République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud MAMMERI, Tizi-Ouzou

Faculté de Génie Electrique et d'Informatique
Département d'Automatique

Mémoire de Fin d'Etudes

En vue de l'obtention du diplôme de
Master académique en automatique
Option commande des systèmes

Thème

Optimisation d'un contrôleur flou par les algorithmes génétiques.

Présentée par :

Nour El Houda KHELOUI. Nabila LEKRIM.

Encadreur : Mr. Mohand A.TOUAT.

Président : Mr. HAMMACHE.

Examinateur : Mr. CHELLI.

Examinateur : Mme. YOUCEFI.

Soutenu le: 01 /10/2016

Promotion 2016

Remerciements

Nous tenons à exprimer toute notre reconnaissance à notre encadreur de mémoire **Mr M.TOUAT**. Nous le remercions de nous avoir orientés, aidés, conseillés et pour la patience dont il nous a fait preuve pendant nos trois années d'étude.

Nous adressons nos sincères remerciements à tous les professeurs intervenants **Mr S.DJENOUN, Mr M.MAIDI** ainsi que **Mr R.KARA**, qui par leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé nos réflexions et ont accepté à nous rencontrer et répondre à nos questions durant notre année d'étude.

Nous remercions nos très chers parents, qui ont toujours été là pour nous, «Vous avez tout sacrifié pour vos enfants n'épargnant ni santé ni efforts. Vous nous avez donné un magnifique modèle de labeur et de persévérance. Nous vous en sommes redevable d'une éducation dont nous en sommes fières».

Nous remercions également nos frères et sœurs pour leur soutien inconditionnel et encouragements qui nous ont toujours accompagnés.

Enfin, nous tenons à remercier tous nos Ami(e)s pour leur sincère amitié, amour, confiance et qui nous ont apporté leur support moral et intellectuel tout au long de ce travail.

À tous ces intervenants, nous leur présentons nos remerciements, notre respect et gratitude en espérant que notre production sera à la hauteur de leur investissement et de leurs attentes.

Nour El Houda & Nabila

SOMMAIRE

SOMMAIRE

Introduction générale

De nos jours, des systèmes de régulation automatique du fonctionnement des processus sont intégrés dans de nombreuses applications, tant dans le domaine scientifique que technologique. Les systèmes devenant de plus en plus complexes, les performances des régulateurs utilisés ne cessent de s'améliorer. Les méthodes de réglage conventionnelles comme la commande optimale, la commande adaptative ou la commande robuste, se basent sur une connaissance plus ou moins précise du modèle mathématique du système à réguler. Lorsque le système est fortement non-linéaire, imprécis ou très complexe, il est parfois impossible de définir un modèle mathématique de son fonctionnement. Dans ce cas, les régulateurs conventionnels sont difficilement utilisables.

La logique floue et les algorithmes génétiques constituent une voie prometteuse pour aborder ce problème particulier de régulation. Ces méthodes font intervenir des mécanismes qui ont été observés et étudiés dans des domaines de recherche initialement très éloignés de l'informatique : linguistique et raisonnement humain pour la logique floue, physiologie humaine et animale pour les réseaux de neurones ou encore génétique et sélection naturelle pour les algorithmes évolutionnistes.

Afin de régler dynamiquement les paramètres d'un système, nous utilisons un contrôleur flou. Ce principe de régulation, basé sur des concepts relativement simples, permet de faire intervenir dans le contrôleur des connaissances acquises par un expert humain. L'intérêt principal de cette méthode est qu'elle ne nécessite pas de connaître le fonctionnement du système, mais simplement la façon de le commander. Les connaissances sont exploitées sous une forme linguistique par l'intermédiaire de règles comme « SI'condition'ALORS'action'». Une méthode de raisonnement (inférence) utilise ces règles pour définir les commandes qui sont envoyées au système. Lorsque les règles de commande ne sont pas connues précisément, la conception du contrôleur flou peut être assimilée à un problème d'optimisation : comment choisir les paramètres du contrôleur afin d'assurer un fonctionnement optimal selon certains critères? De nombreuses techniques d'optimisation ont été décrites dans la littérature (méthodes linéaires, programmation dynamique,…) qui ont trouvé des applications dans la plupart des domaines scientifiques. Dans notre mémoire

nous avons choisi d'utiliser les algorithmes génétiques afin d'optimiser les facteurs d'échelle du contrôleur flou.

Un algorithme génétique est une méthode d'exploration de l'espace des solutions possibles d'un problème exploitant des mécanismes similaires à ceux de la sélection naturelle. A chaque itération, l'algorithme génétique évalue un ensemble limité de solutions du problème, et retient les meilleures d'entre elles (phase de sélection). Ces solutions sont combinées (phase de croisement) et éventuellement légèrement modifiées (phase de mutation) avant d'être à nouveau évaluées. A l'issue de ce processus itératif, on aboutit sous certaines conditions à la solution optimale du problème.

Ce mémoire s'articule autour de quatre chapitres qui se résument comme suit :

Le premier chapitre sera consacré à la logique floue, nous commençons par énoncer ses fondements. Nous voyons comment elle permet d'exprimer selon un formalisme unique des informations très diverses (données incertaines ou imprécises, connaissances exprimées sous forme linguistique...). Ensuite, nous présentons en détails les méthodes de raisonnement flou (propositions, implications et inférence) qui constituent la base de la commande floue. Puis nous décrivons le principe de fonctionnement d'un contrôleur flou.

Dans le deuxième chapitre, nous décrivons le principe des algorithmes génétiques. Nous analysons l'intérêt des différentes phases de traitement : sélection des individus, croisement et mutation. Nous décrivons enfin plusieurs méthodes de représentation des informations traitées par un algorithme génétique sous la forme d'un codage binaire, ou par des nombres réels ou en base n.

L'optimisation d'un contrôleur flou par les algorithmes génétiques fera l'objet du troisième chapitre, en utilisant un algorithme génétique, nous pouvons optimiser les facteurs d'échelle du contrôleur flou.

Dans le quatrième chapitre, nous présentons le système d'un véhicule régulé par le contrôleur flou étudié avec SIMULINK et cela en trois étapes : premièrement une simulation sans optimisation, puis une simulation avec optimisation selon deux méthodes « fminsearch » et les algorithmes génétiques. Enfin une interprétation des résultats ainsi qu'une comparaison seront donnés au final.

1. Introduction :

A partir de concepts mathématiques relativement simples, la logique floue permet de résoudre des problèmes complexes là où les outils de résolution conventionnels peinent. Son succès réside principalement dans son potentiel à résoudre des problèmes sans avoir recours aux modèles mathématiques. L'industrie et les services peuvent bénéficier des atouts de la logique floue, la panoplie des applications à l'échelle planétaire en témoigne. Tous les problèmes de contrôle, de diagnostic ou d'aide à la décision sont des clients potentiels de la logique floue. Ainsi, La logique floue s'est imposée dans des domaines aussi variés que l'électrotechnique, l'industrie, l'automatisme, la robotique, la gestion de la circulation routière, le contrôle aérien, l'environnement et la médecine, afin de résoudre les problèmes d'identification, de régulation de processus, d'optimisation, de classification et de détection de défauts ou de prise de décision. [3]

Le développement des algorithmes flous se fait à travers les méthodes par lesquelles l'homme essaye de copier la nature et de reproduire des modes de raisonnement et de comportement qui lui sont propres.

2. Historique:

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. Il a introduit la notion de sous-ensemble flou pour fournir un moyen de représentation et de manipulation des connaissances imparfaitement décrites, vagues et imprécises. A cette époque, la théorie de la logique floue n'a pas été prise au sérieux et acceptée uniquement par quelques experts [4].

Dès 1975, Mamdani et Assilian publient les premiers résultats permettant une exploitation de cette théorie dans des systèmes de réglage. Utilisant une structure de contrôleur relativement simple, lors de l'étude d'une chaudière à vapeur en utilisant la régulation floue, les variables d'entrées en sont l'erreur en pression PE, l'erreur en vitesse SE, le changement d'erreur en pression CPE et le changement d'erreur en vitesse CSE ; les variables de sortie sont le changement de chaleur HC et le changement de soupape TC. Il comprend 24 règles en deux paquets de 12 correspondant à chacune des variables de sortie. Les univers des variables d'entrées sont discrétisées en 15 points (pour PE et

SE), 13 (pour CPE et CSE) et 5 pour TC. Ainsi ils ont obtenu de meilleurs résultats que ceux fournis par un régulateur standard de type PID [2].

Peu de temps après, en 1977, le danois Ostergaard 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, 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]

3. Logique floue et logique classique :

Dans la logique classique, les variables gérées sont Booléennes. C'est à dire qu'elles ne prennent que deux valeurs 0 ou 1. La logique floue a pour but de raisonner à partir de connaissances imparfaites qui opposent résistance à la logique classique. Pour cela la logique floue se propose de remplacer les variables booléennes par des variables flous.

Autrement dit, dans la théorie ensembliste classique, l'appartenance d'un élément à un ensemble est définie par une valeur logique standard : 1 si l'élément appartient à l'ensemble, 0 sinon.

La logique floue permet de définir des sous-ensembles, en introduisant la possibilité pour une valeur d'appartenir plus ou moins à chacun de ces sous-ensembles, son degré d'appartenance est décrit par une valeur comprise entre 0 et 1. [5]

On constate donc que le raisonnement par la logique floue est identique au raisonnement humain, elle est basée sur la connaissance de l'expert et utilise le langage naturel ce qui rend son utilisation beaucoup plus approprié et plus facile pour la résolution de certains problèmes.

4. Concepts de base :

4.1. Ensembles flous :

Etant donné un ensemble de référence X qui peut être fini ou infini, dénoté par ses éléments $\{x\}$, on peut indiquer les éléments $\{x\}$ qui appartiennent à une certaine classe de X (on leur donne une valeur 1) et ceux qui n'y appartiennent pas (on leur donne une valeur 0), cette classe est alors un sous-ensemble classique de X caractérisé par une fonction caractéristique X_A prenant simplement deux valeurs 0 ou 1 :

$$X_A : X \rightarrow \{0, 1\} \quad \text{..... (1.1)}$$

Si l'appartenance de certains éléments de *X* à une classe n'est pas absolue (l'élément appartient *un peu* au sous-ensemble), on peut remplacer la fonction caractéristique par une *fonction d'appartenance* qui prend ses valeurs dans l'intervalle [0, 1]. Cette classe est appelée sous-ensemble flou de *X*. L'ensemble *X* sera également appelé *univers de discours*. [6]

4.2. Sous ensembles flous:

4.2.1. Définition :

Un sous-ensemble flou *A* dans un univers de discours *X* est caractérisé par sa fonction d'appartenance $\mu_A(x)$ qui associe à chaque élément x de *X* une valeur dans l'intervalle des nombres réels [0, 1].

$$\mu_A(x) : X \rightarrow [0, 1] \quad \ldots\ldots\ldots\ldots\ldots (1.2)$$

Ainsi un sous-ensemble flou *A* dans *X* peut être représenté par un ensemble de couples ordonnés. [5]

$$A = \{(x, \mu_A(x)) | x \in X\} \quad \ldots\ldots\ldots\ldots\ldots (1.3)$$

Le sous-ensemble classique n'est en fait qu'un cas particulier de sous-ensemble flou dont la fonction d'appartenance ne prend que les valeurs 0 ou 1. Un sous-ensemble flou *A* de *X* est aussi souvent représenté par la notation suivante qui indique pour tout élément x de *X* son degré d'appartenance à *A* $\mu_A(x)$:

Si X est continu:

$$A = \int_X \mu_A(x)/x \quad \ldots\ldots\ldots (1.4)$$

Si X est discret:

$$A = \sum_{x_i \in X} \mu_A(x_i)/x_i \quad \ldots\ldots (1.5)$$

Comme les valeurs $\mu_A(x_i)$ représentent les degrés d'appartenance avec lesquels lesx_i appartiennent à A, si $\mu_A(x_i)$ prend la valeur 1 pour tous les éléments de *X*, cela signifie que A est identique à *X*. Au contraire, *A* est vide si $\mu_A(x_i)$ prend la valeur 0 sur tout *X*. [4]

4.2.2. Opérations de base sur les sous ensembles floues :

Considérons 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'égalité, l'inclusion, l'intersection, l'union et le complément grâce à des opérations sur les fonctions d'appartenance. [5]

a) ***L'égalité****:* A et B sont dits égaux, propriété que l'on note *A* = *B*, si leurs fonctions d'appartenance prennent la même valeur en tout point de *X* :

$$\forall x \in X, \quad \mu_A(x) = \mu_B(x) \text{ ..... (1.6)}$$

b) ***L'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 :

$$\forall x \in X, \quad \mu_A(x) \leq \mu_B(x) \text{ ..... (1.7)}$$

Les définitions d'intersection, d'union et de complément de sous-ensembles flous définies ci-dessous font intervenir les opérateurs de minimum, maximum et de complémentation à 1. Cela correspond à une extension des opérateurs ensemblistes standards.

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

$$\forall x \in X, \ \mu_{A \cap B} = \min(\mu_A(x), \mu_B(x)) = \mu_A(x) \wedge \mu_B(x) \text{ ..... (1.8)}$$

Dans cette définition, min et $\wedge$ désignent l'opérateur de calcul du minimum des deux valeurs.

d) ***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:

$$\forall x \in X, \ \mu_{A \cup B} = \max(\mu_A(x), \mu_B(x)) = \mu_A(x) \vee \mu_B(x) \text{ .....(1.9)}$$

Dans cette définition, max et ∪désignent l'opérateur de calcul du maximum de deux valeurs.

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

$$\forall x \in X, \quad \mu_{A^c} = 1 - \mu_A(x) \text{ ..... (1.10)}$$

Comme nous l'avons vu précédemment, la définition d'une opération entre ensembles flous est basée sur une combinaison des fonctions d'appartenance. Les définitions les plus simples utilisent les opérations de minimum, maximum et de complément à 1. Les normes et Co-normes triangulaires constituent une généralisation des opérations de combinaison de type minimum ou maximum.[4]

- ***Norme triangulaire, t-norme*** *:*

Une norme triangulaire T est une fonction définie sur l'ensemble [0, 1] × [0, 1] et prenant ses valeurs dans l'intervalle [0,1], qui satisfait les conditions suivantes :

- ✓ **Commutativité :** $\forall x, y \in [0,1]: \quad T(x,y) = T(y,x) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.11)$
- ✓ **associativité**: $\forall x, y, z \in [0,1]: \quad T(x, T(y,z)) = T(T(x,y), z) \ldots\ldots\ldots\ldots\ldots (1.12)$
- ✓ ***monotonie***: $\forall x, y, z, t \in [0,1]: T(x,y) \leq T(z,t) \; si \; x \leq z \; et \; y \leq t \ldots\ldots\ldots (1.13)$
- ✓ **élément neutre 1**: $\forall x \in [0,1]: \quad T(0,0) = 0, \; T(x,1) = T(1,x) = x \ldots (1.14)$

Intersection définie par une t-norme : On vérifie simplement que l'opérateur minimum est une t-norme. La définition d'une opération d'intersection entre deux sous-ensembles flous se réfère à une t-norme, qui remplace l'opérateur minimum :

$$\forall x \in X, \mu_{A \cap B}(x) = T\,(\mu_A(x), \mu_B(x)) \ldots\ldots\ldots\ldots (1.15)$$

- ***Co-norme triangulaire, t-conorme*** *:*

Une conorme triangulaire ⊥est une fonction définie sur l'ensemble [0, 1] × [0,1] et prenant ses valeurs dans l'intervalle [0, 1], qui satisfait les conditions suivantes :

- ✓ **commutativité**: $\forall x, y \in [0, 1] : \perp (x, y) = \perp (y, x) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.16)$

- ✓ **associativité:** $\forall x, y, z \in [0, 1] : \perp (x, T (y, z)) = \perp (\perp (y, z), z) \ldots\ldots\ldots\ldots (1.17)$
- ✓ **monotonie**: $\forall x, y, z, t \in [0, 1] : \perp (x, y) \leq \perp (z, t)$ si $x \leq z$ et $y \leq t \ldots\ldots\ldots (1.18)$
- ✓ **élément neutre 0:** $\forall x \in [0, 1] : \perp (0, 0) = 0, T (x, 1) = \perp (1, x) = x \ldots\ldots\ldots (1.19)$

Union définie par une t-conorme : On vérifie simplement que l'opérateur maximum est une t-conorme. La définition d'une opération d'union entre deux sous-ensembles flous se réfère à une t-conorme, qui remplace l'opérateur maximum :

$$\forall x \in X, \ \mu_{A \cup B}(x) = \perp (\mu_A(x), \mu_B(x)) \ldots\ldots\ldots (1.20)$$

4.2.3. Caractéristiques D'un Sous Ensemble Flou :

On peut décrire les caractéristiques d'une fonction d'appartenance [4] qui représente un sous-ensemble A sur un univers de discours U par [4]:

- son support *:* $S(A) = \{x \in U | \mu_A(x) \neq 0\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.21)$
- sa hauteur *:* $H(A) = \sup\{\mu_A(x)\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.22)$
- son point de croisement : $C(A) = \{x | \mu_A(x) = 0.5\} \ldots\ldots\ldots (1.23)$
- son noyau *:* $N(A) = \{x \in U | \mu_A(x) = 1\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.24)$

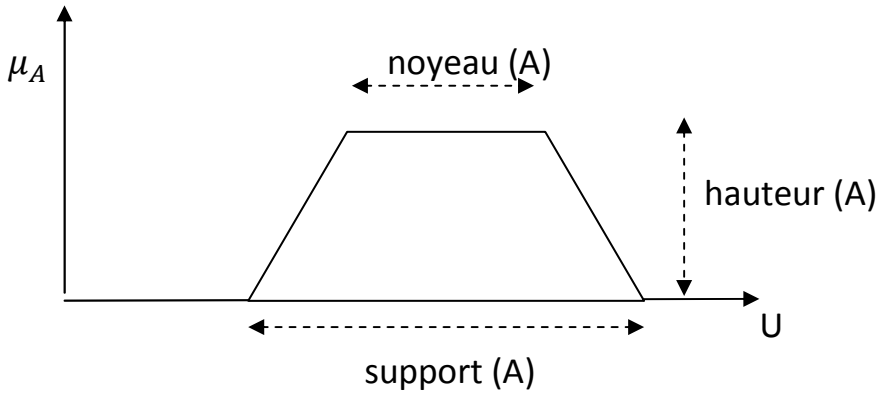


Figure 1.1 : représentation d'un sous-ensemble flou et ses principales caractéristiques.

4.3.Univers de discours :

Partitionnement des univers de discours :

Après le choix des entrées et des sorties du contrôleur flou que l'on cherche à synthétiser, il s'agit de déterminer les symboles qui seront utilisés pour l'écriture de la base de règles.

Leur exploitation ultérieure dans un monde numérique nécessite également de leur attribuer une signification floue. Cette étape correspond au partitionnement flou des différents univers numériques, chaque symbole étant représenté par un sous-ensemble flou défini sur l'univers de discours adéquat.

La Figure (1.2) illustre ce choix pour la variable d'erreur. La notation E_i représente le $i^{éme}$ symbole attaché à l'erreur.

En notant Δa l'écartement entre deux fonctions d'appartenance consécutives, la valeur modale du symbole E_i , notée ε_i s'exprime selon la relation :

$$\varepsilon_i = i\Delta a + \varepsilon_0 \dots\dots\dots\dots\dots\dots (1.25)$$

Un choix similaire pour les autres variables conduit à la définition des symboles dU_jet dU_k dont les valeurs modales respectives vérifient :

$$\delta\varepsilon_j = j\Delta b + \delta\varepsilon_0 \dots\dots\dots\dots (1.26)$$
$$\delta u_k = k\Delta c + \delta u_0 \dots\dots\dots\dots (1.27)$$

Avec Δb et Δc écarts respectifs entre deux fonctions d'appartenance consécutives relatives à la variation d'erreur et à la variation de commande. Par la suite, pour une simplification des calculs, la distribution des fonctions d'appartenance sera supposée symétrique par rapport à zéro, c'est-à-dire $\varepsilon_0 = \delta\varepsilon_0 = \delta u_0 = 0$. Il est à noter que la définition des symboles du_k ne sera exploitée que dans le cas des contrôleurs flous de Mamdani. [10]

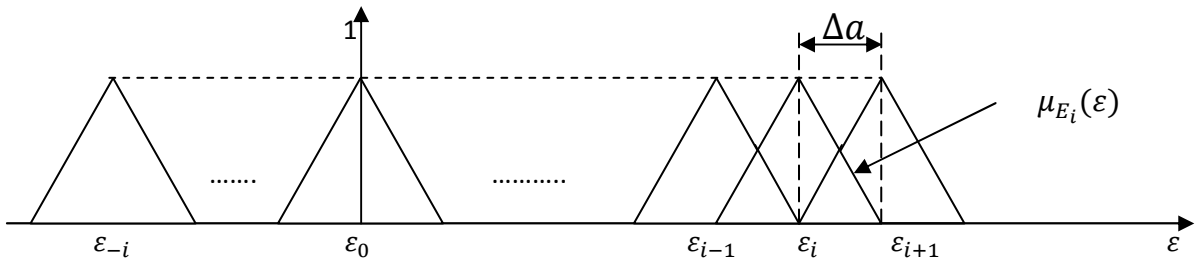


FIGURE 1.2: Partitionnement flou de l'univers de discours associé à l'erreur.

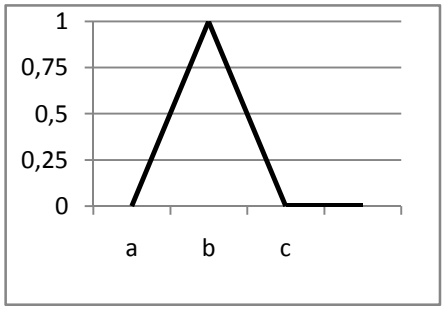
4.4. Fonctions d'appartenances :

En commande floue, les fonctions d'appartenance utilisées peuvent théoriquement être quelconques, pour peu qu'elles soient simples, convaincantes, rapides et efficaces, c'est pour cela qu'on choisit souvent des fonctions triangulaires ou trapézoïdales afin de simplifier les calculs. [9]

La boite à outils de la logique floue (fuzzy logic toolbox) met à notre disposition 11 types de fonctions d'appartenances, elles sont construites à partir de plusieurs fonctions basiques (fonctions linéaires, fonctions à distribution gaussienne, fonctions quadratiques..).
Les fonctions les plus utilisées sont les suivantes:

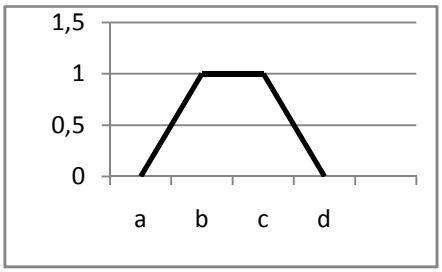
- fonction triangulaire:

$$\mu(x) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \ldots\ldots (1.28)$$



- fonction trapézoïdale:

$$\mu(x) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{c-x}{c-d}\right), 0\right) \ldots\ldots (1.29)$$



5. Raisonnement en logique floue :

Un des apports principaux de la logique standard a été la formalisation des méthodes de déduction, qui sont en quelque sorte un outil de raisonnement.

Les méthodes de déduction utilisées en logique standard permettent de définir une nouvelle certitude à partir d'autres connaissances certaines.

Dans le cadre de la logique floue, il est possible de généraliser les méthodes de raisonnement lorsqu'on dispose de connaissances incertaines ou imprécises.

Avec l'unique schéma de raisonnement (Si les conditions sont remplies, Alors la conclusion est validée) et avec les trois opérateurs Et, Ou et Non, nous pouvons déjà prendre un grand nombre de décisions logiques, comme une nouvelle information (une décision) peut être produite à partir d'informations anciennes. [4]

6. Formulation des lois utilisant « SI-ALORS »

Les ensembles flous et les opérateurs flous sont les sujets et les verbes de la logique floue. Ces "SI-ALORS" règles floues sont utilisées pour formuler les instructions conditionnelles qui constituent la logique floue.

Les lois sont déclarées comme suite: "$SI\ x\ est\ A, ALORS\ y\ est\ B$", où A et B sont respectivement des valeurs linguistiques définis par les ensembles floues dans les univers de discours X et Y.

La première partie ("x est A") est appelé antécédent (prémise), et donc ce qui suit ("y est B") n'est autre que la conséquence (conclusion).

On note que "A" est représenté par un nombre entre (0 et 1), ce qui fait que l'antécédent est une interprétation qui fait référence à une valeur comprise entre 0 et 1 également. D'autre part, "B" est représenté comme étant un ensemble floue, alors la conséquence est une affectation qui attribue l'ensemble floue B à la variable de sortie "y".

L'antécédent peut avoir plusieurs parties, dans ce cas, ces parties seront calculées simultanément et résolues à un seul nombre en utilisant les opérateurs logiques décrit précédemment. La conséquence d'une loi peut aussi être constituée de parties multiples, dans ce cas, toutes les conséquences seront affectées, de façons égales, par le résultat de l'antécédent.

La conséquence fait référence à l'ensemble floue qui sera affecté sur la variable de sortie la fonction d'implication fait en sorte de modifier l'ensemble floue au degré spécifié par l'antécédent. Les façons les plus communes pour modifier la sortie de l'ensemble floue sont *la troncature* utilisant la fonction « *min* » ou *l'écaillage* en utilisant la fonction *« prod »* qui sont toutes les deux prises en charge par la boite à outil de la logique floue. [25]

7. Variables linguistiques :

Pour qu'il soit possible de raisonner simplement sur un problème, il faut tout d'abord spécifier clairement les connaissances disponibles. Les variables linguistiques permettent de décrire dans un cadre très général la connaissance acquise sur une variable, même lorsqu'elle est vague ou imprécise. [4]

Par exemple, si la vitesse est interprétée comme une variable linguistique, alors son ensemble de termes de vitesse lente, moyenne, rapide ... où chaque terme est caractérisé par un ensemble flou. Ces termes peuvent être définis comme des ensembles flous dont les fonctions d'appartenance sont montrées sur la figure suivante *:*

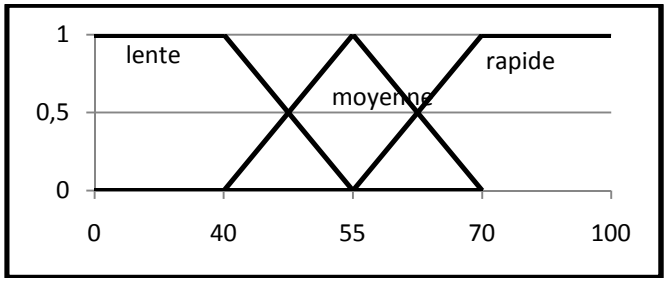


FIGURE 1.3: fonctions d'appartenances

8. Règles floues :

En fait il faut regarder un système de règles de contrôle flou comme une fonction (en général partiellement définie, dont on connait des points imprécis du graphe) de μ_A: {petit, moyen, grand}* {petit, moyen, grand} μ_B{petit, moyen, grand} dans le cas de neuf règles du type [si x est A_1 et y est A_2 alors u est B].

On définit ainsi partiellement une relation R sur des ensembles flous de dénomination maintenant traditionnelles NB (negative big), ... à PB (positive big) définies sur l'univers U*V de (x y).

L'ensemble flou inféré pour u lors de la règle [si (x, y) est A1 et A2 alors u est B] est défini par le modus ponens généralisé :

$$\mu_B(u) = \max(x,y) \in U * V \{\mu_I(T[\mu_{A_1}(x), \mu_{A_2}(y)], \mu_B(u))\} \dots\dots\dots\dots (1.31)$$

La plupart du temps, on aura des règles très simples portant sur E qui est la différence entre la sortie réelle et la consigne, et ΔE = variation de E, avec par exemple «si E est trop grand positif il faut le réduire, d'autant plus que «ΔE» est grand algébriquement. Les transversaux représentants à peu près des situations équivalentes. Par exemple, si E est PS (trop élevé) et ΔE est ZE (il ne varie pas) alors la conclusion NS indique qu'il faut le réduire légèrement.

Δe \ e	NB	NS	ZE	PS	PB
NB	PB	PB	PM	PS	ZE
NS	PB	PM	PS	ZE	NS
ZE	PM	PS	ZE	NS	NM
PS	PS	ZE	NS	NM	NB
PB	ZE	NS	NM	NB	NB

Tableau1.1 : table des règles floues

La conjonction (des prémises) est usuellement interprétée par l'opération 'min' (Zadeh), et la disjonction (des règles) comme le max, ainsi pour deux valeurs précises x et y et trois règles affectées de poids, on aura une troncature de chaque conclusion puis un sous-ensemble flou non obligatoirement normalisé pour la sortie U, et enfin une normalisation («défuzzification» ou «déflousification» par centre de gravité). [8]

La figure ci-dessous schématise en générale u en fonction des entrées x et y

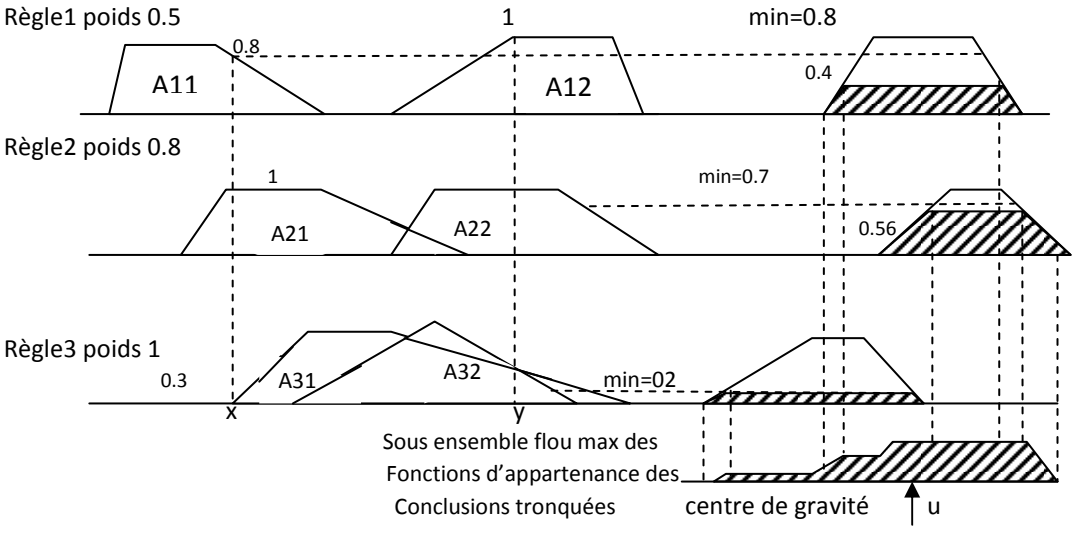


FIGURE 1.4 : système d'inférence floue de Mamdani.

REMARQUE :

S'il est général de choisir «min» pour la conjonction des prémises, il paraît plus naturel de pondérer la conclusion avec le poids de la règle par l'opération «produit», en effet, prenons le premier schéma ci-dessus, en prenant «min», on aurait une troncature identique à 0.5 pour des «nivaux de vérité» des prémisses aussi différents que 1, 0.8, 0.6, 0.5. Si les règles s'appliquent peu ou si leurs poids sont faibles (zone peu élevée) il y aura quand même une valeur u (centre de gravité) déduite.

9. Contrôleurs flous :

Les contrôleurs flous permettent de piloter des systèmes complexes ou difficilement modélisables en utilisant une méthode de raisonnement de type «si 'condition' alors 'action'». Récemment, des auteurs ont prouvé qu'il est possible de reproduire le fonctionnement de n'importe quel contrôleur continu standard – avec une précision arbitraire – grâce à un contrôleur flou. Pourtant, lorsque le système à commander est connu de façon imprécise, la conception du contrôleur flou n'est pas chose triviale.

Le contrôleur flou est constitué de 4 blocs principaux : la base de connaissance, le système d'inférence, l'interface de fuzzification et l'interface de défuzzification. La base de connaissance est composée d'une base des données et d'une base de règles. La base des données contient des faits de la forme: pour les variables linguistiques d'entrée et de sortie du contrôleur flou.

La base des règles caractérise la stratégie de commande émise par l'expert sous forme de règles linguistiques.

Le système d'inférence est capable de raisonner à partir des informations contenues dans la base de connaissance et de faire des déductions.

$$Si\ x_1 est\ A_1\ et\ x_2\ est\ A_2\ alors\ y\ est\ B \dots\dots\dots\dots (1.31)$$

Si B est une valeur linguistique, le contrôleur est dit de **type Mamdani**. Si B est une valeur numérique ou une équation mathématique, alors le contrôleur est dit de **type Takagi-Sugeno**.

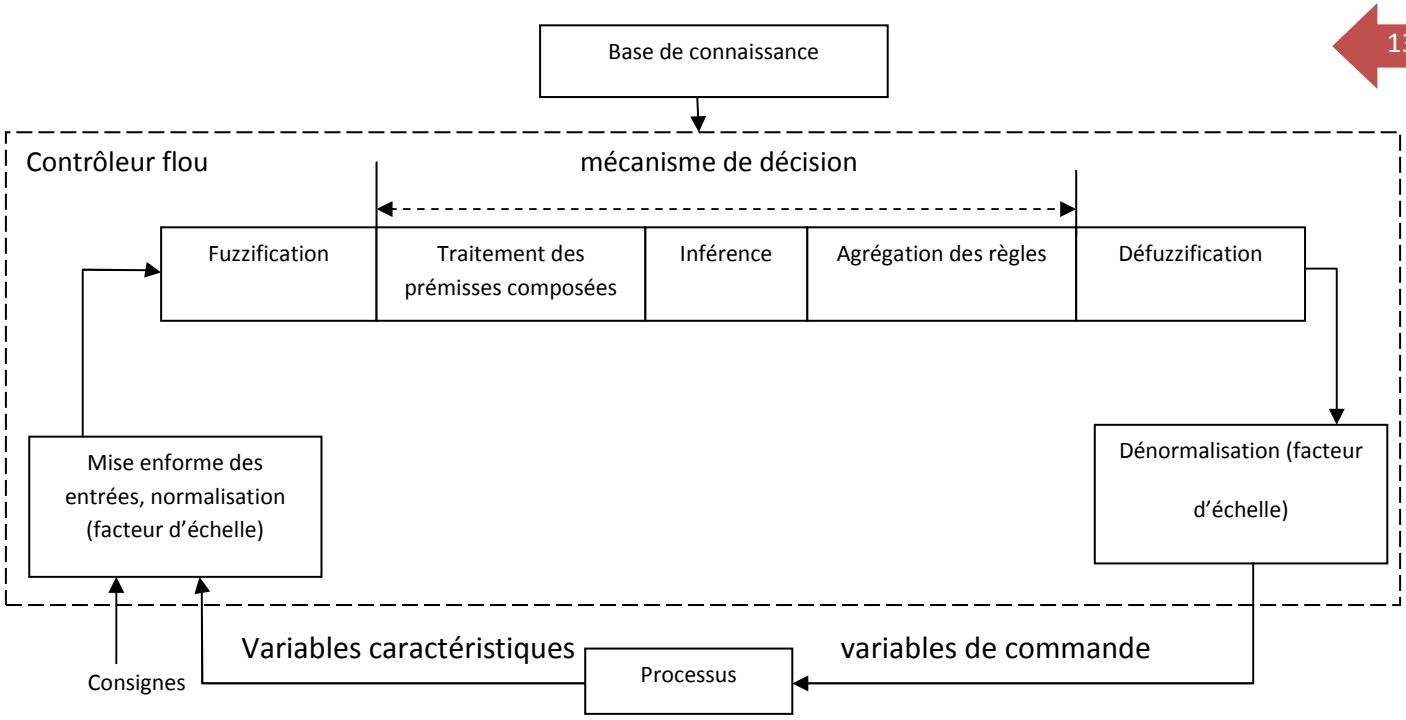


Schéma 1.1 : représentation d'un contrôleur flou.

Le fonctionnement d'un contrôleur flou dépend principalement des caractéristiques de ces trois sous-systèmes [1] :

- ✓ Fuzzification.
- ✓ Règles floues.
- ✓ Défuzzyfication

9.1. Mise en forme des entrées :

➢ **Normalisation :**

Cette première étape permet le traitement des variables d'entrée du contrôleur flou, par exemple, calcul de l'erreur et des variations d'erreurs.

L'utilisation de domaines normalisés (univers de discours compris entre -1 et 1) nécessite une transformation d'échelle transformant les grandeurs physiques des entrées en des valeurs normalisées appartenant à l'intervalle [-1,1].

➢ **Fuzzification :**

C'est l'opération de *projection* des variables physiques réelles sur des ensembles flous caractérisant les valeurs linguistiques prises par ces variables.

Le choix de la forme des fonctions d'appartenance (triangulaires, trapézoïdales, exponentielles, gaussiennes,…) est arbitraire. Les formes

triangulaires facilitent la programmation ce qui explique qu'elles soient le plus fréquemment utilisées. Le nombre de fonctions d'appartenance est généralement impair car elles se répartissent autour de zéro.

➢ **Traitement des prémisses composées :**

En général, les prémisses des règles vont comporter plusieurs clauses liées par des connecteurs « ET », « OU» et « NON». Dans la pratique, pour les opérations de conjonction et de disjonction, on a souvent recours, parmi les normes et conormes triangulaires, aux opérateurs *min* et *max*. Quant à la négation A^c d'un ensemble flou *A* est caractérisée par :

$$\mu_{A^c}(x) = 1 - \mu_A(x) \dots\dots\dots\dots\dots\dots\dots (1.32)$$

9.2. Inférence floue :

Elle repose sur l'utilisation d'un opérateur d'implication permettant d'évaluer le degré de vérité d'une règle *R* de la forme « **Si** *X1* est *A1* et *X2 est A2* **Alors** *Y* est *B* ».

En d'autres termes, cet opérateur quantifie la force de la liaison entre la prémisse et la conclusion de la règle [1].

Les opérateurs les plus répandus en commande sont de type conjonctif :

❖ **L'implication de Mamdani (1975) :**

Supposons que la base de connaissances est constituée de *n* règles d'inférence contenant chacune m prémisses et une conclusion. Le processus d'inférence peut être décrit par le schéma suivant :

Règle 1 : **Si** (x_1 est A_{11}) et $\cdots$ et (x_m est A_{1m}) ; **alors** (*Y* est B_1)
Règle 2 : Si (x_1 est A_{21}) et $\cdots$ et (x_m est A_{2m}) ; alors (*Y* est B_2)
. . .
. . .
. . .

Règle n : Si (x_1 est A_{n1}) et $\cdots$ et (x_m est A_{nm}) ; alors (*Y* est B_n)

Dans lequel x_1, …, x_m sont des éléments des univers du discours X_1, . . ., X_m et A_{ij}, *(j =1, … ,m)*, sont des quantités floues sur l'univers du discours *Xi*, et *Bj , (j = 1, … ,m)*, sont également des quantités floues sur l'univers du discours *Y* .

Afin de définir une seule prémisse pour une règle i, les propositions «x_j *est* A_{ij}», (*j = 1, … ,m*), sont combinées par l'opérateur minimum.
La fonction d'appartenance de cette prémisse unique est donc donnée par :

$$\mu_R(x,y) = \min(\mu_A(x), \mu_B(x)) \dots\dots\dots\dots\dots\dots (1.34)$$

❖ **Inférence floue de Sugeno :**

Sugeno a proposé une méthode d'inférence floue qui garantit la continuité de la sortie. Cette méthode d'inférence s'avère très efficace dans des applications faisant intervenir à la fois des techniques linéaires, d'optimisation et adaptatives [1].

Dans l'inférence de Sugeno, les règles floues sont exprimées de la façon suivante :

Règle i : Si (x_1 est A_{i1}) et $\cdots$ et (x_m est A_{im}) ; alors y = $f_i(x_1, \cdots, x_m)$

Dans laquelle $(x_1, \dots\dots, x_m)$ et y sont des éléments des univers du discours $(X_1, \dots\dots, X_{m)}$; et $(A_{i1}, \dots\dots, A_{im})$ sont des termes linguistiques sur ces mêmes univers du discours. y est une fonction de $(x_1, \dots, x_m)$. Par rapport à l'inférence de Sugeno, celle de Mamdani est plus intuitive, plus générale et elle s'adapte particulièrement bien à l'utilisation de connaissances issues d'une expertise humaine. [5]

➢ **Agrégation des règles :**

Selon le type de l'implication, classique ou conjonctive, l'opérateur utilisé pour agréger les règles est de type conjonctif ou disjonctif. Ainsi, en commande, l'implication étant généralement de type conjonctif, cela revient à considérer que les règles sont liées par un opérateur OU [1]. En pratique, on utilise l'operateur *max* :

$$\mu_{B'}(y) = \max_{i=1,\dots,N} \left(\mu_{R_i}(y)\right) \dots\dots\dots\dots (1.35)$$

Avec N : correspond au nombre de règles activées.

➢ **Défuzzification :**

La défuzzification consiste à transformer l'ensemble flou résultant de l'agrégation des règles en une grandeur de commande précise. Là aussi il existe plusieurs méthodes parmi lesquelles :

- La méthode de la hauteur.
- Le premier des maximas.
- Le dernier des maximas.

- La moyenne des maximas.
- Le centre de gravité.
- Le centre des aires.
- Le centre de la plus grande surface.
- Le centre des maximas.

Les méthodes de défuzzification les plus utilisées en commande floue sont le centre de gravité, le centre des aires et le centre des maximas. [1]

➢ **Dénormalisation :**

Cette dernière étape transforme les valeurs normalisées des variables de commande en des valeurs appartenant à leur domaine physique respectif.

10. Les différents types de la commande à base de la logique floue :

En se basant sur les différentes règles de commande floue et de leurs méthodes de génération [1], les approches de commande en utilisant la logique floue peuvent être classées dans les catégories suivantes :

➢ **La commande floue de type Mamdani :**

Dans un système flou de Mamdani (Conventionnel) les règles sont de type:

$$R(i,j,k): Si\ x\ est\ A_i\ et\ y\ est\ B_j\ Alors\ z\ est\ C_k$$

Où A_i, B_j, C_k représentent des termes linguistiques auxquels est attribuée une signification floue que l'on suppose normalisée.

Pour des entrées numériques précises x_0, y_0, un sous-ensemble flou de sortie, noté *F*, est généré selon la relation :

$$\mu_F(z) = max_{(i,j,k)\in I} \min(\mu_{A_i}(x_0), \mu_{B_j}(y_0), \mu_{C_k}(z)) \dots\dots\dots (1.36)$$

μ est appelée fonction d'appartenance.

L'opération $max_{(i,j,k)\in I}$ correspond à l'application de l'opérateur max sur l'ensemble des règles actives. La valeur numérique de sortie délivrée par le système flou est obtenue par défuzzification, classiquement implantée par la méthode du centre de gravité :

$$z = \frac{\int z\mu_F(z)dz}{\int \mu_F(z)dz} \dots\dots\dots (1.37)$$

Dans ce type de systèmes flous, la prémisse et la conclusion sont floues. Après l'inférence, le résultat est un ensemble flou caractérisé par sa fonction d'appartenance. Afin d'obtenir la valeur réelle de la commande à appliquer il faut passer par une étape de «défuzzification ».

Les règles de la commande floue « Si-Alors » sont obtenues à partir de l'expertise d'un opérateur. Quoique la performance d'une telle technique de commande soit généralement satisfaisante en pratique le non garanti de la stabilité du système de commande en boucle fermée est souvent critiquée.

Plusieurs approches ont été proposées pour étudier le problème de stabilité. L'idée principale de ces approches est de considérer le contrôleur flou comme un contrôleur non linéaire et d'utiliser la théorie des systèmes non linéaire pour l'analyse de la stabilité. [4]

- **La commande floue de type T-S :**

Les contrôleurs flous de Takagi–Sugeno sont, comme ceux de Mamdani, construits à partir d'une base de règles "**Si** ... **Alors** ...". Les prémises sont toujours exprimées linguistiquement et donc similaires à celles utilisées dans un contrôleur flou de Mamdani alors que les conclusions sont de nature numérique.

Un système flou de type Takagi-Sugeno (T-S) utilise des règles écrites de la manière suivante :

$$R_{(i,j,k)}: \; Si\ x\ est\ A_i\ et\ y\ est\ B_j\ Alors\ z = f_k(x) \dots\dots\dots (1.38)$$

Où f_k représente une fonction réelle quelconque.

Les fonctions de sortie f_k, peuvent être en principe des fonctions arbitraires des entrées, mais d'une manière générale elles sont choisies telles qu'elles soient une combinaison linéaire des entrées.

La particularité d'un modèle T-S est que la logique floue est seulement utilisée dans la partie prémise des règles. La partie conclusion est décrite par des valeurs numériques. Pour les valeurs d'entrée précises $x0$ et $y0$, la sortie z est évaluée selon le mécanisme suivant :

$$z = \frac{\sum_{(j,i,k)\in I} w_{i,j} f_k(x_0, y_0)}{\sum_{(i,j,k)\in I} w_{i,j}} \dots\dots\dots (1.38)$$

Avec : $w_{i,j} = \mu_{A_i}(x_0).\mu_{B_j}(y_0)$

Ce type de modèle est aussi très intéressant pour la représentation de systèmes non linéaires tels que les systèmes mécaniques ou chaotiques. [4]

11. Réglage d'un contrôleur flou :

Un contrôleur flou possède de nombreux paramètres de réglage, ce qui peut, à priori, effrayer ses utilisateurs potentiels. En effet, contrairement aux contrôleurs classiques, le contrôleur flou possède un nombre plus conséquent de paramètres, et offre, un nombre de degrés de liberté plus élevé.
On peut distinguer parmi les choix et les réglages à faire :

- L'expression des règles.
- La définition des variables et des valeurs linguistiques, avec leurs fonctions d'appartenance associées,
- La méthode d'implication.
- La méthode d'inférence.
- La méthode de défuzzification.
- Les facteurs d'échelle sur les entrées et les sorties du contrôleur.

Cependant, on constate une certaine insensibilité du résultat au choix des méthodes d'implication, d'inférence et de défuzzification. Pour les autres paramètres, un Réglage séquentiel est possible :

- ✓ Fonctions d'appartenance.
- ✓ Règles.

Le réglage par essais successifs de ces nombreux paramètres étant assez long et fastidieux, diverses techniques d'autoréglage, d'optimisation et d'apprentissage ont été développées ces dernières années. On peut citer, à titre d'exemple, les réseaux de neurones et les algorithmes génétiques. [1]

12. Propriétés d'un réglage par logique floue :

Les propriétés essentielles d'un réglage par logique floue peuvent être résumées de la manière suivante. Aptitude à régler convenablement les systèmes avec un comportement dynamique compliqué, dont la modélisation est difficile, voire impossible. A noter que dans le cas du réglage par logique floue, il n'est pas nécessaire d'établir un modèle. Si, pour un certain système à régler, il existe tout de même un modèle mathématique convenable, on peut l'utiliser pour tester et modifier la stratégie de réglage à l'aide d'une simulation numérique. Cela facilite évidemment la mise en service sur l'installation réelle.

La disponibilité de systèmes de développement efficaces, soit pour microprocesseurs ou PC (solution logicielle), soit pour circuits intégrés (processeurs, fuzzy processors, solution matérielle). Malgré certains inconvénients tels que :

- Le manque de directives précises pour la conception d'un réglage (choix des grandeurs à mesurer, détermination de la fuzzification, des inférences et de la défuzzification).
- La précision de réglage souvent peut élevée etc...

On peut affirmer que le réglage par logique floue présente une alternative valable aux réglages conventionnels. Cela est confirmé non seulement par un fort développement dans beaucoup de domaines d'application, mais aussi par des travaux de recherche sur le plan théorique. [1]

13. Avantages du contrôleur flou :

Les variables linguistiques sont bien adaptées en traduisant le raisonnement qualitatif humain, comme lui, on décrit des situations locales avec des règles représentant chacune une information partielle. Un opérateur expérimenté prend en effet ses décisions seulement sur des situations spécifiques dont il n'a qu'une connaissance imparfaite, mais qu'il agrège le long de son expérience. Le fait d'utiliser des prédicats flous, loin de traduire une pauvreté de l'information, permet d'exprimer des situations graduelles.
Le contrôle flou s'est surtout montré robuste, son comportement vis à vis des situations pathologiques, sa flexibilité (adaptation facile à des domaines, et ils sont nombreux, dont on ne possède pas de modèle mathématique), et bien sûr il est rapide en temps réel.

14. Conclusion

Nous avons vu dans ce chapitre l'intérêt de la logique floue dans le domaine du contrôle de processus. Cette approche permet de tenir compte à la fois des connaissances d'un expert humain et de l'incertitude et de l'imprécision des données traitées par le contrôleur. Les variables linguistiques permettent de traiter ces deux informations initialement très différentes à l'aide d'un formalisme unique.
Pourtant, la conception d'un contrôleur flou n'est pas toujours chose aisée. Lorsqu'on utilise un contrôleur de type standard (par exemple un PID), on dispose de nombreux outils de synthèse permettant de choisir au mieux les

paramètres du régulateur en fonction de la structure ou du modèle du système à commander. Malheureusement, la panoplie d'outils disponibles est beaucoup plus limitée dans le cas des contrôleurs flous.

En fait, le fonctionnement d'un contrôleur flou dépend d'un nombre très important de paramètres (fonctions d'appartenance, règles floues, règles d'inférence, défuzzification) qu'il faut régler lors de la conception. Comme ces paramètres s'influencent mutuellement, il est peu probable qu'une méthode de synthèse traitant indépendamment chaque sous système du contrôleur flou puisse fournir un résultat «optimal».

Dans certains cas, une approche globale de la conception est toutefois possible. Elle est basée sur une méthode d'apprentissage, dans laquelle un système extérieur au contrôleur analyse les performances de ce dernier lorsqu'il utilise un ensemble donné de paramètres. Par essais successifs, le système extérieur peut sélectionner le jeu de paramètres qui assurera le «meilleur» fonctionnement du contrôleur.

1. Introduction :

Les organismes que nous connaissons aujourd'hui sont rendus utile pour la résolution des problèmes d'optimisation dans des divers domaines de la science et en particulier dans celui de l'ingénierie grâce à l'orientation de la recherche vers des techniques d'optimisation basés sur les analogies avec des processus naturels, biologiques ou humain. Parmi ces techniques, on trouve les algorithmes génétiques (AG) qui sont des procédures stochastiques (pseudo aléatoires) inspirées des lois de l'évolution des espèces et de la génétique naturelle. [9]

Les Algorithmes Génétiques sont des méthodes d'exploration de l'ensemble des solutions d'un problème utilisant les mêmes mécanismes que ceux intervenant dans la sélection naturelle. Ils sont utilisés principalement dans les domaines de l'optimisation et de l'apprentissage. Le parallélisme implicite et l'exploration globale de l'espace des solutions sont les deux principaux avantages des algorithmes génétiques.

Tout d'abord, les algorithmes génétiques peuvent traiter des problèmes pour lesquels les solutions potentielles sont situées dans un espace de grande dimension, ce qui ne permet pas d'utiliser des méthodes standard reposant sur une exploration systématique. Ensuite, la recherche d'une solution optimale à un problème peut être réalisée par un algorithme génétique sans nécessiter de connaissance à priori sur la répartition des solutions dans l'espace.

L'AG est une technique d'optimisation basée sur les concepts de la sélection naturelle et génétique. L'algorithme commence avec un ensemble de solutions possibles du problème (individus), constituant une population. L'évolution est guidée par l'utilisation d'opérateurs génétiques.

Bien que les mécanismes exploités dans les algorithmes génétiques aient été découverts par Darwin et Mendel, le livre de John Holland publié en 1975 est considéré comme étant à l'origine des algorithmes génétiques. En 1989, Goldberg a publié un livre sur les AG qui est devenu une référence.

De nos jours, les AG ont trouvé des applications dans des domaines très variés, allant de la biologie jusqu'aux applications liées plus directement à l'informatique (génie des procédés, traitement d'images, optimisation, placement de formes). [4]

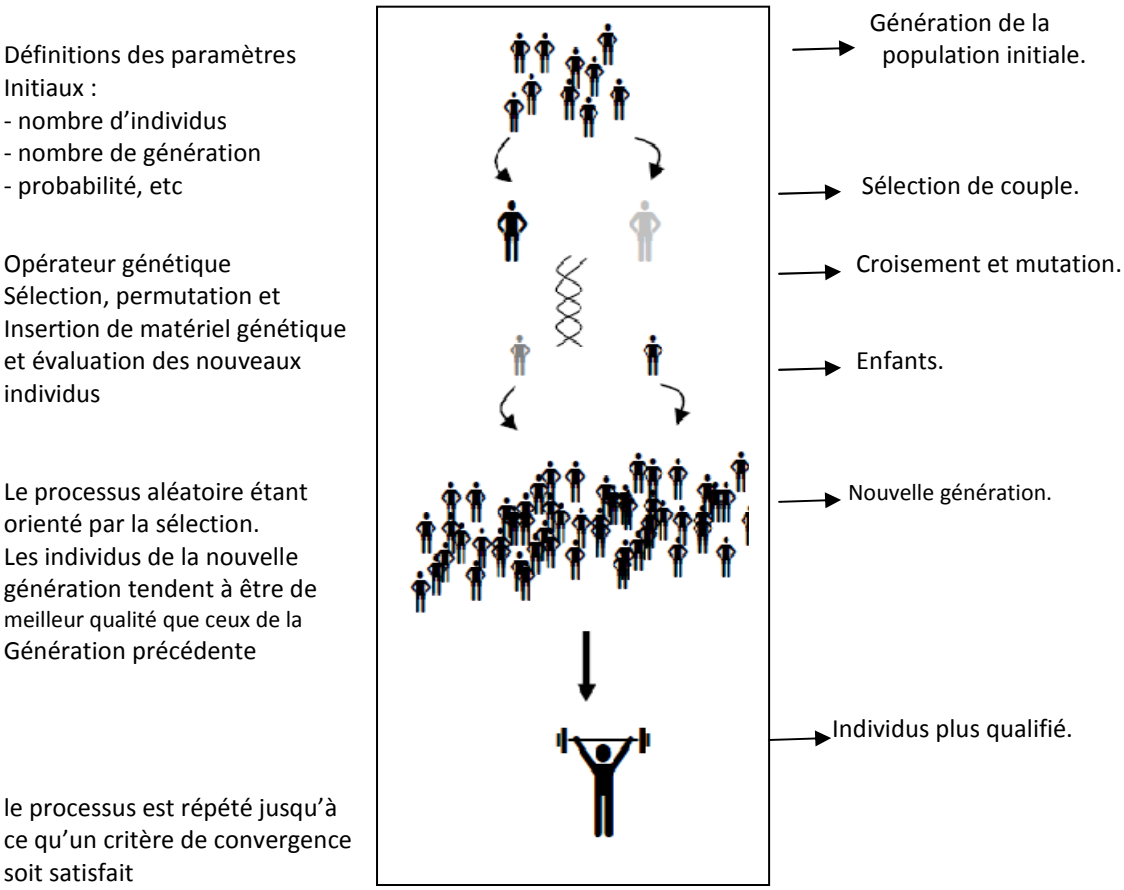


Figure 2.1: Concepts de base d'un Algorithme Génétique.

2. Terminologie :

Avant de décrire le principe des algorithmes génétiques, il est nécessaire de présenter le vocabulaire que nous allons utiliser tout au long de ce travail.

- **Chromosome →chaîne, chromosome** : Dans les systèmes naturels, les chromosomes sont les porteurs de l'information génétique nécessaire à la construction et au fonctionnement d'un organisme. Dans les algorithmes génétiques, les chaînes, ou chromosomes sont analogues aux chromosomes des systèmes biologiques. Ils sont les éléments à partir desquels sont élaborées les solutions.

- **Génotype → structure** : Dans les systèmes naturels, l'ensemble du matériel génétique est appelé le génotype. Dans les algorithmes génétiques, l'ensemble des chaînes est appelé structure. Dans ce cas, l'espace décisionnel peut s'appeler espace génotypique.

- **Phénotype → ensemble de paramètres, solution, point** : Dans les systèmes naturels, l'organisme formé par l'interaction de l'ensemble du matériel génétique avec son environnement est appelé le phénotype. Dans les algorithmes génétiques, les structures décodées forment un ensemble de paramètres donné, ou une solution ou un point dans l'espace des solutions. L'espace objectif peut également être appelé espace phénotypique.

- **Gène → trait, détecteur** : Dans les systèmes naturels, les chromosomes sont constitués par les gènes. Dans les algorithmes génétiques, on dit que les chaînes se composent de traits ou détecteurs.

- **Allèle → valeur de caractéristique** : Dans les systèmes naturels, l'allèle est une, composante du gène. Les allèles sont les différentes valeurs que peuvent prendre les gènes. Dans les algorithmes génétiques, l'allèle est également appelé valeur caractéristique.

- **Locus → position dans la chaîne** : Le locus est la position du gène dans le chromosome. Ce terme est appelé également position dans la chaîne dans les algorithmes génétiques.
- **Individu, organisme → individu, chromosome** : Un organisme ou un individu biologique est une forme qui est le produit de l'activité des gènes. Dans le cadre d'un AG traditionnel, l'individu est réduit à un chromosome, on l'appelle indifféremment individu ou chromosome.
- **Population → population, génération** : Dans les systèmes naturels, la population est un groupe d'individus. Dans les algorithmes génétiques, la population est l'ensemble des individus ou des chromosomes. Les populations sont également appelées des générations.

3. Fonctionnement général des algorithmes génétiques :

3.1. Principe de fonctionnement des algorithmes génétiques :

Les algorithmes génétiques fournissent des solutions aux problèmes n'ayant pas de solutions calculables en temps raisonnable de façon analytique ou algorithmique.

Le schéma général ci-dessous décrit le principe de fonctionnement d'un algorithme génétique [11] :

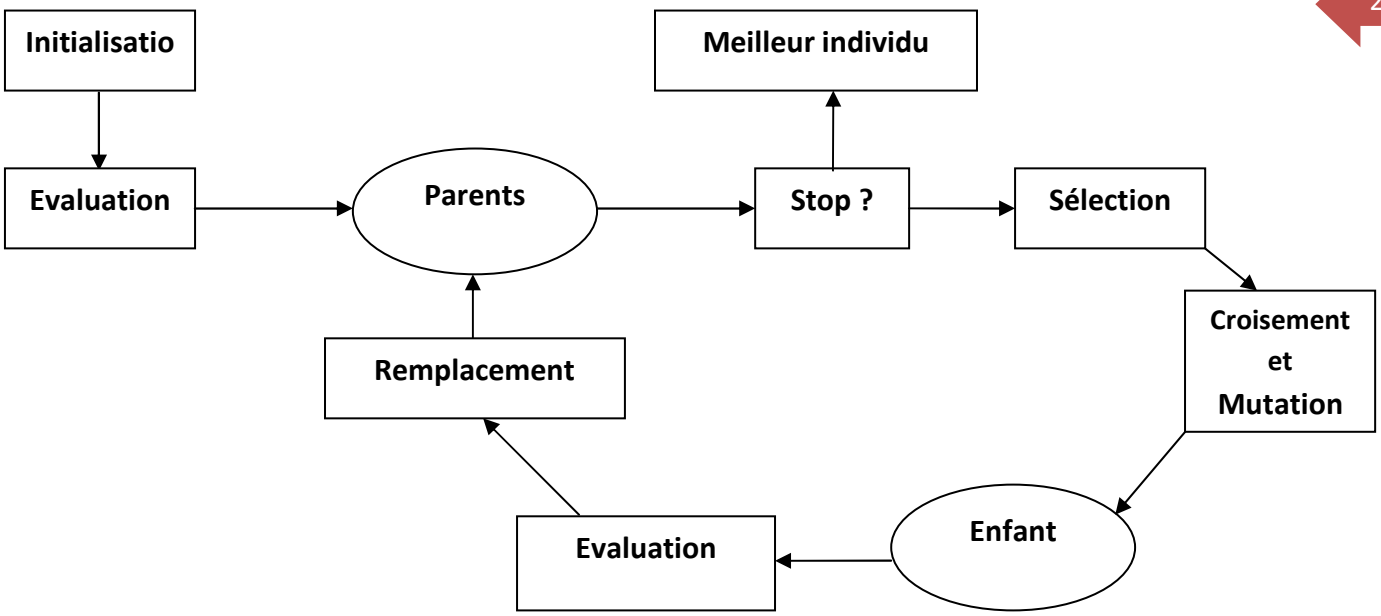


Figure 2.2 : Squelette d'un algorithme

Un algorithme génétique est un algorithme itératif de recherche d'optimum, il manipule une population de taille constante. Cette population est formée de points appelés chromosomes, chaque chromosome représente le codage d'une solution potentielle au problème à résoudre et il est constitué d'un ensemble d'éléments appelés gènes. A chaque itération, appelée génération, est créé une nouvelle population avec le même nombre de chromosomes. Cette génération consiste en des chromosomes mieux adaptés à leur environnement tel qu'il est représenté par la fonction sélective. Au fur et à mesure des générations, les chromosomes vont tendre vers l'optimum de la fonction sélective.

La création d'une nouvelle population à partir de la précédente se fait par l'application des opérateurs génétiques qui sont : *la sélection, le croisement et la mutation*. La sélection des meilleurs chromosomes, est la première opération dans un algorithme génétique. Au cours de cette opération, l'algorithme sélectionne les éléments pertinents qui optimisent mieux la fonction.

Le croisement permet de générer deux chromosomes nouveaux « enfant » à partir de deux chromosomes « parents », tandis que la mutation réalise l'inversion d'un ou plusieurs gènes d'un chromosome.

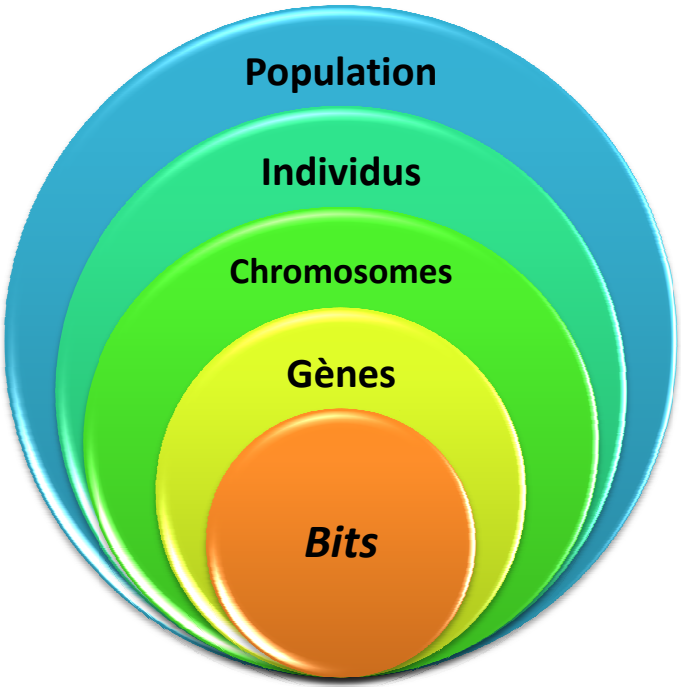


Figure 2.3: principaux éléments d'une population.

4. Codage :

Lors de la génération d'une population initiale, tous les individus doivent être codés selon une représentation spécifique. Le choix du codage à adopter est en fonction des caractéristiques du présent problème. Il y a trois principaux types de codage, et on peut passer de l'un à l'autre facilement. [7]

4.1. Codage binaire :

HOLLAND a proposé de représenter les chromosomes comme étant des chaînes de bit, appelées chaînes binaires. Il a montré que cette méthode de codage est en principe la plus efficace, car elle fait intervenir un alphabet comportant uniquement deux symboles. Cependant, le codage binaire présente un inconvénient majeur. Lorsque le nombre de paramètres à coder est important et que la précision souhaitée sur la solution est élevée, la chaîne binaire devient très longue.

4.2. Codage en nombres réels :

En termes d'occupation mémoire, ce codage est moins intéressant puisqu'il nécessite autant de bits en mémoire pour représenter les variables. Par rapport au codage binaire, la représentation en nombre réels est mieux adaptée aux problèmes imposant une précision. Dans un algorithme génétique utilisant un codage en nombre réels, chaque chromosome est en fait un vecteur dont les cordonnées sont les variables du processus d'optimisation. [7]

- ✓ Ce type de codage est rarement utilisé pour l'optimisation des règles floues de **Mamdani** à cause de la nature des conclusions qu'elles présentent.

4.3. Codage en base N :

Certains auteurs ont proposé une autre méthode de codage appelée codage en base n, et ont décrit les opérateurs génétiques associés. Dans cette méthode, les éléments constituant un chromosome sont des chiffres exprimés dans une base de numérotation n. Ce qui permet de représenter n valeurs discrètes. Les opérateurs de croisement et de mutation adaptés au codage en base n sont de simples extensions des opérateurs standard. Pour le croisement, les chromosomes sont scindés puis recombinés entre deux éléments codés en base n. Pour la mutation, l'élément initial est remplacé par un chiffre en base n tiré au sort. [7]

- ✓ Le codage en base N, c'est le codage le plus utilisé pour la représentation des règles floues de **Mamdani** .

5. Opérateurs génétiques :

Les opérateurs génétiques jouent un rôle prépondérant dans la réussite d'un algorithme génétique. Nous en dénombrons trois principaux : l'opérateur de sélection, de croisement et de mutation. Si le principe de chacun de ces opérateurs est facilement compréhensible, il est toutefois difficile d'expliquer l'importance isolée de chacun de ces opérateurs dans la réussite de l'algorithme génétique. Cela tient en partie, au fait que chacun de ces opérateurs agit selon divers critères qui lui sont propres (valeur sélective des individus, probabilité d'activation de l'opérateur, etc.…). [5]

5.1 Opérateur de sélection :

Cet opérateur peut- être le plus important, puisque il permet aux individus d'une population de survivre, de se reproduire ou de mourir. En règle générale, la probabilité de survie d'un individu sera directement reliée à son efficacité relative au sein de la population. [4]
La probabilité de sélection d'un individu C_i est :

$$\text{Pselection } (C_i) = \text{Fevaluation } (C_i) / \sum_{j=1}^{n} Fevaluation(C_i)$$

- ✓ n : la taille de la population de l'algorithme génétique.
- ✓ Pselection (C_i) : probabilité de sélection de l'individu i.
- ✓ Fevaluation (C_i) : la valeur d'évaluation de l'individu i.

Il existe plusieurs méthodes de sélection, la plus connue est la roue de loterie biaisée (roulette wheel) de Goldberg (1989). Selon cette méthode, chaque chromosome sera dupliqué dans une nouvelle population proportionnellement à sa valeur d'adaptation. On effectue, en quelque sorte, autant de tirages avec remises qu'il y a d'éléments dans la population.

L'inconvénient majeur de cette méthode, repose sur le fait qu'un individu n'étant pas le meilleur peut tout de même dominer la sélection. Elle peut aussi engendrer une perte de diversité par la domination d'un super individu.
Il existe d'autres méthodes de sélection, les plus connues sont :

- ➢ **Sélection par élitisme :**

Elle consiste à copier un ou plusieurs des meilleurs chromosomes dans la nouvelle génération pour ne pas les perdre. Ensuite en génère le reste de la population selon le mécanisme de reproduction usuel. Cette méthode améliore considérablement les algorithmes génétiques car elle permet de ne pas perdre les meilleures solutions.

- ➢ **Sélection par tournoi** :

Cette technique consiste à choisir aléatoirement deux individus et à comparer leur fonction d'adaptation. On accepte la meilleure solution pour accéder à la génération intermédiaire. On répète cette opération jusqu'à remplir la génération intermédiaire (N/2 composants). Les individus qui

gagnent à chaque fois peuvent être copiés plusieurs fois. Cela favorise la pérennité de leurs gènes. (N : La taille de la population).

5.2. Opérateur de croisement :

L'opérateur de croisement utilisé par les algorithmes génétiques est la transposition informatique du mécanisme qui permet, dans la nature, la production de chromosomes qui héritent partiellement des caractéristiques des parents.

Tout d'abord, deux individus, qui forment alors un couple, sont tirés au sein de la nouvelle population issue de la reproduction. Puis un site de croisement est choisi aléatoirement. Enfin, selon une probabilité Pc que le croisement s'effectue, les segments finaux (dans le cas d'un seul site de croisement) des deux parents sont alors échangés autour de ce site. Cet opérateur permet la création de deux nouveaux individus.

Un individu sélectionné lors de la reproduction ne subit pas nécessairement l'action d'un croisement. Le croisement ne s'effectue qu'avec une certaine probabilité. Plus cette probabilité est élevée, plus la population subira de changement [5].

Il existe différents types de croisement :

- **Croisement selon un point** : détermine aléatoirement un point de coupure et échange la deuxième partie des deux parents

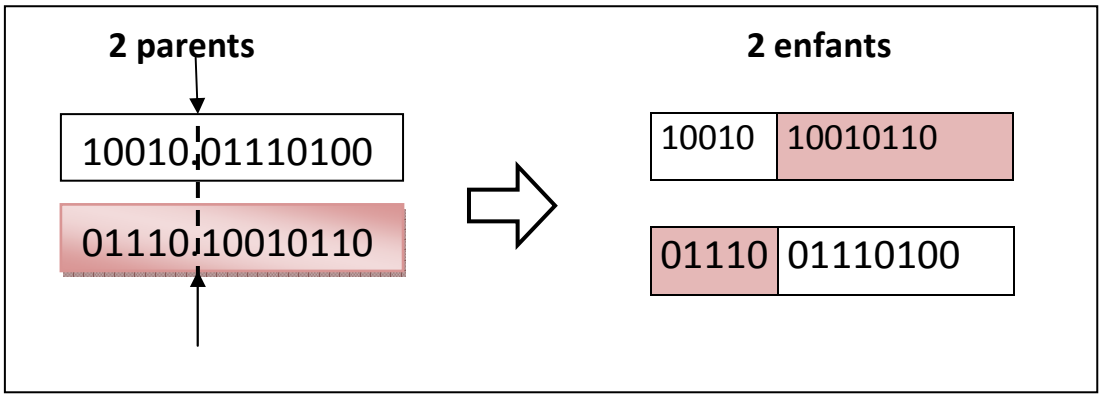


Figure 2.4: croisement selon un point.

- **Croisement selon deux points** (peut être étendu à x points) : possède deux points de coupure qui sont déterminés aléatoirement.

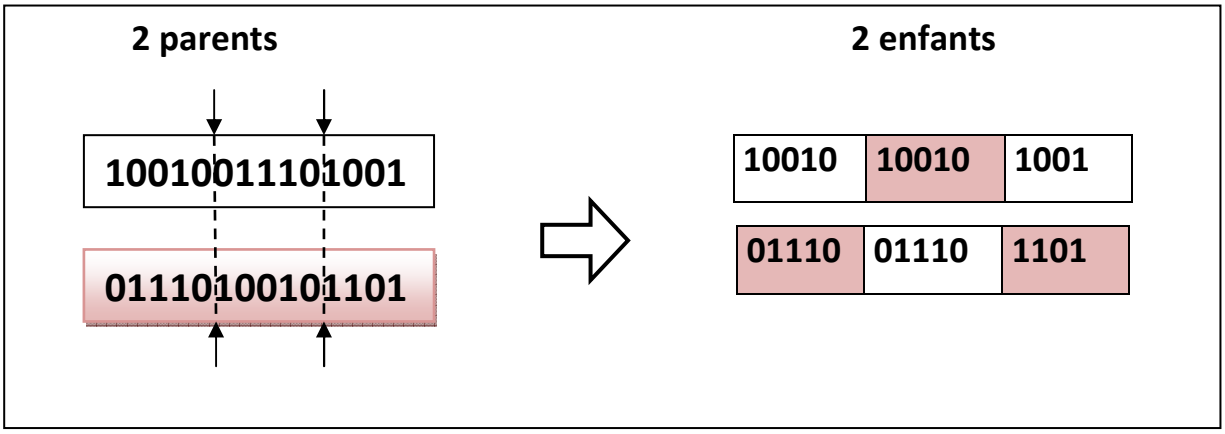


Figure 2.5: croisement selon deux points.

- **Croisement uniforme** : échange chaque bit avec une probabilité fixé à 0,5.

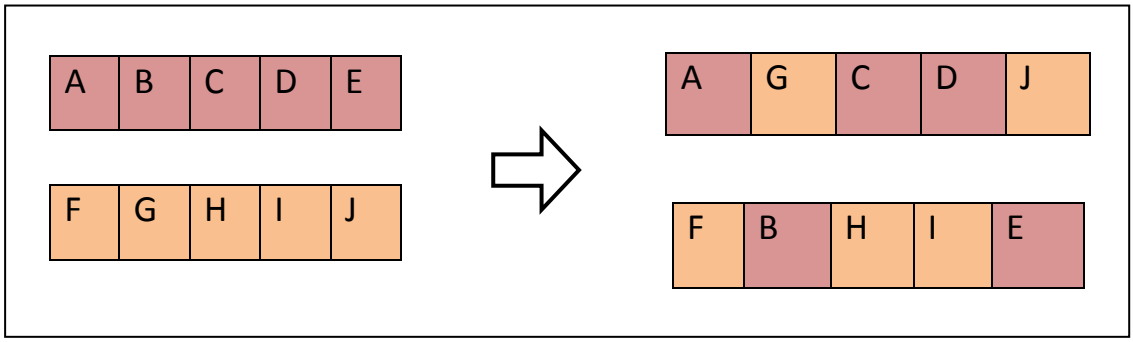


Figure 2.6: croisement uniforme.

5.3. Opérateur de mutation :

En biologie, une mutation est une modification d'un chromosome qui n'est pas issu d'une opération de croisement. Il est assez difficile de cerner l'effet d'une mutation dans le cas des organismes vivants. D'après les expérimentations réalisées, la mutation évite qu'une population soit stagnée lorsque tous ses individus soient génétiquement identiques .En D'autres

termes, l'opération de mutation permet de faire apparaître de nouveaux gènes.[5]

La mutation originale est celle appliquée à des chromosomes codés en binaire. Chaque bit est remplacé selon la probabilité pm par son inverse. On génère un nombre aléatoire entre (0 et 1), si ce nombre est supérieur ou égale à pm alors:

Si la valeur du bit est égale à 0 alors elle sera remplacée par 1 et si elle est égale à 1 alors elle sera remplacée par 0.

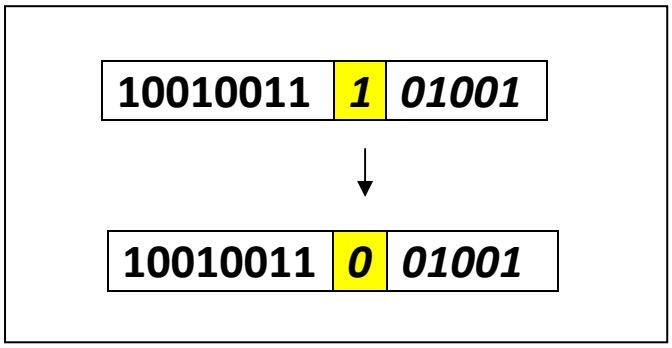


Figure 2.7: une mutation.

6. Sélection finale :

Cette dernière étape du processus itératif consiste en l'incorporation des nouvelles solutions dans la population courante. Les nouvelles solutions sont ajoutées à la population courante en remplaçant totalement ou partiellement les anciennes solutions. Généralement, les meilleures solutions remplacent les plus mauvaises. Il en résulte une amélioration de la population. [5]

7. Recherche des paramètres optimaux :

Les paramètres qui influent sur l'AG :

Les opérateurs génétiques sont guidés par un certain nombre de paramètres fixés à l'avance. La valeur de ces paramètres influe sur la réussite ou non d'un algorithme génétique. Ces paramètres sont les suivants :

- ✓ **La taille de la population N** :

Si N est trop grand, le temps de calcule de l'algorithme peut s'avérer très important, et s'il est trop petit, il peut converger trop rapidement vers un mauvais chromosome. Cette importance de la taille est essentiellement dû à la

notion de parallélisme implicite qui implique que le nombre d'individus traité par l'algorithme est au moins proportionnelle au cube du nombre d'individus.

- ✓ **La probabilité de croisement Pc** :

Elle dépend de la forme de la fonction sélective, son choix est généralement heuristique. Plus elle est élevée, plus la population subit de changements importants. [5]

- ✓ **La probabilité de mutation Pm :**

Ce taux est généralement faible, puisqu'un taux élevé risque de conduire à une solution sous- optimale. Plutôt que réduire Pm, une autre façon d'éviter que les meilleurs individus soient altérés, est d'utiliser la reconduite explicite de l'élite dans une certaine proportion. Ainsi, bien souvent, les meilleurs 5% par exemple, de la population sont directement reproduits à l'identique, l'opérateur de reproduction ne jouant alors que sur les 95% restant. Cela est appelé stratégie élitiste.

La notion « d'autorégulation » avec l'idée de «contrôle auto-intégré » des paramètres de l'AG a été introduite et utilisée par plusieurs chercheurs, afin de trouver des méthodes systématiques pour concevoir un AG avec des paramètres optimaux [4]. On peut résumer le concept d'adaptation des paramètres de l'AG par un autre AG par le schéma suivant :

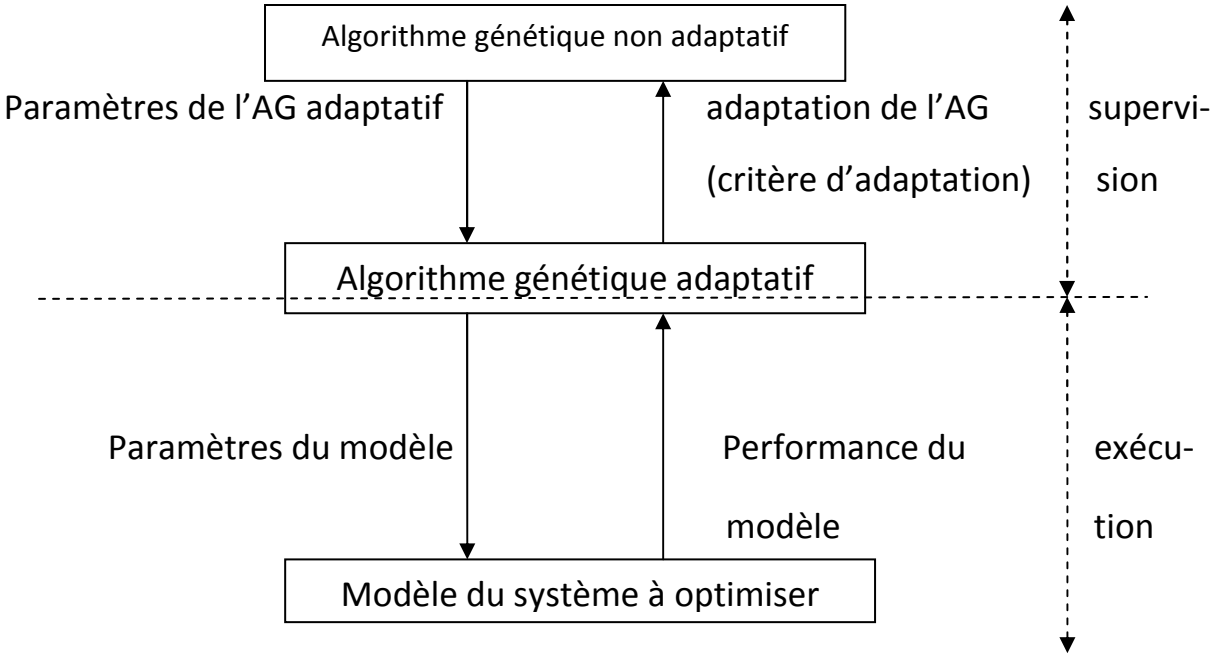


Schéma 2.2: le concept d'adaptation des paramètres de l'algorithme genétique.

➢ Les principaux paramètres d'un AG sont :

- ✓ La probabilité de croisement **pc** .
- ✓ La probabilité de mutation **pm**.
- ✓ La taille de la population **N**.

La taille de la population joue un rôle très important dans l'emploi des AG comme procédures d'optimisation. Si la taille est petite, cela implique une exécution rapide et une convergence incertaine. Si la taille est importante, le problème du temps de calcul apparaît.

La sélection des meilleurs individus pour la reproduction et la diversité dans la population sont deux facteurs nécessaires pour le déroulement de l'AG dans un sens positif. Ces deux derniers sont liés directement à la **taille de la population** et aux valeurs de **pm** et **pc**.

8. Critère d'arrêt :

Les étapes d'évaluation reproduction/sélection sont appliquées d'une manière itérative sur la population, jusqu'à satisfaction d'une condition Fin ; cette condition est soit un nombre maximum de générations, soit une valeur de fitness minimale, soit une convergence vers la meilleure solution ou fixé un temps d'arrêt [12].

9. Etapes de construction d'un Algorithmes génétiques.

Début
1. Générer une population aléatoire initialement de n chromosomes.
2. Evaluer la fitness des chromosomes avec la fonction f(x).
3. appliquer l'opération de sélection.
4. Appliquer l'opération de croisement avec une probabilité pc.
5. Appliquer l'opération de mutation avec une probabilité pm.
6. Ajouter les nouveaux chromosomes à la nouvelle population.
7. Calculer la fonction fitness f(x), pour tout chromosome x.
8. appliquer l'opération de remplacement.
9. Jusqu'à la satisfaction des conditions de terminaison.
FIN

10. Avantages et Inconvénients :

➢ Avantage :

• élimination de solutions non valides.
• Permet de traiter des espaces de recherche important (beaucoup de solutions, pas de parcourt exhaustif envisagé).
• Nombre de solutions important.
• Relativité de la qualité de la solution selon le degré de précision demandé.

➢ Inconvénients :

• Nécessitent plus de calculs que les autres algorithmes méta heuristiques (notamment la fonction évaluation).
• Paramètres difficiles à fixer (taille population, % mutation).
• Choix de la fonction d'évaluation délicat.
• Pas assuré que la solution trouvée est la meilleure, mais juste une approximation de la solution optimale.
• Problèmes des optimums locaux si paramètres mal évalués.[22]

➢ Exemple :

Cet exemple est dû à Goldberg (1989). Il consiste à trouver le maximum de la fonction f(x) = x sur l'intervalle [0, 31] où *x* est un entier naturel. [22]

On a 32 valeurs possibles pour *x* on choisit donc un codage discret sur 5 bits : on obtient donc la séquence 01101 pour 13, la séquence 11001 pour 25, etc... On initialise la population initiale de manière aléatoire et on fixe sa taille à

4 individus. On définit simplement la fitness comme étant la valeur de *x*, vu qu'on en cherche la valeur maximum sur l'intervalle [0, 31] plus la valeur de *x* sera élevée plus on se rapprochera du maximum de la fonction identité et donc plus la fitness sera grande. Soit la population initiale suivante : [22]

Individus	Séquence	Fitness	% du total
1	01011	11	18.6
2	10011	19	32.2
3	00101	5	8.5
4	11000	24	40.7
Total		59	100

Tableau 2.1 : la population initiale

On opte pour une sélection par la méthode la loterie biaisée :

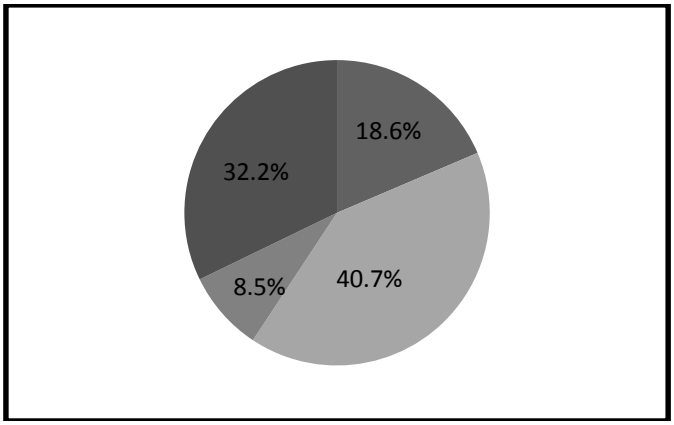


Schéma 2.3: application de la méthode de sélection de la loterie biaisée sur la population.

On fait tourner la roue 4 fois de suite, en général on fait tourner *n / 2* fois, soit 2 fois dans ce cas, mais le nombre 2 étant trop petit on décide de la faire tourner 4 fois. On obtient la nouvelle population :

Individus	Séquence
1	11000
2	00101
3	11000
4	01011

Tableau 2.2: individus sélectionnés par la méthode de la loterie biaisée.

On applique l'opérateur de croisement en utilisant un seul point de croisement. Normalement chaque couple donne 2 enfants qu'on ajoute à notre nouvelle population P' la faisant passer de *n / 2* individus à n individu, mais vu que dans le cas de notre exemple nous avons déjà atteint nos *n* individus, les 2 enfants du couple remplaceront leurs parents. [22]

Deux couples sont formés de manière aléatoire :

- couple 1 : l'individu 2 avec l'individu 3
- couple 2 : l'individu 1 avec l'individu 4.

Les point de croisement sont eux aussi tirer au hasard .On obtient le résultat suivant :

Parents	Enfants
00 | 101 01 | 011	01011 00101
11 | 000 01 | 011	01011 11000

Tableau 2.3: résultat de l'application de l'opérateur de croisement avec un point de croisement sur les individus sélectionnés par la loterie biaisée.

On applique l'opérateur de mutation qui choisit de manière aléatoire si on doit faire une mutation et sur quel locus la faire :

Individu	Séquence	Fitness	% du total
1	11011	27	38
2	00101	5	7
3	01111	15	21.1
4	11000	24	33.8
Total		71	100

Tableau 2.4 : la nouvelle population après application des différents opérateurs.

En une seule génération le maximum est passé de 24 à 27, et la fitness globale de la population a relativement augmentée pour passer de 59 à 71. On

s'arrête ici pour cet exemple mais dans la réalité on continuera à engendrer des générations successives jusqu'à obtenir le maximum global.

11. Conclusion :

Dans ce chapitre, nous avons décrit brièvement le principe des algorithmes génétiques, qui permettent d'aborder la majorité des problèmes d'optimisation en utilisant un formalisme simple. Dans le chapitre suivant, nous verrons comment utiliser les algorithmes génétiques pour optimiser la structure et les paramètres de normalisation (facteurs d'échelle) d'un contrôleur flou.

Nous avons vu également que les paramètres régissant l'évolution de la population (nombre d'individus, probabilités de croisement et de mutation) ont une influence importante sur la convergence d'un algorithme génétique.

1. Introduction :

Comme nous l'avons vu dans le chapitre1, les contrôleurs flous permettent de piloterdes systèmes complexes ou difficilement modélisables en utilisant une méthode de raisonnement de type «SI 'condition' ALORS 'action'». Récemment, des auteurs ont prouvé qu'il est possible de reproduire le fonctionnement de n'importe quel contrôleur continu standard (avec une précision arbitraire) grâce àun contrôleur flou, Pourtant, lorsque le système à commander est connu de façon imprécise, la conception du contrôleur flou n'est pas une chose triviale [13].

En effet, Au cours des dernières années, plusieurs chercheurs ontconceptualisé les méthodes de réglage des performances des systèmes flous par les algorithmes génétiques. La mise au point génétiques des processus se concentrent sur l'adaptation de la base de données floue des paramètres de codage des fonctions d'appartenance et/ou facteurs d'échelle pour les variables d'entrées et de sorties Ilssont basés sur l'hypothèse que la base de règles a étédéjà conçue, soit par un expert humain ou par un avantprocessus d'apprentissage. [24]

2. Optimisation des contrôleurs flous de type Mamdani par algorithmes génétiques :

On considère un contrôleur à deux entrées, l'erreur $e(t)$ et sa variation $\Delta e(t)$ et une sortie $u(t)$, comme illustré sur le schéma qui suit :

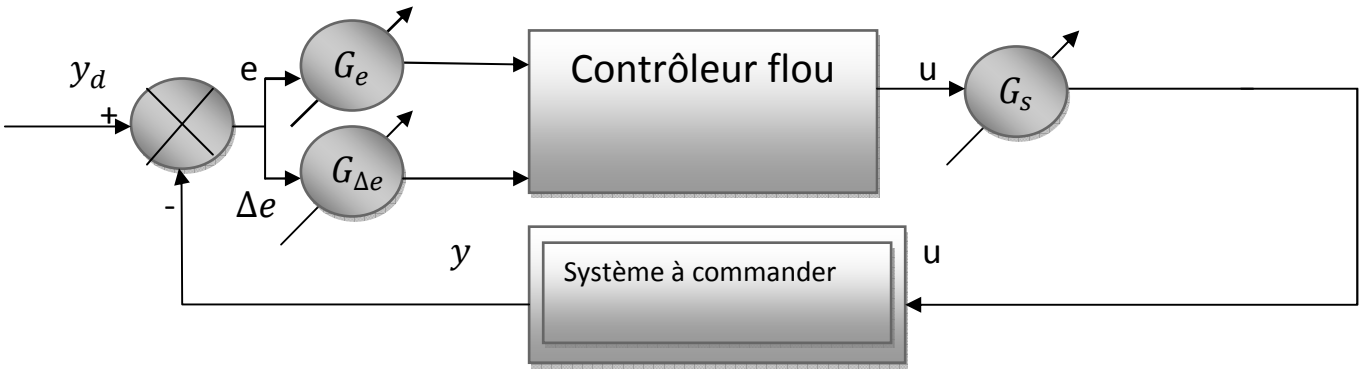


Figure3.1 : contrôleur flou à deux entrées.

Où $e(t) = y_d(t) - y(t)$ est la différence entre la sortie désirée $y_d(t)$ et la sortie mesurée du système commandé $y(t)$ .Pour avoir une flexibilité dans l'implantation du régulateur, les univers de discours des entrées et de sortie sont limités à un intervalle [-1,1] déterminé par la normalisation des entrées et de sortie pour ce faire, des gains d'adaptations (facteurs d'échelles) sont utilisés pour avoir la dynamique désirée. [16].

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

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

Dans cette section, le réglage des facteurs d'échelles sera étudié par les algorithmes génétiques. Comme ces derniers travaillent sur une population de chromosomes sur lesquels sont codés les paramètres à optimiser, il est nécessaire de définir la structure du chromosome qui prend en compte les paramètres cités précédemment.

3. La base des règles floues :

Les règles floues constituant la base du contrôleur, dans ce cas, possèdent deux prémisses et une seule conclusion:

$$Règle\, i_1, i_2 : \text{Si } (e \text{ est } A_{1i1}) \text{ ET } (\Delta e \text{ est } A_{2i2}) \text{ Alors } u \text{ est } B_j$$

Où: $(A_{1i1} \mid i_1 = 1,\ldots,i_{N_e})$, $(A_{2i2} \mid i_2 = 1,\ldots,i_{N_{\Delta e}})$ et $(B_j \mid j = 1,\ldots,N_s)$ sont les termeslinguistiques qualifiant respectivement l'erreur *e,* la variation de l'erreur Δ*e* et la variation de la commande *u*.

$N_e, N_{\Delta e}$ et N_S représentent respectivement le nombre de termes linguistiques associés à *e*, Δ*e* et *u*.

On définit alors $N_e \times N_{\Delta e}$ règles floues pour ce contrôleur, organisées couramment sous forme d'un tableau et dont le contenu correspond aux conclusions de toutes les configurations possibles de prémisses.

La conclusion d'une règle repérée par les indices i_1 et i_2 est alors représentée par un nombre $R_{i_1 i_2}$ défini comme suit : [17]

$$R_{i_1 i_2} = \begin{cases} 0 & \text{Si la règle}(i_1, i_2) \text{ est inutile.} \\ j = \text{j}\in\{1,\ldots,\text{N}_{\text{ts}}\} & \text{Si la conclusion de la règle } (i_1, i_2) \text{ est } B_j. \end{cases} \quad \text{..... (3.2)}$$

	$A_{21}A_{22} \ldots A_{2i2} \ldots A_{2N_{te}}$
A_{11}	$R_{11}R_{12} \ldots R_{N_{te}}$
A_{12}	$R_{21}R_{22} \ldots R_{2N_{te}}$
.	. . . .
.	. . . .
A_{1n}	.... R_{ni2}.... ....
.	. . . .
.	. . . .
$A_{1N_{te}}$	$R_{N_{te}1}$ .... $R_{N_{te}N_{te}}$

Tableau3.1 : table de règles floues.

4. Méthodes d'optimisation :

- **Structure du chromosome :**

Pour appliquer les AG directement ou couplée avec d'autres méta heuristiques, le problème de la représentation du chromosome est posé. La première décision que le concepteur doit faire est de savoir comment représenter une solution dans une structure chromosomique.

Selon la figure 3.8, le chromosome proposé est composé de trois paramètres. Ces paramètres représentent les deux facteurs d'échelle en entrée et le facteur d'échelle en sortie du contrôleur flou :

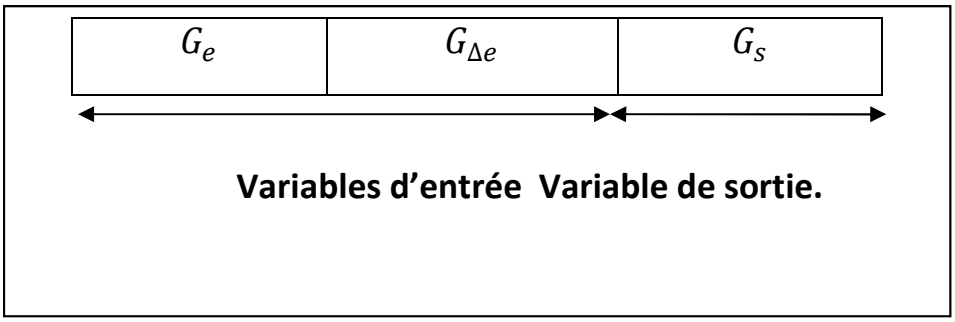


FIGURE 3.8 : *Structure du chromosome de l'AGS.*

Il existe trois méthodes de codage des chromosomes :

- ✓ **Codage simple :**les parties constituant le chromosome de l'AG sont codées par un seul type de codage.
- ✓ **Codage hybride :**dans ce cas,on distingue deux types de codage utilisés pour les trois parties du chromosome de l'AG.
- ✓ **Codage mixte :** on utilise trois types de codage pour les trois parties des chromosomes.

5. initialisation des chromosomes :

Habituellement, les chromosomes de la première population sont initialisés avec des valeurs aléatoires ou avec des valeurs qui décrivent le savoir-faire des experts pour accélérer la convergence de l'algorithme d'optimisation.

6. Opérateurs génétiques :

Comme le chromosome d'optimisation de contrôleur flou comprend trois chaînes.il convient alors de spécifier pour chacune d'elle les opérateurs génétiques qui leurs convient [4].

6.1 Opérateur de Croisement :

- ✓ ***Croisement en base n.***
- ✓ ***Croisement des chaînes réelles.***
- ✓ ***Croisement des chaînes binaires.***

6.2 Opérateur de Mutation :

- ✓ ***Mutation des chaînes en base n.***
- ✓ ***Mutation des chaînes réelles.***

7. Conclusion :

Dans ce chapitre nous avons décrit la méthode pour la conception d'un contrôleur flou par algorithmes génétiques.
Nous avons donc présenté la méthode de synthèse d'un contrôleurflou de type *Mamdani* basée sur un algorithme génétique. Cette méthode consiste en l'optimisation des facteurs d'échelle. Pour cela, nous avons d'abord proposé une méthode de représentation de l'ensemble des paramètres du contrôleur.
Ensuite nous avons décrit l'étape d'initialisation des chromosomes en s'attardant sur lapartie du chromosome correspondant aux facteurs d'échelle du contrôleur flou.Pour finir nous avons cité les opérateurs génétiques permettant une meilleure exploration de l'espace de recherche.

1. INTRODUCTION :

La biologie moderne a fait un saut extraordinaire dans la compréhension des mécanismes biologiques intervenant dans le monde du vivant. Cette connaissance ouvre actuellement un champ d'investigation très intéressant qui permet de mettre en œuvre les méthodes expérimentales et théoriques pour l'observation et la modélisation des systèmes.

L'objectif de ce chapitre est d'optimiser les paramètres du système proposé en utilisant une méthode stochastique d'optimisation par les algorithmes génétique nous permettant de converger éventuellement vers un optimum globale. La procédure de cette méthode sera développée par un programme sous MATLAB.

2. Le modèle du système :

➢ Modèle mécanique :

Le système à optimiser est le contrôleur de vitesse du véhicule illustré dans la figure suivante :

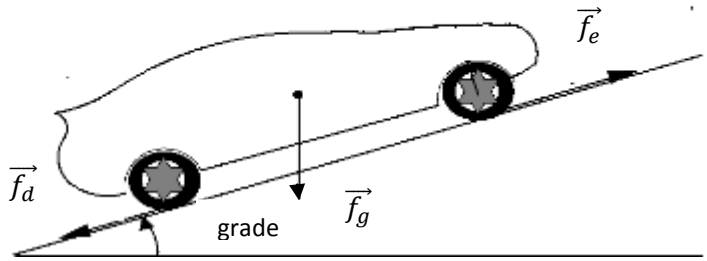


Figure 4.1 : Système à commander.

Représenté par le système d'équation ci-dessous : [19]

$$m\dot{v} = f_e(\theta) - f_d(v) - f_g. \quad \text{..... (4.1).}$$

Avec :

Et les forces externes définies comme suite:

- ✓ Force motrice : $f_e(\theta) = f_i + \gamma\sqrt{\theta}.$ ..... (4.2).

- ✓ Force de trainage : $f_d(v) = \alpha v^2 sign(v)$. ..... (4.3).
- ✓ Force de gravité : $f_g = mgsin(grade)$. ..... (4.4).

Où :

Symboles	Explication	Valeur
M	Masse du véhicule	1000 Kg
γ	Constante road grade	12500 Newtons
θ	Position de l'étrangleur	/
f_i	engine idle force	1000 N
α	Constante	4 N/ (m/s)²
Grade	Pente	10%
G	Gravité	9.81 m/s²
Consigne (V_c)	Vitesse	40 m/s

Tableau 4.1 : paramètres mécaniques du véhicule.

➢ Modèle sur SIMULINK :

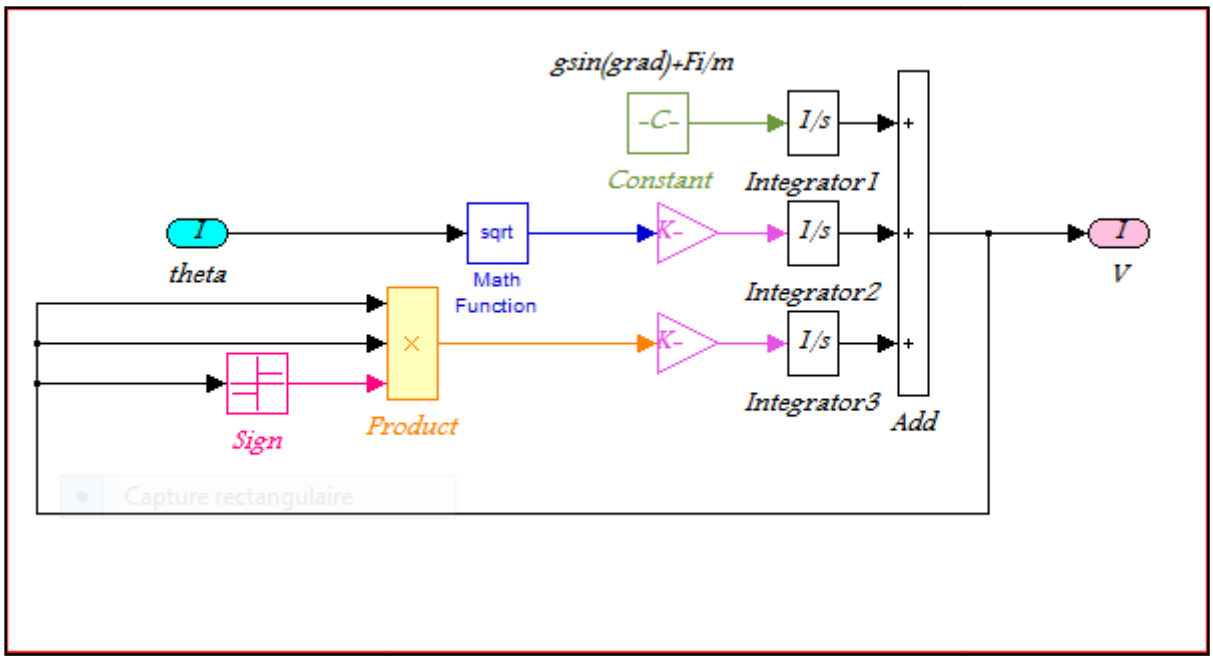


Figure 4.2 : Modèle du système.

3. Représentation du système contrôlé :

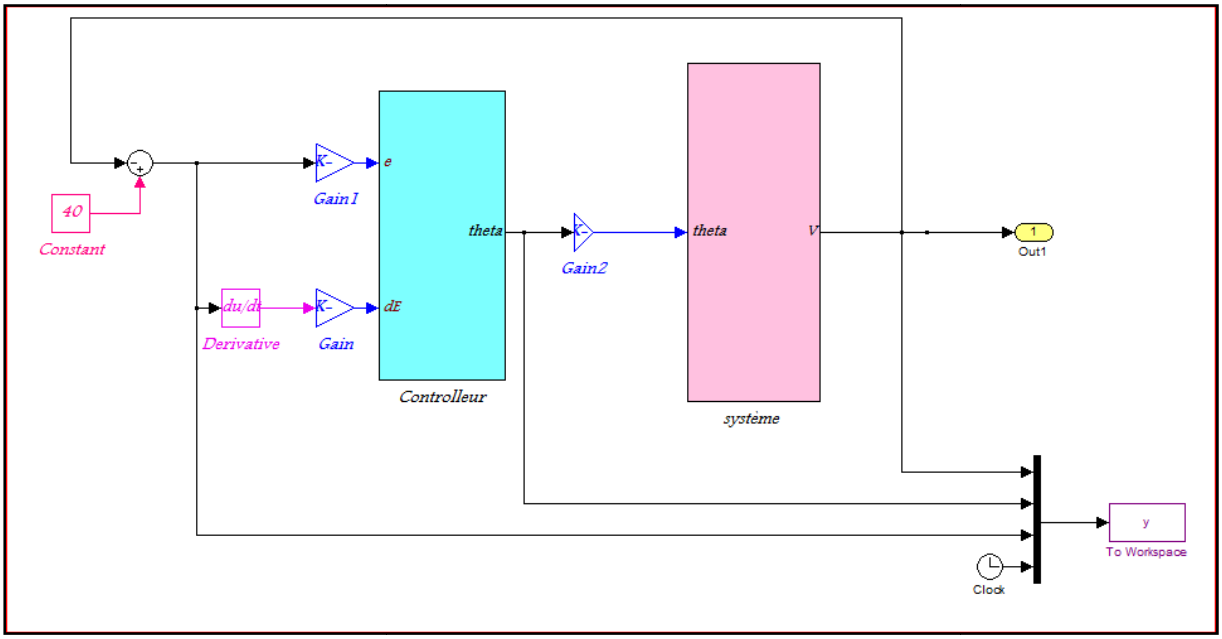


Figure 4.3 : Schéma bloc du système régulé.

4. Représentation du contrôleur flou :

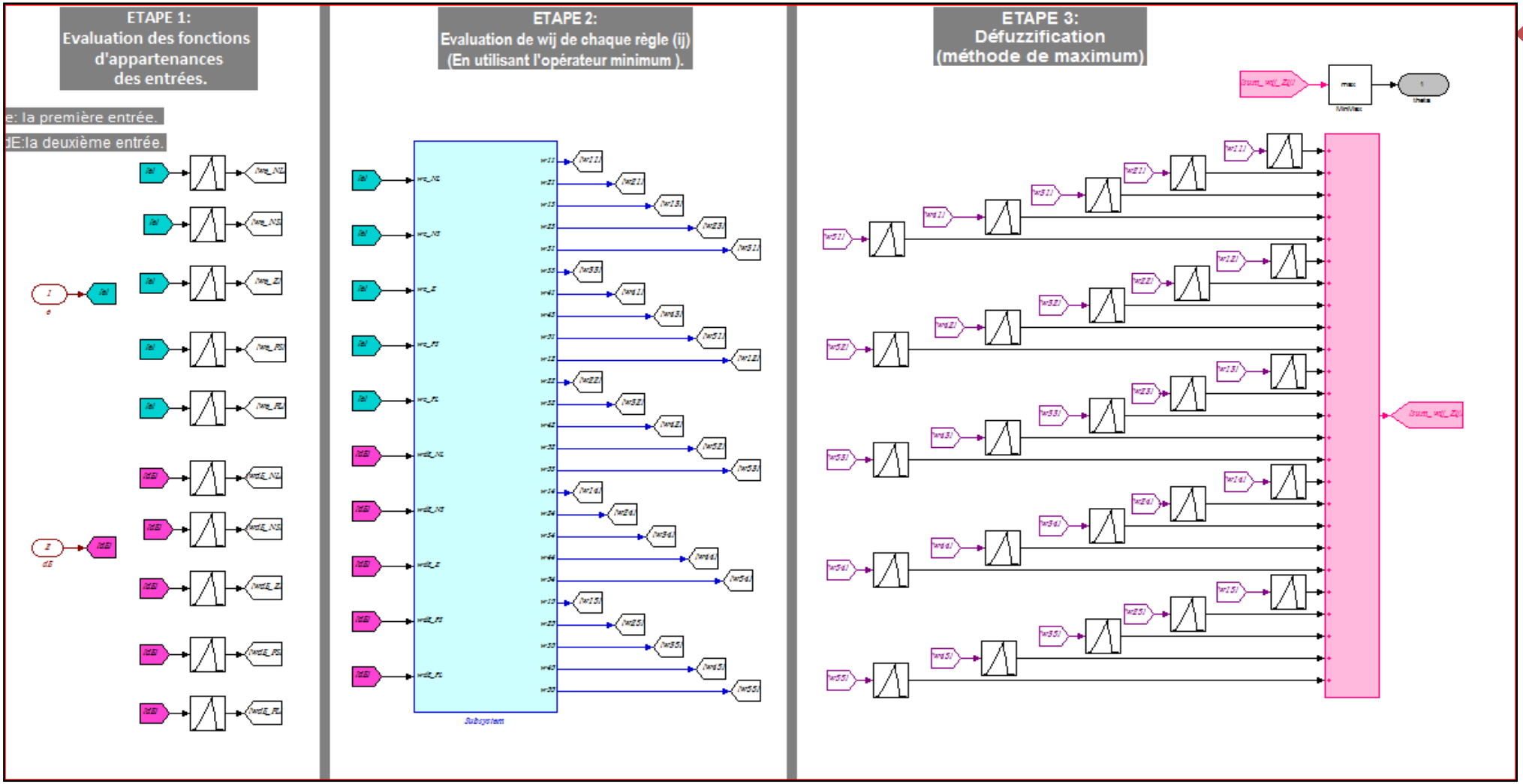


Figure 4.4 : Contrôleur flou sous SIMULINK.

Notre travail consiste en l'optimisation d'un contrôleur flou par les algorithmes génétiques, pour ce faire, nous devons optimiser pour soit les fonctions d'appartenances et/ou les règles floue, cependant, en raison de la complexité du programme (beaucoup de paramètres à prendre en considération) et pour garder les fonctions d'appartenance de sotie dans un univers de discours borné c'est-à-dire [-1,1], nous avons donc opter pour l'optimisation des facteurs d'échelle.

Les règles floues utilisées sont de types triangulaires, elles se présentent en cinq règles pour l'erreur et cinq règles pour la dérivée de l'erreur, ce qui résulte en 25 règles en sortie. Ces dernières sont représentées dans le tableau suivant :

dE \ e	NL	NS	Z	PS	PL
NL	NL	NL	NS	NS	NS
NS	NL	NS	Z	Z	Z
Z	NL	NS	Z	PS	PL
PS	Z	Z	Z	PS	PL
PL	PS	PS	PS	PL	PL

Tableau 4.2 : Les règles floues.

- **Règles floues sous SIMULINK :**

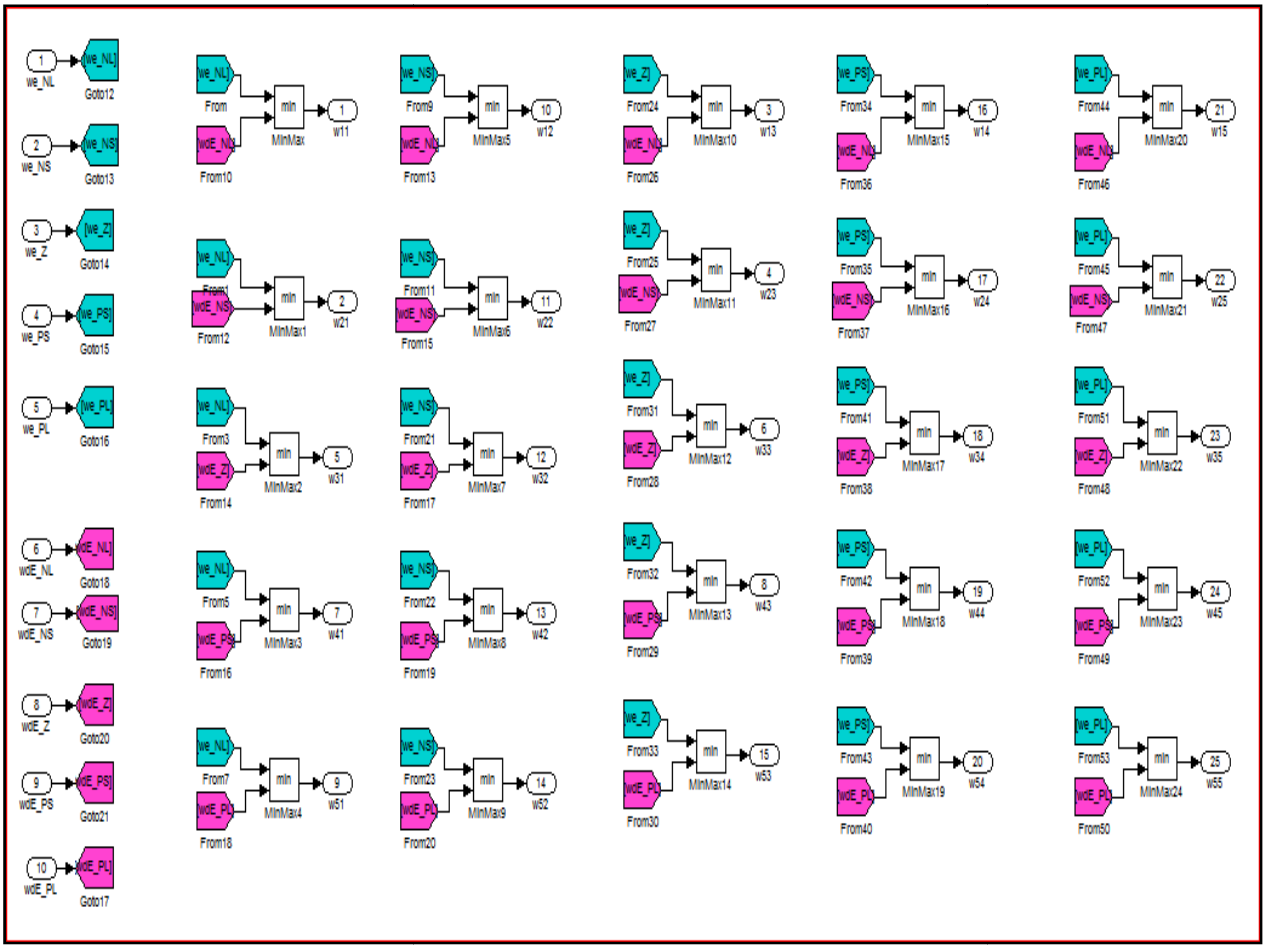


Figure 4.5: Les règles floues sous SIMULINK.

- **Surface non linéaire du contrôleur flou :**

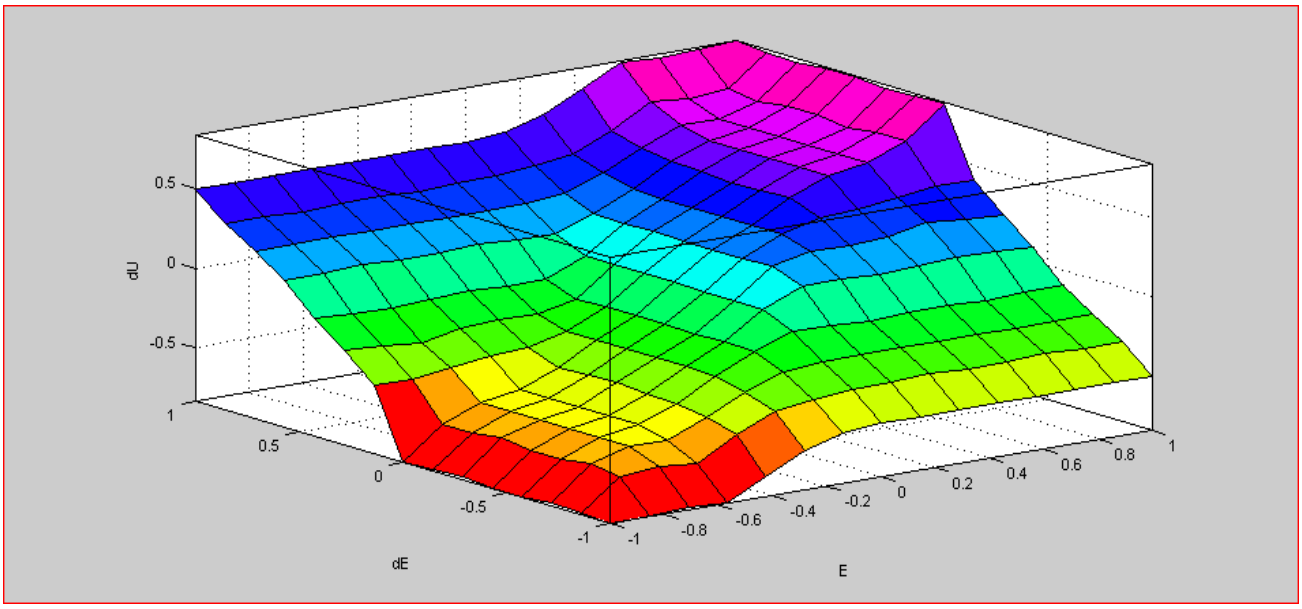


Figure4.6 : Surface non linéaire du contrôleur flou.

5. Simulation :

5.1. Simulation sans optimisation:

Nous avons imposé les facteurs d'échelle suivants :

k= [K_p K_d K_u]= [1.9 1 0.1].

Avec :

K_p , K_d : gains en entrée du contrôleurs flou.

K_u :gain en sortie du contrôleur flou.

- **Resultats de simulation :**

Nous avons obtenu les resultats suivants :

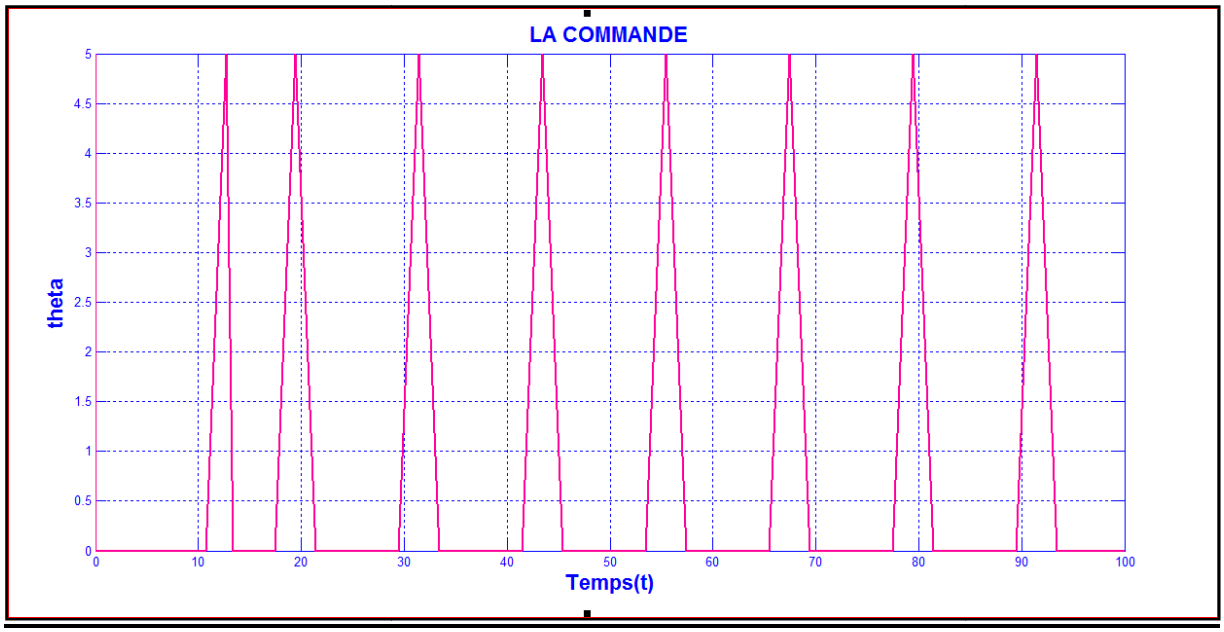


Figure4.7 : Représentation de signal de commande avant l'optimisation.

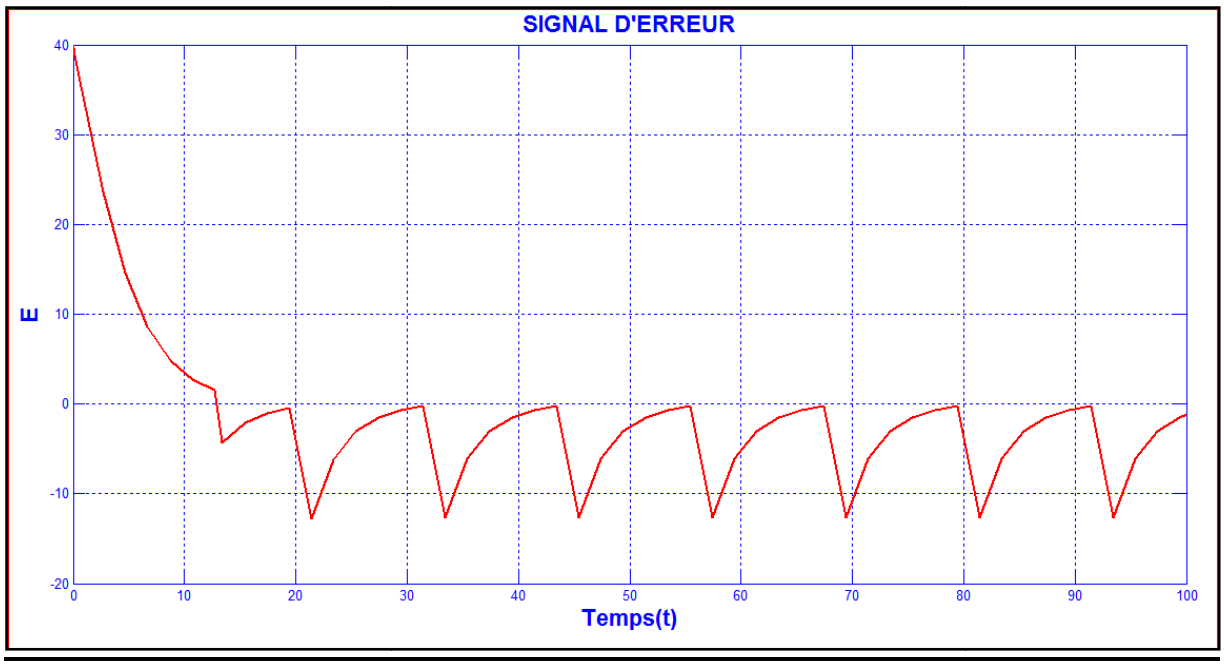


Figure 4.8 : Représentation de signal de l'erreur avant l'optimisation.

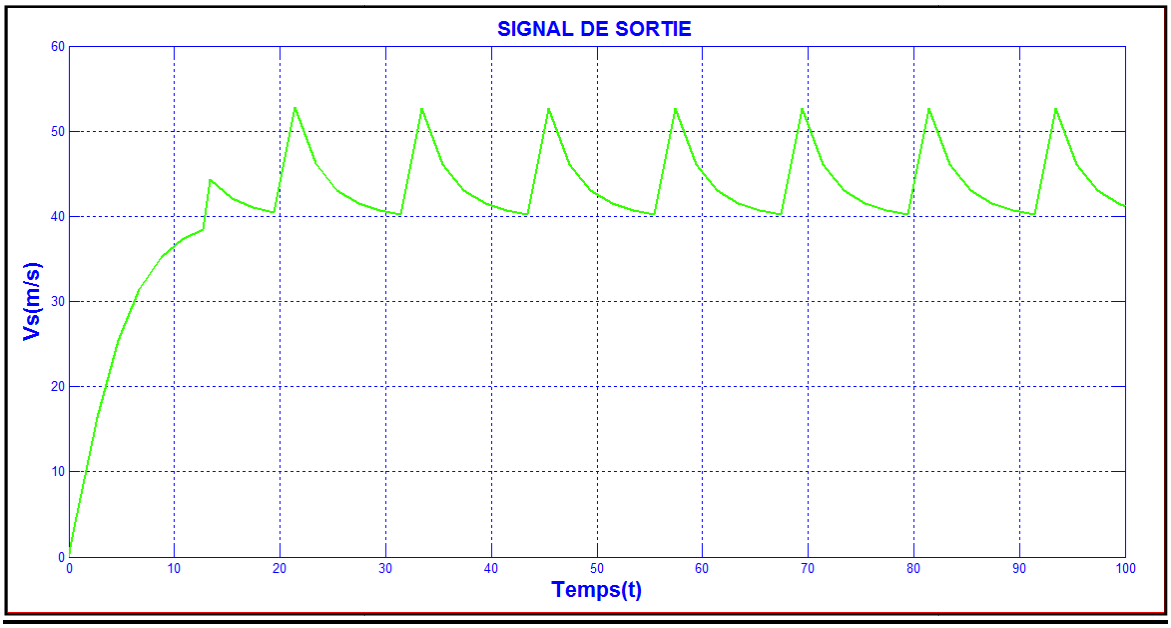


Figure 4.9 : Représentation de signal de la sortie avant l'optimisation.

➢ **Interprétation des résultats :**

D'après les résultats, on remarque que le système est instable vue que la sortie et l'erreur ne convergent pas vers une valeur finie.

5.2. Simulation avec optimisation :

Pour obtenir les résultats désirés nous allons procédé à une optimisation des facteurs d'échelle du contrôleur, par deux méthodes.

5.2.1. Optimisation des paramètres par « fminsearch » :

a. Présentation :

Cette fonction trouve le minimum d'une fonction à plusieurs variables, à partir d'une première estimation. Ceci généralement fait référence à une optimisation non-linéaire sans contrainte. Elle utilise la méthode de recherche de simplex de Lagarias et al,qui est une méthode de recherche directe qui n'utilise pas des gradients numériques ou analytiques, ce pendant, elle ne garantie pas une convergence vers un minimum global.

b. Résultats de simulation :

1er essai :

On utilise les mêmes paramètres qu'on a utilisés dans la partie précédente (sans optimisation).

k= [1.9 1 0.1].

Les résultats obtenus sont illustrés ci-dessous :

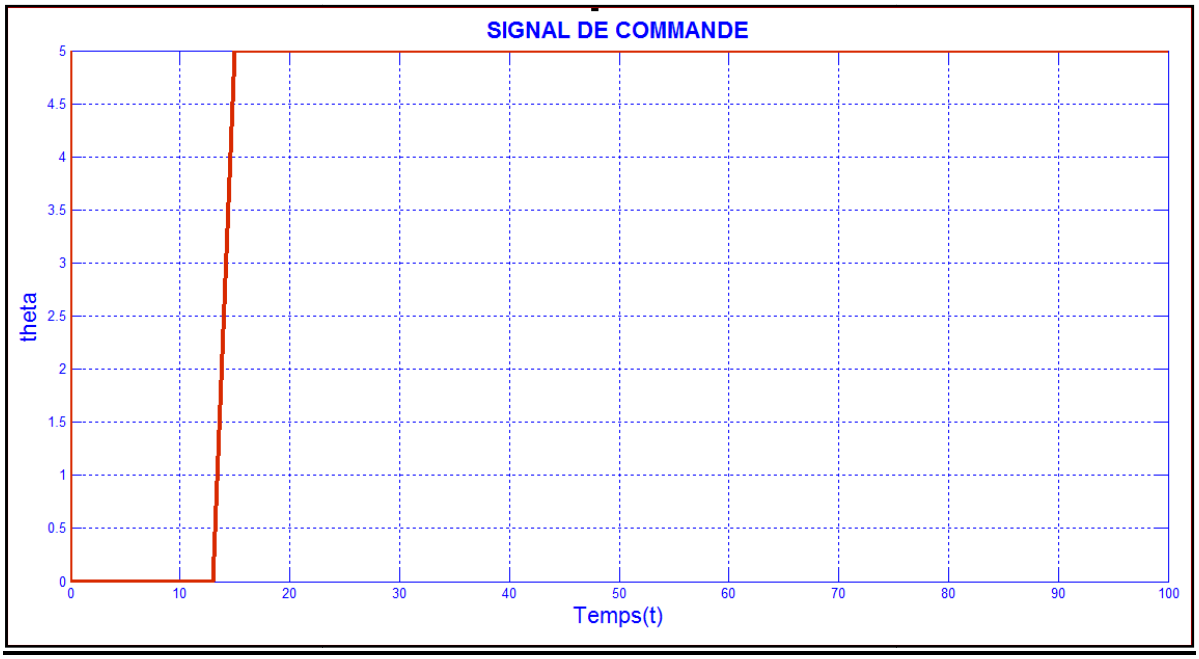


Figure 4.10: Représentation de signal de commande après l'optimisation.

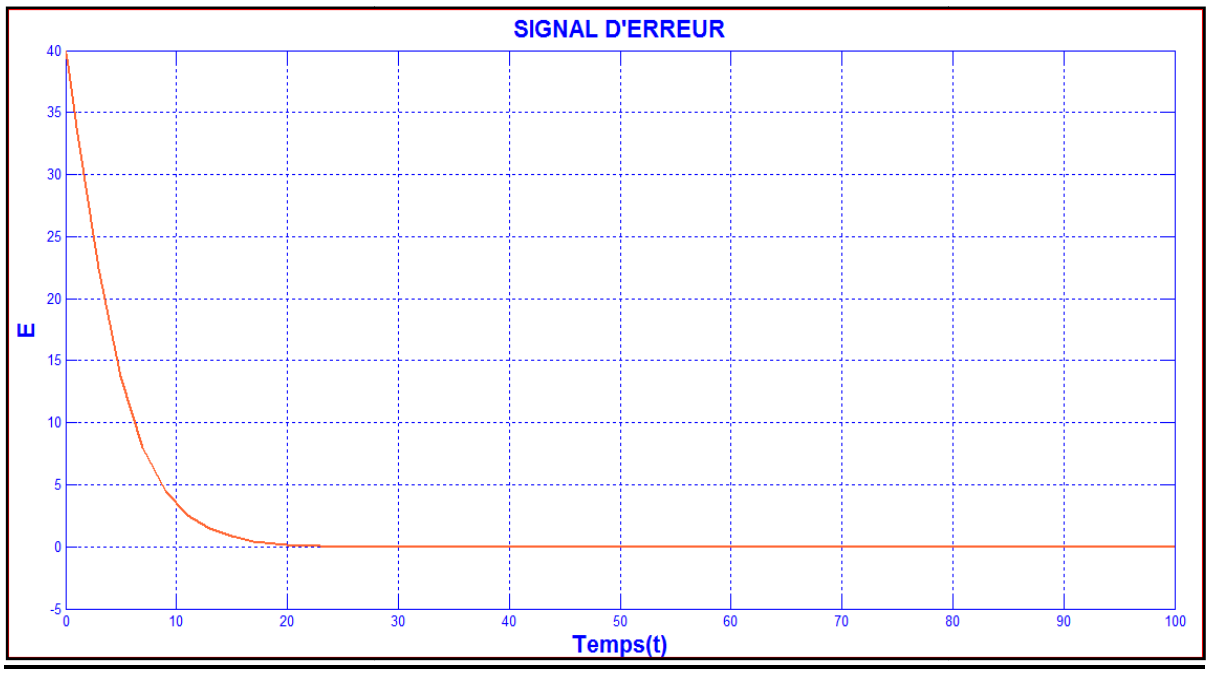


Figure 4.11: Représentation de signal de l'erreur après l'optimisation.

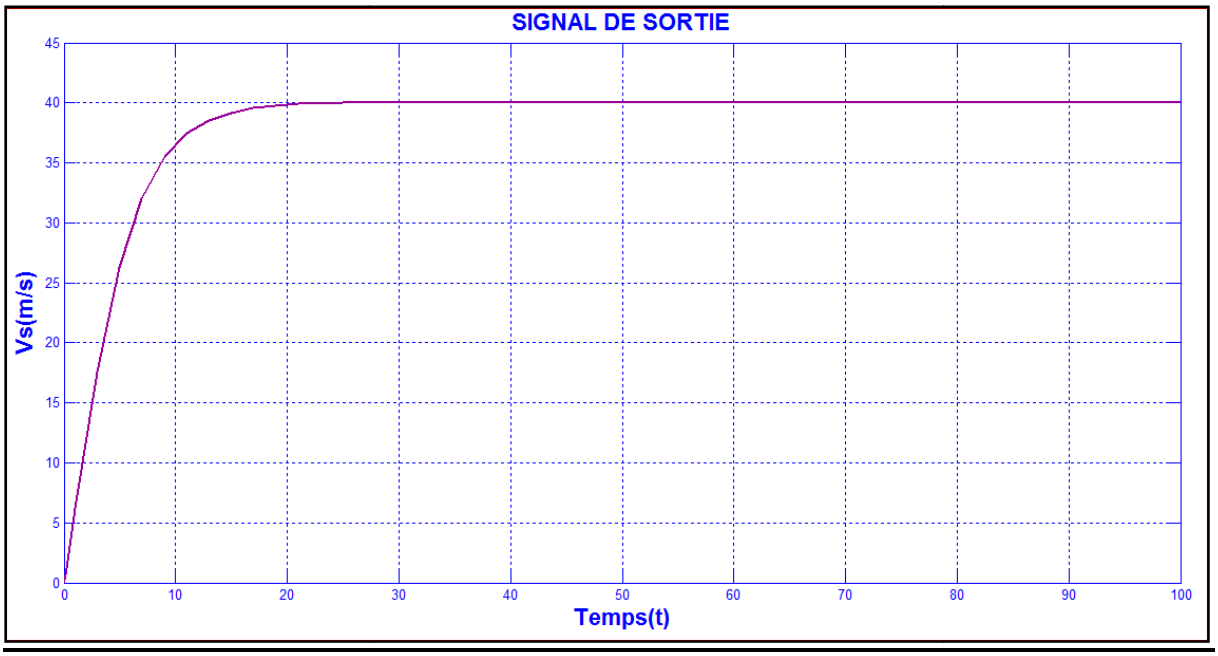


Figure 4.12 : Représentation de signal de sortie après l'optimisation.

Les valeurs obtenues du 1er essai :

Paramètres	K_p	K_d	K_u
Valeur	1.5695	2.05790	0.0000
Fitness(fval)	8.215625889e+003		

Tableau 4.3 : Les valeurs obtenues.

2ème essai :

Cette fois ci on injecte les paramètres optimaux obtenu du le premier essai :

K= [1.5695 2.05790 0.0000]

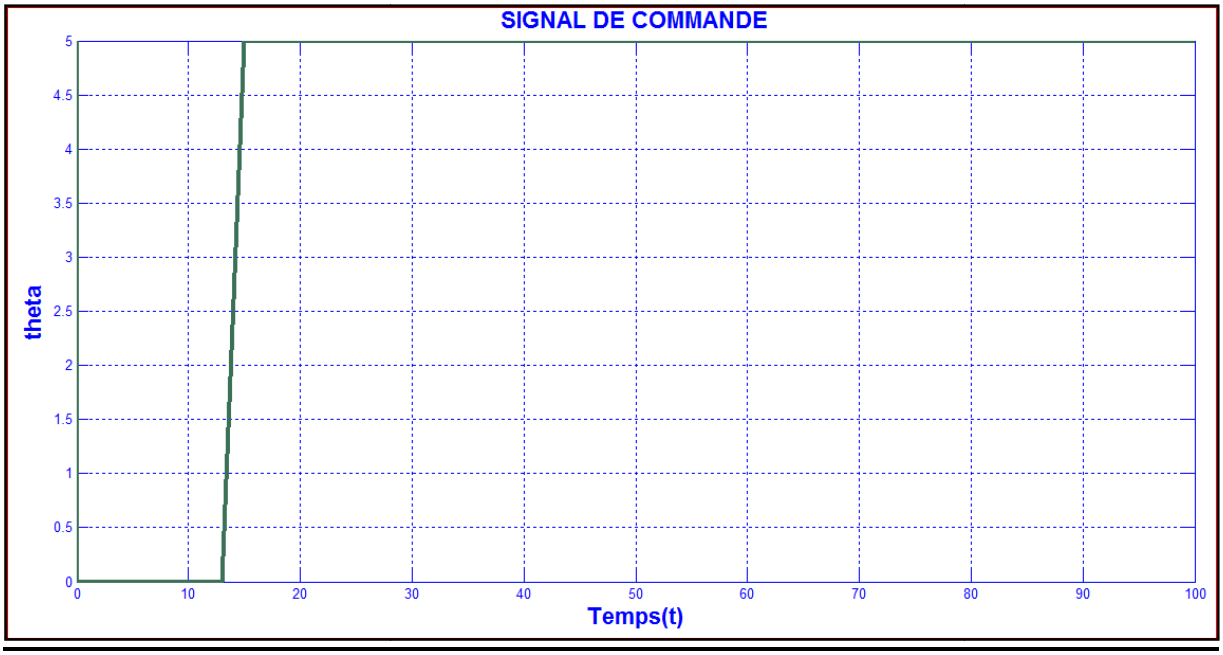


Figure 4.13: Représentation de signal de commande après l'optimisation.

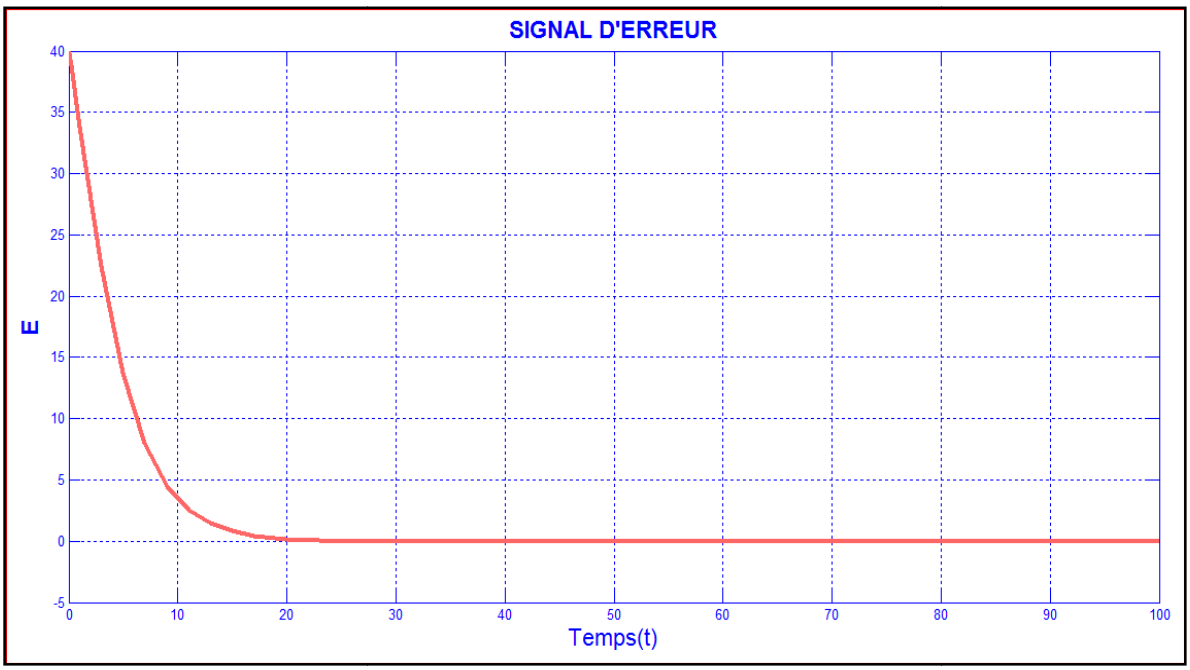


Figure 4.14: Représentation de signal de l'erreur après l'optimisation.

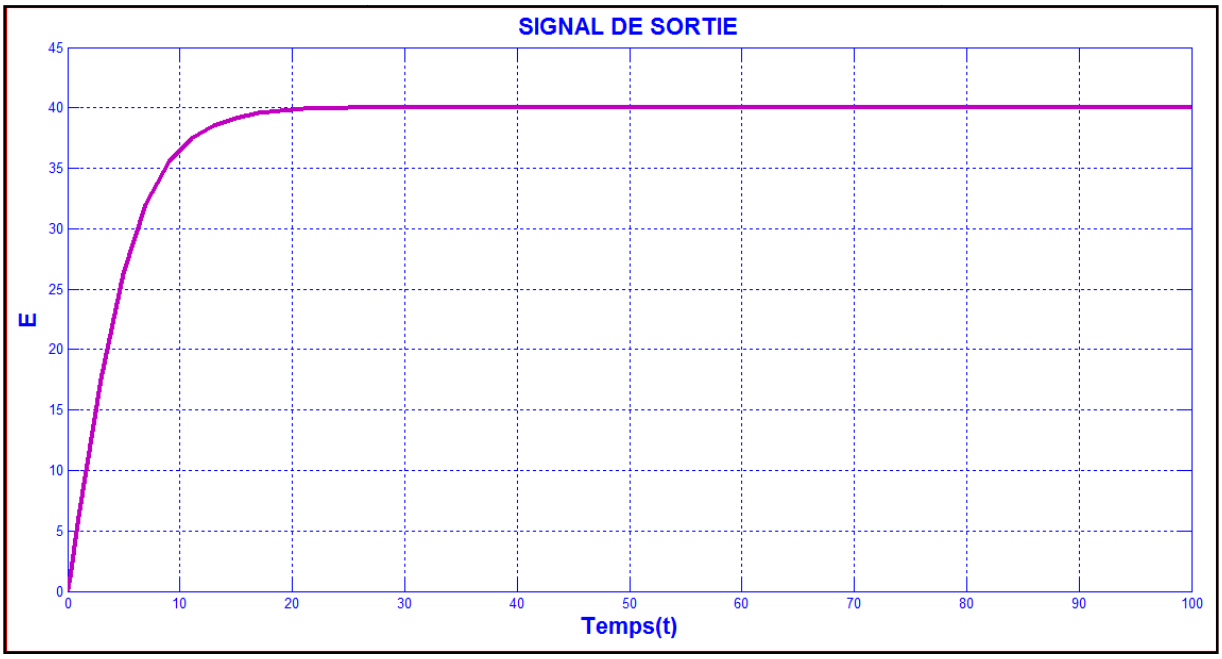


Figure 4.15 : Représentation de signal de sortie après l'optimisation.

Les résultats obtenus du 2ème essai :

Paramètres	K_p	K_d	K_u
Valeur	1.6251	2.0479	0.0000
Fitness (fval)	8.215625808e+003		

Tableau 4.4: Les valeurs obtenues.

c. Interprétation des résultats :

Dans le deux cas, on remarque que l'erreur et la sortie convergent vers des valeurs finies, ce qui nous permet de dire que notre système s'est stabilisé, juste que dans le deuxième essai on a obtenu un résultat meilleurs vis-à-vis de la valeur de l'erreur (fval).

5.2.2. Optimisation des paramètres par les algorithmes génétique :

a. Présentation :

Les **(AG**), utilisent un vocabulaire similaire à celui de la génétique, cependant, les processus auxquels ils font référence sont beaucoup plus complexes. En imitant ce principe, les algorithmes génétiques appliqués à un problème d'optimisation font évoluer un ensemble de solutions utilisent un mécanisme de sélection naturelle. Ainsi, les AG ne se basent pas sur un **individu**, mais sur une **population** d'individus qui vont évoluer de génération en génération pour obtenir un résultat se rapprochant de la solution optimale.

Pour un problème d'optimisation donné, un individu représente un point de l'espace d'état ou une solution possible du problème donné il est composé d'un ou plusieurs chromosomes. Les chromosomes sont eux-mêmes constitués de gènes qui contiennent les caractères héréditaires de l'individu. A chaque individu est attribué un **"fitness"** qui mesure la qualité de la solution qu'il représente, souvent c'est la valeur de la fonction à optimiser. Ensuite, une nouvelle population des solutions possibles est produite en sélectionnant les parents parmi les meilleurs de la **"génération"** actuelle pour effectuer des **croisements** et des **mutations.**

La sélection a pour but de favoriser les meilleurs éléments de la population, tandis que le croisement et la mutation assurent une exploration efficace de l'espace d'état. Les meilleurs individus d'une génération vont créer une nouvelle génération plus adaptée au problème dont la nouvelle population contient une plus grande proportion de caractéristiques des meilleurs individus de la génération précédente.

b. Les opérateurs génétiques :

L'algorithme génétique proposé utilise :

· Un opérateur de sélection par roulette de loterie.

· Un opérateur de croisement en deux points.

· Un opérateur de mutation consiste en un changement aléatoire du gène dans l'intervalle associe.

c. La fonction objective :

La fonction objective est l'erreur quadratique entre la réponse du système donné par le constructeur et celle du système ayant les paramètres optimisés par l'algorithme génétique. Elle s'exprime comme suite :

$$J = \int e^2 \, dt \quad \text{..... (4.5)}$$

Avec :

$$e = V_s - V_c \quad \text{..... (4.6)}$$

V_s: sortie mesurée.

V_c: consigne.

La plage de variation des paramètres utilisés :

UB= [1.9001 1.0001 0.1002]

LB= [1.8999 1.0000 0.10000]

d. Résultats de simulation :

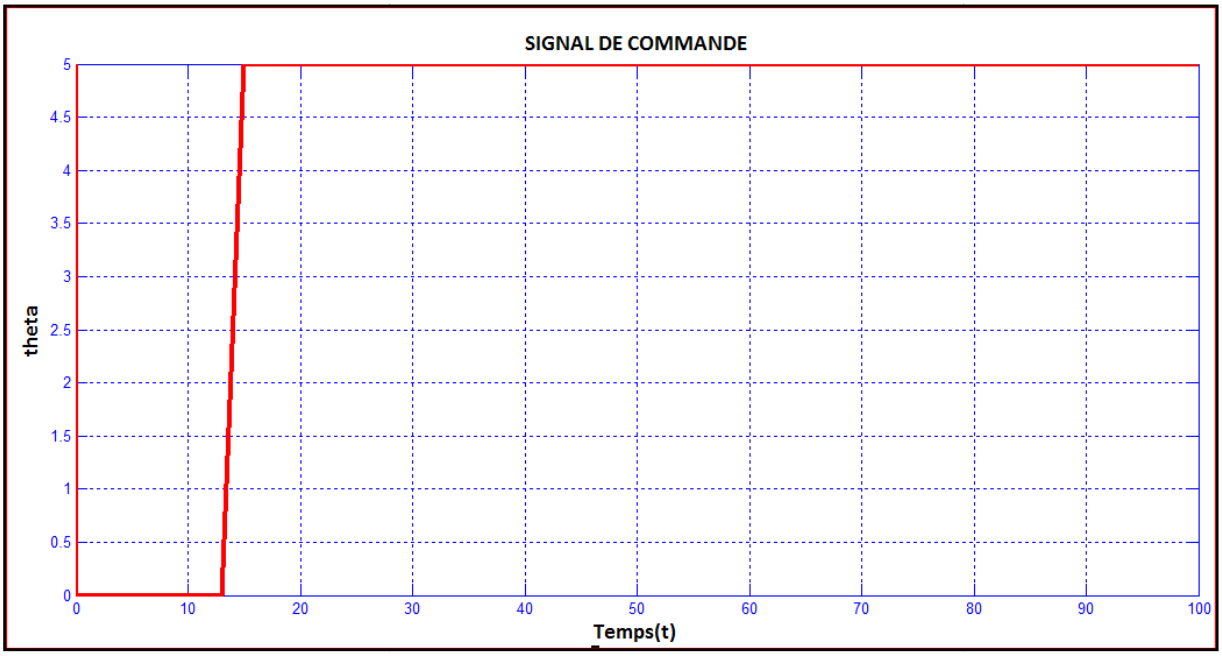


Figure 4.16: Représentation de signal de commande après l'optimisation.

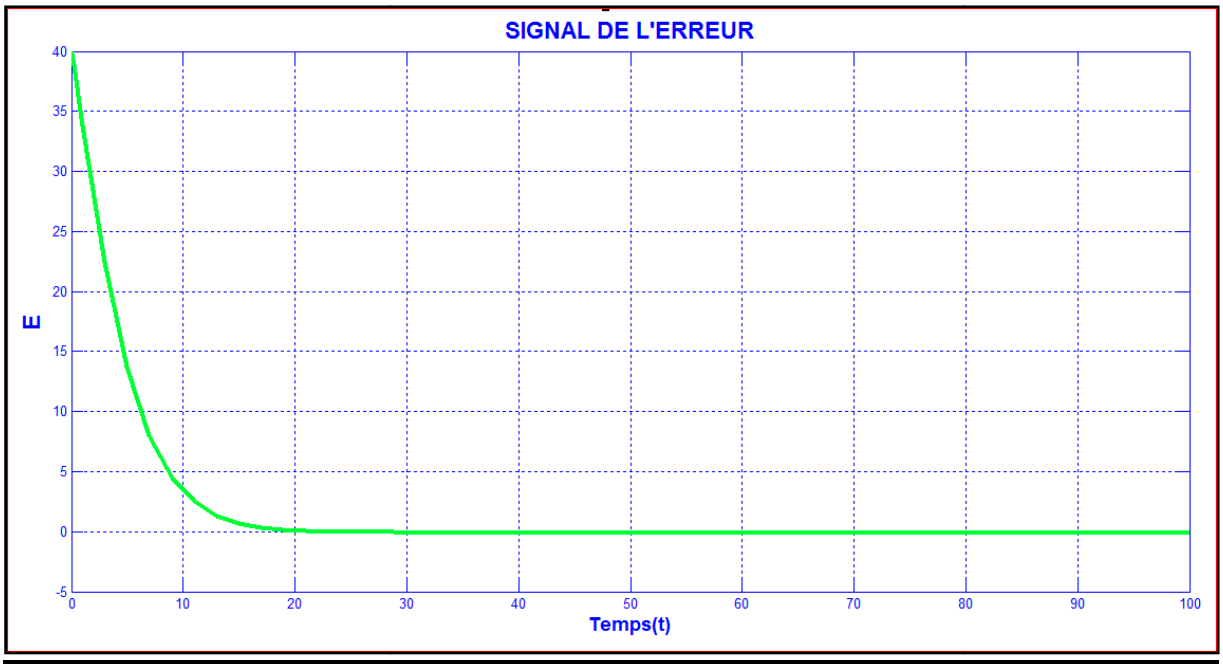


Figure 4.17: Représentation de signal de l'erreur après l'optimisation.

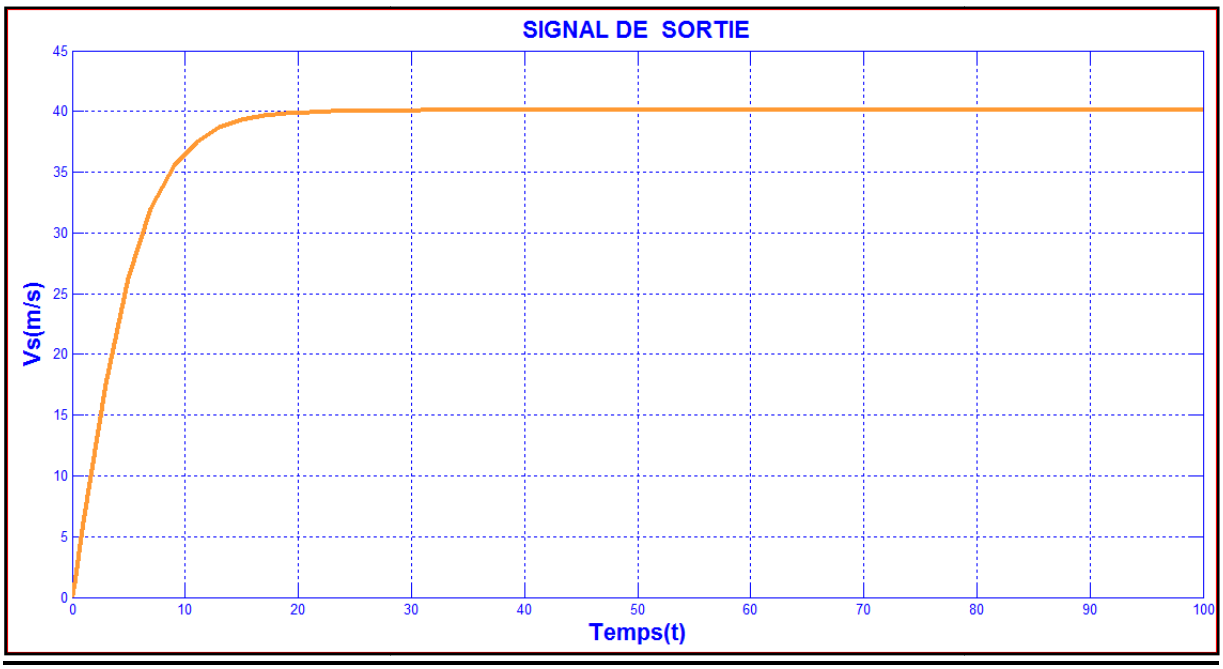


Figure 4.18 : Représentation de signal de sortie après l'optimisation.

e. Interprétation des résultats :

D'après les résultats, on remarque que le système est sable vue que la sortie et l'erreur convergent vers une valeur finie.

f. Organigramme de l'AG :

L'organigramme de l'algorithme génétique utilisé dans l'optimisation des paramètres du contrôleur est le suivant :

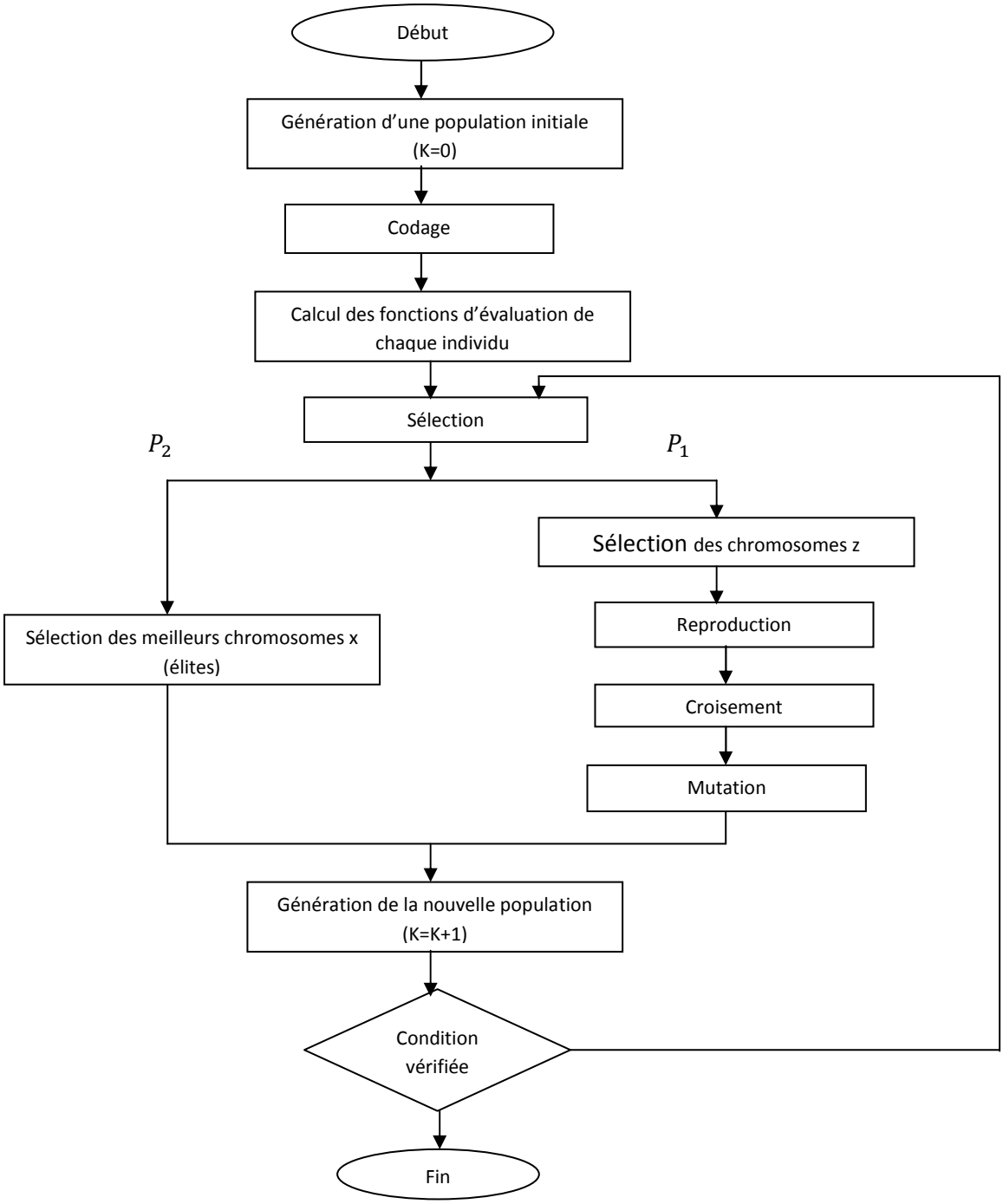


Schéma 4.1: organigramme de l'algorithme génétique.

g. Fonctions utilisées :

Fonction	Signification
ga	Trouver minimum de fonction en utilisant un algorithme génétique.
gaoptimset	Créer structure génétique des options d'algorithme.

Tableau 4.5: fonction utilisées dans le programme d'optimisation.

h. Optimisation des paramètres :

Champs	Valeur	Commentaire
PopInitRange	UB=[1.9001 1.0001 0.1002] LB=[1.8999 1.0000 0.10000]	Plage de variation des paramètres (individus)
PopulationSize	200	Taille de la population
ElitCount	1	Taux d'élitisme
Generations	80	Nombre de génération
StallTimeLimit	800	Critère d'arrêt (temps)
InitialPopulation	200	Choix aléatoire
Migrationfraction	0.5	Emigration des individus.
SelectionFcn	Selectionroulette (selection par roulette).	Fonction de séléction des meilleurs individus.
CrossoverFcn	Crossovertwopoint (croisement en deux points)	Type de croisement
MutationFcn	Mutationadaptfeasible (mutation adaptative)	Type de mutation
PlotFncs	✓ Gaplotbestf. ✓ Gaplotbestindiv. ✓ gaplotdistance. ✓ gaplotselection.	Représentation graphique

Tableau 4.6: Paramètres des algorithmes génétique utilisés.

i. Résultats obtenus :

❖ La figure suivante montre les meilleurs individus ainsi leurs fitness après 52 générations.

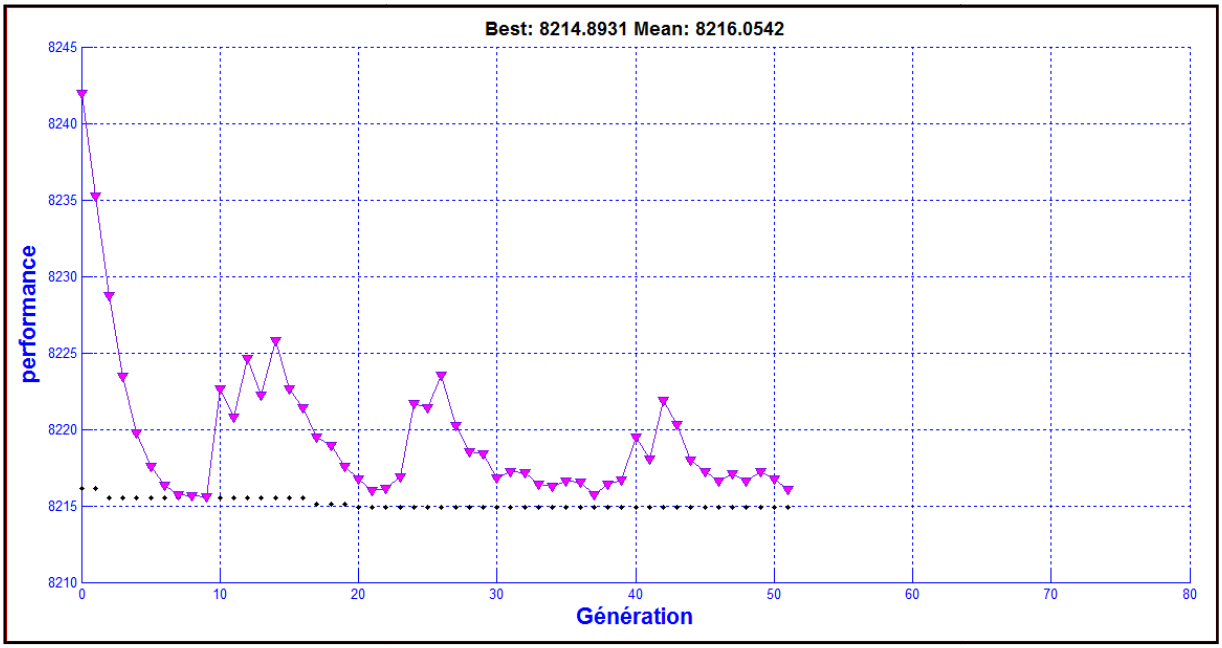


Figure4.19: Evolutions des performances en fonction des générations.

➢ Interprétation des résultats :

Pour 80 générations et une population de 200 individus par génération, les opérateurs du croisement et de mutation sont appliqués, générant de nouveaux individus.

Le meilleur individu pour le quel le minimum global est égal à 8216.0542

- En violet : performances moyennes de la population.
- En noir: meilleur performance.

❖ La courbe inférieure montre la distance moyenne entre les individus à chaque génération, ce qui est une bonne mesure de la diversité d'une population.

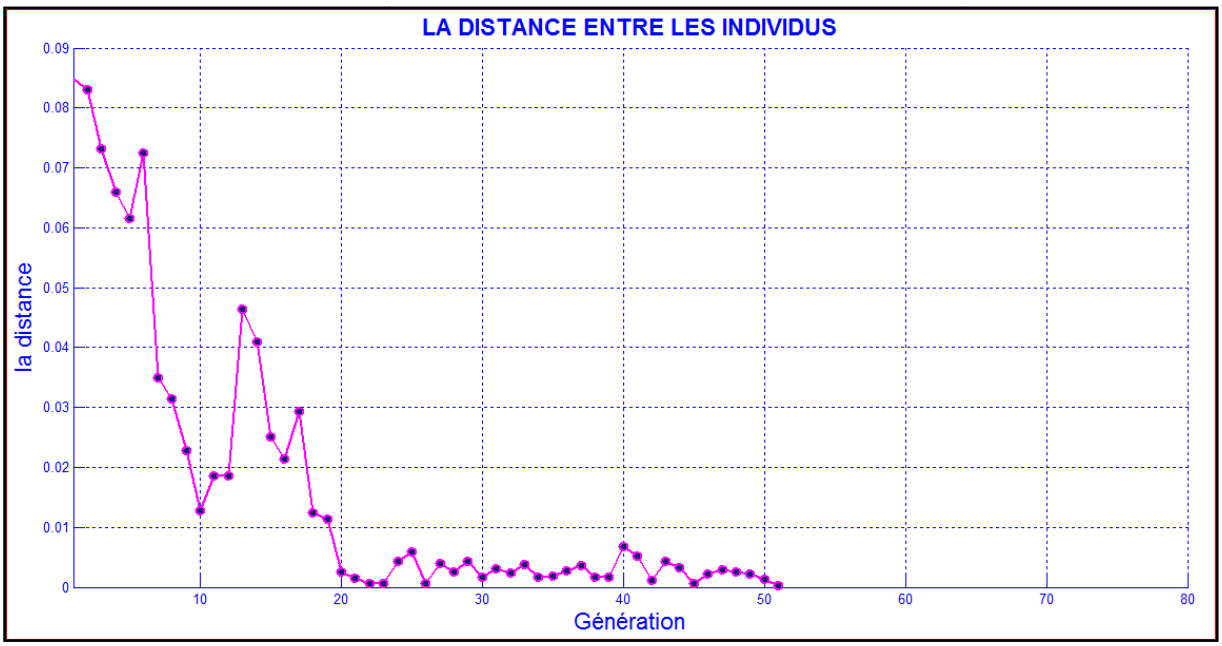


Figure 4.20 : La distance entres les individus en fonction des générations.

❖ La figure suivante représente le meilleur individu de chaque paramètre :

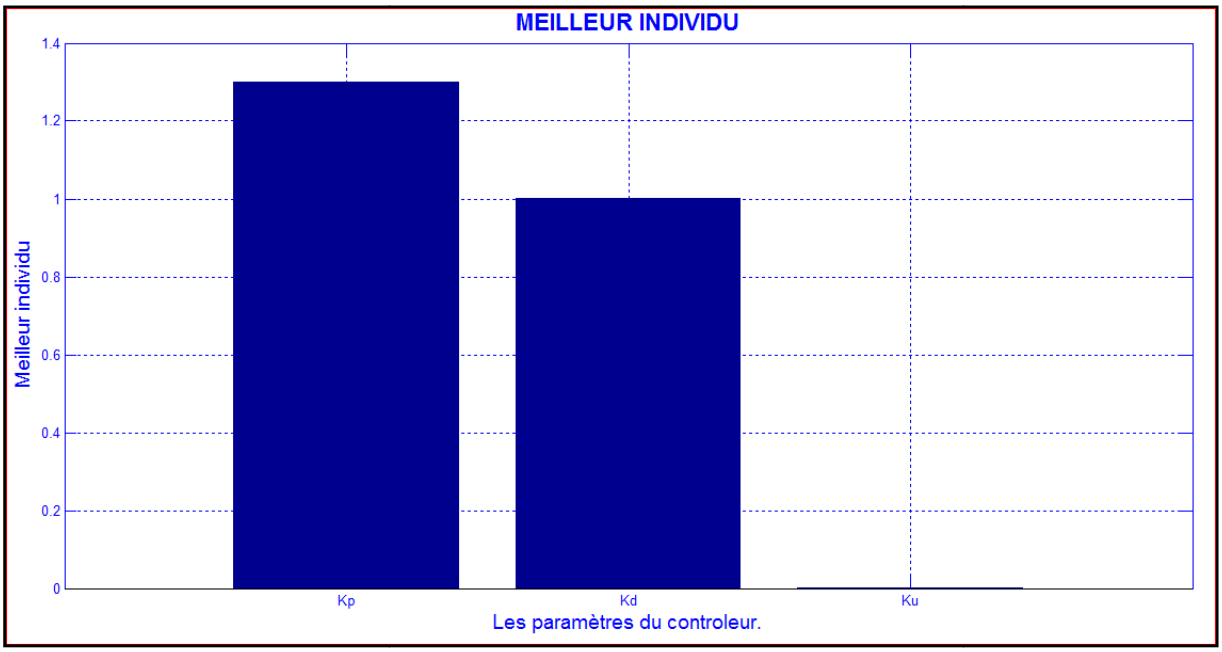


Figure 4.21: Le meilleur individu en fonction des paramètres (K_p, K_d, K_u).

Les résultats obtenus du 1er essai :

Paramètres	K_p	K_d	K_u
Valeur	1.299990247121750	1.000010171012110	0.000009003298911
Fitness (fval)	8.214893137992249e+003		

Tableau 4.7 : résultats finals.

2ème essai :

La plage de variation des paramètres utilisés :

UB= [1.30 1.0000200 0.0000009];
LB= [1.29998 1.000000 0.00001];

- La figure suivante montre les meilleurs individus ainsi leurs fitness après 52 générations.

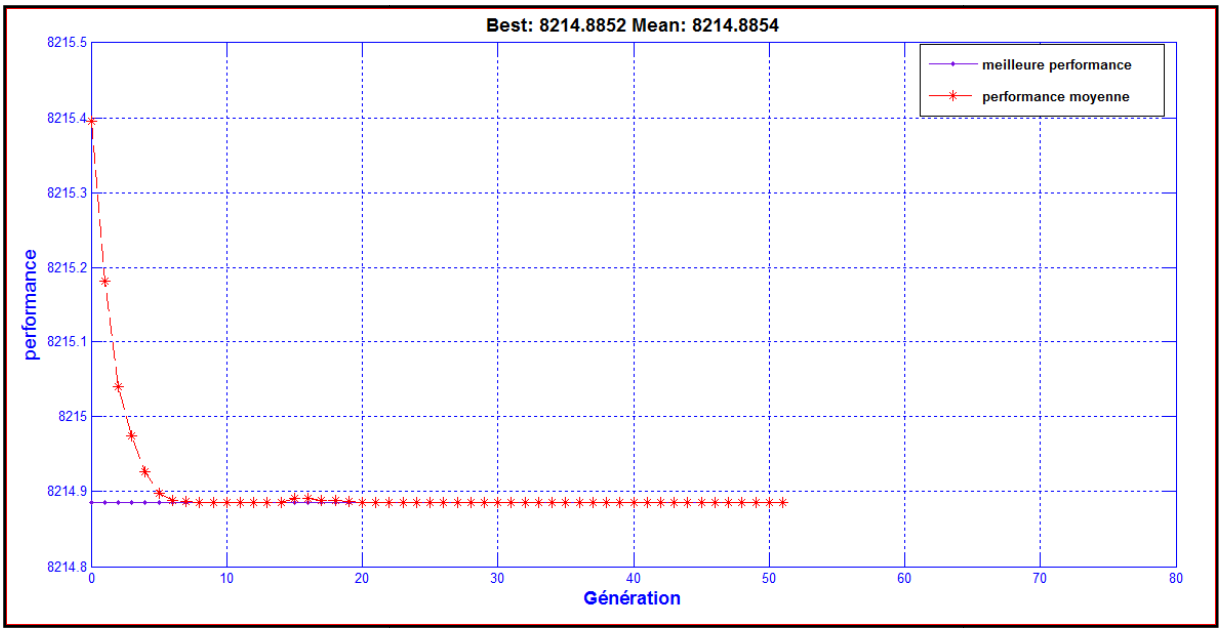


Figure4.22: Evolutions des performances en fonction des générations.

➢ Interprétation des résultats :

Pour 80 générations et une population de 200 individus par génération, les opérateurs du croisement et de mutation sont appliqués, générant de nouveaux individus.

Le meilleur individu pour le quel le minimum global est égal à 8214.8852

- En rouge : performances moyennes de la population.
- En bleu: meilleur performance.

La convergence est atteinte après environ de 52 générations mais d'excellente résultats sont déjà obtenu après 20 générations.

❖ La courbe inférieure montre la distance moyenne entre les individus à chaque génération, ce qui est une bonne mesure de la diversité d'une population.

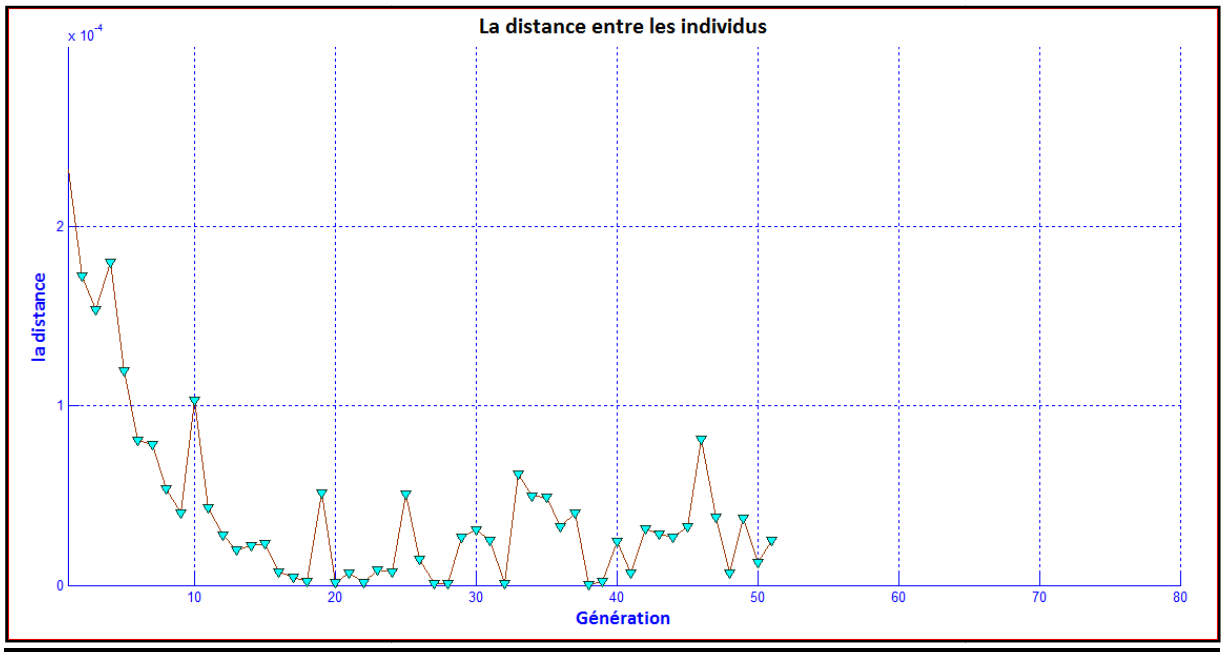


Figure 4.23: La distance entres les individus en fonction des générations.

Autrement dit, si la distance moyenne entre les individus est grande, la diversité est élevée, par contre, si la distance moyenne est faible, la diversité est faible.

De ce fait, si la diversité est trop élevée ou trop faible, l'algorithme génétique pourrait ne pas bien performer

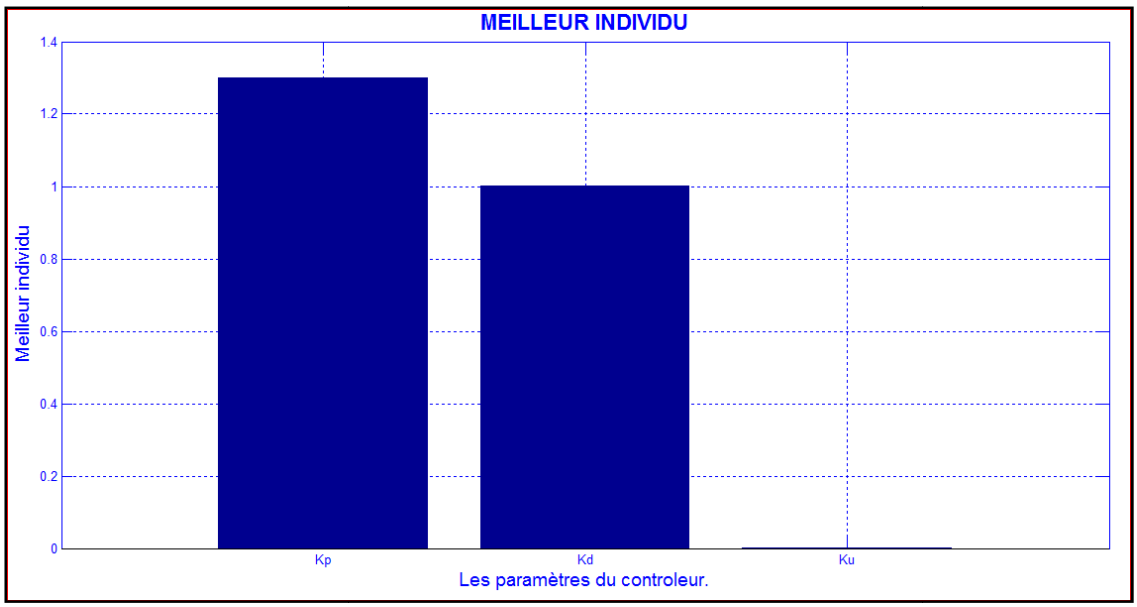


Figure 4.24: Le meilleur individu en fonction des paramètres (K_p, K_d, K_u).

Les valeurs obtenues du 2ème essai :

Paramètres1	K_p	K_d	K_u
Valeur	1.299980000000000	1.000004744468084	0.000010000000000
Fitness(fval)	8.214885167262059e+003		

Tableau 4.8 : résultats finals.

6. Conclusion :

Chacune des méthodes simulées s'avère applicable du moment qu'elles donnent des résultats satisfaisants quoi que les algorithmes génétique permettent de converger vers un minimum globale, cependant, l'utilisation de cette méthode reste délicate car de nombreux paramètres doivent être fixés :

- La taille de la population et le nombre de génération.
- Les probabilités de croisement et de mutation.
- La génération de la population initiale.
- Les différents opérateurs de sélection, croisement, mutation.

Les algorithmes génétiques fournissent des solutions proches de la solution optimale grâce à ses opérateurs, ils sont applicables à de nombreux problèmes, mais associés à d'autres méthodes classiques ils deviennent plus efficaces.

Pour ce qui est de « fminsearch », elle prend un vecteur de paramètres initiaux et s'arrête au premier minimum local trouvé, pour converger vers d'autres, il faut changer à chaque fois le point de départ.

Conclusion générale

Nous avons vu dans ce chapitre l'intérêt de la logique floue dans le domaine du contrôle de processus. Cette approche permet de tenir compte à la fois des connaissances d'un expert humain, de l'incertitude et de l'imprécision des données traitées par le contrôleur. Les variables linguistiques permettent de traiter ces deux informations initialement très différentes à l'aide d'un formalisme unique.

Pourtant, la conception d'un contrôleur flou n'est pas toujours chose aisée. Lorsqu'on utilise un contrôleur de type standard (par exemple un PID), on dispose de nombreux outils de synthèse permettant de choisir au mieux les paramètres du régulateur en fonction de la structure ou du modèle du système à commander. Malheureusement, la panoplie d'outils disponibles est beaucoup plus limitée dans le cas des contrôleurs flous.

Toutefois une approche globale de la conception de ce dernier est possible. En effet, nous avons présenté dans cette thèse une méthode de synthèse d'optimisation d'un contrôleur flou par un algorithme génétique qui optimise ses facteurs d'échelle, nous l'avons comparé à une autre méthode d'optimisation simplex qui est « fminsearch », les résultats obtenus durant les essais montrent l'efficacité de cette méthode qui assure le meilleur fonctionnement du contrôleur.

BIBLIOGRAPHIE

[1] : Amieur Toufik, «*Commande des Systèmes Non Linéaires par Mode Glissant Flou*» Mémoire de Fin D'étude Université Mohamed KHEIDER Biskra, Algérie, 2009.

[2] : livre, « *application de MATLAB 5 et SIMULINK 2* » par M.Mokhtari et M.Marie , édition springer, 1998.

[3]: K.M. Passino and S. Yurkovich, «*Fuzzy Control*» , Reading, MA:Addison-Wesley, 1998.

[4]: MOKEDDEM Diab**, «***Contrôle Flou des Processus Biotechnologiques à Base d'Algorithmes Génétiques*», thèse de doctorat, Université Ferhat Abbas DE SETIF, 2010.

[5] : Merabti Halim, «*Etude des systèmes flous à intervalle*», Mémoire de Fin D'étude Université MENTOURI de Constantine, 2008**.**

[6] : Samir MERADI, «*Conception d'un régulateur flou pour la commande vectorielle de la machine asynchrone*», Mémoire de Fin D'étude Université Mohamed KHIDER de Biskra, 2007.

[7] : Wei WU. «*Synthèse d'un contrôleur flou par Algorithme Génétique : Application au réglage dynamique des paramètres d'un système*», thèse de doctorat de l'Université de Lille1, 1998.

[8] : S.K. Tso and Y.H. Fung, «***Synthesis and Stability Analysis of Linguistic Fuzzy Controlled Systems***», Centre for Intelligent Design, Automation and Manufacturing City University of Hong Kong, HONG KONG.

[9]: Randy, L., Haupt, S. E. «*Practical genetic algorithms*». *John Wiley & Sons,* 2004.

[10] : Sylvie Galichet, « Contrôle flou : de l'interpolation numérique au codage de l'expertise », Université de Savoie, 2001.

[11]: M.Oulmahdi, «*Algorithmes évolutionnaires dans les systèmes de parole*», thèse de doctorat, Université Abderrahmane Mira de Bejaïa, 2011.

[12] : N.Lassouaoui, L.Hamami, N.Nouali, « *Les algorithmes génétiques application la segmentation d'image »*, Article, Ecole Nationale Polytechnique, laboratoire Signal & Communication, Algérie, 2004.

[13]: Gang, F. «*An Approach to adaptive control of fuzzy dynamic systems***»** IEEE Transactions on Fuzzy Systems*,* Vol.10, No.2, 268-275, 2002.

[14]: «*Design of Adaptive Fuzzy Logic Controller Based on Linguistic-Hedge Concepts and Genetic Algorithms***»** Bin-Da Liu, Senior Member, IEEE, Chuen-Yau Chen, Student Member, IEEE, and Ju-Ying Tsao

[15]: Herrera, F., Lozano, M., Verdegay, J.L. «*Turning fuzzy logic controllers by genetic algorithms***»**. International Journal of Approximate Reasoning, 12: 299–315, 1995.

[16] : Talbi Nesrine : «*Conception des Systèmes d'Inférence Floue par des Approches Hybrides,* a*pplication pour la Commande et la Modélisation des Systèmes Non linéaires* » Mémoire de Fin D'étude, Université de Constantine 1, 2014.

[17]: Ouahib GUENOUNOU , « *Méthodologie de conception de contrôleurs intelligents par l'approche génétique- application à un bioprocédé »* thède de doctorat, Université de Toulouse, 2009.

[18]: Dan SIMON, «*sum normal optimization of fuzzy membership functions»,* Cleveland State University, Department of Electrical and Computer Engineering, 1960.

[19]: Lagarias, JC, JA Reeds, MH Wright et PE wright, « *Propriétés de convergence de la Simplex Méthode Nelder-Mead en bas Dimensions* » SIAM Journal of Optimization. Vol.9, numéro 1, 1998, pp. 112-147.

[20] : SAHRAOUI Dallal «*Identification paramétrique d'un pendule inversé simple par un algorithme génétique»,* Mémoire de Fin D'étude, Université Mohamed Khider Biskra, 2014.

[21] :RENDERS Jean Michel « Algorithme génétique et réseaux de neurones»,1995.

[22] : Nadia. OA « *conception et réalisation d'un système de clustering basé sure les algorithmes génétiques* » exposé, Université des Sciences et de la Technologie Houari Boumedièn (USTHB), Algerie, 2014

[23] : www.mathworks.com

[24]: Piero.P.Bonissone and Yuto Chen, «*genetic algorithm for automated tuning of fuzzy controllers : a transportation application* », Information Technology Laboratory, USA.

[25]: Kazuo Tanaka and Hua O. Wang *« fussy control system design and analysis, A Linear Matrix Inequality Approach*», 2001 John Wiley & Sons, Inc.