REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOHAMED BOUDIAF - M'SILA

FACULTE DE TECHNOLOGIE
DEPARTEMENT D'ELECTRONIQUE
N° :……..

DOMAINE : SCIENCES TECHNOLOGIE
FILIERE : ELECTRONIQUE
OPTION : INSTRUMENTATION

Mémoire présenté pour l'obtention

Du diplôme de Master Académique

Par :

BARKATI Hamza

Intitulé

Modélisation floue d'un système Non linéaire

Soutenu devant le jury composé de :

Dr. N. GUERMAT	Université Mohamed Boudiaf- M'sila	Président
Dr. N. BENAIDJA	Université Mohamed Boudiaf- M'sila	Rapporteur
Dr. M .DJERIOUI	Université Mohamed Boudiaf- M'sila	Examinateur

Année universitaire : 2019 /2020

Remerciement :

Ce mémoire a pu le jour grâce à l'aide de nombreuses personnes à qui

On tien à adresser nos reconnaissance et mes remerciements.

En premier lieu, on tien à remercier Dieu le tout puissant pour la volonté, la santé et la patience qu'il m'a données durant toutes ces années d'études

En suit on adresse nous vifs et sincères remerciements à notre encadreur << Dr. BENAIDJA NOURI >> pour sa patience, le sujet de ce mémoire, son enseignement, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à alimenter notre réflexion.

On tient à remercier sincèrement les membres jury << Dr. N. GUERMAT et Dr. M .DJERIOUI >> qui nous font le grand honneur d'évaluer ce travail.

Ainsi que tous nos professeurs qui nous ont enseigné durant nos études à la faculté de technologie

Et nous exprimons également tous le bonheur du monde à nos collègues de promotion 2019/2020.

Dédicace :

Je désire dédier les fruits de ce modeste travail

A la mémoire de ma sœur Hassiba

Qui est toujours dans mon cœur

A mon âme, mon bonheur et'd ma vie la femme la plus précieuse au monde Ma chère Mère Rahima

A celui qui m'a donné la vie, il fait partie de moi

Mon cher Père Abd el hamid

A mes adorables frères :

Amr (sa femme Rim et son garçon Firas) et Badri.

A mes très chères sœurs :

Wissam (son garçon Ahmad et sa fille Tasnim), Aya et Rahma

A toutes la famille BARKATI et la famille LAYACHI.

A mes proches amis e surtout Abdou, Abd el hak et Choayb

A tous mes professeurs respectueux du primaire au supérieur

Et à tous ceux qui me sont chères

Sommaire

Chapitre II : Les Méthodes d'Optimisation

Chapitre III. Optimisation de la base de règles floues pour la Modélisation des Systèmes Non linéaires par l'Algorithme Hybride AG-RS

Liste des symboles et abréviations

SIF : Système d'Inférence Floue.

AG : Algorithme Génétique.

AGE : Algorithme génétique avec élitisme.

TS : Takagi-Sugeno.

AG-RS : L'algorithme hybride des AG et la Recuit Simulé.

SF : Système flou.

EQM : Erreur Quadratique Moyenne.

REQM : Racine de l'Erreur Quadratique Moyenne.

RS : Recuit Simulé.

Liste des figures

Chapitre I : Les Systèmes d'Inférence Floue

Chapitre II : Les Méthodes d'Optimisation

Chapitre III. Optimisation de la base de règles floues pour la Modélisation des Systèmes Non linéaires par l'Algorithme Hybride AG-RS

Liste des tableaux

Chapitre I : Les Systèmes d'Inférence Floue

Chapitre III. Optimisation de la base de règles floues pour la Modélisation des Systèmes Non linéaires par l'Algorithme Hybride AG-RS

Introduction générale

Introduction générale

Une bonne maîtrise d'un système passe en général par une information fiable sur ce dernier. Cependant, la plupart des systèmes industriels possèdent des comportements non linéaires ce qui a incité les chercheurs à développer des outils d'identification performants.

Les outils intelligents sont de plus en plus utilisés dans la conception, la modélisation et la commande de systèmes complexes tels que les robots, les procédés biologiques, les instruments … . On entend par outils intelligents les techniques du soft computing à savoir : la logique floue, les algorithmes génétiques et le recuit simulé.

La logique floue introduite par Zadeh (1965) dans les années soixante constitue un outil très puissant pour la représentation des termes et des connaissances vagues. Elle est issue de la capacité de l'homme à décider et à agir d'une manière intelligente malgré l'imprécis et l'incertitude des connaissances disponibles.

Les algorithmes génétiques sont des méthodes stochastiques basées sur une analogie avec des systèmes biologiques. Ils reposent sur un codage de variables organisées sous forme de structures chromosomiques et prennent modèle sur les principes de l'évolution naturelle de Darwin pour déterminer une solution optimale au problème considéré. Ils ont été introduits par Holland (Holland, 1975) pour des problèmes d'optimisation complexe. Contrairement aux méthodes d'optimisation classique, ces algorithmes possèdent la capacité d'éviter les minimums locaux pour effectuer une recherche globale.

le recuit simulé est une méthode de programmation empirique (métaheuristique) inspirée d'un processus utilisé en métallurgie. On alterne dans cette dernière des cycles de refroidissement lent et de réchauffage (*recuit*) qui ont pour effet de minimiser l'énergie du matériau. Cette méthode est transposée en optimisation pour trouver les extrema d'une fonction. Elle a été mise au point par trois chercheurs de la société IBM, S. Kirkpatrick, C.D. Gelatt et M.P. Vecchi en 1983, et indépendamment par V. Černy en 1985.

Une hybridation des trois paradigmes est utilisée dans ce mémoire afin d'aboutir à un modèle flou représentant en mieu un système non linéaire type :

Le premier chapitre est consacré aux systèmes flous. Nous rappelons d'abord les notions de bases sur lesquelles reposent ses systèmes puis nous décrivons leur principe de fonctionnement et leurs différentes composantes.

Le deuxième chapitre est divisé en deux parties. La première présente une description détaillée des algorithmes génétiques simples dans laquelle nous rappelons les définitions relatives à leur fonctionnement. Dans la deuxième partie nous introduirons le recuit simulé, version améliorée de la déscente stochastique.

Le troisième chapitre constitue le noyau de ce mémoire. Nous commençons par une description de la méthode du modèle de référence. Nous procédons ensuite à l'dentication par algorithmes génétiques simples. Enfin nous proposons un algorithme hybride pour l'identification du système flou de type Sugeno.

Chapitre I. Propriétés générales Les Systèmes d'Inférence Floue

I.1. Introduction

De nos jours, la logique floue (fuzzy logic) est un axe de recherche important sur lequel se focalisent de nombreux scientifiques. Des retombées technologiques sont d'ores et déjà disponibles, tant dans le domaine grand public (appareils photos, machines à laver, fours à micro-onde), que dans le domaine industriel (classification, aide à la décision, réglage et commande de processus, complexes liés à l'énergie, aux transports, à la transformation de la matière, à la robotique, aux machines-outils).

Les bases théoriques de la logique floue ont été formulées en 1965 par le professeur Lotfi A. Zadeh, de l'Université de Berkeley en Californie **[1]**. Il a introduit la notion de sous-ensemble flou pour fournir un moyen de représentation et de manipulation des connaissances imparfaitement décrites, vagues ou imprécises.

A cette époque, la théorie de la logique floue n'a pas été prise au sérieux excepté par quelques experts.

Dès 1975, Mamdani et Assilian publient les premiers résultats permettant une exploitation de cette théorie dans des systèmes de réglage **[2]**. En utilisant une structure de contrôleur relativement simple, ils ont obtenu de meilleurs résultats lors de la commande de certains processus que ceux fournis par un régulateur standard de type PID.

Peu de temps après, en 1977, le danois Ostergaard **[3]** a appliqué la logique floue à la commande de tubes broyeurs pour la fabrication de ciment. A cette époque, la plupart des études concernant les systèmes de régulation exploitant la logique floue ont été réalisées en Europe. A partir de 1985 environ, ce sont les Japonais qui commencent à utiliser largement la logique floue dans des produits industriels et de consommation pour résoudre des problèmes de réglage et de commande **[4]**.

Ce chapitre est consacré à la description des éléments de base de la théorie des systèmes flous. Ce concept constitue la plateforme pour les différents travaux exposés dans ce travail.

I.2. Les Systèmes d'Inférence Floue

I.2.1. Définition

Les systèmes d'inférence floue (SIF) peuvent être considérés comme des systèmes logiques qui utilisent des règles linguistiques pour établir des relations entre leurs variables d'entrée et de sortie. Aujourd'hui, les applications des SIF sont très nombreuses outre la commande, ils sont largement utilisés pour la commande, la modélisation, le diagnostic et la reconnaissance de formes **[5]**. Pour une meilleure compréhension, nous présentons quelques notions de base de ces systèmes ainsi que leurs types et leurs caractéristiques **[6]**.

I.2.2. Les Ensembles Flous

Mathématiquement, un ensemble flou A d'un univers de discours U, est caractérisé par une fonction d'appartenance, notée μ_A, à valeur dans l'intervalle [0,1] et qui associe à chaque élément x de U un degré d'appartenance $\mu_A(x)$ indiquant le niveau d'appartenance de x à A. $\mu_A(x) = 1$ et $\mu_A(x) = 0$ correspondent respectivement à l'appartenance et la non-appartenance. L'univers de discours ou le référentiel est l'ensemble des valeurs réelles que peut prendre la variable floue x.

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

$$\text{Supp}(x) = \{x \in A / \mu_A(x) > 0\} \qquad \textbf{(I.1)}$$

Un ensemble flou dont le support est un singleton flou dans U avec

$\mu_A(x) = 1$ est appelé « singleton flou ».

b. Sa hauteur : qui est sa plus grande valeur prise par sa fonction d'appartenance. Elle est définie par :

$$h(x) = \{\text{Sup}\, x \in A.\, \mu_A(x)\} \qquad \textbf{(I.2)}$$

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

$$\text{Noy}(x) = \{x \in A / \mu_A(x) = 1\} \qquad \textbf{(I.3)}$$

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

La figure I.1 illustre les caractéristiques d'un ensemble flou : support, hauteur et noyau.

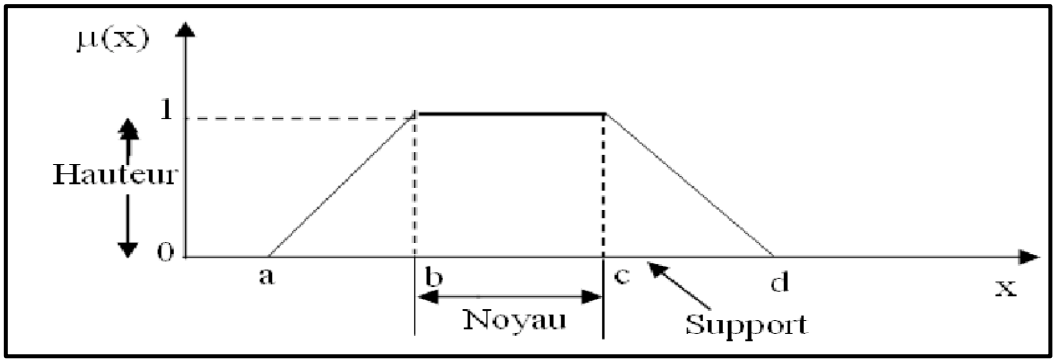

Figure I.1: Les caractéristiques d'un ensemble flou.

I.2.2.1. Variable linguistique

La notion de variable linguistique permet de modéliser les connaissances imprécises ou vagues sur une variable dont la valeur précise est inconnue. Une variable linguistique, ou variable floue, est donc une variable dont les valeurs floues appartiennent à des ensembles flous pouvant représenter des mots du langage naturel. Ainsi une variable floue peut prendre simultanément plusieurs valeurs linguistiques. Le domaine sur lequel ces termes et ces variables sont définis, constitue l'univers de discours.

Le découpage de cet univers de discours par les termes flous est appelé une partition floue. Lorsque l'univers de discours est totalement recouvert par les termes flous, et que pour toutes valeurs, la somme des degrés d'appartenance est égale à 1, on parle alors de partition floue forte.

La variable linguistique peut être représentée par un triplet (x, T(x), U) dans lequel x est le nom de la variable linguistique, T(x) l'ensemble des valeurs linguistiques de x et U l'univers de discours.

La figure I.2 illustre un exemple de variable linguistique 'vitesse' avec trois termes linguistiques : petite, moyenne et grande.

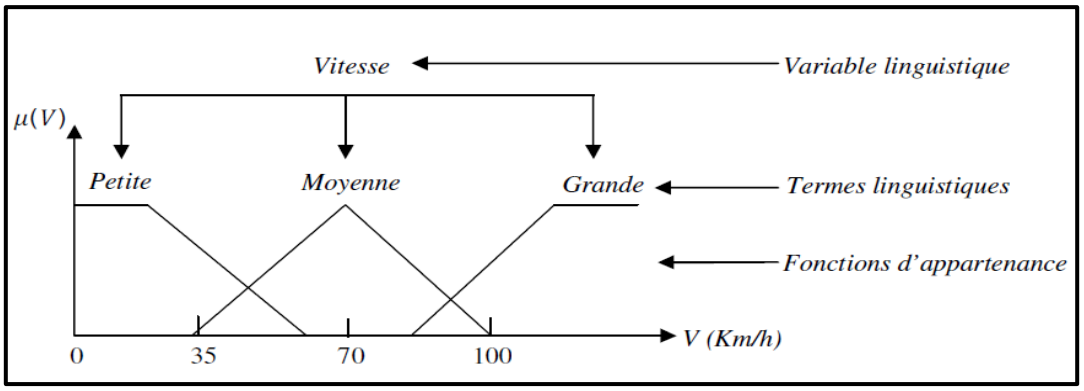

Fig. I.2 : Variable linguistique.

I.2.2.2. Fonction d'appartenance

Soit un ensemble E et un sous-ensemble A de E (A ⊂ E), et x un élément de E appartenant à A (x ∈ A). Pour illustrer cette caractéristique, on utilise la fonction d'appartenance μ_A (x) compris entre 0 et 1, qui représente **le degré d'appartenance** de x à l'ensemble flou A. Le plus souvent, la fonction d'appartenance est déterminée par l'une des fonctions suivantes (figure I.3):

a. Fonction triangulaire

Elle est définie par trois paramètres {a, b, c}, qui déterminent les coordonnées des trois sommets (figure I.3-a).

$$\mu_A(x) = \begin{cases} \frac{x-a}{b-a}, & a \le x \le b; \\ \frac{c-x}{c-b}, & b < x \le c; \\ 0, & x < a \text{ ou } x > c. \end{cases} \equiv \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \qquad \textbf{(I.4)}$$

b. Fonction trapézoïdale

Elle est définie par quatre paramètres {a, b, c, d}, (figure I.3-b) :

$$\mu_A(x) = \begin{cases} 0, & x < a; \\ \frac{x-a}{b-a}, & a \le x < b; \\ 1, & b \le x \le c; \\ \frac{d-x}{d-c}, & c < x \le d; \\ 0, & x > d. \end{cases} \equiv \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \qquad \textbf{(I.5)}$$

c. Fonction gaussienne

Elle est définie par deux paramètres $\{\sigma, m\}$, (figure I.3-c) :

$$\mu_A(x) = \exp^{(-\frac{(x-m)^2}{2\sigma^2})} \quad \textbf{(I.6)}$$

d. Fonction sigmoïde

Elle est définie par deux paramètres {a, c}, (figure I.3-d).

$$\mu_A(x) = \frac{1}{1+exp^{(-a(x-c))}} \quad \textbf{(I.7)}$$

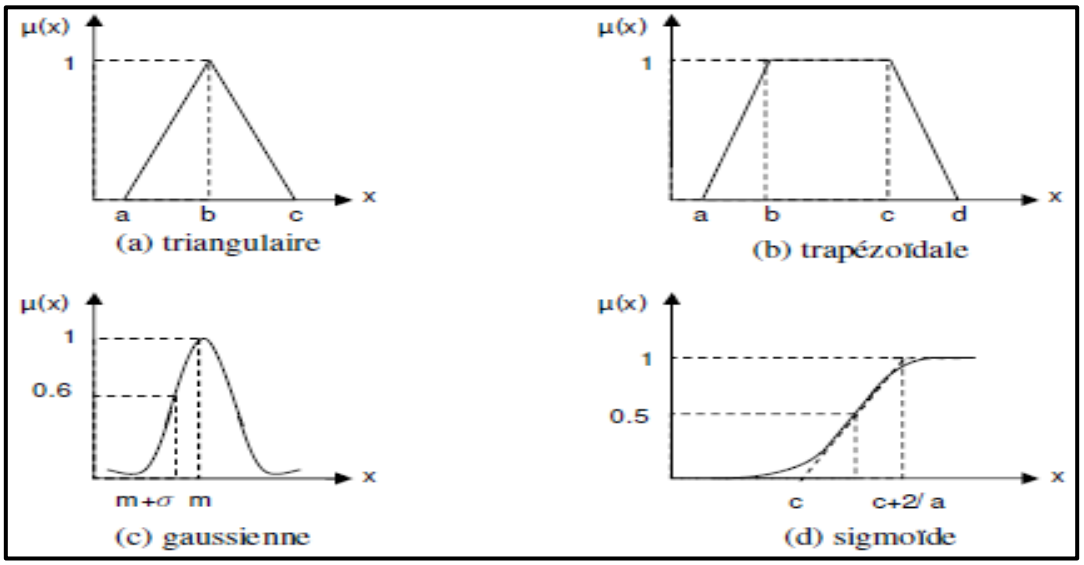

Fig. I.3 : Formes usuelles des fonctions d'appartenance.

I.2.2.3 Opérations sur les ensembles flous [7]

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

a. Egalité

Deux sous-ensemble flous A et B de X sont dits égaux s'ils ont des fonctions d'appartenance égales en tout point de X. formellement, A= B si et seulement si :

$$\forall x \in X, \mu_{A(X)} = \mu_{B(X)} \quad \textbf{(I.8)}$$

b. Inclusion

Soient A et B deux sous – ensembles flous de X. si pour n'importe quel élément x de X, x appartient toujours moins à A qu'à B, alors on dit que A est inclus dans B $(A \subset B)$. formellement, $A \subset B$ si et seulement si :

$$\forall\, x \in X\,, \mu_{A(X)} \leq \mu_{B(X)} \qquad \textbf{(I.9)}$$

c. Union

L'union de deux sous-ensembles flous A et B de X est le sous-ensemble flous constitué des éléments de X affectés du plus grand des degrés avec lesquels ils appartiennent à A et B. formellement, $A \cup B$ est exprimé par :

$$\mu_{A(X)} \cup B(x) \;=\; max\,(\mu_{A(X)}, \mu_{B(X)})\,) \qquad \textbf{(I.10)}$$

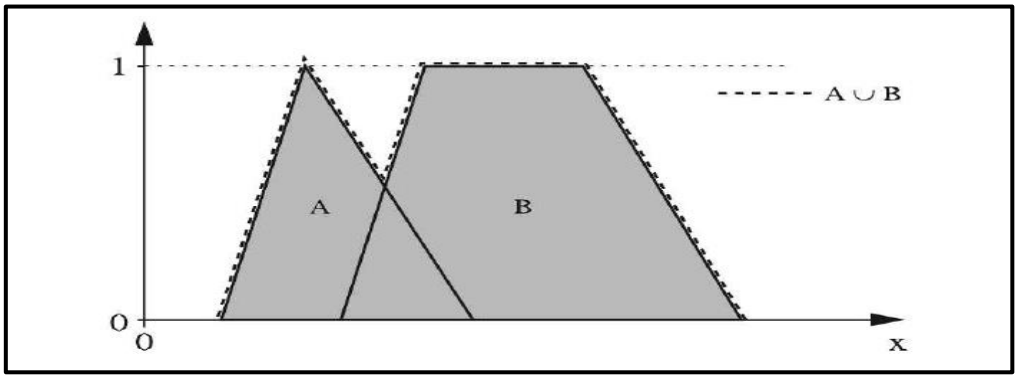

Fig. I.4 : Union de deux sous-ensembles flous avec l'opérateur max.

d. Intersection

L'intersection de deux sous-ensembles flous A et B de X est le sous-ensemble flou constitué des éléments de X affectés du plus petit des degrés avec lesquels ils appartiennent à A et B. formellement, $A \cap B$ est exprimé par :

$$\mu_{A(X)} \cap B(x) \;=\; min\,(\mu_{A(X)}, \mu_{B(X)})\,) \qquad \textbf{(I.11)}$$

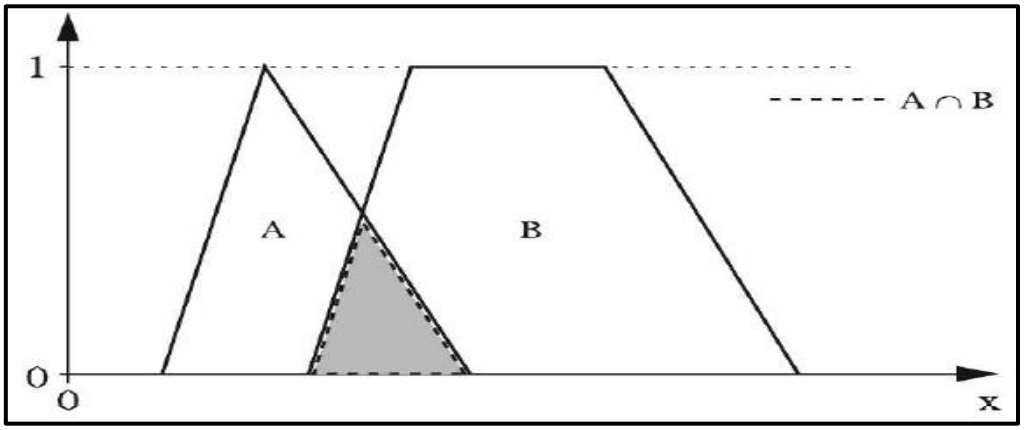

Fig. I.5 : Intersection de deux sous-ensembles flous avec l'opérateur min.

e. Complément

Le complément d'un sous-ensemble flou A de B est noté Ā. Il est défini à partir de la fonction d'appartenance de A par :

$$\forall\, x \in X\,, \mu_{\bar{A}(X)} = 1 - \mu_{A(X)} \qquad \textbf{(I.12)}$$

I.2.3. Les systèmes flous

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

1.2.3.1. Description générale d'un système flou

La configuration de base d'un système flou comporte quatre blocs principaux :

(La figure I.6)

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

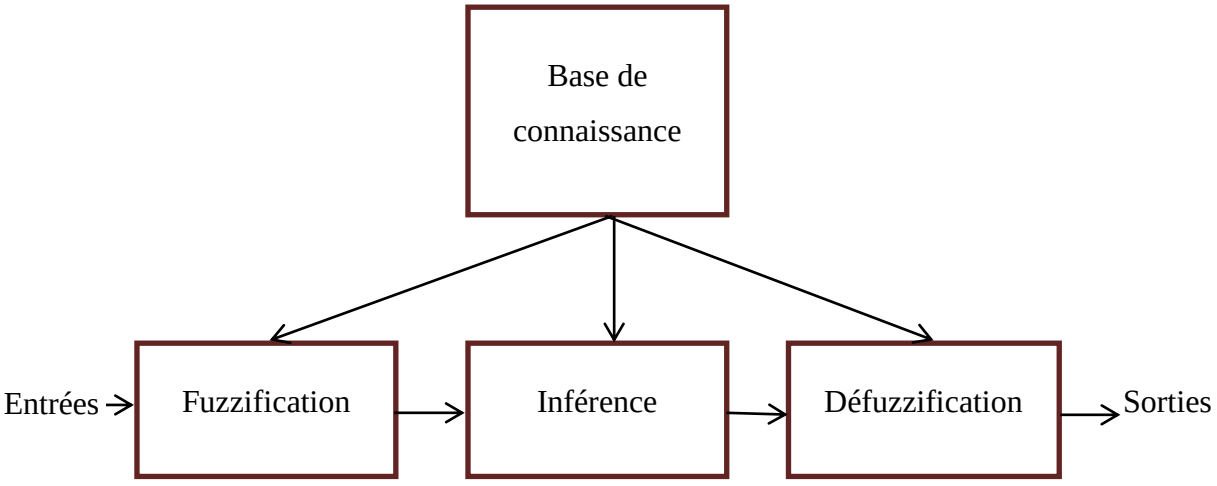

Fig. I.6 : Schéma synoptique d'un système flou.

Le rôle de chaque bloc peut être résumé comme suit :

Le bloc ***base de connaissance*** contenant une connaissance dans le domaine d'application et le résultat de commande prévu. Il consiste en "base de données" et en "base de règles linguistiques (floues) de commande". Il contient :

- la base de données fournissant les informations nécessaires pour les fonctions de normalisation d'un RF;
- la base de règles constitue un ensemble d'expressions symboliques formalisés auprès d'une connaissance d'expert. Une règle indique que, si une condition est présente dans le fonctionnement du système alors une décision est nécessaire pour mettre le système dans l'état de fonctionnement désiré, et est de type :
  Si ″ condition ″ Alors ″ conséquence″.

Le bloc ***fuzzification*** effectue les fonctions suivantes :

- établir les plages de valeurs pour les fonctions d'appartenance à partir des valeurs des variables d'entrées,
- transformer les données numériques en valeurs linguistiques utilisant une désignation. Chaque désignation est un ensemble flou de l'univers de discours
- décrit par une fonction caractéristique appelée la fonction d'appartenance.

Le bloc ***inférence*** est le cœur d'un RF, il s'appelle mécanisme de décision et permet de calculer l'ensemble flou associé au système à l'aide de l'implication floue et des règles d'inférence dans la logique floue.

Le bloc ***défuzzification*** effectue la fonction suivante :

- établir les plages de valeurs pour les fonctions d'appartenance à partir des valeurs des variables de sortie,

a. Base de connaissance

Elle comprend la base de données et la base des règles floues

❖ **La base de données**

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

❖ **La base des règles floues**

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

$$R_l: \textbf{Si}\ X_1\ est\ A_1^l\ et \ldots etX_n\ est\ A_n^l\ \textbf{Alors}\ \ u_l\ est\ B^l \qquad \textbf{(I.13)}$$

Avec :$[X_1, \ldots, X_n]$ les entrées du système.

u_l : La sortie du système.

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

- La complétude : une base de règles d'un système flou est dite complète si, pour chaque vecteur d'entrée, il existe au moins une règle floue activée. Afin d'assurer cette propriété, les fonctions d'appartenance doivent couvrir toutes les plages possibles des variables d'entrée. L'utilisation de fonctions d'appartenance triangulaires régulièrement réparties respecte la propriété de complétude.
- La consistance : une base de règles d'un système flou est dite inconsistante, s'il existe deux règles floues ayant la même prémisse mais des conclusions différentes. La propriété de consistance permet d'éviter les contradictions dans une base de règles.

b. Interface de fuzzification

La première étape de la fuzzification consiste à la normalisation des grandeurs d'entrées à l'aide de facteurs de normalisation ou d'échelle, généralement normalisées sur un intervalle bien déterminé appelé univers de discours. Ensuite pour qualifier chaque variable, l'expert doit définir un univers de discours constitué d'un nombre fini

de sous-ensemble flous. Pour chacun d'entre eux, on fait correspondre une fonction d'appartenance.

La figure I.7 représente un exemple de choix des fonctions d'appartenance d'une variable normalisée, qui est composée de sept ensembles flous, NG (Négative Grande), NM (Négative Moyenne), NP (Négative Petite), EZ (Environ Zéro), PP (Positive Petite), PM (Positive Moyenne), PG (Positive Grande).

Les fonctions d'appartenance peuvent être symétriques ou non symétriques. Toutefois, il faut éviter les lacunes entre les fonctions d'appartenance de deux ensembles voisins, puisque cela provoque des zones non traitées par le régulateur (zones mortes), ce qui entraîne une instabilité de réglage. Le choix du nombre d'ensembles dépend de la solution et de l'intervention du réglage désiré.

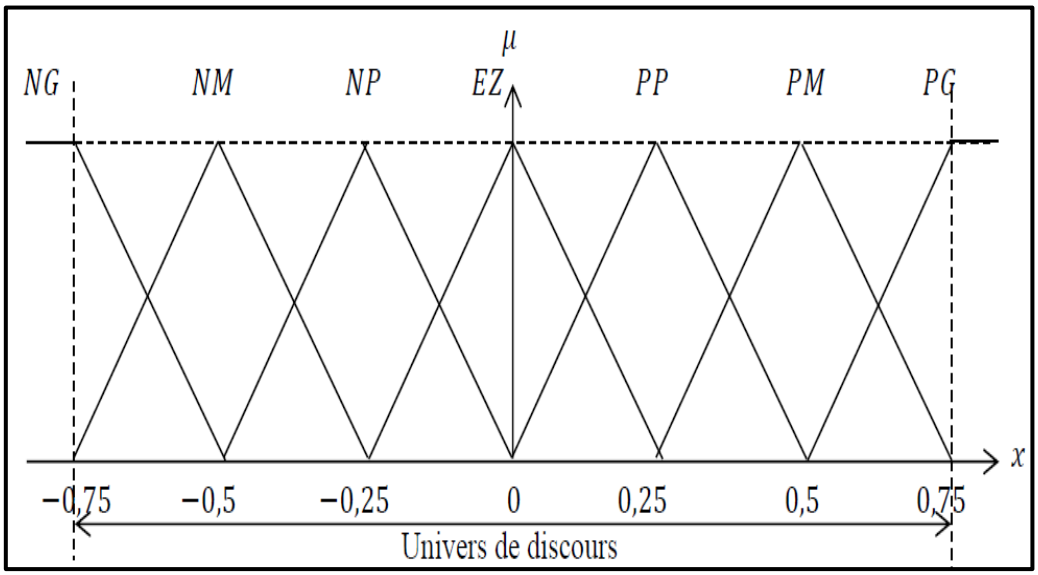

Fig. I.7 : Fonctions d'appartenance symétriques.

c. Interface (composition des règles)

Les sous-ensembles flous issus de l'inférence sont regroupés pour obtenir un seul ensemble représentatif des différentes conclusions des règles floues. Cette opération est réalisée par le moteur d'inférence, qui combine les règles floues. Selon les principes de la logique floue, une transformation est effectuée à partir des ensembles flous dans l'espace d'entrées vers des ensembles flous dans l'espace de sortie.

Pour le réglage par logique floue, on utilise en générale une des méthodes suivantes: **[8]**

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

❖ **Méthode d'inférence max-min**

Elle consiste à caractériser l'ensemble de sorties par une fonction d'appartenance égale au maximum des fonctions d'appartenance des sous-ensembles flous. La méthode d'inférence max-min, sa dénomination dite " Max-Min" ou " implication de Mamdani ", est due à la façon de réaliser les opérateurs "ALORS" et "OU" de l'inférence. Par la fonction " Min", cette méthode réalise l'opérateur "ET" et la conclusion "ALORS" de chaque règle ; par contre la liaison entre toutes les règles (opérateur OU) est réalisée par la fonction "Max".

Pour illustrer cette méthode (Max-Min), la figure. I.8 présente Trois règles, qui sont données par la forme générale :

Règle (1): SI x est négatif petit (NP) ET y est environ zéro (EZ)

ALORS z est positif petit (PP)

Règle (2): SI x est environ zéro (EZ) ET y est environ zéro (EZ)

ALORS z est environ zéro (EZ)

Règle (3): SI x est environ zéro (EZ) ET y est positif petit (PP)

ALORS z est négatif petit (NP)

Avec: NP, EZ et PP sont des sous-ensembles flous.

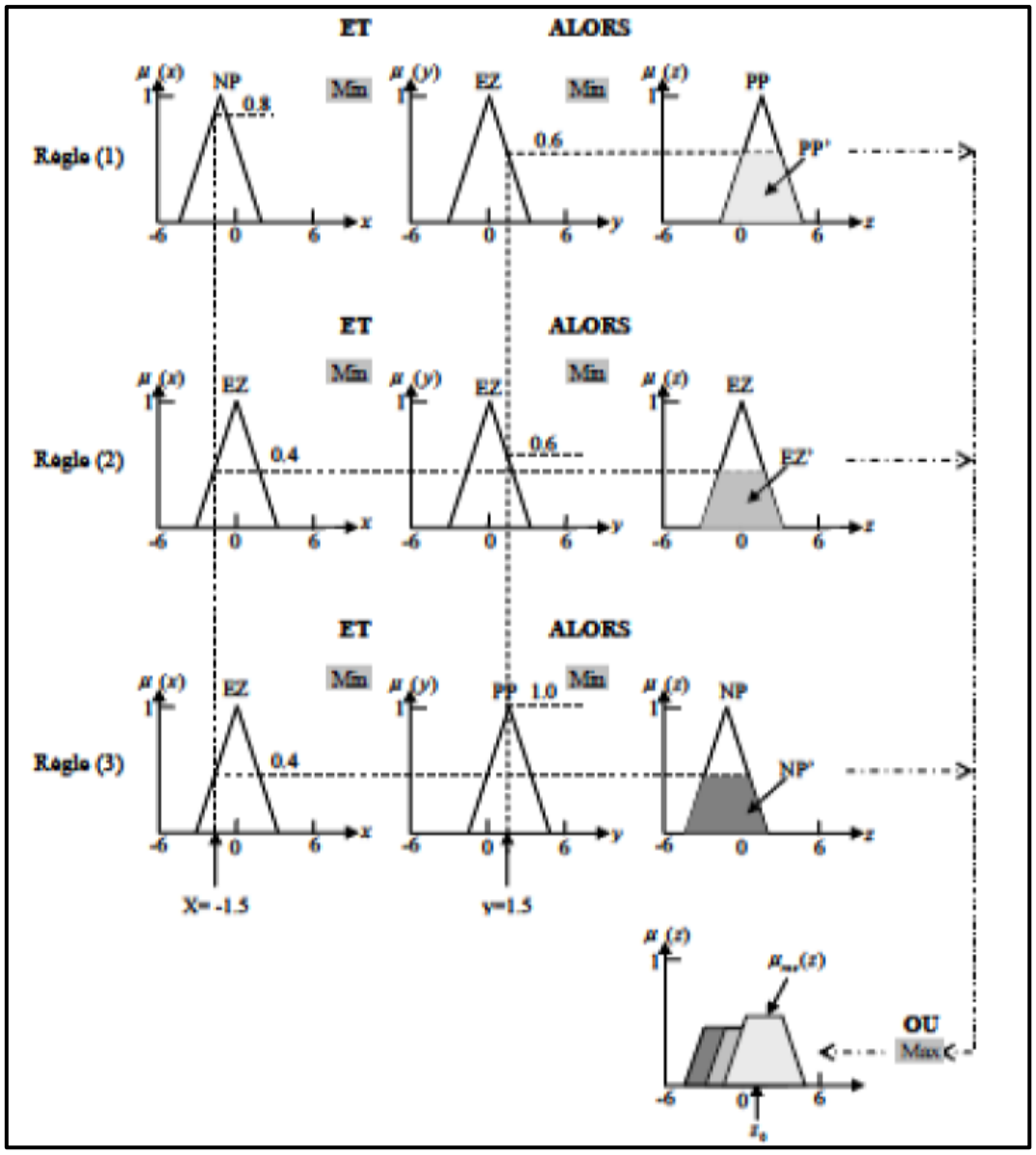

Fig. I.8 : Méthode d'inférence Max-Min (Mamdani)

Dans cette figure, pour la première règle la condition :(Si x NP ET y EZ) donne pour x=−1,5 et y=1,5 les degrés d'appartenance μ_{NP}= 0,8 et μ_{EZ}= 0,6, ce qui implique que le minimum des deux valeurs c'est le degré d'appartenance 0,6. On obtient ainsi une « fonction d'appartenance partielle » $\mu_{pp'}(z)$ comme indiquée à la figure I.8.

La fonction d'appartenance résultante de sortie correspond au maximum des trois fonctions d'appartenance partielles puisque les règles sont liées par l'opérateur OU.

❖ **Méthode d'inférence max-prod**

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

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

La figure I.9 montre un exemple de la méthode d'inférence Max-Produit.

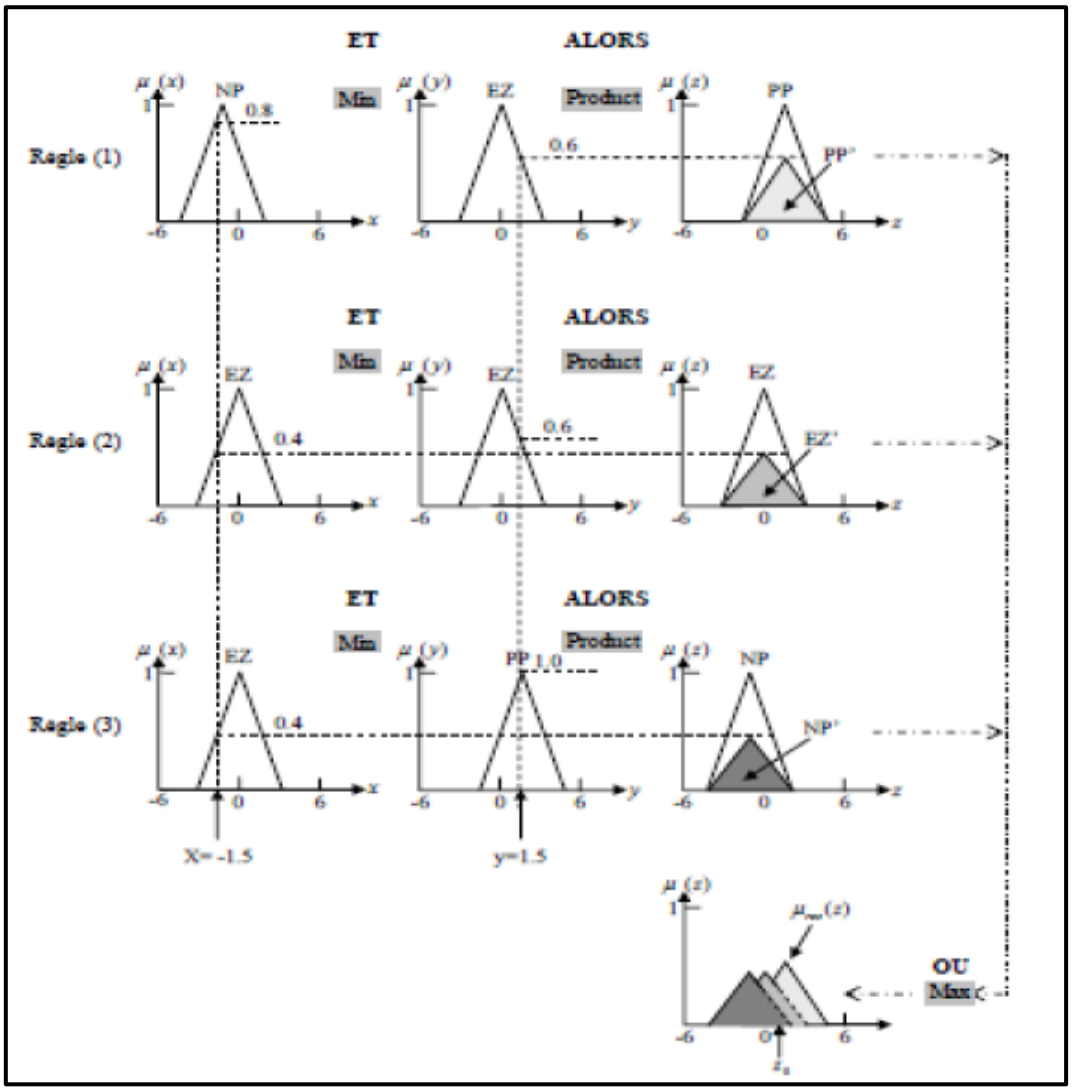

Fig. I.9 : Méthode d'inférence Max-Produit (Larsen)

❖ **Méthode d'inférence somme-prod**

Elle consiste à faire la somme de fonctions d'appartenance des sous-ensembles issus de l'inférence. La méthode d'inférence Som-Produit, au niveau de la condition, par la formation de la somme ; plus précisément par la valeur moyenne réalise l'opérateur "OU" tandis que, par la formation du produit réalise l'opérateur "ET". La conclusion de chaque règle, précédée par "ALORS", liant le facteur d'appartenance de la condition avec la fonction d'appartenance de la variable de sortie par l'opérateur " ET" est réalisée par la formation du produit. Dans ce cas, l'opérateur "OU", qui lie les différentes règles est réalisé par la formation de la somme, et "ALORS" est réalisé par la formation du produit ; d'où la désignation de cette méthode d'inférence par Somme-Produit.

d. Interface de défuzzification

La défuzzification permet de générer une valeur numérique à partir de l'ensemble obtenu par composition des règles. Il existe plusieurs méthodes de défuzzification (au moins une dizaine). Les méthodes les plus couramment utilisées sont :

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

❖ **Méthode de centre de gravité**

La défuzzification par centre de gravité consiste à calculer l'abscisse du centre de gravité de la fonction d'appartenance résultante μr de la phase de composition selon la relation suivante:

$$y^* = \frac{\int y.\mu r(y).dy}{\int \mu r(y).dy} \qquad \textbf{(I.14)}$$

En pratique, on estime le centre de gravité en calculant la moyenne d'un certain nombre de points échantillonnés sur la fonction par la relation suivante:

$$y^* = \frac{\int y_i.\mu r(y_i).dy}{\int \mu r(y_i).dy} \qquad \textbf{(I.15)}$$

❖ **Méthode du maximum**

Cette méthode s'applique seulement dans le cas où la fonction d'appartenance associée à l'ensemble de sortie n'admet qu'un seul maximum. On choisit comme sortie l'abscisse y^*correspondant à ce maximum.

❖ **Méthode de la moyenne des maxima**

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

$$y^* = \left(\frac{\inf\ (M)+\sup\ (M)}{2}\right) \qquad \textbf{(I.16)}$$

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

1.2.3.2. Différents types de systèmes d'inférence floue

Dans la littérature, il existe plusieurs types de système d'inférence floue qui peuvent être appliqués dans les systèmes flous. Les deux types qui sont largement utilisées en pratique sont le type de Mamdani et le type de Sugeno. Ces deux types de systèmes d'inférence varient quelque peu dans la façon dont les résultats sont déterminés.

a. Système d'inférence floue de type Mamdani

En 1974, E.H Mamdani avait présenté pour la première fois la technique de commande par logique floue. Celle-ci consiste à déterminer un ensemble de règles qui maîtrisent le comportement dynamique du système à commander. L'obtention de ces règles est facile auprès des experts qui connaissent bien le système. Le système de Mamdani est caractérisé par des règles à prémisses et conclusions symboliques, l'inférence (Max-Min), et la défuzzification par centre de gravité.

La forme de l'implication floue est de la forme :

$$\text{«}Si\ x_1\ est\ A\ et\ x_2\ est\ B\ \textbf{\textit{Alors}}\ y\ est\ C\ \text{»}$$

La conséquence de ce type de système est une valeur floue. Une structure de ce type de contrôleur est représentée sur la figure I.10.

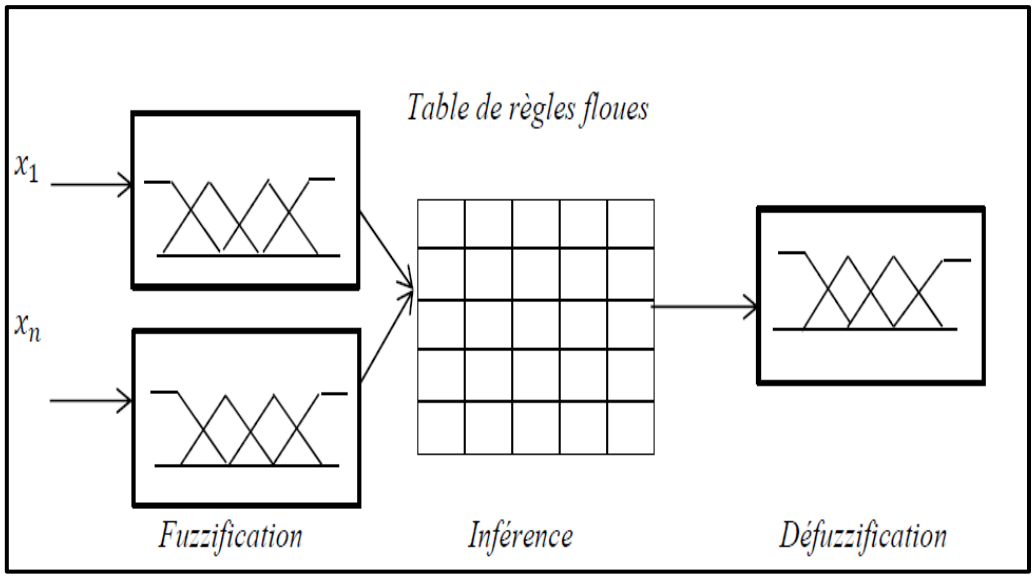

Fig. I.10 : Système d'inférence de Mamdani.

b. Système d'inférence floue de type Takagi-Sugeno

Les systèmes flous au sens de Sugeno diffèrent des systèmes au sens de Mamdani portant sur la partie défuzzfication.

Sugeno propose de remplacer la défuzzification des variables de sortie par une combinaison linéaire des variables d'entrée (le plus souvent, cette combinaison linéaire se réduit à une constante « $Z_1 = k$ »).

La figure I.11 montre le modèle du système d'inférence Sugeno.

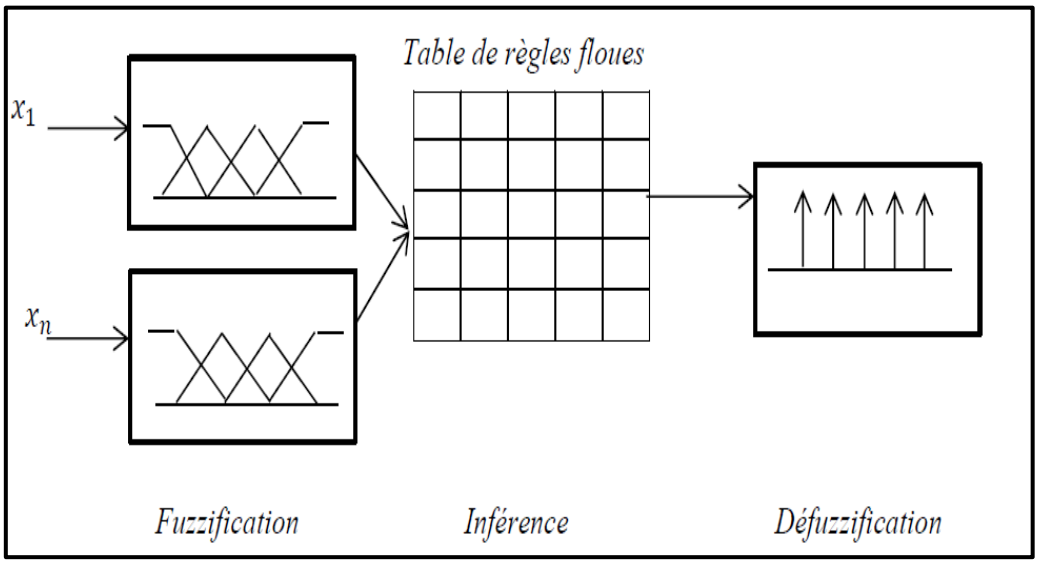

Fig. I.11 : Système d'inférence de *Sugeno*.

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

x= [x_1, x_2,…x_n] : Sont les entrées du système flou, y sa sortie. Pour chaque x_i est associé m_i ensemble flou F_i^J dans X_i tel que $x_i \in X_i$

La base de règles complète du système flou est de la forme :

R_k : ***si*** X_1 *est* F_1^J *et* X_2 *est* F_2^J *et …et* X_n *est* F_n^J ***Alors*** $y = F_K(x)$;

$k = 1,2,..,n$ **(I.17)**

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

$$F_K(x) = a_0^k + \sum_{i=1}^{n} a_i^k xi \quad \textbf{(I.18)}$$

Si $F_K(x)$ est une constante :

$$F_K(x) = a_0^k \quad \textbf{(I.19)}$$

On a donc un système flou de type Takagi-Sugeno d'ordre zéro.

Etant donné que chaque règle possède une conclusion numérique, on ne passe pas par la procédure de défuzzification. La sortie du système flou est donnée par la relation suivante :

$$y(x) = \frac{\sum_{k=1}^{n} \mu_k(x).F_k(x)}{\mu_k(x)} \quad \textbf{(I.20)}$$

Avec :

$$\mu_k(x) = \prod_{i=1}^{n} F_i^K \quad \textbf{(I.21)}$$

$F_i^k \in \{F_i^1, F_i^2, \dots, F_i^{mi}\}$ Représente le degré d'activation de la règle R_k

L'avantage plus facile de déterminer les relations entrées-sorties.

De la structure de Sugeno réside dans le fait qu'elle permet une mise en équation nettement plus simple que la structure de Mamdani, il est en particulier.

I.2.4. La Caractéristiques des systèmes d'inférence floue

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

I.2.4.1. Les caractéristiques structurelles

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

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

I.2.4.2. Les caractéristiques paramétrique

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

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

I.2.5. Avantages et inconvénients des logiques floues

Le tableau (I.1) suivant représenté les avantages et les inconvénients des logiques floues.

Le réglage par logique floue présente une solution désirable par rapport aux réglages classiques. Cela est confirmé par un fort développement dans beaucoup de domaines d'application.

Avantages	**Inconvénients**
• Le modèle mathématique non requis. • La connaissance antérieure sur les règles peut être utilisée. • Une interprétation et implémentation simple. • Adaptation difficile au changement de l'environnement	• Les règles doivent être disponibles • Ne peut pas apprendre • Aucunes méthodes formelles pour l'ajustement

Tableau I.1 : les avantages et les inconvénients des logiques floues.

I.2.6. Conclusion

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

Les SIF exigent de préférence la disponibilité d'une expertise humaine ; par conséquent, les performances de ces derniers sont étroitement liées aux techniques d'acquisition de connaissances et la justesse des informations acquises. Cependant, pour réaliser un SIF, il va falloir :

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

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

Chapitre II.
Techniques Expérimentales:
Les Méthodes d'Optimisation

II.1. Introduction

Les métaheuristiques sont apparues dans les années 1980 et forment une famille d'algorithmes d'optimisation visant à résoudre des problèmes d'optimisation difficile, pour lesquels on ne connaît pas de méthode classique plus efficace. Elles sont généralement utilisées comme des méthodes génériques pouvant optimiser une large gamme de problèmes différents, sans nécessiter de changements profonds dans l'algorithme employé. L'un des intérêts majeurs des métaheuristiques est leur facilité d'utilisation dans des problèmes concrets sans nécessité de connaissances particulières sur le problème d'optimisation à résoudre**.**

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

II.2. Les Algorithmes Génétiques

II.2.1. Introduction

Les algorithmes génétiques développés par J. Holland présentent des qualités intéressantes pour la résolution de problèmes d'optimisation complexes. Leurs fondements théoriques furent exposés par Goldberg **[9].** Ils tentent de simuler le processus d'évolution des espèces dans leur milieu naturel: soit une transposition artificielle de concepts basiques de la génétique et des lois de survie énoncés par Darwin.

C'est actuellement la méthode la plus diffusée et la plus utilisée dans la résolution des problèmes d'optimisations dans de nombreux domaines d'applications.

II.2.2. Principe de fonctionnement des algorithmes génétiques

Un algorithme génétique est construit de manière tout à fait analogue au processus d'évolution d'une population. Dans l'ensemble des solutions d'un problème d'optimisation, une population de taille N est constituée de N solutions (les individus de la population) convenablement marquées par un codage qui les identifie complètement. Une procédure d'évaluation est nécessaire à la détermination de la force de chaque individu de la population. Viennent ensuite une phase de sélection (en sélectionnant les individus selon leur force) et

une phase de recombinaison (opérateurs artificiels de croisement et de mutation) qui génèrent une nouvelle population d'individus, qui ont de bonnes chances d'être plus forts que ceux de la génération précédente. De génération en génération, la force des individus de la population augmente et après un certain nombre d'itérations, la population est entièrement constituée d'individus tous forts, soit de solutions quasi-optimales du problème posé.

II.2.3. Structure de l'algorithme génétique

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

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

L'organigramme fonctionnel de la figure II.1 illustre la structure de l'algorithme génétique. Les diverses phases et les mécanismes associés à chacune d'entre elles seront présentés dans les sections suivantes.

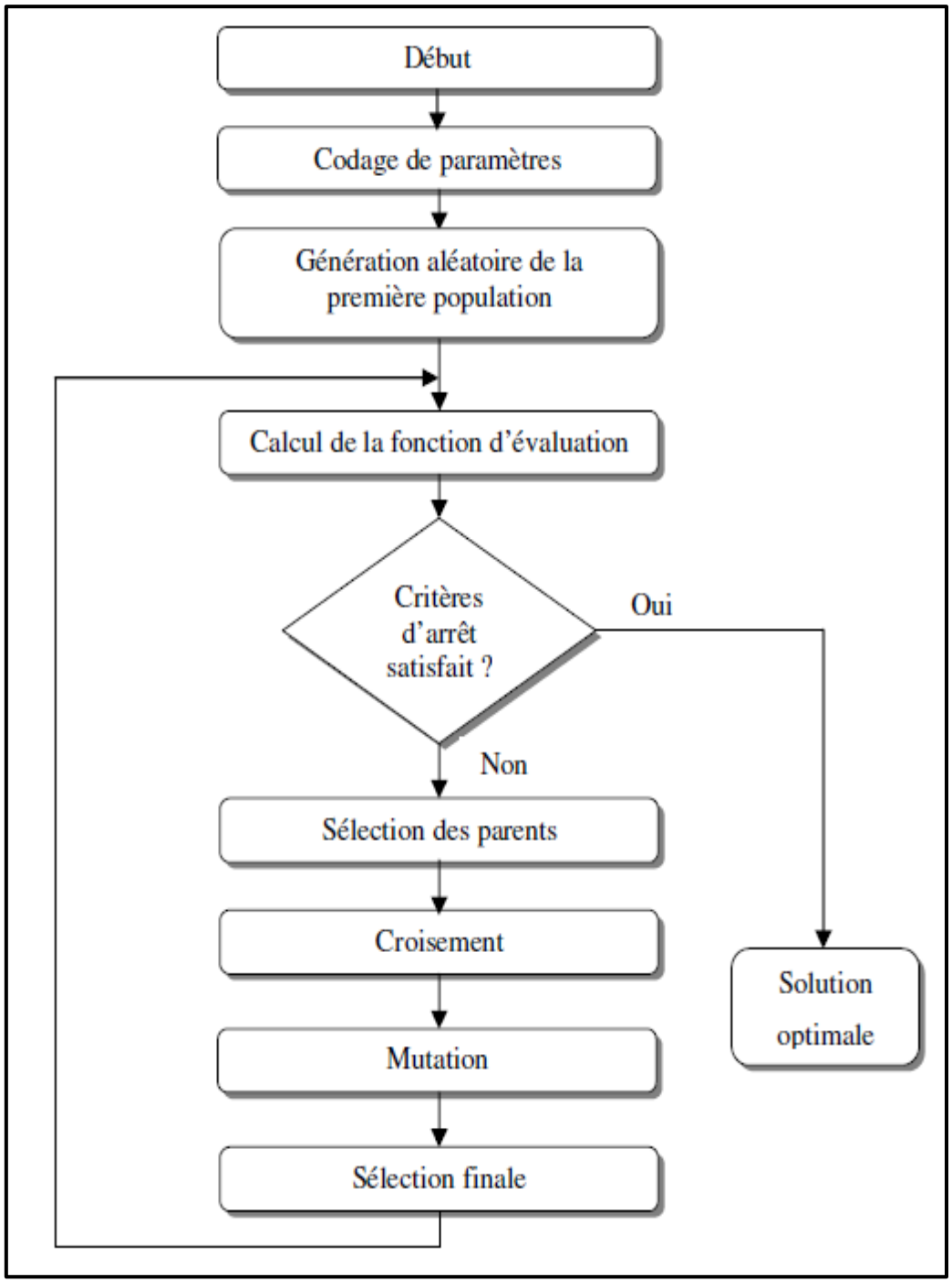

Figure II.1: Organigramme général de l'algorithme génétique.

II.2.4. Le codage

Premièrement, il faut représenter les différents états possibles de la variable, dont on cherche la valeur optimale sous forme utilisable pour un (**AG**) : c'est le codage.

Cela permet d'établir une connexion entre la valeur de la variable et les individus de la population de manière à imiter la transcription génotype-phénotype qui existe dans le monde vivant. Il y a plusieurs type de codage : binaire, réel, codage de gray et codage dynamique.

II.2.4.1. Le codage binaire

Ce codage a été le premier à être utilisé dans le domaine des **AG**. Il présente plusieurs avantages : alphabet minimum {0,1}, facilité de mise en point d'opérateurs génétiques et existence de fondements théoriques (théorie sur les schémas). Néanmoins ce type de codage présente quelques inconvénients :

- Les performances de l'algorithme sont dégradées devant les problèmes d'optimisation de grande dimension à haute précision numérique. Pour de tels problèmes, les **AG** basés sur les chaînes binaires ont de faibles performances.

- La distance de Hamming entre deux nombres voisins (nombre de bits différents) peut être assez grande dans le codage binaire : l'entier 7 correspond à la chaîne 0111 et la chaîne 1000 correspond à l'entier 8. Or la distance de hamming entre ces deux chaînes est de 4, ce qui crée bien souvent une convergence, et non pas l'obtention de la valeur optimale.

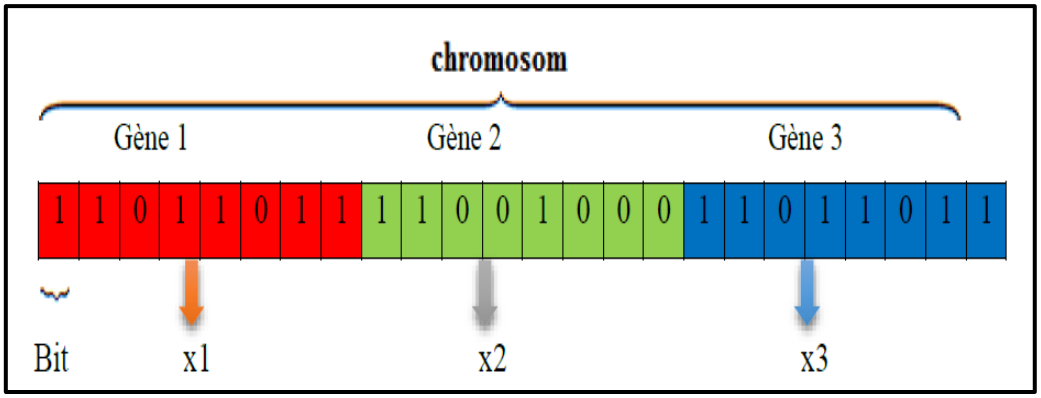

Figure II.2: Illustration du codage des variables d'optimisation.

II.2.4.2. Le codage réel

Il a le mérite d'être simple. Chaque chromosome est en fait un vecteur dont les composantes sont les paramètres du processus d'optimisation. Par exemple, si on recherche

l'optimum d'une fonction de n variables $f\ (x_1,\ x_2,\ldots,\ x_{n-1},\ x_n)$, on peut utiliser tout simplement un chromosome **Ch**. contenant les n variables :

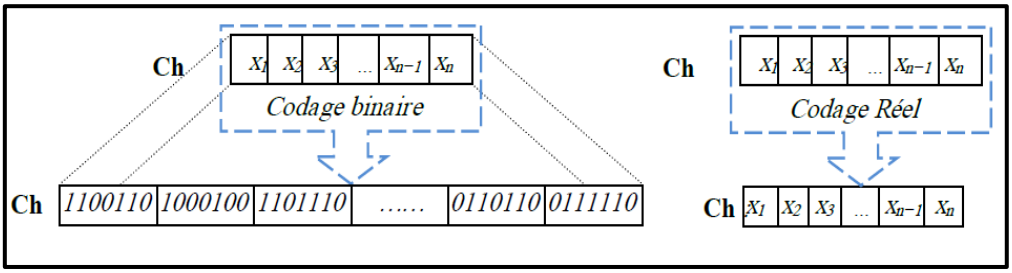

Figure II.3: Le codage réel et le codage binaire.

Avec ce type de codage la procédure d'évaluation des chromosomes est plus rapide vu l'absence de l'étape de transcodage (du binaire vers le réel). Les résultats donnés par Michalewicz **[10]** montrent que la représentation réelle aboutit souvent à une meilleure précision et un gain important en termes de temps d'exécution.

II.2.5. Génération de la population initiale

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

II.2.6. Fonction d'évaluation

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

$$max_{x\,\epsilon\,E}\quad f\,(x) \qquad \textbf{(II.1)}$$

Dans beaucoup de problèmes, l'objectif est exprimé sous forme de minimisation d'une fonction coût $\boldsymbol{h}$:

$$min_{x\,\epsilon\,E}\quad h\,(x) \qquad \textbf{(II.2)}$$

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

$$f(\mathrm{x}) = \frac{1}{(1+\mathrm{h}\,(x)\,)} \qquad \textbf{(II.3)}$$

II.2.7. Les mécanismes d'un AG

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

II.2.7.1. La sélection

La sélection est la première étape du fonctionnement d'un algorithme génétique. Cette étape permet de choisir les individus qui vont accéder à la génération intermédiaire, pour se reproduire et former la nouvelle génération.

La sélection doit favoriser les meilleurs éléments selon le critère à optimiser (minimiser ou maximiser), Cet opérateur ne crée pas de nouveaux individus mais identifie les individus sur la base de leur fonction d'adaptation.

On trouve essentiellement cinq types de méthodes de sélection différentes :

1) La méthode de la "loterie biaisée" (roulette Wheel) de Gold Berg,
2) La sélection par tournois,
3) La méthode "élitiste"
4) La sélection universelle stochastique.
5) La sélection par range.

Mais les plus connues et les plus utilisées étant la « roue de loterie» est la plus ancienne et la « sélection par tournoi».

a. Sélection par roue de loterie biaisée

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

$$p_i(C_i) = \frac{f(ch_i)}{\sum_{j=1}^{N} f(ch_j)} \quad \textbf{(II.4)}$$

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

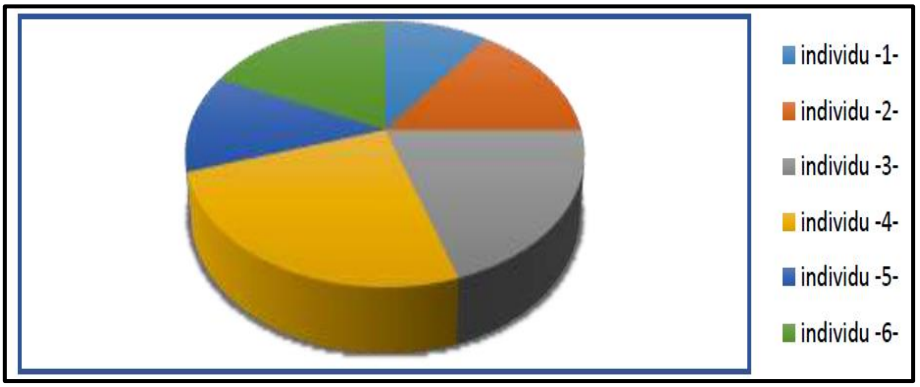

Figure II.4: Représentation de sélection par roulette.

b. La méthode élitiste

La stratégie élitiste consiste à conserver le meilleur individu à chaque génération. Ainsi l'élitisme empêche l'individu le plus performant de disparaître au cours de la sélection ou que ses bonnes combinaisons soient affectées par les opérateurs de croisement et de mutation.

Après chaque évaluation de la performance des individus à une génération g donnée, le meilleur individu de la génération précédente $(g - 1)$ est réintroduit dans la population si aucun des individus de la génération g n'est meilleur que lui. Par cette approche, la performance du meilleur individu de la population courante est monotone de génération en génération.

c. La sélection par tournois

Choisir aléatoirement deux individus et on compare leur fonction d'adaptation (combattre) et on accepte la plus adapte pour accéder à la génération intermédiaire, et on répète cette opération jusqu'à remplir la génération intermédiaire (N/2 composants).les individus qui gagnent à chaque fois on peut les copier plusieurs fois ce qui favorisera la pérennité de leurs gènes**.**

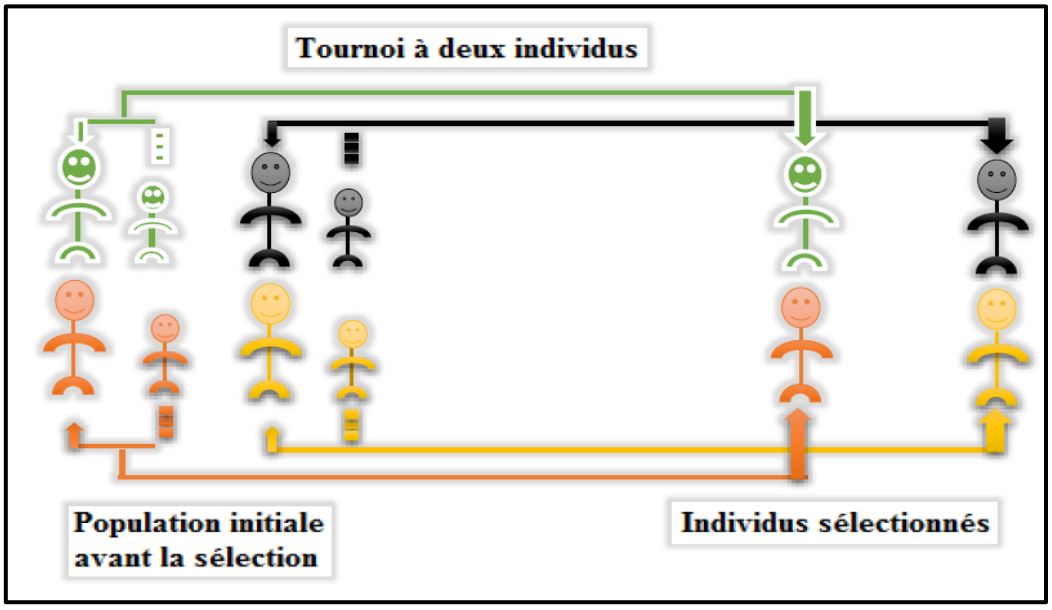

Figure II.5: Représentation d'une sélection par tournoi d'individus pour un critère de maximisation (chaque individu représente une solution possible).

II.2.7.2. Le croisement

a. Le croissement binaire

Le croisement est un processus aléatoire de probabilité P_C appliqué séquentiellement à des couples de parents pris au hasard dans la population. Il consiste à échanger une partie du matériel génétique des parents pour former deux nouveaux individus (enfants).

Dans les algorithmes génétiques, le croisement est considéré comme le principal opérateur pour produire des nouveaux chromosomes. Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes.

Classiquement, les croisements sont envisages avec deux parents et génèrent deux enfants. Initialement, le croisement associé au codage par chaines de bits est le croisement à découpage de chromosomes. Pour effectuer ce type de croisement sur des chromosomes

constitues de m gènes, on tire aléatoirement une position dans chacun des parents. On échange ensuite les deux sous-chaines terminales de chacun des deux chromosomes, ce qui produit deux enfants **C1** et **C2.**

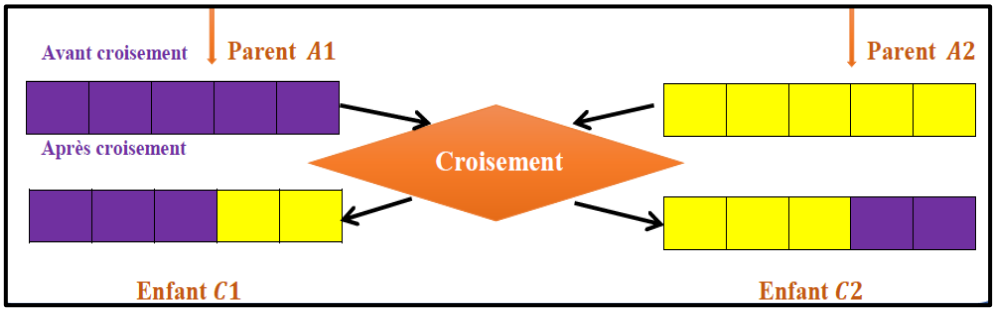

Figure II.6: Croisement en seul point.

On peut étendre ce principe en découpant le chromosome non pas en 2 sous-chaînes mais en 3, 4, etc.

A la différence du croisement seul point, ce type de croisement s'applique en plusieurs points (m points) et chaque chromosome sera ainsi découpé en (m+1) segments.

La position de chaque point *Mi*, se détermine aléatoirement, Ce type de croisement à découpage de chromosomes est très efficace pour les problèmes discrets.

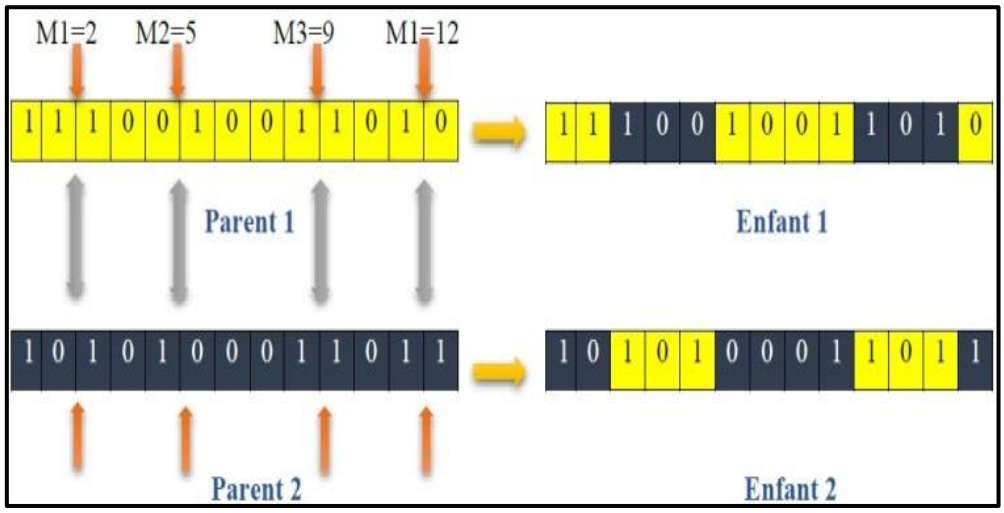

Figure II.7: Croisement en multipoints (m = 4).

❖ Le croissement uniforme

Le croisement uniforme est obtenu à partir d'un masque binaire initialisé aléatoirement et possédant un nombre de bits égal au nombre de bits des individus de la population.

Le première enfant est créé en prenant les gènes de premier parent lorsque les bits correspondant dans le masque valent 1 et les gènes du deuxième parent si ces derniers

valent 0.

Le deuxième enfant s'obtient de la même manière en complémentant le masque.

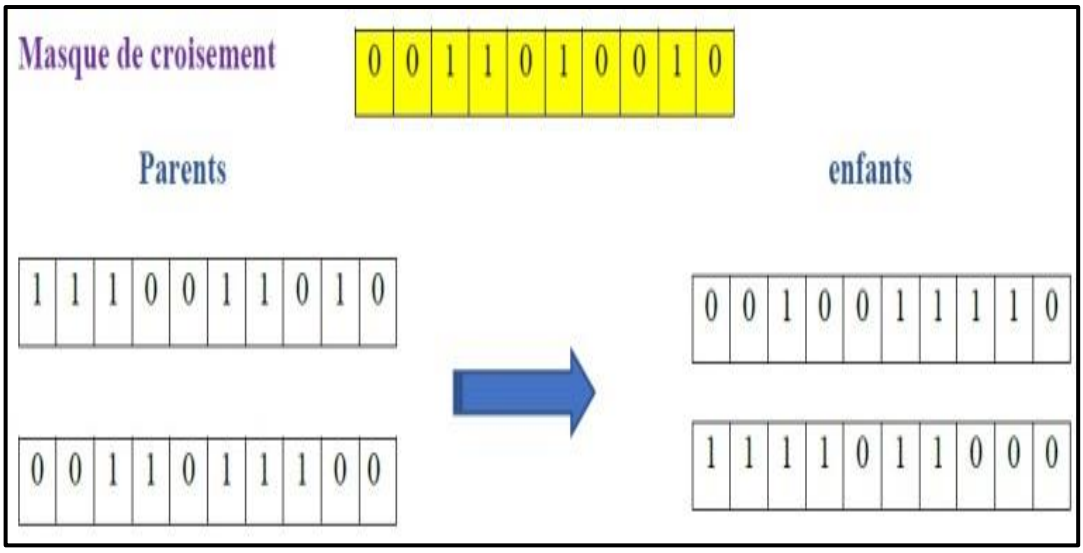

Figure II.8: Le croisement uniforme.

La probabilité de croisement a une influence considérable sur la vitesse de convergence d'un algorithme génétique. Plus elle est grande et plus elle favorise la recombinaison des individus tout en favorisant de tomber dans un optimum local. Les valeurs classiques pour ce paramètre varient 0.6 à 0.95.

b. Le croissement réel

Le croisement réel ne se différencie du croisement binaire que par la nature des éléments qu'il altère : ce ne sont plus des bits qui sont échangés à droite du point de croisement, mais des variables réelles.

Le croissement arithmétique

Le croisement arithmétique est propre à la représentation réelle.il s'applique à une paire de chromosomes et se résume à une moyenne pondérée des variables des deux parents.

II.2.7.3. La mutation

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

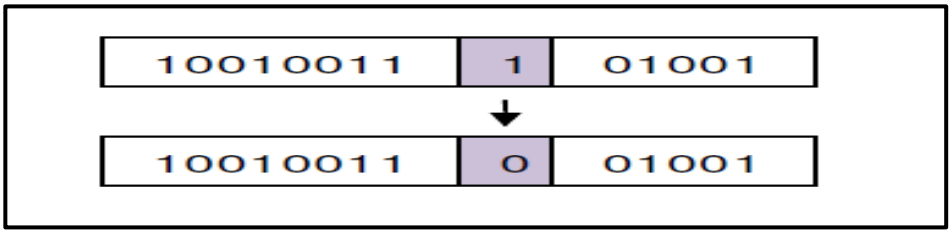

Figure II.9: Exemple d'une opération de mutation.

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

$$P_m = {}^{1}/_{L}. \qquad \textbf{(II.5)}$$

Où L est la longueur du chromosome.

Il existe plusieurs méthodes de mutation parmi lesquels on trouve :

a. Mutation binaire

La mutation revient à modifier aléatoirement la valeur d'un paramètre. Elle constitué un opérateur de recherche secondaire qui favorise l'apparition de nouvelles caractéristiques génétiques. Difficiles à obtenir par le seul opérateur de croisement.

Le rôle principal de la mutation est remédié à ce type de dégénérescence. Une implémentation possible est la complémentation d'un bit dans un chromosome.

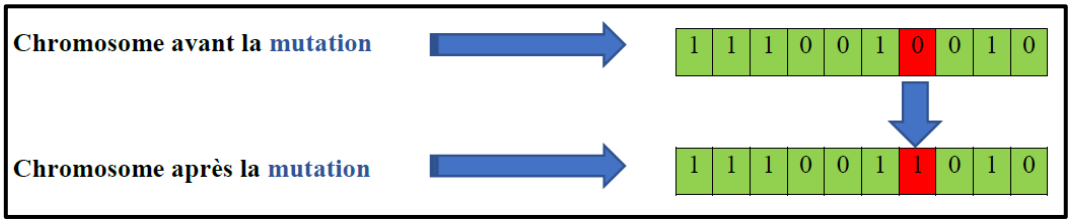

Figure II.10: La mutation binaire.

Elle consiste à altérer un ou plusieurs gènes du chromosome d'un enfant donné, introduisant de ce fait une diversité dans la structure de la population. Cet opérateur permet ainsi l'exploration de l'espace des solutions. Il est caractérisé par la probabilité de mutation (P_m) qui détermine si un enfant doit subir une mutation.

La probabilité de mutation doit être assez faible par rapport à celle du croisement de manière à ne pas perturber l'évolution de l'algorithme. Une valeur élevée transformera l'algorithme en une recherche aléatoire, alors qu'une valeur très faible rendra impossible l'extraction des optimums locaux. Les valeurs classiques pour ce paramètre varient de 0.001 à 0.2.

b. Mutation réelle

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

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

c. Mutation uniforme

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

d. Mutation non uniforme

Le calcul de la nouvelle valeur d'un gène est un peu plus complexe. Le gène X_k subit des modifications importantes durant les premières générations, puis graduellement décroissantes au fur et à mesure que l'on progresse dans le processus d'optimisation. Pour une génération **t**, on tire au sort une valeur binaire qui décidera si le changement doit être positif ou négatif. La nouvelle valeur X'_k du gène X_k est donnée par :

$$X'_K = \begin{cases} X_K + \Delta(t, X_K^{max} - X_K) \;\; Si \quad r = 0 \\ X_K - \Delta\left(t, X_K - X_K^{min}\right) \;\; Si \quad r = 1 \end{cases} \qquad \textbf{(II.5)}$$

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

II.2.8. Critère d'arrêt

Les opérateurs de reproduction peuvent être mis en œuvre de différentes façons qui ont une influence forte sur le comportement de l'algorithme.

Le critère d'arrêt est une caractéristique essentielle des algorithmes génétiques. Plusieurs critères sont proposés :

- un nombre maximum de générations.
- une valeur de fitness minimale.
- une convergence vers la meilleure solution.
- Le temps de calcul atteint une valeur prédéterminée.
- …etc.

II.2.9. Domaine d'application des algorithmes génétiques

Les applications des **AG** sont multiples :

- optimisation de fonctions numériques difficiles (discontinues…).
- traitement d'image (alignement de photos satellites, reconnaissance de suspects…).
- optimisation d'emplois du temps.
- optimisation de design.
- contrôle de systèmes industriels.
- apprentissage des réseaux de neurones
- Les AG peuvent être utilisées pour contrôler un système évoluant dans le temps (chaîne de production, centrale nucléaire…).
- Les AG sont également utilisées pour optimiser des réseaux (câbles, fibres optiques, mais aussi eau, gaz…), des antennes
- TSP (voyageur de commerce)
- Bin Packing-Remplissage de boites rectangulaires.
- Construction de réseaux de communication.
- Contrôle de pipe-lines et d'autres systèmes complexes.
- Constitution des équipes de travail.

II. 3. L'algorithme du recuit simulé :

II.3.1. Définition

L'idée principale du recuit simulé tel qu'il a été proposé par Metropolis en 1953 est de simuler le comportement de la matière dans le processus du recuit très largement utilisé dans la métallurgie. Le but est d'atteindre un état d'équilibre thermodynamique, cet état d'équilibre (où l'énergie est minimale) représente - dans la méthode du recuit simulé – la solution optimale d'un problème ; L'énergie du système sera calculée par une fonction coût (ou fonction objectif) spécifique à chaque problème. La méthode va donc essayer de trouver la solution optimale en optimisant une fonction objectif, pour cela, un paramètre fictif de température a été ajouté par Kirk Patrick, Gelatt et Vecchi **[11]**. En gros le principe consiste à générer successivement des configurations à partir d'une solution initiale $\boldsymbol{S_0}$ et d'une température initiale $\boldsymbol{T_0}$ qui diminuera tout au long du processus jusqu'à atteindre une température finale ou un état d'équilibre (optimum global).

II.3.2. Principe de fonctionnement l'algorithme du recuit simulé

Le recuit simulé applique itérativement l'algorithme de Metropolis, pour engendrer une séquence de configurations qui tendent vers l'équilibre thermodynamique :

1) Choisir une température de départ $T=T_0$ et une solution initiale $S=S_0$;

2) générer une solution aléatoire dans le voisinage de la solution actuelle ;

3) comparer les deux solutions selon le critère de Metropolis ;

4) répéter 2 et 3 jusqu'à ce que l'équilibre statistique soit atteint ;

5) décroitre la température et répéter jusqu'à ce que le système soit gelé.

Dans un premier temps, T étant généralement choisie très grande, beaucoup de solutions - même celles dégradant la valeur de f - sont acceptées, et l'algorithme équivaut à une visite aléatoire dans l'espace de solutions. Mais à mesure que la température baisse, la plupart des solutions augmentant l'énergie sont refusées, et l'algorithme se ramène à une amélioration itérative classique.

A température intermédiaire, l'algorithme autorise de temps en temps des transformations qui dégradent la fonction objectif. Il laisse ainsi une chance au système de s'extraire d'un minima local.

Notons aussi que si la température est égale à 0, seules les solutions optimisant *f* sont acceptées. L'algorithme se comportera donc comme la méthode de la descente du gradient.

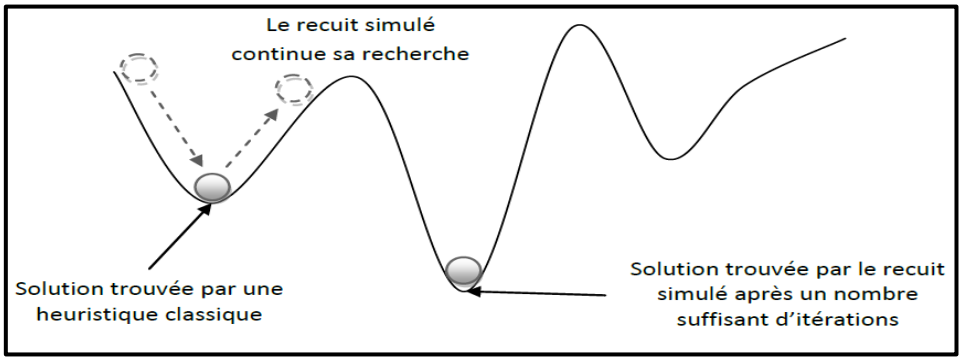

Figure II.11: comparaison entre le recuit simulé et une heuristique classique.

II.3.2.1. Etat initial de l'algorithme

La solution initiale peut être prise au hasard dans l'espace des solutions possibles, elle peut aussi être générée par une heuristique classique, telle que la descente du gradient ou l'algorithme glouton.

La température initiale doit être assez élevée, car c'est elle qui fixe la probabilité d'accepter ou de refuser les solutions défavorables à l'optimisation de la fonction *f*.

II.3.2.2. Variation de la température

Deux approches sont possibles pour décroitre la température :

a) décroissance par paliers :

Pour chaque valeur de la température, on itère l'algorithme de Metropolis jusqu'à atteindre un équilibre statistique, puis on diminue la température.

b) Décroissance continue :

On fait baisser la température d'une façon continue, le plus courant est d'utiliser la loi suivante : $T_i+1=\alpha.T_i$ / $\alpha<1$ (en général $\alpha=0.9$ à 0.99)

Remarque : Le paramètre α est à choisir avec précaution. En effet, s'il est choisi trop grand, la température baissera très rapidement et l'algorithme pourra être bloqué dans un minima local. Si au contraire il est choisi trop petit, la température baissera très lentement et le temps de calcul sera très grand.

II.3.2.3. Amélioration :

Cet algorithme est parfois amélioré en ajoutant une variable qui mémorise la meilleure valeur rencontrée jusqu'à présent ; sans cela, l'algorithme pourrait converger vers une certaine solution, alors qu'on avait visité auparavant une solution meilleure.

II.3.3. Domaines d'applications

Comme pour toute méta-heuristique, la méthode du recuit simulé peut être appliquée dans de nombreux problèmes d'optimisation, les chercheurs l'ont utilisé essentiellement dans :

- La conception des circuits intégrés
- Le routage des paquets dans les réseaux ;
- La segmentation d'images ;
- Le problème du voyageur de commerce ;
- Et, le problème du sac à dos.

II.3.4. Avantages et Inconvénients

➢ **Avantages :**

- Facile à implémenter ;
- Donne généralement de bonnes solutions par rapport aux algorithmes de recherche classiques ;
- Peut être utilisé dans la plupart des problèmes d'optimisation ;
- Il converge vers un optimum global (lorsque le nombre d'itérations tend vers l'infini).

➢ **Inconvénients**

- La difficulté de déterminer la température initiale :
  - Si elle est trop basse, la qualité de recherche sera mauvaise,
  - Si elle est trop haute, le temps de calcul sera élevé.

- L'impossibilité de savoir si la solution trouvée est optimale ;
- Dégradation des performances pour les problèmes où il y a peu de minimas locaux (comparé avec les heuristiques classiques comme la descente du gradient par exemple).

II .4. Conclusion

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

Chapitre III.

Optimisation de la base de règles floues pour la Modélisation des Systèmes Non linéaires par l'Algorithme Hybride AG-RS

III.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 **[12-14]**. 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 **[15-17]** 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 hybride proposé AG-RS. Le modèle flou est à base de règles prédéterminée et fixe.

III.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).

III.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 cette 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.

III.2.2. Identification de paramètres

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

Plusieurs chercheurs ont développé des techniques basées sur les métaheuristiques pour identifier les paramètres d'un modèle flou. Par exemple dans **[18]**, la base de règles floues est construite en utilisant la recherche Tabou. Dans **[19]**, l'optimisation par les colonies de fourmis est utilisée pour la conception de la base de règles floues de modèles flous,

L'hybridation des algorithmes génétiques et la recherche Tabou pour optimiser une base de règles floues pour un modèle flou de Takagi-Sugeno d'ordre zéro est proposée dans **[20]** tandis que dans **[21]** une approche hybride de l'optimisation par essaim particulaire et la recherche Tabou est utilisée. Une hybridation des algorithmes génétiques et le recuit simulé est proposée et décrite dans ce qui suit.

III.3. Structure du modèle flou à identifier

III.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 caractérisées par des fonctions d'appartenance triangulaires, et la variable de sortie est caractérisée par des singletons flous.

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

$$R_i : si \;\; X_1(k) \; est \; A_1^i \; Et \; X_2(k) \; est \; A_2^i \; Alors \; y_m \; (k) \; est \; S^i \qquad \textbf{(III.1)}$$

Où R_i est la $i^{éme}$ 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 (III.2), et S^i sont des singletons flous.

$$A_j^i(x_j) = \max\left(\min\left[\frac{x_{j1}-a_{j1}}{a_{j2}-a_{j1}}, \frac{a_{j3}-x_j}{a_{j3}-a_{j2}}, 0\right]\right) \quad \textbf{(III.2)}$$

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

- $R_1: si\ X_1(k)\ est\ N(-1, a_{11}, a_{12}) Et\ X_2(k)\ est\ N(-1, a_{21}, a_{22})$
  $Alors\ y_m(k) = S_1$.
- $R_1: si\ X_1(k)\ est\ N(-1, a_{11}, a_{12}) Et\ X_2(k)\ est\ Z(a_{21}, a_{22}a_{23})$
  $Alors\ y_m(k) = S_2$.
- $R_2: si\ X_1(k)\ est\ Z(a_{11}, a_{12}, a_{13}) Et\ X_2(k)\ est\ N(-1, a_{21}, a_{22})$
  $Alors\ y_m(k) = S_3$.
- $R_2: si\ X_1(k)\ est\ Z(a_{11}, a_{12}, a_{13}) Et\ X_2(k)\ est\ Z(a_{21}, a_{22}a_{23})$
  $Alors\ y_m(k) = S_4$.
- $R_3: si\ X_1(k)\ est\ P(a_{12}, a_{13}, 1) Et\ X_2(k)\ est\ Z(a_{21}, a_{22}a_{23})$
  $Alors\ y_m(k) = S_5$.
- $R_3: si\ X_1(k)\ est\ P(a_{12}, a_{13}, 1) Et\ X_2(k)\ est\ P(a_{22}, a_{23}, 1)$
  $Alors\ y_m(k) = S_6$.

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

III.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 $\overrightarrow{X} = (x_1, x_2)$, les degrés d'activation $\beta_i(\overrightarrow{X})$ de la règle i est calculée par :

$$\beta_i(\overrightarrow{X}) = \prod_{j=1}^{2} A_j^i(x_j) \quad \textbf{(III.3)}$$

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

soit :

$$y_m = \frac{\sum_{i=1}^{r} \beta_i(\vec{X})\, S_i}{\sum_{i=1}^{r} \beta_i(\vec{X})} \qquad \textbf{(III.4)}$$

Où S_i est la valeur de la conclusion de $i^{éme}$ 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 III.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.

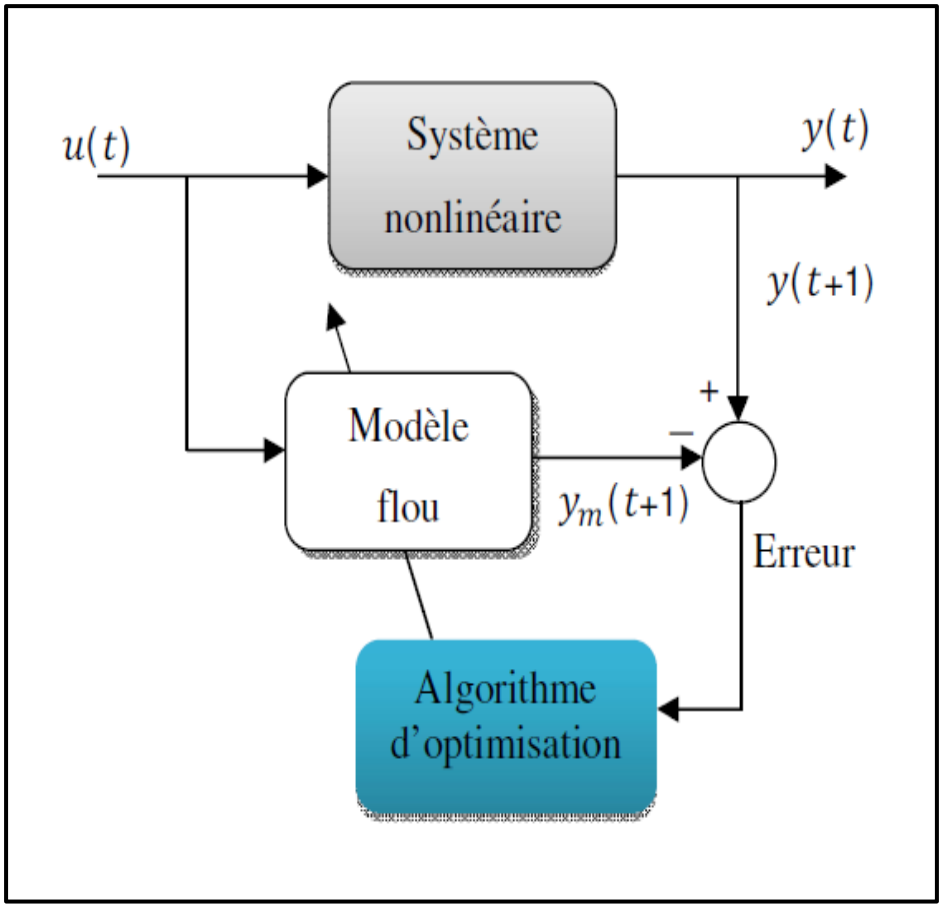

Figure III.1 : Structure de modélisation.

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

- l'erreur quadratique moyenne définie par :

$$fit = EQM = \left(\frac{1}{nT}\right)\sum_{k=1}^{n} e^2\,(K) \qquad \textbf{(III.5)}$$

Où, n est le nombre total d'échantillons, $e(k) = \mathrm{y(k)} - \mathrm{y_m}\,(k)$ et T est le

Temps d'échantillonnage.

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

$$fit = \sqrt{EQM} = \sqrt{\left(\frac{1}{nT}\right)\sum_{k=1}^{n} e^2\,(K)} \qquad \textbf{(III.6)}$$

III.3.3. Système non linéaire de Narendra et Parthasarathy

On considère un système non linéaire décrit par **[22]** :

$$\mathrm{y(k+1)} = \frac{\mathrm{y(k)}}{1+\mathrm{y^2(k)}} + u^2(\mathrm{k}) \qquad \textbf{(III.7)}$$

Où $u(\mathrm{k})$ et $\mathrm{y(k)}$ sont les entrées et sorties du système, respectivement, à l'index de temps k (figure III.2).

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

$$u(\mathrm{k}) = \sin\left(\frac{\pi k}{50}\right) * \cos\left(\frac{\pi k}{30}\right), \quad 0 \le k \le 250 \qquad \textbf{(III.8)}$$

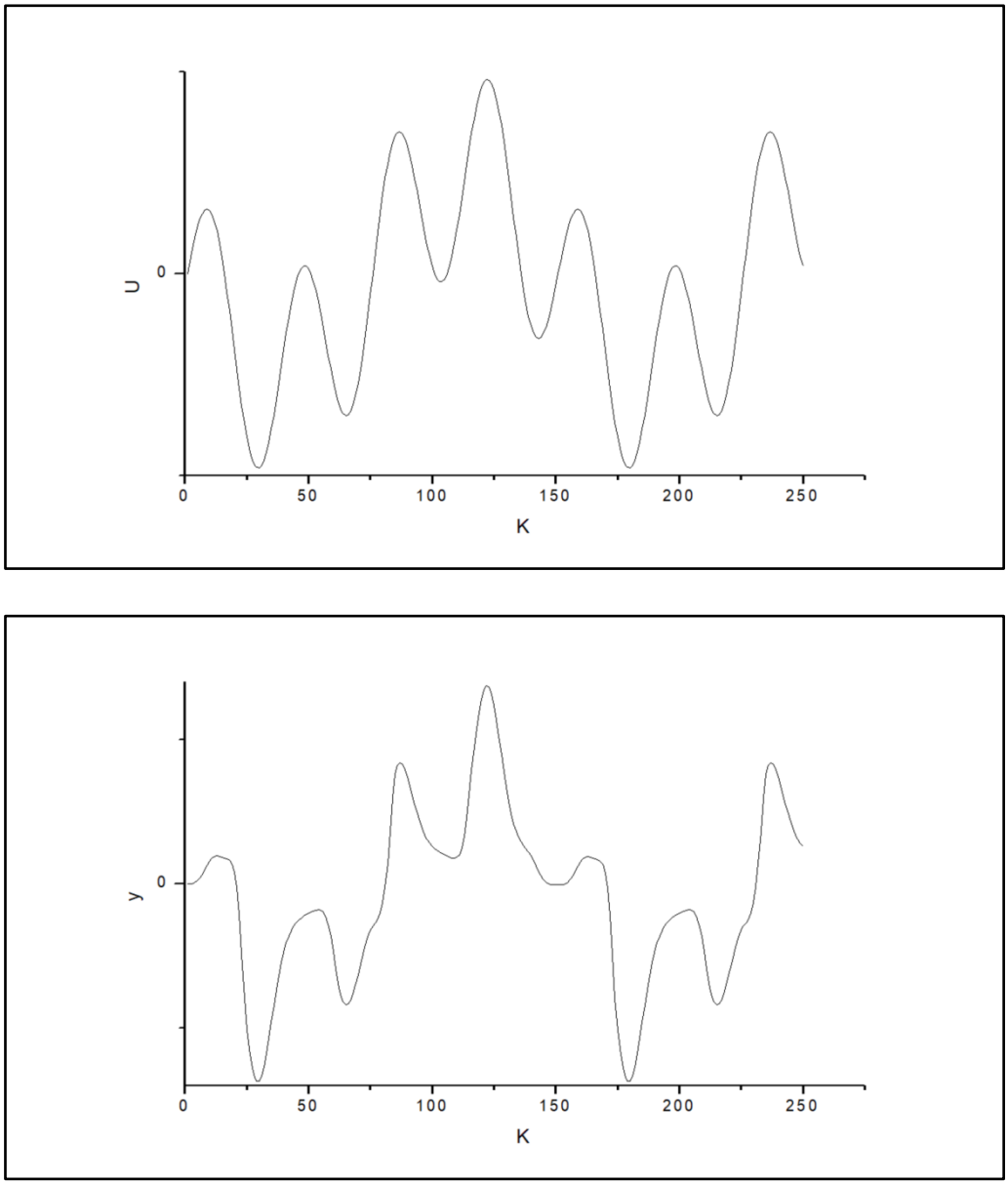

Figure III.2 : Données d'entrée et de sortie du système de Narendra et Parthasarathy.

Les entrées du model flou sont u(k) et y(k) tandis que la sortie est

$y_m\ (k+1)$. La fonction d'évaluation fit est donnée par :

$$fit = \sqrt{\frac{\sum_{k=0}^{249}(y_m(k+1) - y(k+1))^2}{250}} \qquad \textbf{(III.9)}$$

III.4. Simulations

Pour illustrer les performances de l'algorithme hybride d'optimisation AG-RS dans le domaine de conception de modèles flous, l'exemple de Narendra and Parthasarathy **[22]**, est utilisé.

III.4.1. Identification des modèles flous par l'approche AG

Avant de tester les performances de l'algorithme hybride **AG-RS** pour l'identification des modèles flous, il faut préparer une plateforme pour l'algorithme principal des algorithmes génétiques. Pour cela les caractéristiques suivantes ont été considérées pour les **AG** :

- Le codage utilisé pour le chromosome est le codage réel.
- La population initiale sera générée aléatoirement pour couvrir tous l'espace de recherche.
- La méthode de la "loterie biaisée" est choisie pour l'opération de sélection.
- Le croisement en 1 point est utilisé.
- La mutation uniforme est choisie car elle est la plus simple et la plus adaptée au codage réel.
- La sélection finale pour la nouvelle génération sera par compétition, pour que les meilleurs individus (enfants et parents) participent dans la nouvelle génération.

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

Les valeurs spécifiques de l'algorithme **AG** pour l'identification du modèle flou de Narendra et Parthasarathy sont données par le tableau III.1. La figure III.3 donne l'évolution de l'indice de performance durant l'exécution de l'algorithme **AG**.

Caractéristique	Valeur
Taille de la population (N)	10
Nombre de générations (max _ gen)	5000
Probabilité de croisement (Pc)	0.9
Probabilité de mutation (Pm)	0.1
Univers de discours	[-1, 1]

Tableau III.1 : Valeurs spécifiques de l'algorithme d'optimisation **AG**.

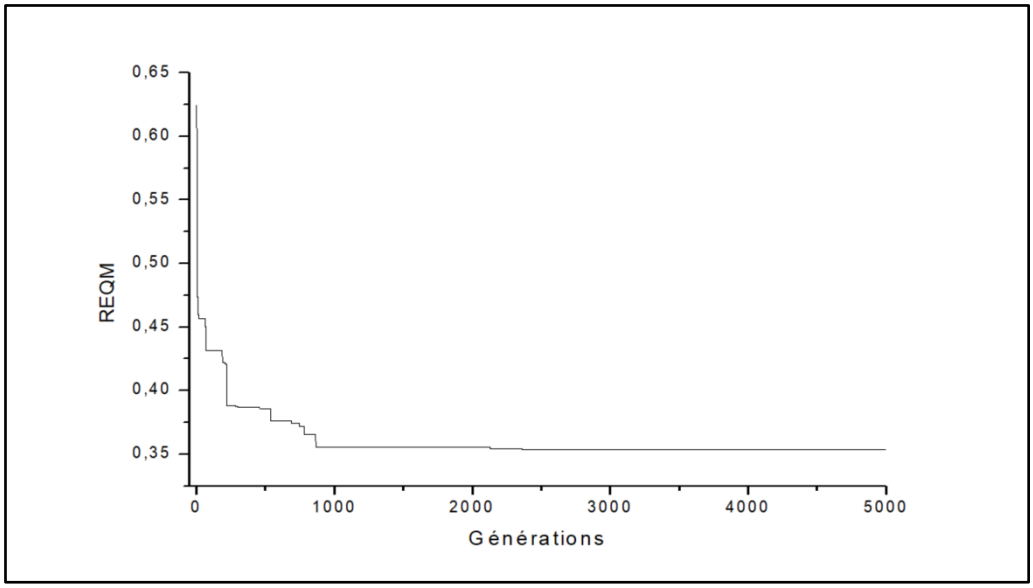

Figure III.3 : Evolution de la fonction d'évaluation.

A la fin de l'exécution de l'algorithme **AG** on obtiendra les valeurs suivantes :

a11 = -0.86; a12 = -0.025; a13 = 0.87;

a21 = -0.26; a22 = -0.25; a23 = 0.1;

S1 = -0.1; S2 = -0.37; S3 = -0.37; S4 = -0.37; S5 = 0.134; S6 = 0.85.

Le schéma de la figure III.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.

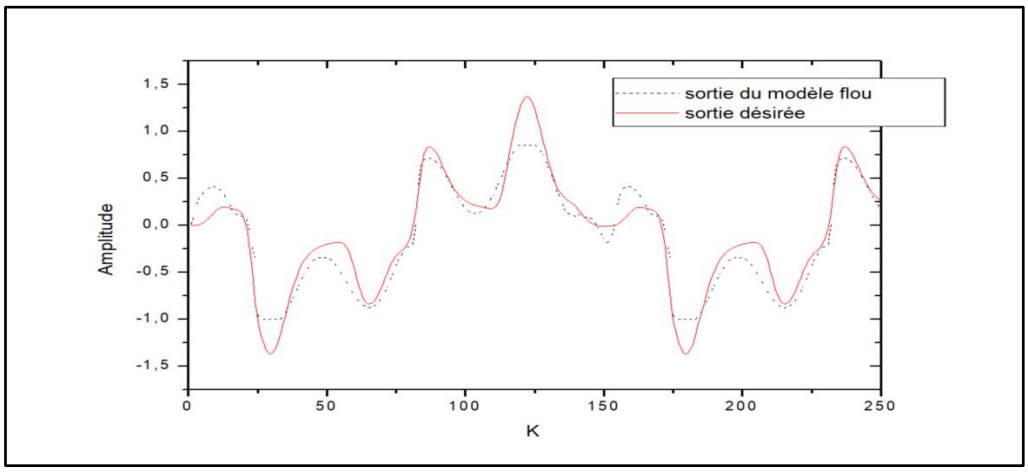

Figure III.4 : Résultats d'optimisation du modèle flou par **AG**

III.4.2. Identification des modèles flous par l'approche AG-RS

Les valeurs spécifiques de l'algorithme hybride **AG-RS** pour l'identification du modèle flou de Narendra et Parthasarathy sont données par le tableau III.2.

Caractéristique	**Valeur**
Taille de la population (N)	10
Nombre de générations (max _ gen)	5000
Probabilité de croisement (P_c)	0.9
Probabilité de mutation (P_m)	0.1
Univers de discours	[-1, 1]
Température initiale (T_0)	100
Coefficient de température (λ)	0.9
Nombre de solutions voisines (N_v)	24

Tableau III.2 : Valeurs spécifiques de l'algorithme hybride d'optimisation **AG-RS**.

La figure III.5 donne l'évolution de l'indice de performance durant l'exécution de l'algorithme **AG-RS**.

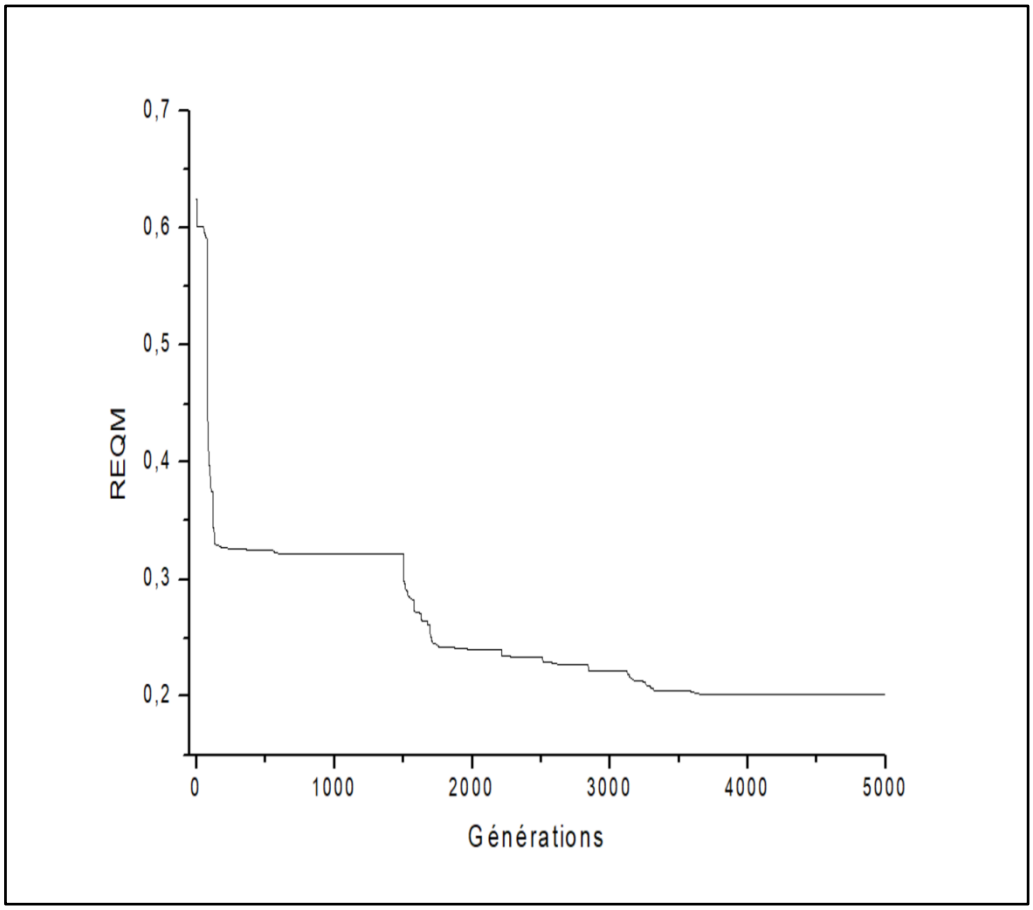

Figure III.5 : Evolution de la fonction d'évaluation.

A la fin de l'exécution de l'algorithme hybride **AG-RS** on obtiendra les valeurs suivantes :

a11 = -0.375; a12 = -0.17; a13 = 0.97;

a21 = -0.648; a22 = 0.603; a23 = 0.644;

S1 = -0.74; S2 = -0.028; S3 = 0.28; S4 = 0.2; S5 = 0.82; S6 = 1.

Le schéma de la figure III.6 montre la sortie originale du système ainsi que la réponse du modèle flou optimisé. On remarque une meilleure convergence de la réponse du modèle flou vers la sortie désirée.

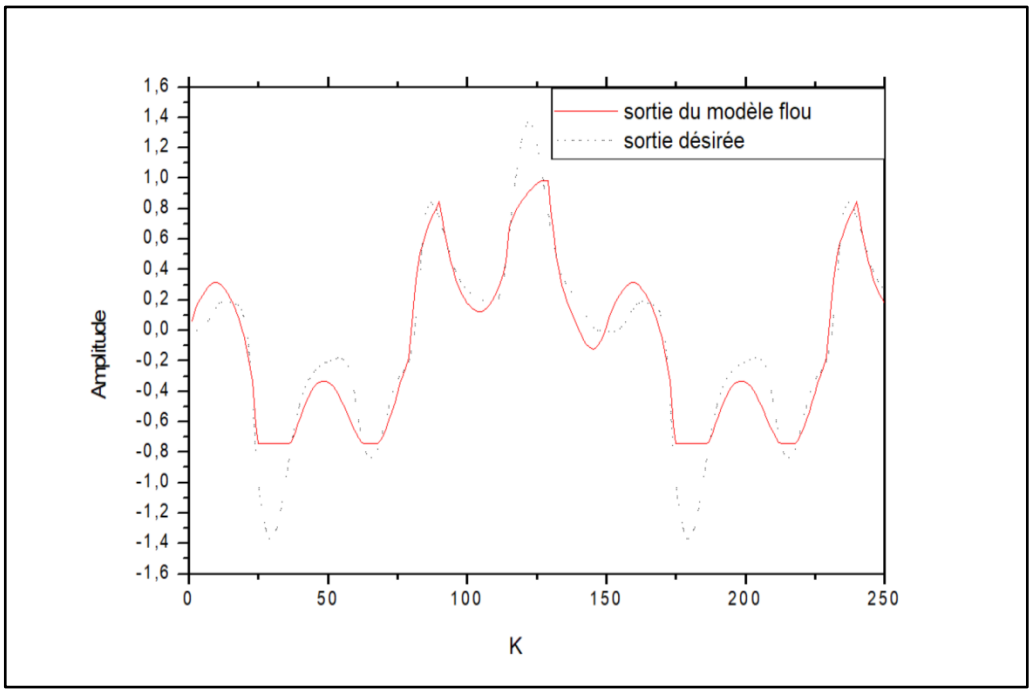

Figure III.6 : Résultats d'optimisation du modèle flou par **AGE-RS**

Les résultats d'identification du modèle flou de Narendra et Parthasarathy par **AG et AG-RS** sont comparés. En se basant sur les résultats du tableau III.3, on note que l'hybridation améliore la fonction d'évaluation.

Méthode d'optimisation	Nombre de règles floues	REQM
AG	6	0.353
AG-RS	6	0.202

Tableau III.3 : Comparaison des performances des méthodes d'identification des modèles flous de type TS d'ordre zéro pour le modele de Narendra et Parthasarathy.

III.5. Conclusion

Dans ce chapitre l'algorithme hybride **AG-RS** 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 accomplissent efficacement les performances désirées. L'algorithme **AG** simple et hybride **AG-RS** ont de bonnes performances (grande précision), en utilisant une structure simple du modèle flou (nombre minimum de règles floues et d'ensembles flous).

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

Conclusion générale

Conclusion générale

Le travail présenté dans ce mémoire concerne la modélisation floue des systèmes non-linéaires en utilisant les techniques avancées de l'automatique telles que la logique floue et les outils d'optimisation tels que les algorithmes génétiques.

Les approches proposées reposent sur l'hybridation d'un algorithme évolutionnaire et un autre algorithme dit de recherche locale. Notre objectif est d'exploiter les avantages de chacun d'entre eux pour aboutir à un algorithme de recherche globale qui ne tombe jamais dans les optimums locaux.

La simplicité d'implantation du recuit simulé, son efficacité à cause de sa tolérance graduée des solutions dégradant la fonction objectif, la qualifie d'être la meilleure méthode de recherche locale pour l'hybridation avec les AG.

Les deux approches ont été testées et validées pour le modèle de Narendra & Parthasarathy, en simulation.

Les différentes possibilités qu'offrent les modèles flous du type TS d'ordre zéro pour la modélisation des systèmes non-linéaires ont donné une bonne approximation des systèmes avec une grande précision.

Dans le cadre de développement futur de ce travail, il est intéressant d'élaborer une loi d'identification optimisée par d'autres techniques d'optimisation (colonies de fourmis, colonies d'abeilles, etc.) pour les systèmes multi variables.

Références bibliographiques

Références bibliographiques

[1] Zadeh, L. A. Soft computing and fuzzy logic. IEEE Software, 11(6), pp 48-56, 1994.

[2] Mamdani, E.H., Assilian, S. An experiment in linguistic synthesis with a fuzzy logic controller. Int. J. Man Mach. Studies, 7(1) : 1–13, 1975.

[3] Ostergaad, J.J. Fuzzy logic control of a heat exchange process. Dans Fuzzy Automata and Decision Processes, M.M. Gupta, G.N. Saridis, and B.R. Gaines,Eds., pp 285–320.

[4] Willaeys, D., Malvache, N. Use of fuzzy model for process control. IEEE International Conference on Cybernetics and Society, 1978.

[5] Joao M. C. Sousa, Uzay Kaymar, Fuzzy Decision Making in Modeling and Control, World Scientific, 2002.

[6] George J. Klir, Bo Yuan, Fuzzy Sets and Fuzzy Logic, Theory and Applications, Prentice Hall, 1995.

[7] Bernadette Bouchon-Meunier - Logique floue, principes, aide à la décision, éditions Lavoisier, 2003.

[8] Kumar S Ray, Soft Computing and Its Applications, Volume II: Fuzzy Reasoning and Fuzzy Control, Apple Academic Press, 2014.

[9] D. E. Goldberg, "Algorithmes génétiques : Exploration, optimisation et apprentissage automatique", Addison Wesley France, 1996.

[10] Z. Michalewicz,"Genetic Algorithms + Data Structures = Evolution Programs",

Springer-Verlag, 1992.

[11] Kirkpatrick, Gelatt et Vecchi," Optimization by Simulated Annealing", Science, New Series, pp. 671-680 , Mai 1988.

[12] B. Kosko, "Fuzzy systems as universal approximators", *IEEE Transactions on*

Computers, vol.43, 1994, pp.1329-1333.

[13] J. Buckley, "Universal Fuzzy Controllers", *Automatica*, Vol. 28, No. 6, 1992, pp. 1245-1248.

[14] J. Buckley, "Sugeno type Controllers are Universal Controllers", *Fuzzy sets and Systems 53,* pp. 299-303, 1993.

[15] X.J. Zengand M.G. Singh, "Fuzzy Bounded Least Squares Method for the Identification of Fuzzy Systems", *Proc. of the IEEE Conf. on Fuzzy Systems (Fuzz´IEEE 97),* pp. 403-408, Bercelone, Espagne, 1997.

[16] E. Kim, P.Park, S.Ji and M.Park, "A New Approch to Fuzzy Modeling", *IEEE Trans.on Fuzzy Systems,* Vol. 5, No. 3, August 1997, pp. 328-337.

[17] P.Y. Glorennec, "Algorithme d'apprentissage pour systèmes d'inférence floue", *Edition Hermès,* Paris, 1999.

[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*

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

[22] K.S. Narendra, K. Parthasarathy, "Identification and control of dynamical systems using neural networks", *IEEE Trans Neural Network,*1990, 1(1), pp. 4–27.

Résumé:

Les travaux présentés dans ce mémoire s'articulent, essentiellement, autour des principaux axes de la modélisation floue du type Takagi-Sugeno d'ordre zéro pour des systèmes dynamiques, complexes et fortement non-linéaires.

En premier lieu, une méthode d'identification a été proposée en utilisant plusieurs algorithmes hybrides basés sur la combinaison des algorithmes de recherche globale, les Algorithmes Génétiques (AG) et l'algorithme de recherche locale, le Recuit Simulé (RS). L'hybridation consiste à combiner les caractéristiques de ces méthodes (AG-RS) pour tirer des avantages permettant d'aboutir à une bonne solution en un temps de calcul réduit, tous en garantissant la précision des systèmes identifiés.

Les résultats de modélisation ont été comparés via un certain critère de performance.

***Mots-clés* :** Base de règles floues**,** Recuit Simulé, Algorithmes Génétiques,Modélisation floue.

Abstract *:*

The work presented in this thesis is essentially articulated around the main axes of fuzzy modeling of the Takagi-Sugeno type of zero order for dynamic, complex and strongly non-linear systems. First, an identification method has been proposed using several hybrid algorithms based on the combination of global search algorithms, Genetic Algorithms (GA) and the local search algorithm, Simulated Annealing (RS). Hybridization consists in combining the characteristics of these methods (AG-RS) to derive advantages allowing a good solution to be obtained in a reduced computation time, all while guaranteeing the precision of the systems identified. The modeling results were compared using a certain performance criterion.

Keywords: Fuzzy rule base, Simulated Annealing, Genetic Algorithms, fuzzy modeling.

ملخص:

العمل المقدم في هذه المذكرة يتمحور أساسا حول المجالات الرئيسية للنمذجة والمراقبة الغامضة من نوع

Takagi- Sugeno رتبة صفر من أجل أنظمة ديناميكية معقدة وغير خطية للغاية.

أولاً، تم اقتراح طريقة تحديد باستخدام عدة خوارزميات هجينة بناءً على مزيج من خوارزميات البحث العالمية والخوارزميات الجينية (GA) وخوارزمية البحث المحلي، التلدين المحاكي (RS). يتكون التهجين من الجمع بين خصائص هذه الطرق (AG-RS) من أجل الحصول على مزايا تسمح بالحصول على حل جيد في وقت حساب مخفض ، مع ضمان دقة الأنظمة المحددة. تمت مقارنة نتائج النمذجة باستخدام معيار أداء معين.

الكلمات المفتاحية: قاعدة القاعدة الغامضة، التلدين المحاكي، الخوارزميات الجينية، النمذجة الغامضة.