreur quadratique moyenne définie par :

$$fit = EQM = \left(\frac{1}{nT}\right) \sum_{k=1}^n e^2(k) \quad (3.5)$$

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

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

$$fit = \sqrt{EQM} = \sqrt{\left(\frac{1}{nT}\right) \sum_{k=1}^n e^2(k)} \quad (3.6)$$

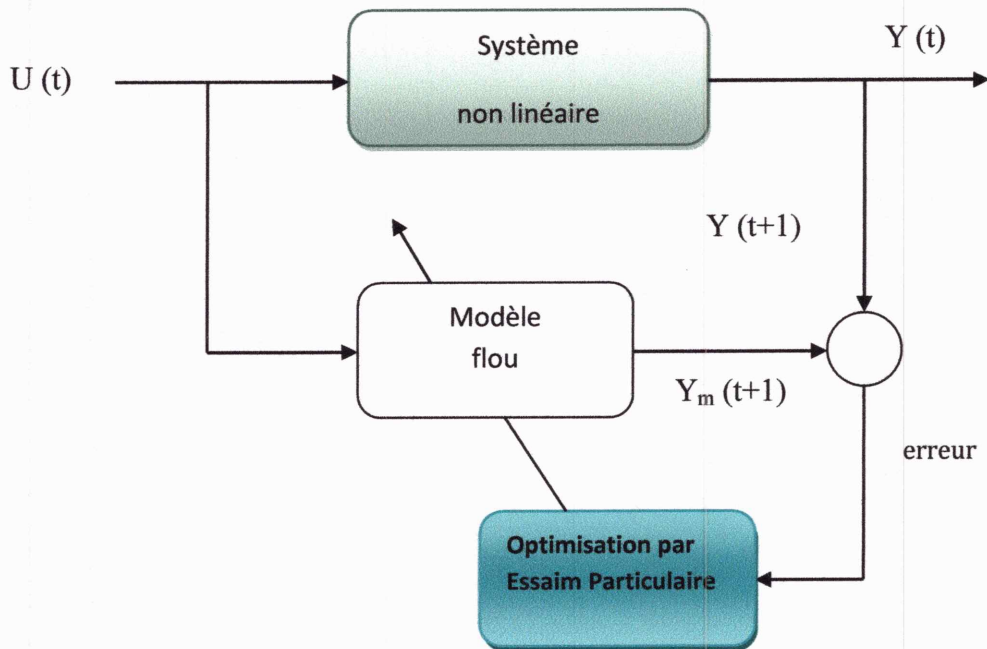


Figure.3.1 : Structure de modélisation.

4. Simulations :

Pour illustrer les performances de l'algorithme d'optimisation OEP dans le domaine de conception de modèles flous, deux exemples sont utilisés : l'exemple de Narendra and Parthasarathy [21], et l'exemple de l'équation non-linéaire :

4.1. Système non linéaire de Narendra et Parthasarathy 1:

On considère un système non linéaire décrit par [21] :

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k) \quad (3.7)$$

Où $u(k)$ et $y(k)$ sont les entrées et sorties du système, respectivement, aux fonction des itérations (figure 3.2).

Les performances de l'algorithme OEP sont évaluées en testant un ensemble de 250 points générés à partir d'un signal d'entrée sinusoïdal de la forme:

$$u(k) = \sin(\pi k/50) * \cos(\pi k/30), \quad 0 \leq k \leq 250 \quad (3.8)$$

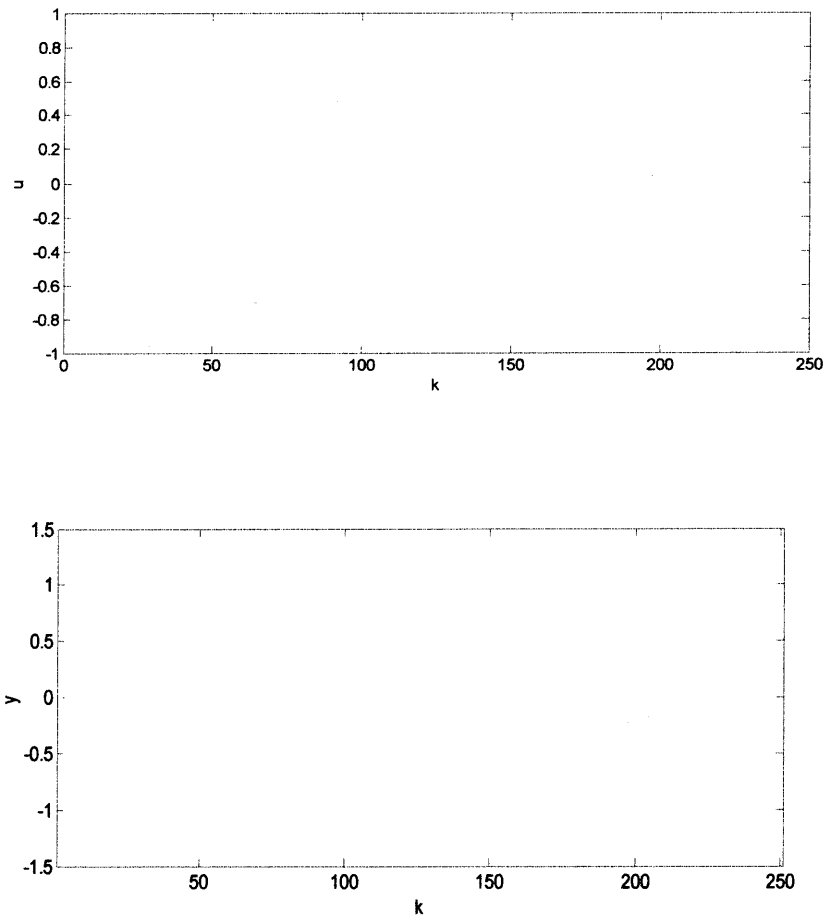


Figure. 3.2: Données d'entrée et de sortie du système de Narendra et Parthasarathy.

La fonction d'évaluation est donnée par :

$$fit = \sqrt{\frac{\sum_{k=0}^{249} (y_m(k+1) - y(k+1))^2}{250}} \quad (3.9)$$

4.2. Identification du Modèle flou :

Les valeurs spécifiques de l'algorithme d'optimisation OEP pour l'identification du modèle flou de Narendra et Parthasarathy sont données par le tableau 3.2. L'évolution de la

fonction coût au cours de l'exécution de l'algorithme d'optimisation OEP est illustrée par la figure 3.2.

| Caractéristique | Valeur |
|--------------------------------|--------|
| Nombre de particules (N) | 10 |
| Nombre d'itérations (max_iter) | 100 |
| C_1, C_2 | 1.5 |
| w | 0.9 |
| Univers de discours | [-1,1] |

Tableau 3.2 : Valeurs spécifiques de l'algorithme d'optimisation OEP.

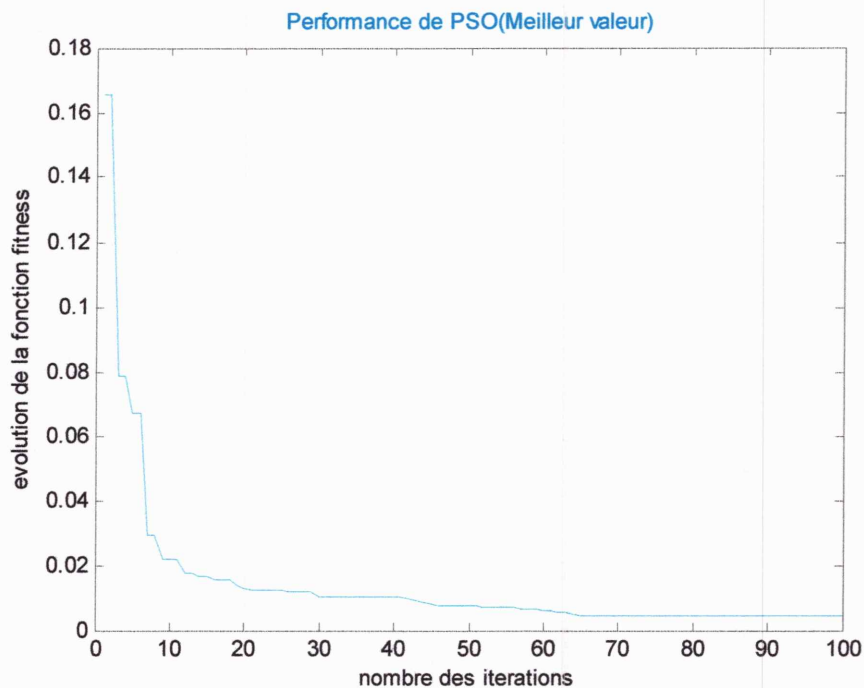


Figure. 3.3 : Evolution de la fonction d'évaluation.

Le tableau suivant donne les paramètres optimaux obtenus par l'OEP après 100 itérations :

| Les points modaux des fonctions d'appartenance | | | |
|--|--------------------------|-------------------------|-------------------------|
| u | -0.42 (C ₁₁) | -0.10(C ₁₂) | 0.54 (C ₁₃) |
| y | -0.72 (C ₂₁) | 0.5 (C ₂₂) | 0.83 (C ₂₃) |

Tableau 3.3 : paramètres des prémisses optimaux obtenus par l'OEP.

La figure suivante représente les formes des fonctions d'appartenance des entrées et de sortie trouvées après optimisation :

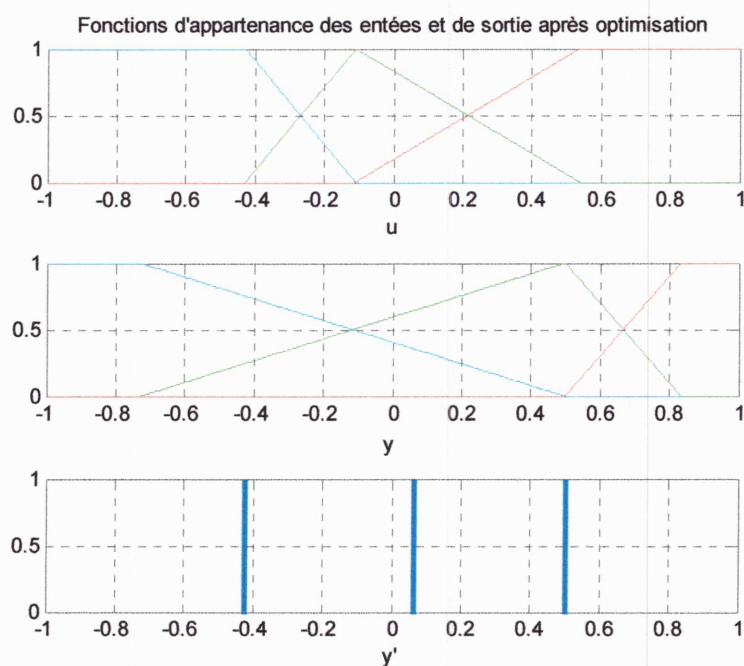


Figure 3.4 : Disposition et formes des fonctions d'appartenance après optimisation

La figure 3.4, montre la sortie originale du système ainsi que la réponse du modèle flou optimisé. On remarque une convergence de la réponse du modèle flou vers la sortie désirée.

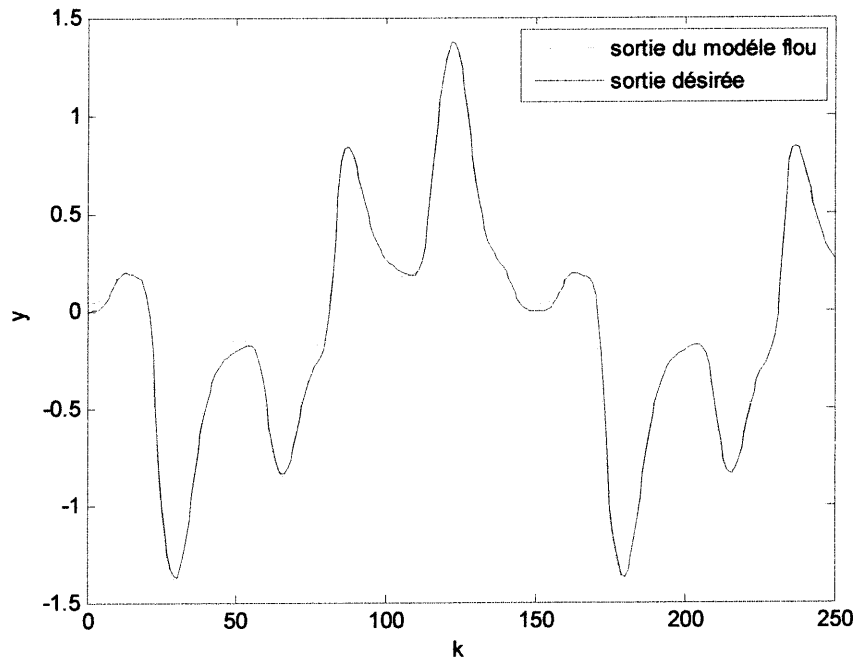


Figure 3.5 : Résultats d'identification du modèle flou par OEP.

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

$$u(k) = 0.5 * \left(0.6 * u(k - 1) + 0.6 * \sin\left(\frac{2\pi k}{25}\right) + 0.8 * \sin\left(\frac{\pi k}{32}\right) \right) \quad (3.10)$$

La figure 3.6 montre le test de performance de l'algorithme OEP. D'après la figure 3.6, il est clair que l'algorithme d'optimisation OEP est capable concevoir un modèle flou optimisé avec une bonne approximation du système à identifier.

Les résultats d'identification du modèle flou de Narendra et Parthasarathy sont comparés avec ceux existés dans la littérature. Le tableau 3.4 montre les résultats d'optimisation par ACO, EGA, OEP pour le même problème d'identification. En se basant sur les résultats du tableau 3.4, on note que la fonction d'évaluation est minimale en comparant avec les autres méthodes.

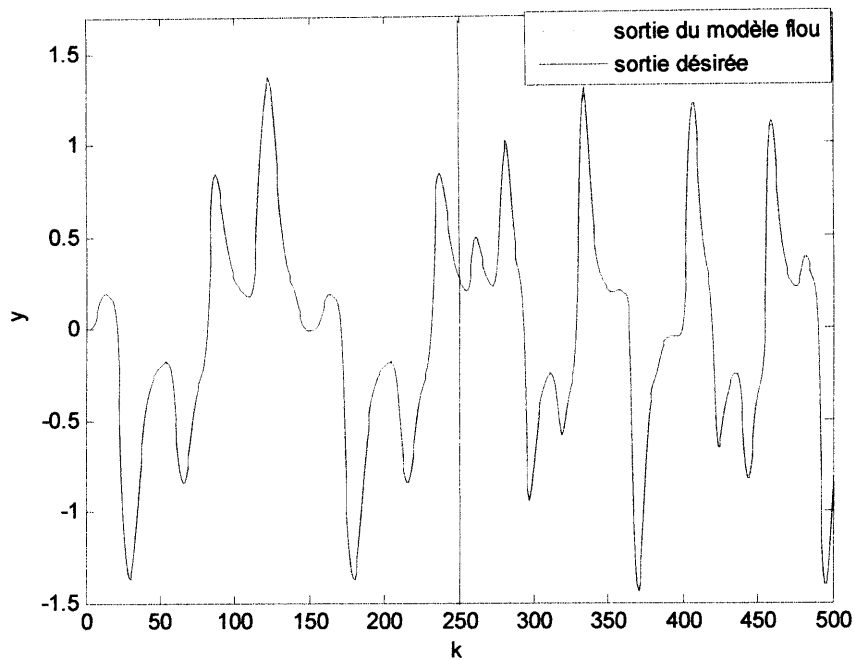


Figure 3.6 : La validation du modèle flou de Narendra et Parthasarathy.

| Méthode d'optimisation | Nombre de règles floues | REQM |
|------------------------|-------------------------|-------|
| ACO | 5 | 0.130 |
| EGA | 5 | 0.041 |
| OEP | 9 | 0.004 |

Tableau 3.4: Comparaison des performances de différentes méthodes d'identification des modèles flous de type TS d'ordre zéro pour l'exemple 1.

4.3. Système non linéaire de Narendra et Parthasarathy 2:

Cet exemple est pris de Narendra et de Parthasarathy [22]. C'est un système dans fortement non-linéaire de second ordre défini par l'équation suivante:

$$y(k) = \frac{y(k-1) \cdot y(k-2) \cdot (y(k-1) + 2.5)}{1 + y^2(k-1) + y^2(k-2)} + u(k) \quad (3.11)$$

Avec les entrées u et y ont été choisis dans les intervalles $[-2, 2]$, $[-5, 5]$ respectivement.

Les performances de l'algorithme OEP sont évaluées en testant un ensemble de 500 points générés à partir d'un signal d'entrée aléatoire uniformément distribué (figure (3.7)). L'évolution de la fonction coût est illustrée par la figure (3.8). On note une bonne convergence de l'algorithme vers la solution globale avec une bonne approximation du modèle réel.

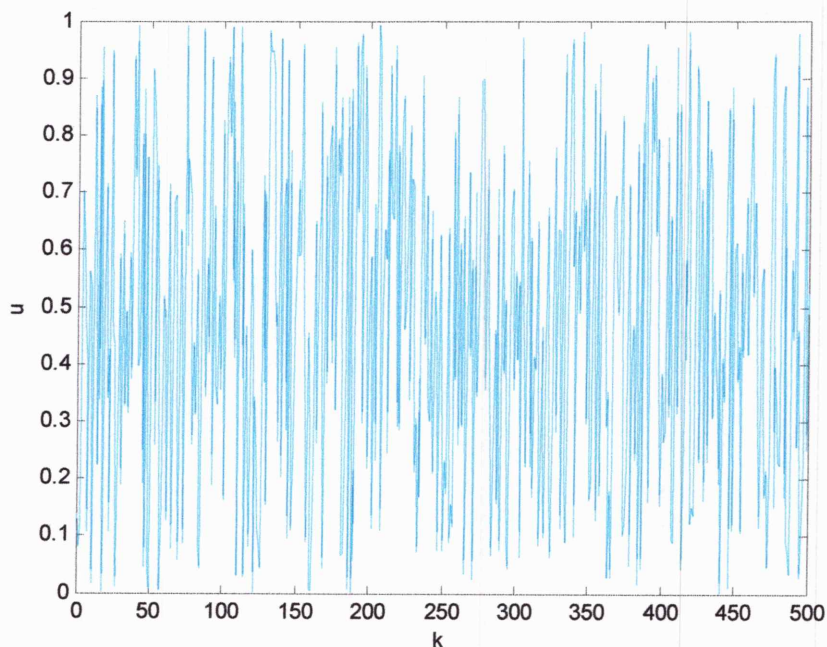


Figure 3.7 : Signal d'entrée aléatoire $u(t)$.

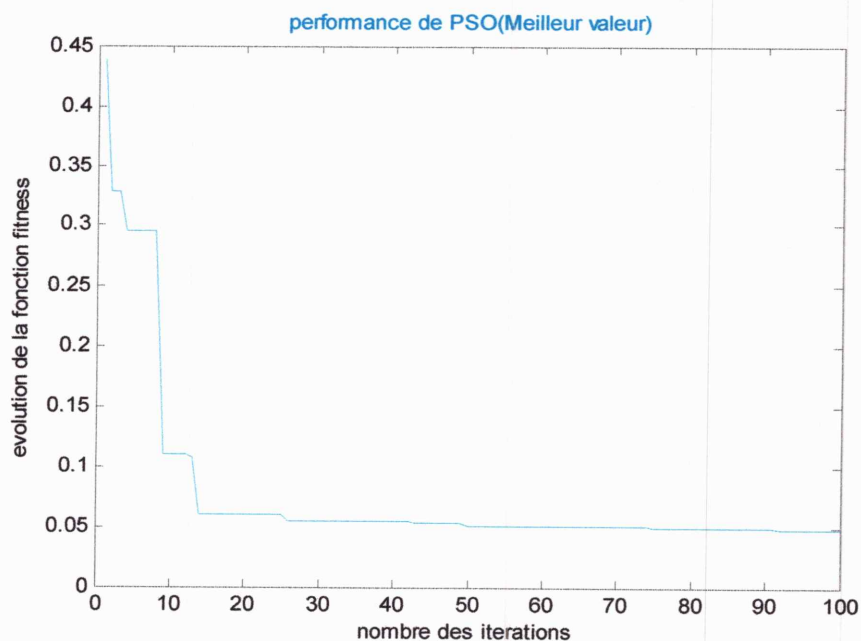


Figure 3.8 : Evolution de la fonction d'évaluation.

Le tableau suivant donne les paramètres optimaux obtenus par l'OEP après 100 itérations :

| Les points modaux des fonctions d'appartenance | | | |
|--|--------------------------|------------------------|-------------------------|
| u | -0.82 (C ₁₁) | 0.5 (C ₁₂) | 0.70 (C ₁₃) |
| y | -0.02 (C ₂₁) | 0.7 (C ₂₂) | 1.00 (C ₂₃) |

Tableau 3.5 : paramètres des prémisses optimaux obtenus par l'OEP.

La figure suivante représente les formes des fonctions d'appartenance des entrées trouvées et de sortie après optimisation :

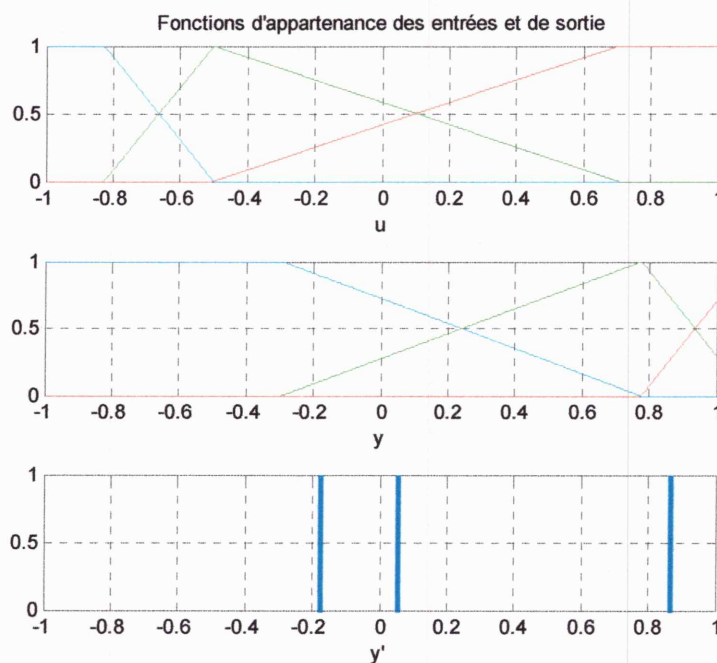


Figure 3.9 : Disposition et formes des fonctions d'appartenance après optimization.

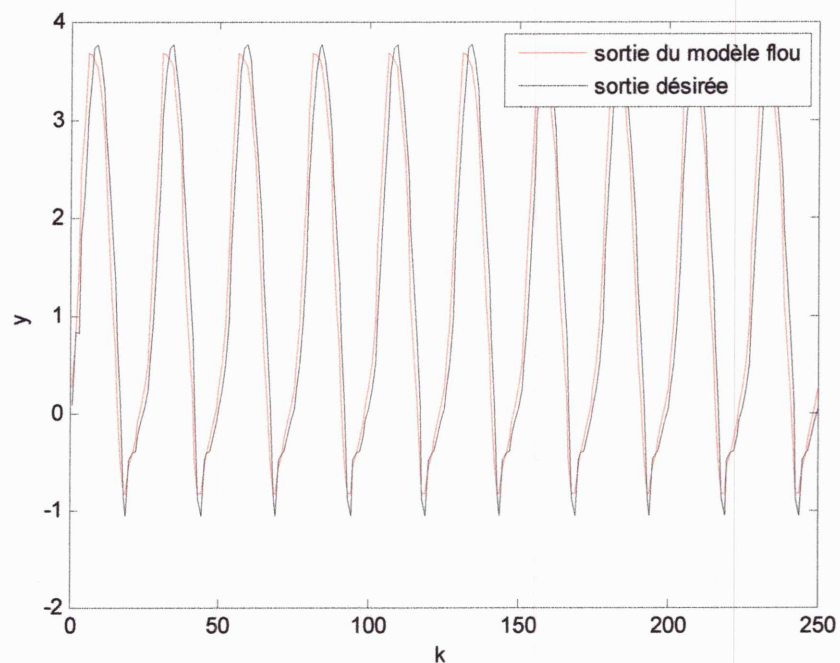


Figure 3.10 : Résultats d'identification du modèle flou par OEP.

Le nombre de règle du modèle flou est 9. La figure 3.10, montre la sortie originale du système ainsi que la réponse du modèle flou optimisé. On remarque une convergence de la réponse du modèle flou vers la sortie désirée.

Le test de performance est réalisé en insérant un autre signal d'entrée $u(k)$ le système (figure 3.11), ce signal est un signal sinusoïdal :

$$u(k) = \sin\left(2\pi \cdot \frac{k}{25}\right). \quad (3.12)$$

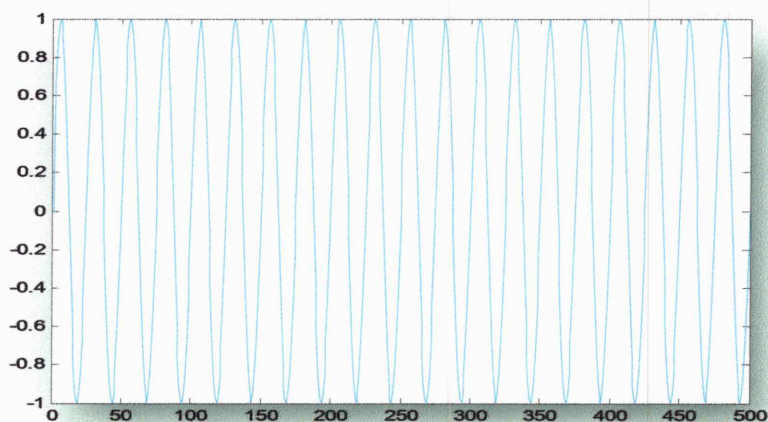


Figure 3.11 : Signal d'entrée sinusoïdal $u(t)$.

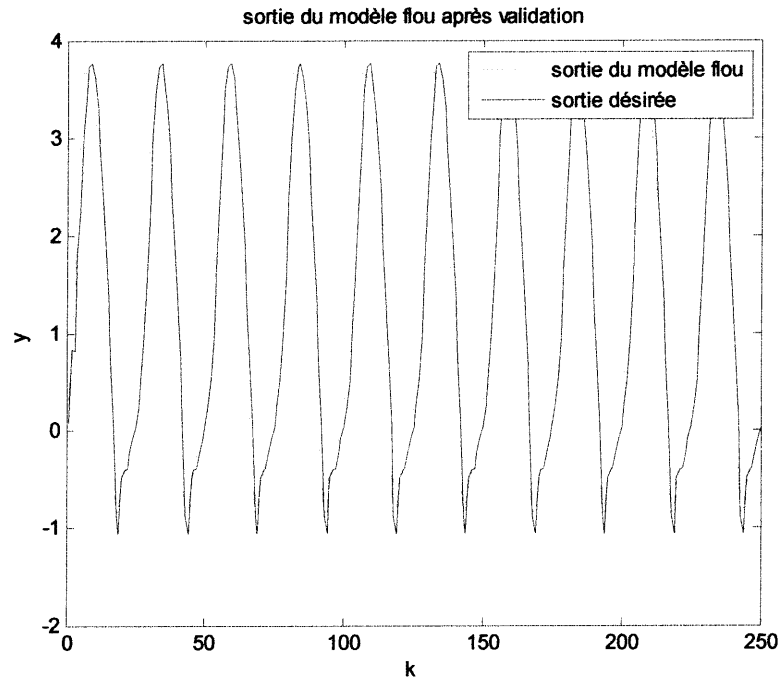


Figure 3.12: Test de performance pour une entrée sinusoïdale.

D'après la figure (3.12), il est clair que l'algorithme d'optimisation OEP est capable concevoir un modèle flou optimisé avec une bonne approximation du système à identifier.

Les résultats d'identification du modèle flou du deuxième exemple non linéaire sont comparés avec ceux existés dans la littérature. Le tableau (3.6) montre les résultats d'optimisation par [Model de Sugeno (1991,1993), Model de Farag (1998), Model de Wang (1999)] pour le même problème d'identification. En se basant sur les résultats du tableau, on note que la fonction d'évaluation est minimale en comparant avec les autres méthodes.

| Méthode d'optimisation | Nombre de règles floues | REQM |
|-------------------------------------|-------------------------|--------|
| Model de Sugeno (1991,1993) [23,24] | 12 | 0.5072 |
| Model de Farag (1998) [25] | 75 | 0.0374 |
| Model de Wang (1999) [26] | 8 | 0.6184 |
| OEP | 9 | 0.0236 |

Tableau 3.6: Comparaison des performances de différentes méthodes d'identification des modèles flous de type TS d'ordre zéro pour l'exemple 2.

5. Conclusions :

Dans ce chapitre l'algorithme d'optimisation OEP a été appliqué pour l'optimisation des prémisses et de conclusion de la base de règle floues pour une bonne approximation des systèmes non linéaires. Les résultats ont montré que l'approche proposée pour l'identification des modèles flous accomplit efficacement les performances désirées. En comparant l'approche proposée avec d'autres existées dans la littérature, l'algorithme d'optimisation OEP A de bonnes performances (grande précision), en utilisant une structure simple du modèle flou

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



Conclusion générale

Les fondations nécessaires à la maîtrise des techniques intelligentes sont établies. Après avoir introduit des concepts de base sur lesquelles reposent les systèmes d'inférence floue, la structure générale d'un SIF (système d'inférence floue) ainsi que les différents types de modèles flous ont été présentés.

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

Les métaheuristiques, comprenant notamment la méthode d'optimisation par essaim particuliaire est un algorithme d'optimisation de type stochastique qui progresse vers un optimum par échantillonnage d'une fonction cout dont le but est la résolution de problèmes d'optimisation. Grâce à la simplicité et la souplesse de leur principe, il peut être un outil d'optimisation et de conception des systèmes de contrôle des processus complexes dont la dynamique n'est pas encore maîtrisée. De plus les différents tests de simulation ont montré que les systèmes obtenus sont robustes en ce sens qu'ils satisfont les critères de performances dans les limites préétablies.

Bibliographie :

[1] : Université Polytechnique de Bucarest (faculté d'automatique et d'informatique, laboratoire AC PC) : projets de recherche, accord de coopération + école d'été en automatique à Bucarest (cours : « logique floue et commande »), 21_25 mai 2012.

[2] : Claude Rosental, « Histoire de la logique floue. Une approche sociologique des pratiques de démonstration », Revue de Synthèse, vol. 4, 4, octobre-décembre 1998, pp. 575-602.

[3] : S. Zribi Boujelbene, D. Ben Ayed Mezghani et N. Ellouze « Systèmes à Inférences Floues pour la Classification Phonémique », 4th International Conférence: Sciences of Electronique, Technologies of Information and Télécommunications, March 25-29, 2007 – Tunisia.

[4] : Victor-Hugo Grisales Palacio « Modélisation et Commande floues de type Takagi-Sugeno appliquée a un bioprocédé de traitement des Eaux usées », thèse de Doctorat de l'Université Paul Sabatier - Toulouse III et l'Université de los Andes, Colombie, le 22 février 2007.

[5] :Mr. Merabti Halim « Etude des systèmes flous à intervalle », Mémoire de magistère, Université de Mentouri de Constantine.

[6]: W. Dullaert, M. Sevaux, K. Sörensen et J. Springael, « Applications of metaheuristics », numéro spécial de l'European Journal of Operational Research, n° 179, 2007.

[7]: Kennedy, J. and Eberhart, R. (1995). « Particle Swarm Optimization », In Proceedings of IEEE International Conference on Neural Network, volume IV, pages 1942–1948.

- [8] : Eberhart, R. C. et Kennedy, J., « *A new optimizer using particle swarm theory* », Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan. pp. 39-43, 1995.
- [9]: Toumi Abida « restauration Adaptative d'image par les méthodes intelligentes », thèse de doctorat en sciences, Université Mohamed Khider – Biskra, Faculté des Sciences et de la technologie, 05 Février 2013.
- [10]: Van den Bergh, F. (2002). « An Analysis of Particle Swarm Optimizers », PhD thesis, Department of Computer Science, University of Pretoria.
- [11] : Abbas EL DOR «Perfectionnement des algorithmes d'Optimisation par Essaim Particulaire. Applications en segmentation d'images et en électronique », THESE DE Doctorat en informatique, Université Paris-Est, Thèse soutenue le 5 décembre 2012.
- [12]: J. Buckley, « Universal Fuzzy Controllers », *Automatica*, Vol. 28, No. 6, 1992, pp.1245-1248.
- [13]: J. Buckley, « Sugeno type Controllers are Universal Controllers », *Fuzzy sets and Systems* 53, pp. 299-303, 1993.
- [14]: X.J. Zeng and M.G. Singh, « Fuzzy Bounded Least Squares Method for the Identification of Fuzzy Systems », *Proc. of the IEEE Conf. on Fuzzy Systems (Fuzz'IEEE 97)*, pp. 403-408, Barcelone, Espagne, 1997.
- [15]: E. Kim, P.Park, S.Ji and M.Park, « A New Approach to Fuzzy Modeling », *IEEE Trans.on Fuzzy Systems*, Vol. 5, No. 3, August 1997, pp. 328-337.
- [16] : P.Y. Glorennec, «Algorithme d'apprentissage pour systèmes d'inférence floue», *Edition Hermès*, Paris, 1999.
- [17]: K.S. Narendra, K. Parthasarathy, « Identification and control of dynamical systems using neural networks », *IEEE Trans Neural Network*, 1990, 1(1), pp. 4–27.

- [18] A. Bagis, « Fuzzy rule base design using tabu search algorithm for nonlinear system modeling », *ISA Transactions*, vol 47, 2008, pp. 32–44.
- [19] C. F. Juang, P. H. Chang, « Designing fuzzy-rule-based systems using continuous Ant colony optimization », *IEEE trans., Fuzzy Syst.*, vol. 18, no. 1, 2010, pp. 138-149.
- [20] N. Talbi and K. Belarbi, « Automatic Generation of fuzzy rule base by a hybrid Approach: Application to Control and modeling », *International Review of Automatic Control (I.R.E.A.CO.)*, Vol. 5, N. 2, March 2012, pp. 284-291.
- [21]: C. F. Juang, C. M. Lu, C. Lo, and C. Y. Wang, « Ant colony optimization algorithm for fuzzy controller design and its FPGA implementation », *IEEE Trans. Ind. Electron.*, vol. 55, no. 3, 2008, pp. 1453–1462.
- [22]: N.K. Kasabov and Q. Song, « DENFIS: Dynamic Evolving Neural-Fuzzy Inference System and Its Application for Time-Series Prediction », *IEEE Trans. on Fuzzy Systems*, vol. 10, n°. 2, 2002, pp. 144-154.
- [23]: L.X. Wang, « Stable Adaptive Fuzzy Control of Nonlinear Systems », *IEEE, Trans. On Fuzzy Systems*, Vol. 1, No. 2, 1993, pp. 146-155.
- [24]: C. C. Lee, « Fuzzy logic in control systems: fuzzy logic controller. Part2 », *IEEE Transactions on systems Man and Cybernetics*, Vol. 20, 1990, pp. 419-435.
- [25]: A. Chatterjee, K. Pulasinghe, K. Watanabe, and K. Izumi, « A particles swarm-optimized fuzzy-neural network for voice-controlled robot systems », *IEEE Trans. Ind. Electron.*, vol. 52, no. 6, 2005, pp. 1478–1489.
- [26]: L.X. Wang, « Design and Analysis of Fuzzy Identifiers of Nonlinear Dynamic Systems », *IEEE Transactions on Automatic Control*, 40(1), 1995, pp. 11-23.