République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Jijel

Faculté des Sc[illegible] de Technologie

Département d'Electronique

Projet de fin d'Etudes pour l'Obtention du Diplôme de Master II en Électronique

Option : Électronique et Analyse des Systèmes (EAS)

Thème

Conception et Optimisation d'un Système de Commande par les Algorithmes Evolutionnaires

Réalisé Par :

- *DJEDDAI Nawal*
- *BOUMANKAR Malika*

Encadré Par:

Pr. Abdelkrim BOUKABOU

Promotion Juin 2015

Remerciements

Nous remercions tout d'abord Dieu le tout puissant, qui nous a donné du courage et de la volonté, d'avoir réussit dans nos études.

Nos plus vifs remerciements sont adressés à notre encadreur ***Pr. Abdelkrim BOUKABOU*** *pour sa patience toute la durée du travail, sa gentillesse et ses encouragements.*

Nous exprimons notre vive reconnaissance à l'ensemble des membres du jury pour l'intérêt qu'ils ont porté à ce travail.

Nous adressions également nos remerciements les plus respectueux à toutes les personnes qui nous ont aidées à préparer ce travail et en particulier ***Dr. Ammar SOUKKOU,***

Enfin, nous remercions nos familles et tous nos proches pour leurs ***encouragements et leurs soutiens dans tous les instants, ainsi nos amies*** *et collègues de promotion, pour leur collaboration et pour les bons* ***moments partagés tout au long de notre formation.***

NAWAL & MALIKA

Dédicace

*Grâce à **Allah** voilà notre*

travail terminé et il est temps pour

moi de partager ma joie avec tous ceux qui

m'ont soutenu et encourager

A mes très chers parents

ce que je ne jamais remercier assez en modeste signe de reconnaissance pour leurs sacrifices, leur compréhension amour et leur tendresse, mon respect et mon profond amour,

que dieu les garde et protège.

A mes chers frères.

A mes chères sœurs.

*A tout la famille **Djeddai***

A mes amies chacune de son nom

A mes collègues de la promotion

*de Master en électronique **2015***

Nawal

Dédicace

*Grâce à **Allah** voilà notre*

travail terminé et il est temps pour

moi de partager ma joie avec tous ceux qui

m'ont soutenu et encourager

A mes très chers parents

ce que je ne jamais remercier assez en modeste signe de reconnaissance pour leurs sacrifices, leur compréhension amour et leur tendresse, mon respect et mon profond amour,

que dieu les garde et protège.

A mes chers frères.

A mes chères sœurs.

*A tout la famille **Boumankar***

A mes amies chacune de son nom

A mes collègues de la promotion

*de Master en électronique **2015***

Malika

TABLE DES MATIERES

Liste des figures

Liste des tableaux

INTRODUCTION GENERALE

L'optimisation est l'une des branches les plus importantes des mathématiques appliquées, et de nombreuses recherches, à la fois pratiques et théoriques, lui sont consacrées. Il existe deux grandes approches de l'optimisation. L'une est dite déterministe : les algorithmes de recherche utilisent toujours le même cheminement pour arriver à la solution, et on peut donc déterminer à l'avance les étapes de la recherche. L'autre est aléatoire : pour des conditions initiales données, l'algorithme ne suivra pas le même cheminement pour aller vers la solution, et peut même proposer différentes solutions. C'est vers cette seconde approche, que va s'orienter notre travail, et plus particulièrement vers un type bien précis d'algorithme de recherche aléatoire, les algorithmes du type évolutionnaires.

Les algorithmes génétiques font partie de cette famille, ils permettent d'explorer des domaines possédant de très nombreuses solutions, et leur efficacité pratique a été prouvée bien avant que les résultats de convergence théorique soient établis. Toutefois le choix des nombreux opérateurs génétiques, intervenant dans la mise en place de l'algorithme reste à l'appréciation de l'utilisateur, c'est pour cela qu'un domaine de recherche très actif est consacré à l'étude de ces derniers et à la mise en place de nouvelles techniques. Surtout que l'utilisation de ces algorithmes est souvent coûteuse en temps de calculs et les performances de ces algorithmes dépendent beaucoup des différents opérateurs génétiques.

Dans le cadre de ce mémoire, nous nous sommes intéressés à l'étude des algorithmes génétiques utilisant de nouveaux mécanismes inspirées de processus biologiques, tels que la transformation, la transposition génétique et la stratégie de la recherche de niche et leur incidence sur leurs performances de l'algorithme d'optimisation. Pour confirmer l'efficacité de ces nouveaux mécanismes, nous nous sommes intéressés aux problèmes d'optimisations des systèmes dynamiques non linéaires modélisés par des équations du type équations différentielles.

L'élaboration d'une loi de commande pour un procédé physique nécessite la prise en compte de certains paramètres tels que le suivi de la consigne, le rejet de la perturbation, une marge de robustesse vis-à-vis de certains paramètres de procédé à commander. La littérature propose une multitude de structure de commande. Chacune d'elle possède son application et également des propriétés (cas linéaire, cas non linéaire, procédé stable,...etc). Commander un processus, c'est déterminer les commandes à lui appliquer, de manière à assurer aux variables

d'états ou aux sorties qui nous intéressent un comportement précisé par un cahier des charges. En particulier, un régulateur PID ou correcteur PID (pour « proportionnel intégral dérivé ») est un organe de contrôle permettant d'effectuer une régulation en boucle fermée d'une grandeur physique d'un système industriel ou « procédé » (voir Automatique). C'est le régulateur le plus utilisé dans l'industrie, et il permet de régler un grand nombre de grandeurs physiques.

Ainsi, dans le cas d'un fonctionnement en mode de régulation (consigne fixe), il faut choisir des réglages permettant à la grandeur réglée de retourner dans un temps raisonnable à sa valeur de consigne. Dans le cas de fonctionnement de la boucle en mode d'asservissement (consigne variable), choisir des réglages permettant de limiter le ou les éventuels dépassements (over shoot) de la grandeur réglée. De ce fait, le choix des paramètres du contrôleur PID doit être fait d'une manière optimale. Pour cela, on adopte une procédure d'optimisation par algorithme génétique (AG).

Tout au long de ce mémoire, on s'intéressera à l'optimisation de la commande PID par algorithme génétique. Outre l'introduction et la conclusion générales, ce mémoire est organisé en trois chapitres répartis comme suit : Le premier chapitre est dédié à des notions fondamentales des algorithmes génétiques. Le deuxième chapitre est consacré à la commande PID par les algorithmes génétiques. On commence par donner un aperçu général sur les principaux critères et les domaines d'application de la commande PID ainsi que les critères de choix des paramètres PID. Le troisième chapitre proposera une étude détaillée sur trois cas pratiques, on commence par l'étude d'un circuit électrique, on va modéliser le système, et après sa linéarisation, on va appliquer la loi de commande pour minimiser un certain critère de performance afin d'obtenir une commande optimale. On va suivre les mêmes étapes pour le deuxième cas (un processus thermique) et on termine par l'application de la loi de commande sur un système réel de réglage de surtension dans un système de pompage. Enfin, on termine par une conclusion générale.

Chapitre I

LES ALGORITHMES EVOLUTIONNAIRES ET LES ALGORITHMES GENETIQUES

I.1 Introduction

La biologie et l'observation des phénomènes biologiques ont toujours été une source d'inspiration très riche pour les scientifiques.

L'idée d'évolution est très ancienne, plusieurs auteurs ont proposé des interprétations évolutionnistes du monde, à base de phénomènes d'adaptation au milieu et de lutte pour la vie. A cet effet des courants différents et variés ont marqué la recherche durant les quarante dernières années, et sont toujours le sujet de débats parfois très passionnés. Historiquement [1], les premières expériences en évolution artificielle proposées par Friedberg à la fin des années cinquante. L'idée consistait à faire évoluer des programmes informatiques, écrits en langage machine suivant un processus similaire au processus de l'évolution naturelle. Des parties de ces programmes étaient mutées par des variations aléatoires puis sélectionnées pour constituer de nouveaux programmes.

A la suite de ces premières expériences plusieurs chercheurs ont commencé à s'intéresser au sujet, et certains d'autre eux ont tenté d'adopter ce principe dans les années 60, qui consiste à s'inspirer des mécanismes de l'évolution naturelle et à utiliser le concept de populations d'individus ou solutions pour résoudre des problèmes du monde réel. Toute fois la faible puissance des machines de l'époque et des connaissances de la génétique naturelle n'a pas permis d'avoir de résultats convaincants. La mise en œuvre relativement aisée de ces algorithmes ainsi que les nombreux succès qu'elles ont obtenus ont contribué à leur développement spectaculaire ces vingt dernières années. Ce développement porte à la fois sur leur diffusion dans de très nombreux domaines d'application ainsi que sur l'étude et l'exploration des mécanismes d'évolution eux-mêmes. De nombreuses applications de ces méthodes concernent le domaine des systèmes de production (Pierreval, 2003) et notamment le domaine de l'ordonnancement (Portmann, 2001) [1, 2].

I. 2 Les algorithmes évolutionnaires (AE)

Les algorithmes évolutionnaires sont des méthodes stochastiques d'optimisation globale basées sur la théorie Darwinienne de l'évolution des espèces biologiques. Les origines des algorithmes évolutionnaires remontent à la fin des années 50, et depuis 1970, plusieurs méthodes évolutionnaires ont été proposées, principalement concernant les algorithmes génétiques, la programmation génétique et les stratégies d'évolution [3]. Toutes ces approches opèrent sur un ensemble de solutions candidates. Ils Utilisent à la fois les principes de la survie des individus les mieux adaptées et ceux de la propagation du patrimoine génétique qui

s'inspirent des mécanismes de sélection naturelle et des phénomènes génétiques tel que des **mécanismes d'évolution de la nature : croisements, mutations, sélections.**

Pour utiliser ces algorithmes, il faut disposer d'une population d'individus. Chaque individu dispose d'une chaîne chromosomique qui dirige son comportement. Cette chaîne s'apparente à l'ADN dans les organismes vivants. Comme dans les systèmes naturels, des **croisements sont réalisés périodiquement et permettent à l'algorithme de créer la génération** suivante d'individus, ainsi des mutations sont aussi effectuées. Ces mutations évitent à l'ensemble de la population de converger vers une solution qui ne serait pas optimale. Il existe **plusieurs types de ces algorithmes mais l'idée essentielle est la même : simuler l'évolution** d'une population dans un espace de recherche à l'aide de trois opérateurs: sélection, croisement, mutation [4,5].

On distingue quatre grandes familles d'algorithmes évolutionnaires [6]:

- **Les algorithmes génétiques (AG) :**

Les algorithmes génétiques sont des techniques de recherche inspires par l'évolution biologique des espaces. Introduits par J.H. Holland au début des années 1970, ils ont d'abord **suscité un intérêt limité, du fait de leur important coût d'exécution. Ils connaissent, depuis** les années 1990, un développement considérable, notamment suite l'apparition des architectures massivement parallèles, qui exploitent leur (parallélisme intrinsèque).

- **Les Stratégies d'Evolution (SE)**

Les Stratégies d'Evolution (SE) développés par (Rechenberg et H.P. Schwefel, 1965, Berlin). Elles ont été développées pour résoudre des problèmes d'optimisation à **variables réelles posés au milieu industriels et pour les quels n'existe pas de fonction objectif** analytique; le contexte étant l'optimisation paramétrique. Ce sont les meilleurs algorithmes pour les problèmes purement numériques. Ce modèle de stratégies d'évolution utilise le **principe de mutation sur les réels du modèle de la programmation évolutive avec un taux de** mutation plus grand. Cette augmentation peut être interprétée par le fait que si la proportion de mutation réussie est élevée, l'espace de recherche exploré est limité autour d'un optimum local; il faut donc diversifier la population en augmentant le taux de mutation. Ces approches utilisent un opérateur de sélection de type déterministe, les solutions dont la fonction d'adaptation est mauvaise sont éliminées de la population. En outre, dans le modèle originel, les populations des parents et de leurs descendants sont généralement de taille différente.

❖ **Programmation Evolutionnaire (PE)**

Programmation Evolutionnaire (PE) développés par (L.J. Fogel, 1964 et D.B.Fogel, 1991, 1995, Californie, USA). Ce modèle évolutionnaire accentue l'utilisation de la mutation et n'utilise pas dans sa version originale la recombinaison des individus par croisement. **Développé à l'origine pour l'évolution des automates à état fini, ce modèle est souvent** appliqué à la résolution de problèmes d'optimisation à variables réelles dans d'espaces de recherche très variés. L'idée consiste à faire subir des mutations importantes aux mauvais individus et des mutations faibles aux bons individus. L'opérateur de sélection est de type **probabiliste. Il est à noter que la représentation des individus n'a pas une forme spécifique** de génome telle que dans une représentation linéaire de type chaîne binaire par exemple.

❖ **Programmation Génétique (PG)**

Programmation Génétique (PG) développés par (J. Koza, 1990, Californie, USA). Apparue initialement comme une extension du modèle d'apprentissage des algorithmes génétiques, ils sont devenus une branche à part entière (conférence, journal). La (*PG*) permet **de générer des fonctions informatiques à partir des principes évolutionnaires, la population** est un ensemble de codes de base de programmes informatiques. La spécificité des (*PG*) est l'espace de recherche, Les individus formant une population sont donc des programmes **candidats à la résolution d'un problème. Ces programmes sont exprimés sous la forme** d'arbres sur lesquels les opérateurs génétiques produisent des transformations en vue d'obtenir un programme qui satisfaisant la résolution du problème choisi. Les(*PG*) cherchent à atteindre un des vieux rêves des programmeurs, faire écrire le programme par un autre **programme.**

I.2.1 Principes de base des algorithmes évolutionnaires (AE)

En général, un AE est caractérisé par trois faits :

1. Un ensemble de solutions candidates est maintenu,
2. Celui-subi un processus de sélection
3. Est manipulé par des opérateurs génétiques, le plus souvent le croisement et la mutation.

Par analogie avec l'évolution naturelle, les solutions candidates sont appelées individus, et l'ensemble des solutions candidates est appelé la population. Chaque individu représente une solution possible, c'est-à-dire, dans un sens, représente un vecteur de décision du problème. Cependant, un individu n'est pas à proprement parlé un vecteur de décision mais

encode plutôt un vecteur de décision selon une structure appropriée. Cette structure est supposée être un vecteur, un vecteur de bits ou bien un vecteur de réel, bien que d'autres structures comme les arbres [5] puissent également être utilisées ; l'ensemble de tous vecteurs possibles constitue l'espace des individus *I*. Dans cette terminologie, la population est un ensemble de vecteur $i \in I$, pour être plus précis, un ensemble multiple de vecteurs puisque il peut contenir plusieurs individus identiques.

Dans le processus de sélection, qui peut être soit stochastique, soit totalement déterministe, les individus de faible qualité sont supprimés de la population, tandis que les individus de haute qualité sont reproduits. Le but est de diriger la recherche sur certaines portions de l'espace de recherche et d'augmenter la qualité moyenne à l'intérieur de la population. La qualité d'un individu est représentée par une valeur numérique, également appelée fitness. Notions que puisque la qualité est relative à la fonction d'objectifs et aux contraintes, un individu doit d'abord être décodé avant que sa fitness puisse être calculée. Soit un individu $i \in I$. Une fonction de mappin g_m encapsule l'algorithme de décodage pour obtenir le vecteur de décision $x=m(i)$ à partir de i Appliquer f à x fournie le vecteur d'objectifs correspondant sur la base duquel une valeur de fitness est assignée à *i*.

Le croisement et la mutation ont pour but la génération de nouveaux candidats à l'intérieur de l'espace de recherche par la variation des solutions existantes. L'opérateur de croisement prend un certain nombre de parents et crée un certain nombre d'enfants par recombinaison des parents et crée un certain nombre d'enfants par recombinaison des parents. Pour imiter la nature stochastique de l'évolution, une probabilité de croisement est associée à cet opérateur. En contraste, l'opérateur de mutation modifie les individus en changeant certaines portions du vecteur de décision associé selon une probabilité de mutation donnée. Le croisement et la mutation s'appliquent à des individus, c'est à dire dans l'espace de recherche, et non sur les vecteurs de décision décodés.

L'évolution naturelle est simulée par un processus itératif. D'abord, une population initiale est créée de manière aléatoire. Cette population est un point de départ du processus d'évolution. Ensuite une boucle consistant les étapes d'évolutions (assignation des valeurs de fitness aux individus), de sélection, de croisement et/ou de mutation est exécutée un certain nombre de fois. Chaque itération est appelée une *génération*, et souvent un nombre maximum prédéfini de génération sert de critère de terminaison de la boucle. Mais d'autres conditions, stagnation dans la population ou existence d'individus de la valeur de fitness suffisante, peuvent être utilisées pour stopper l'exécution.

I.2.2. La boucle évolutionnaire

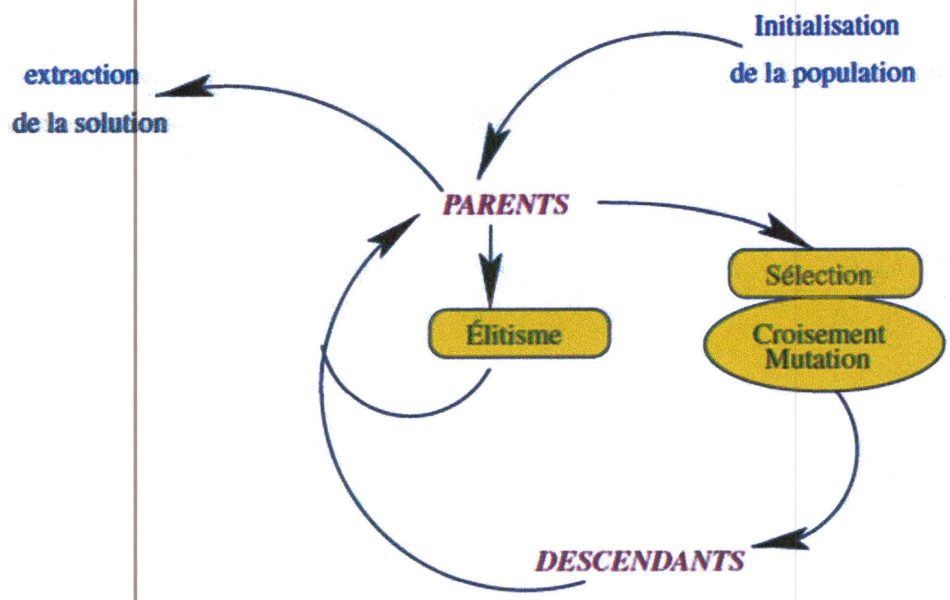


Figure I. 1: *Organigramme de l'algorithme évolutionnaire simple.*

Le premier point est une boucle générationnelle de populations d'individus (représentés sous forme discrète ou continue, à l'aide de chromosomes ou gènes) correspondant chacun à une solution au problème considéré [8, 9, 10]. Cette boucle est représentée dans la figure I.1 et ses étapes principales sont les suivantes.

a) L'initialisation

Elle est usuellement aléatoire (diverses stratégies sont d'ailleurs envisageables pour échantillonner correctement un espace de recherche complexe ou de grande dimensionnalité). C'est là que l'on peut injecter la ou les solutions initiales au problème que l'on a pu obtenir par ailleurs (par exemple à l'aide d'autres techniques de résolution). Si l'initialisation a théoriquement peu d'importance (on converge en limite toujours vers l'optimum global), on constate expérimentalement que cette étape influence énormément la variance des résultats (et aussi la rapidité d'obtention de ceux-ci). Il est ainsi souvent très avantageux d'injecter le maximum de connaissances *a priori* sur le problème par le biais de l'initialisation : placer dans la population initiale des individus que l'on sait proches par différents aspects de l'optimum recherché ne peut qu'accélérer la convergence de l'algorithme.

b) La sélection

Cet opérateur a pour rôle de détecter quels individus de la population courante seront autorisés à se reproduire (les « parents»).La sélection est fondée sur la qualité des individus, estimée à l'aide d'une fonction, nommée « fitness», « fonction d'évaluation», ou encore « **performance.» Dans le schéma canonique de l'AG «à la Goldberg »,** 2 parents donnent 2 enfants, ainsi on sélectionne un nombre de parents égal au nombre d'enfants désirés, mais évidemment bien d'autres schémas moins conventionnels peuvent être programmés (2 parents pour 1 enfant, *n* parents pour *p* enfant). Le paramètre principal de cette étape de sélection est **ce que l'on appelle la *pression sélective* qui correspond globalement au quotient de la** probabilité de sélection du meilleur individu sur la probabilité de sélection de l'individu moyen de la population courante. Ce paramètre, comme nous le verrons plus loin gère la **rapidité de concentration de la population autour de son meilleur individu.**

c) La reproduction

Les parents sélectionnés sont utilisés pour générer des descendants. Les deux opérations principales sont le croisement, qui combine les gènes de 2 parents, et la mutation qui consiste **en une légère perturbation du génome. Ces opérations sont appliquées aléatoirement, à l'aide** de deux paramètres, la probabilité de croisement *pc* et la probabilité de mutation *pm*. Ces probabilités sont des paramètres très importants, qui influent de façon considérable sur la **qualité des résultats globaux (convergence et qualité des résultats).**

d) L'évaluation

Cette étape consiste à calculer (ou estimer) la qualité des individus nouvellement créés. C'est là, et uniquement là qu'intervient la fonction à optimiser. Aucune hypothèse n'est faite **sur la fonction elle-même, excepté le fait qu'elle doit servir de base au processus de sélection** (elle doit pouvoir permettre de définir une probabilité ou au minimum un ordonnancement des solutions).

e) Le remplacement

Le remplacement gère la constitution de la génération n+1. Ila été prouvé (expérimentalement et théoriquement) que la stratégie simpliste qui consiste à remplacer l'ensemble des parents par tous les descendants produits est assez in efficace pour des applications d'optimisation. Le maintien d'un certain taux d'élitisme est nécessaire, tout simplement pour ne pas perdre la mémoire des bons individus visités. Les stratégies usuelles

de remplacement consistent à transmettre directement un pourcentage des meilleurs individus de la population courante dans la population suivante [11], par exemple, (emploie un paramètre qui gère le pourcentage de renouvellement de la population, le « Génération gap »). Cette proportion de remplacement est un paramètre essentiel du comportement de convergence de l'algorithme évolutionnaire. Par exemple les stratégies d'évolution (μ, λ) et $(\mu + \lambda)$ [12] [13] [14] signifient que λ descendants sont générés à partir d'une population de μ individus. La stratégie consiste à gérer l'élitisme par le biais de la différence $\mu - \lambda$ (on garde les $\mu - \lambda$ meilleurs individus de la population courante et on complète par les λ descendants), tandis que la stratégie «+» est une version plus adaptative dans sa gestion de l'élitisme : à partir d'un ensemble intermédiaire de taille $\mu + \lambda$ constitué des μ individus de la population courante et des λ descendants, on sélectionne les μ meilleurs individus de la génération suivante.

f) L'arrêt

Stopper le processus au bon moment est essentiel du point de vue pratique. Si l'on a peu ou pas d'informations sur la valeur cible de l'optimum recherché (ce qui autorise un arrêt dès que cette valeur est atteinte par le meilleur individu de la population courante), il est délicat de savoir quand arrêter l'évolution. En l'absence de toute information, une stratégie couramment employée consiste à stopper l'algorithme dès qu'un nombre maximal d'itérations est atteint. Il est aussi possible de tester la dispersion de la population.

I.2.3 Les outils évolutionnaires de base des AE

Les éléments d'un algorithme évolutionnaire « canonique » peuvent être décrits simplement, mais ce que nous présentons ci-dessous doit être vu comme une « recette ». Les applications efficaces à base d'algorithmes évolutionnaires sont évidemment plus complexes, le problème essentiel étant d'adapter, de créer même, les opérateurs répondant aux spécificités du problème. De même qu'une recette d'un livre de cuisine nécessite d'être adaptée avec finesse au matériel et ingrédients disponibles, aux goûts des convives, pour être vraiment appréciée

I.2.4 Application des algorithmes évolutionnaires

Les algorithmes évolutionnaires sont des algorithmes d'optimisations stochastiques, qui consistent à faire évoluer artificiellement un ensemble de solutions potentielles à un problème donné. Cette évolution qui a pour but la découverte des optima de la fonction à optimiser. Le

principe général de ces algorithmes, consiste donc à faire évoluer artificiellement un ensemble de solutions potentielles de manière à favorisé les meilleures solutions; ce qui correspond à la recherche du maximum d'une fonction. Aujourd'hui les algorithmes évolutionnaires sont très utilisés pour la résolution de problème d'optimisation ils sont reconnus comme des outils très efficaces, plus particulièrement lorsque les problèmes à résoudre sont compliqués et que la fonction à optimiser n'est pas régulière ou que ses dérivées soient inaccessibles, mal conditionnées ou complexes à calculer, et que les méthodes d'optimisation classiques soient totalement inadaptés [15].

Le succès pratique de ces méthodes vient du fait qu'elles représentent des outils d'optimisation adaptés à des fonctions, des problèmes difficiles, complexes, irréguliers. Le mécanisme évolutionnaire a cependant un coût calculatoire (c'est une méthode itérative, une recherche par tâtonnement, aléatoire dirigé) qui peut devenir important. Ces méthodes sont en fait complémentaires des méthodes d'optimisation plus standard comme les méthodes d'optimisation déterministes qui font le plus souvent des hypothèses de régularité sur les fonctions à optimiser [16].

Le champ d'application des algorithmes évolutionnaires est en fait très large, il vade applications réelles complexes comme le contrôle du flux de pipelines de gaz, le routage aérien ou la planification des trajectoires de robots, à des problèmes plus théoriques et combinatoires, en modélisation économique, en finance, en commande de processus et pour l'apprentissage. On peut cependant dire que l'intérêt et l'efficacité des algorithmes évolutionnaires ont été globalement prouvés d'un point de vue théorique [3].

I.2.5 Représentations et opérateurs génétiques

Les opérateurs génétiques dépendent directement du choix de la représentation, c'est en fait cet aspect qui distingue les divers courants d'algorithmes évolutionnaires (algorithmes génétiques, stratégies d'évolution, programmation génétique, évolution grammaticale). Nous présentons rapidement ci-dessous les représentations, les opérateurs, les stratégies de sélection et de remplacement les plus classiques, mais on pourra trouver dans la littérature bien d'autres déclinaisons de ces composantes dans des espaces de recherche non standard, en fonction des différentes applications (espaces de listes, de graphes).

I.2.5.1 Algorithmes Génétiques et représentation discrète

Les algorithmes génétiques sont initialement fondés sur l'emploi d'une représentation binaire des solutions, assez rapidement étendue ensuite à une représentation discrète3.Chaque individu de la population est représenté par une chaîne de longueur fixe, dont les éléments (gènes ou allèles) sont choisis dans un alphabet fini [17].

D'un point de vue théorique, il a été prouvé qu'un AG converge vers l'optimum global de son espace de recherche pour une taille de population finie, et pour une probabilité de croisement fixée, si sa probabilité de mutation *pm(k)* décroît à chaque génération en respectant la borne minimale [19] : $Pm(k) \geq 1/2 * k^{-1/M*L}$, où M est la taille de la population, L est la longueur des chromosomes et k le numéro de *génération*[4]. En pratique, une décroissance (plus rapide que le modèle théorique) de *pm* permet d'améliorer l'efficacité de l'AG en permettant dans les premières générations une exploration large de l'espace de recherche, puis une convergence plus rapide lors des dernières générations de l'AG (ce qui correspond à une variation graduelle des capacités d'exploration et d'exploitation de l'algorithme).

I.2.5.2 Stratégies d'évolution et représentation continue

La représentation continue, ou représentation réelle, est historiquement liée aux approches de type stratégies d'évolution, elle a été ensuite étendue aux algorithmes génétiques. Dans cette approche, la recherche s'effectue dans Rn ou une partie de celui-ci. Les opérateurs génétiques associés à cette représentation sont soit des extensions continues des opérateurs discrets, soit directement des opérateurs continus.

Les croisements discrets consistent à mélanger les gènes réels d'un chromosome, sans toucher à leur contenu, on peut ainsi adapter directement les opérateurs de croisement binaires précédents, par exemple les croisements à un point, plusieurs points ou les croisements uniformes.

L'avantage de la représentation continue est certainement mieux exploitée avec des opérateurs spécialisés, dits croisements continus qui mélangent plus intimement les composantes des vecteurs parents pour fabriquer de nouveaux individus : *croisement barycentrique*, encore appelé croisement intermédiaire ou arithmétique, qui produit un descendant $x\_$ à partir d'un couple de vecteurs (x, y) de Rn grâce au tirage aléatoire d'une constante α, usuellement choisie de façon uniforme dans [0 1]ou $[-\varepsilon, 1 + \varepsilon]$ pour une version étendue du croisement (BLX-ε),tel que :

$$\forall i \in 1,.., n, x_i' = ax_i + (1 - \alpha) y_i$$

La constante α peut être tirée une fois pour toute pour l'ensemble des coordonnées de $x\_$, ou tirée indépendamment pour chacune de ses coordonnées (variante dite « uniforme» de ce croisement).

La généralisation de ce croisement à un croisement à plus de 2 parents, voire même à l'ensemble de la population (croisement« global ») est assez directe [20].

Beaucoup d'opérateurs de mutation ont été proposés et expérimentés pour la représentation réelle, nous donnons ci-dessous les plus classiques.

- La mutation Gaussienne consiste à ajouter un bruit Gaussien aux composantes du vecteur-individu concerné, ce qui implique l'ajustement d'un paramètre supplémentaire σ, la déviation standard de ce bruit :

$$\forall i \in 1,..., n, x_i' = xi + N(0, \sigma)$$

L'ajustement de σ est relativement complexe (trop petit, il ralentit l'évolution, trop grand, il perturbe la convergence de l'AE), de nombreuses stratégies ont été proposées, consistant à rendre ce paramètre variable au cours de l'évolution, soit en fonction du temps, de la valeur de fitness, dépendant des axes de l'espace de recherche (mutations non isotropes), ou encore auto-adaptatif, comme ci-après. Des études ont aussi été conduites sur l'emploi de bruits non Gaussiens.

- La mutation Log-normale auto-adaptative est une des grandes innovations des stratégies évolutionnaires, et consiste à faire gérer le paramètre σ directement par l'AE, en l'intégrant au code génétique. Les individus sont donc des vecteurs (x, σ), et la mutation se traduit simplement comme suit :

$$\forall i \in 1,..., n, \sigma_i' = \sigma_i * exp(N(0, \tau))$$

Et

$$x_i' = xi + N(0, \sigma_i')$$

De nombreux travaux théoriques et expérimentaux ont montré l'intérêt et la puissance de ces méthodes auto-adaptatives.

- La mutation uniforme, inspirée des approches AG consiste à tirer uniformément la nouvelle valeur de la composante x_i de l'individu x dans un intervalle Min_i, Max_i, c'est un opérateur

plus « brutal », mais qui peut être efficace malgré tout lorsqu'il convient de maintenir une bonne diversité (en complément, avec d'autres opérateurs de mutation, par exemple).

- Il existe de nombreuses autres mutations spécialisées, par exemple pour le traitement des contraintes, particulièrement utiles lorsque les solutions recherchées se trouvent près des limites de l'espace contraint. Une excellente revue des divers opérateurs génétiques se trouve dans [21].

I.2.5.3 Programmation génétique, et représentation par arbre

La programmation génétique (PG) correspond à une représentation de structures de longueurs variables sous forme d'arbres.

La GP a été imaginé à l'origine pour manipuler des programmes codés en LISP, dans le but créé des programmes qui puissent résoudre des problèmes pour lesquels ils n'ont pas été explicitement programmés. La richesse de la représentation arborescente de taille variable est l'une des clés du succès de ce courant dans le milieu évolutionnaire, en effet beaucoup de problèmes d'optimisation, de commande ou de contrôle peuvent se formuler comme un problème d'induction de programme [22].

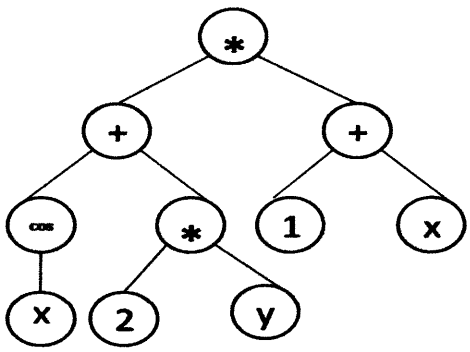


Figure I. 2 : *Exemple d'une représentation arborescente de la fonction1 ((Cos(x) + 2 * y) * (1 + x)).*

Un algorithme de PG explore un espace de programmes récursivement composés à partir d'éléments d'ensembles de fonctions, de variables et de terminaisons (données, constantes). Les individus de la population sont des programmes qui, lorsqu'ils sont exécutés, produisent les solutions au problème posé. Les croisements sont le plus souvent des échanges de sous arbres. Les mutations sont plus complexes, on en distingue plusieurs suivant leur action sur la structure du génome.

- suppression/ajout de nœud.

- modification de la fonction d'un nœud en conservant son arité.
- mutation des constantes (valeurs continues), par exemple par ajout d'un bruit Gaussien,
- mutation des variables discrètes, par permutation ou tirage aléatoire uniforme dans un ensemble de valeurs.

Les applications de la programmation génétique sont très nombreuses, par exemple [22] :
- en contrôle optimal : exemple du contrôle par un programme de la force appliquée sur un chariot, de façon à lui faire atteindre un point donné en un minimum de temps.
- en planification de trajectoires et d'actions en robotique : exemple de la fourmi artificielle qui doit trouver de la nourriture le long d'un chemin irrégulier.
- pour la régression symbolique : faire évoluer une expression mathématique (c'est-à-dire un programme) de façon à modéliser au mieux un ensemble fini de données.

I.3 L'algorithme génétique

I.3.1 Introduction

L'algorithme génétique (AG) est un algorithme de recherche basé sur les mécanismes de la sélection naturelle et de la génétique. Il combine une stratégie de "survie des plus forts" avec un échange d'information aléatoire mais structuré. Pour un problème pour lequel une solution est inconnue, un ensemble de solutions possibles est créé aléatoirement. On appelle cet ensemble la population. Les caractéristiques (ou variables à déterminer) sont alors utilisées dans des séquences de gènes qui seront combinées avec d'autres gènes pour former des chromosomes et par après des individus. Chaque solution est associée à un individu, et cet individu est évalué et classifié selon sa ressemblance avec la meilleure, mais encore inconnue, solution au problème [23]. Il peut être démontré qu'en utilisant un processus de sélection naturelle inspiré de Darwin, cette méthode convergera graduellement à une solution.

I.3.2 Optimisation par les algorithmes génétiques :

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 [25].

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.

I.3.3 Principe des algorithmes génétiques

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

Nous avons représenté en figure suivante le principe d'un algorithme génétique [26].

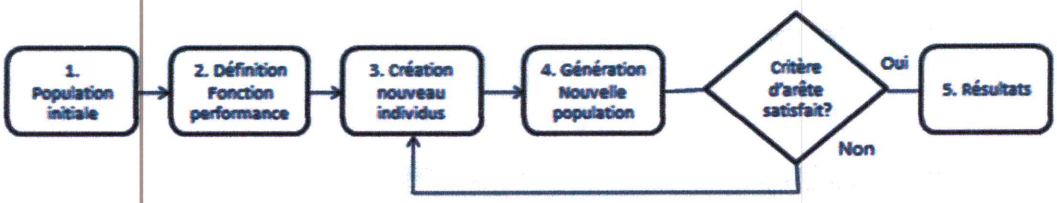


Figure I.3 : *étapes de l'algorithme génétique*

I.3.4 Mécanismes de fonctionnement d'un (AG)

I.3.4.1 Génération de la population initiale

La première étape consiste à construire une population initiale constituée des N individus. Chaque individu représente une architecture particulière d'un stratifié, et à l'image du monde de la génétique, ces caractéristiques sont codées sous la forme de chromosomes. Une constitution spécifique, nommée phénotype, correspond à un arrangement de gènes, ou degrés de liberté. Chaque individu possède un nombre de gènes égale à deux fois le nombre de couches TV du stratifié, un gène correspond au matériau et un second correspond à la porosité de la couche. Le regroupement de gènes reliés aux matériaux se nomme chromosome de matériaux et similairement, l'ensemble des gènes de porosité se nomme chromosome de porosité. Ces gènes sont encodés grâce à une suite de nombres binaires. Par conséquent, l'alphabet des gènes comprend deux valeurs, soit 0 soit 1. Le domaine associé à une variable est donc fractionné en 2^{N} *bit*, où Nb_n représente le nombre de bits associé à un gène.

Une fois que l'ensemble des variables est défini, l'étape suivante consiste à décoder les chromosomes afin de connaître l'architecture poreuse des différents individus [27].

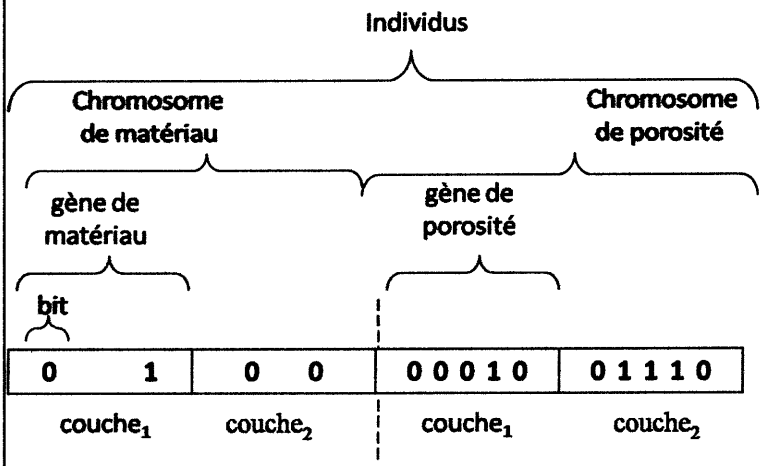


Figure I. 4 : *Représentation binaire d'un individu, N=2*

Selon cette modélisation, un stratifié constitué de dix couches (N=10) peut acquérir 2^{70} architectures différentes. Dans ce contexte, l'emploi d'un AG s'avère être un outil puissant pour trouver efficacement une solution à un problème d'optimisation.

I.3.4.2 La fonction fitness

En général, chaque programme possède une valeur numérique (fonction fitness) qui détermine sa qualité par rapport au problème pose. La fitness peut se mesurer de plusieurs

manière : une mesure d'*erreur* pour les problèmes de régression symbolique ou on calcule la différence entre le résultat obtenu par le programme et la valeur attendue, un temps d'exécution pour ramener le système dans un état voulu (cela peut faire référence au cout d'argent ou cout d'énergie), ou une mesure de précision dans les problème de classification, ou encore la mesure d'une conformité d'une structure au critère du design donne par l'utilisateur, ou bien la mesure du gain qu'un programme peut faire dans un scenario de jeu, Le fitness d'un individu s'effectue en évaluant son exécution.

Le processus d'interprétation d'un arbre passe par l'évaluation des nœuds de cet arbre, dans un ordre qui garantit qu'aucun nœud ne sera exécuté avant de connaitre les valeurs de ces enfants ou arguments s'il en a. Ceci est généralement réalise entra versant l'arbre de manière récursive, il est également possible de commencer par des feuilles au nœud racine. Les deux méthodes sont équivalentes dans le cas ou aucun nœud de l'arbre n'a d'effet indésirable.

I.3.4.3 Codage et décodage des variables

- **Codage**

Le codage est une partie très importante des algorithmes génétiques. Il permet de représente l'individu sous la forme d'un chromosome. Ce chromosome est constitué de gènes qui prennent des valeurs dans un alphabet binaire ou non. La figure ci-dessous donne les 5 niveaux d'organisation dans un AG.

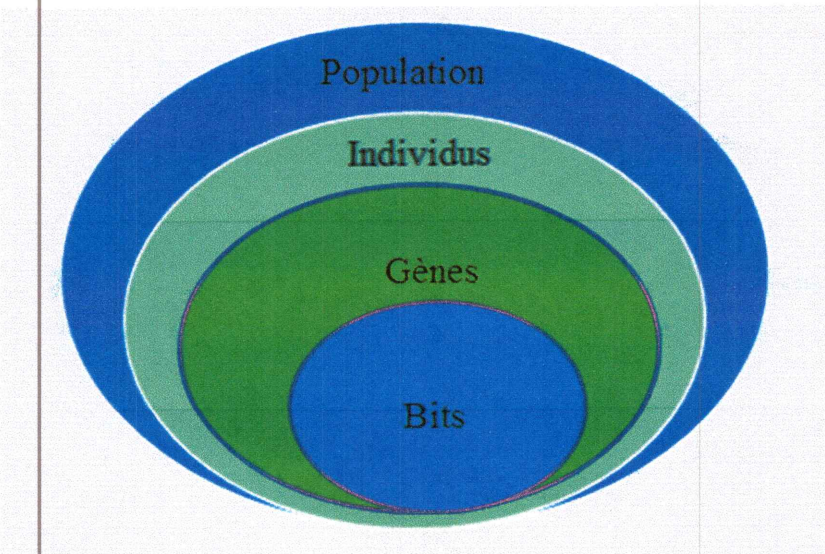


Figure I.5 *: Les niveaux d'organisation dans un AG*

a) Codage binaire

Goldberg et Holland ont démontré qu'il est idéal de représenter le chromosome en une chaîne binaire. C'est pourquoi les AG utilisent généralement cette représentation Les individus sont représentés sous forme de chaînes de bits contenant toute l'information **nécessaire à la description d'un point dans l'espace. Ce type de codage a pour intérêt de** permettre la création d'opérateurs de croisement et de mutation simples [28].

A chaque variable d'optimisation x_i correspond un ***gène.*** Un ***chromosome*** sera donc un ensemble de gènes. Chaque point est représenté par un ***individu*** doté d'un ***génotype*** constitué **d'un ou de plusieurs chromosomes. La *population* est un ensemble de *N* individus qui vont** évoluer d'une génération à une autre. Du point de vue informatique, un gène est un entier long de **K** bits. Un chromosome est un tableau de gènes. Un individu est un tableau de **chromosomes. La population est un tableau d'individus [28]. Dans le codage binaire le gène** est codé par un caractère binaire, 0 ou 1. C'est le plus courant et celui qui a été employé lors de la première application des algorithmes génétiques. Un des avantages du codage binaire est que l'on peut ainsi facilement coder toutes sortes d'objets : des réels, des entiers, des valeurs **booléennes, des chaînes de caractères... Cela nécessite simplement l'usage de fonctions de** codage et décodage pour passer d'une représentation à l'autre [25].

Pour chaque paramètre xi situé dans l'intervalle $[X_{i\,min}, X_{i\,max}]$, on associe une chaîne binaire $b_0 b_1 \ldots b_{lxi}$ définie sur lxi bits. A cette chaîne correspond une valeur entière naturelle [25]:

$$N(X_i) = \sum_{i=1}^{lxi-1} 2^{lxi-i-1} * b_i \qquad (I.1)$$

Le paramètre réel X_i de l'espace de recherche relatif à $N(X_i)$ est obtenu par interpolation linéaire.

$$x_i = x_{i\,min} + \frac{X_{i\,min} - X_{i\,max}}{2^{lxi}-1} * N(X_i) \qquad (I.2)$$

La longueur totale du chromosome est donnée par :

$$L = \sum_{i=1}^{m} l_{xi} \qquad (I.3)$$

m : le nombre des paramètres

Exemples

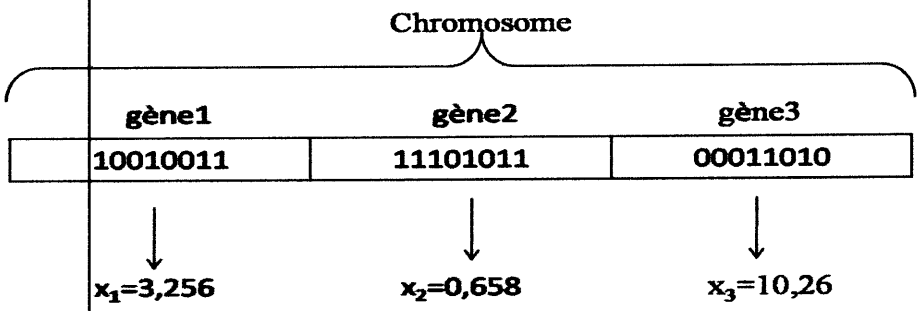
Chromosome A:	1	0	1	1	0	0	1	1
Chromosome B:	1	0	0	0	1	0	0	1

b) Codage réel

Davis, Janikow et Michalewicz ont effectué une comparaison entre la représentation binaire et la représentation réelle. Ces auteurs ont trouvé que la représentation réelle donne de meilleurs résultats d'après leur problème à résoudre. Dans ce codage le génome est un vecteur réel et l'espace de recherche est un sous-ensemble d'IR. Cette représentation est aujourd'hui très utilisée dans les problèmes d'optimisation cardans de nombreuses applications du monde réel, ces problèmes sont naturellement formulés sous forme paramétrique i.e. Les premiers travaux qui ont utilisé ce type de représentation ont été ceux de Rechenberg et de Schwefel quand ils ont introduit les stratégies d'évolution [1] [25].

Le codage réel peut-être utile notamment dans le cas où l'on recherche le maximum d'une fonction réelle.

Exemples



c) Codage de Gray

Dans le cas d'un codage binaire on utilise souvent la "distance de Hemming" entre les codages binaires de deux nombres réels proches, comme mesure de la dis similarité entre deux éléments de population, cette mesure compte les différences de bits de même rang de ces deux séquences. Et c'est là que le codage binaire commence à montrer ses limites. En effet, deux éléments voisins en termes de distance de Hemming ne codent pas nécessairement deux éléments proches dans l'espace de recherche. Cet inconvénient peut être évité en utilisant un "codage de Gray" : le codage de Gray est un codage qui a comme propriété qu'entre un élément n et un élément n+1, donc voisin dans l'espace de recherche, un seul bit diffère [29].

- **Décodage**

Sans indication sur la structure et la signification des chaînes de bits, on ne peut pas associer une caractéristique physique du stratifié telle une porosité à un gène. Par conséquent,

pour décoder un chromosome on doit connaître le domaine et le système numéral de chaque variable. Cette étape s'effectue à la suite de la création d'une nouvelle population, comme l'indique la figure I.6qui peut prendre une valeur comprise entre 0 et 1, est codée sur 5 bits et est traitée comme une variable discrète. Le fractionnement du domaine est linéaire ce qui implique que la différence entre deux valeurs de porosité successives est constante. Ainsi, la précision sur cette variable est fonction du nombre de bits utilisé pour coder un gène. Dans ce cas, la résolution est de $1/(2)^{5}$, soit environ 3%. Le système numéral employé pour représenter ce type de gêne est le code Gray. Ce choix est particulièrement approprié aux algorithmes génétiques, car les nombres d'un même voisinage diffèrent très peu dans leur représentation binaire. Cette représentation permet d'améliorer la convergence et le temps de calcul de l'AG soumis à une optimisation globale sur des paramètres continus [30].

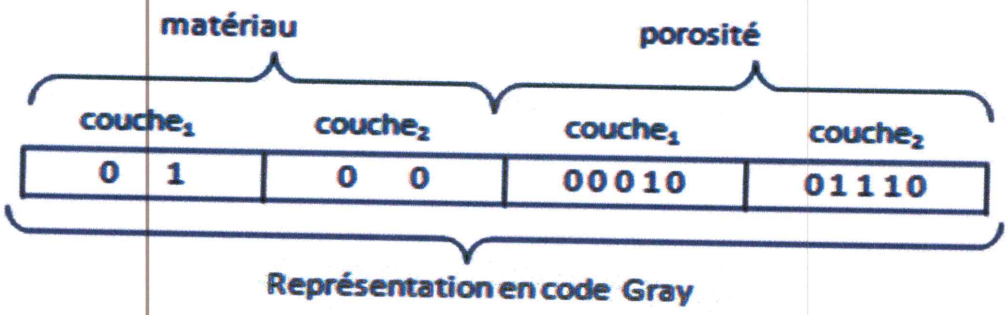


Figure I.6 : *Exemple de décodage d'une chaine binaire, N=2*

I.3.4.4Les opérateurs génétiques

1. La sélection des parents

La sélection des parents à pour but de deviner les individus de la population courante qui seront autorisés à se reproduire (les "**parents**"). La sélection est fondée sur la qualité des individus, estimée à l'aide de fonction d'adaptation. Cette opération est peut-être la plus importante puisqu'elle permet aux individus d'une population de sur vivre, 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. Il existe plusieurs méthodes pour la reproduction. , on citera à titre d'exemple:

- La sélection par roulette ou proportionnelle
- La sélection par tournoi.
- La sélection à reste stochastique.
- La sélection stochastique à reste stochastique.

- La sélection par le rang.
- La sélection uniforme.

Parmi ces différents types de sélection la méthode la plus connue et la plus utilisée reste la roulette biaisée, proposée par Goldberg (1989).

a) La sélection par roulette

La phase de sélection spécifie les individus de la population qui doivent survivre. La méthode de base, appelée roue de loterie attribue à chaque individu une probabilité de survie proportionnelle à son adaptation dans la population. Lors de la phase de sélection, les individus sont sélectionnés aléatoirement en respectant les probabilités ***pi*** associées pour former la population de la nouvelle génération.

Cette méthode consiste à dupliquer chaque individu de la population proportionnellement à son milieu. Ainsi, les individus ayant la plus grande valeur de fitness auront plus de chance d'être choisis. Dans une population de N individus, la fonction de sélection est la suivante [25]:

$$P_S(X_i) = \frac{F(X_i)}{\sum_{j=1}^{N} F(X_i)} \qquad \text{(I.4)}$$

En utilisant cette probabilité de reproduction, on peut créer une roue de loterie biaisée. Chaque individu de la population occupe une section de la roue proportionnellement à son adaptation et qui indique aléatoirement quel individu peut se reproduire. Cette méthode n'assure pas la sélection des meilleurs individus et peut être une cause de la convergence prématurée

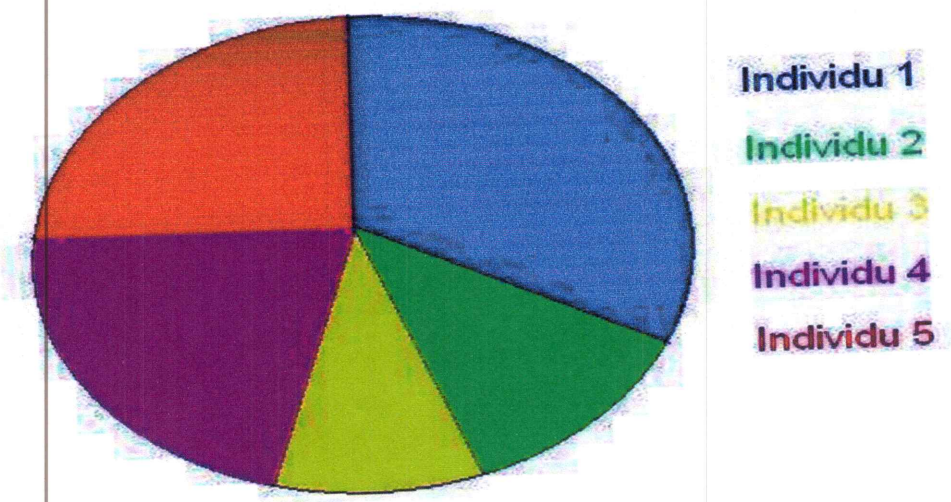


Figure I .7 : *Exemple de sélection par roulette*

b) La Sélection par tournoi

C'est la méthode la plus facile à mettre en œuvre. Cette technique utilise la méthode de la roulette biaisée pour sélectionner deux individus. On récupère celui dont la valeur de la fonction d'adaptation est la plus grande. Cette méthode choisit toujours une valeur de la fonction d'adaptation plus élevée par rapport à la technique de la roulette biaisée [25].

c) La sélection par élitiste

Cette méthode consiste à sélectionner les n individus dont on a besoin pour la nouvelle génération P' en prenant les n meilleurs individus de la population P après l'avoir triée de manière décroissante selon la fitness de ses individus.

Il est inutile de préciser que cette méthode est encore pire que celle de la loterie biaisée dans le sens où elle amènera à une convergence prématurée encore plus rapidement et surtout de manière encore plus sûre que la méthode de sélection de la loterie biaisée ; en effet, la pression de la sélection est trop forte, la variance nulle et la diversité inexistante, du moins le peu de diversité qu'il pourrait y avoir ne résultera pas de la sélection mais plutôt du croisement et des mutations. Là aussi il faut opter pour une autre méthode de sélection.

d) La sélection stochastique

Contrairement aux méthodes déterministes, les méthodes stochastiques associent aux individus une probabilité de sélection, généralement fonction croissante de leur fonction d'adaptation [25].

2. L'opérateur de croisement

Le croisement est un opérateur qui assure le brassage et la recombinaison des gènes parentaux pour former des descendants aux potentialités nouvelles, il correspond à un échange des matériels génétiques entre deux reproducteurs (parents) choisis d'une manière aléatoire parmi les génitures sélectionnés précédemment, pour former deux nouveaux chromosomes (ou enfants). L'opérateur de croisement étant aléatoire, il se produit selon une probabilitéa_x fixée par l'utilisateur en fonction du problème à optimiser. Il existe plusieurs types d'opérateurs de croisement dont nous citons ci-dessous les plus utilisés :

-Croisement a un point : C'est l'opérateur de croisement le plus simple. Il consiste à choisir au hasard deux parents (individus) de la population contenant des chromosomes précédemment sélectionnés et un site de croisement (un nombre de1 à $L-1$, avec L la longueur du chromosome et k l'indice de ce site), on obtient deux enfants en prenant d'une

part les k premiers allèles du premier parent et les $(L-k)$ derniers allèles du second parent et pour le deuxième enfant, les k premiers allèles du second parent et les $(L-k)$ derniers allèles du premier parent. Ceci est illustré dans l'exemple suivant en considérant que parent 1 et parent 2 ont été sélectionnés aléatoirement pour subir un processus de croisement à un point avec une position de coupure choisie aléatoirement entre1 $<= k <= 9$.

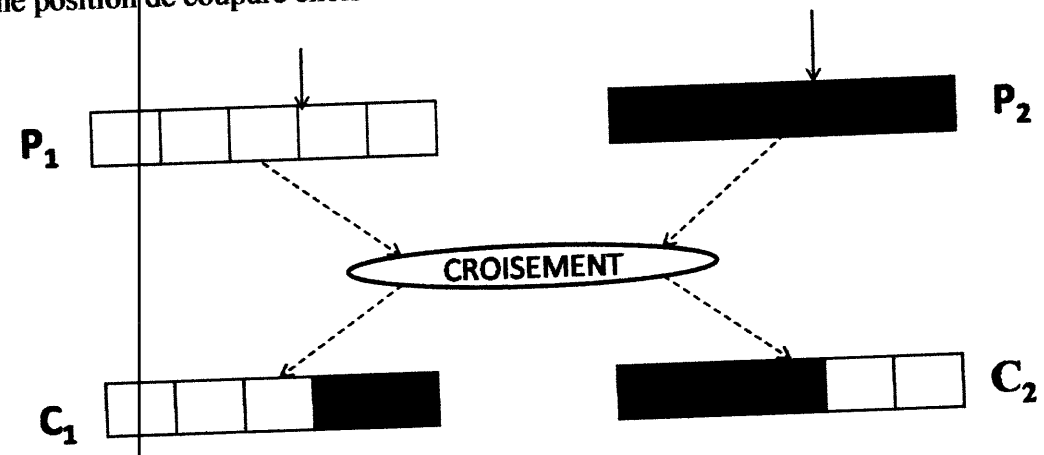


Figure I.8 : *Couper le croisement à un point.*

Tableau I.1 : *Schéma de croisement à un point*

Parent1	1 0 0 0 1 1 1 1 1 0
Parent2	1 1 1 1 0 0 0 0 0 1
Position aléatoire	K=4
Enfant1	1 0 0 0 0 0 0 0 0 1
Enfant2	1 1 1 1 1 1 1 1 1 0

-Croisement multipoints : Il est semblable à l'opérateur de croisement à un point sauf que dans ce cas plusieurs sites de croisement sont sélectionnés. Les enfants seront construits par mélange des parties appartenant aux deux parents. Nous considérons dans l'exemple qui suit un croisement à deux points, supposons que les deux points de coupure se trouvent aux positions 3 et 5.

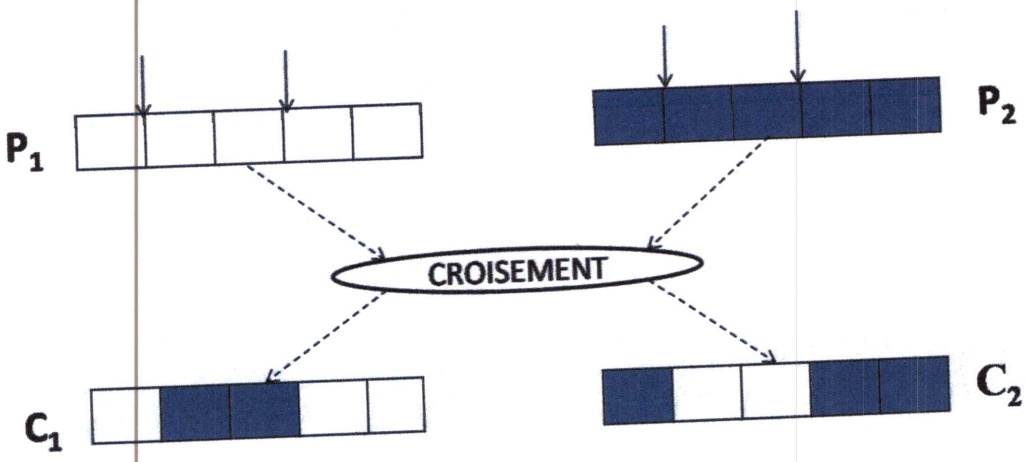


Figure I.9 : *Couper le croisement 2 points.*

Tableau I.2 : *Schéma de croisement à deux points*

Parent1	1 0 0 0 1 1 1 1 1 0
Parent2	1 1 1 1 0 0 0 0 0 1
Positions aléatoires	K1=3 K2=5
Enfant1	1 0 0 1 0 1 1 1 1 0
Enfant2	1 1 1 0 1 0 0 0 0 1

Tableau I.3 : *Les différents gènes pour les parents et les enfants.*

Individus parents			
Gène1	Gène2	Gène3	Gène4
Gène1	Gène2	Gène3	Gène4
Gène1	Gène2	Gène3	Gène4

Individus parents			
Gène1	Gène2	Gène3	Gène4
Gène1	Gène2	Gène3	Gène4
Gène1	Gène2	Gène3	Gène4

-Croisement uniforme : Ce croisement a été proposé par Syswerda. Il consiste à choisir avec la même probabilité un allèle de l'un ou l'autre parent pour transmettre à valeur à la même position des enfants. Pour bien comprendre ce type de croisement nous présentons l'exemple suivant :

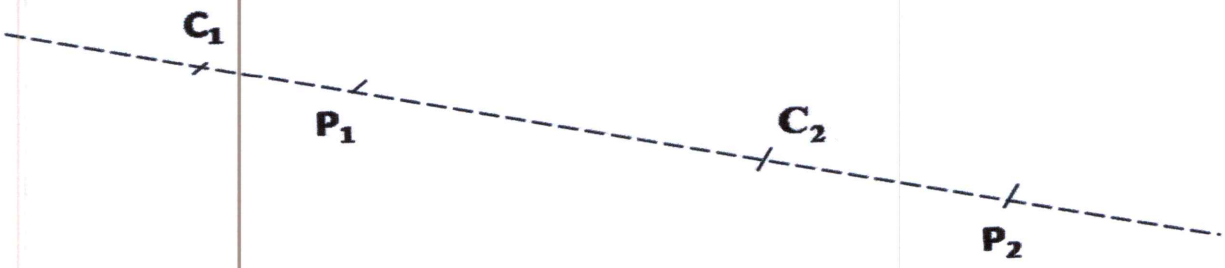


Figure I.10: *Croisement barycentrique.*

Tableau I.4: *schéma de croisement uniforme.*

Parent1	1 0 0 0 1 1 1 1 1 0
Parent2	1 1 1 1 0 0 0 0 0 1
Manque	1 0 0 1 1 1 0 0 1 0
Enfant1	1 1 1 0 1 1 0 0 1 1
Enfant2	1 0 0 1 0 0 1 1 0 0

3. l'opérateur de mutation

La mutation prend une place de plus en plus importante dans les algorithmes génétiques, alors qu'il y a encore quelques années son rôle était encore considéré comme accessoire. Comme les individus les mieux adaptés sont les plus susceptibles d'être choisis lors de la sélection, la perte de certains gènes est inévitable avec le temps. La mutation est l'opérateur qui permet d'éviter la dégénérescence de la population. Cette dégénérescence peut se traduire par une convergence des individus vers un optimum local, d'où l'importance de la mutation. Ce phénomène génétique d'apparition de "mutants" est rare mais permet d'expliquer les changements dans la morphologie des espèces, toujours dans le sens d'une meilleure adaptation au milieu naturel. On utilise une fonction censée nous retourner *true* avec une probabilité p_m.

Pour chaque locus **faire**

Faire appel à la fonction

Si cette fonction nous renvoie *true* **alors**

on inverse le bit se trouvant à ce locus

FinSi

FinPour

La mutation classique appliquée à une chaine binaire consistera à changer un << 1 >> en << 0 >> ou vice versa. Pour illustrer ce processus, supposons qu'un chromosome noté A ainsi que la position $K = 5$, ont été aléatoirement choisis, il vient :

Tableau I.5: *Schéma d'une mutation classique.*

Avant mutation	1 0 0 0 1 1 1 1 1 0
Position aléatoire	K=5
Après mutation	1 0 0 0 0 1 1 1 1 0

Comme pour le cas du croisement, il y a plusieurs types d'opérateurs de mutation, nous citons:

-Transposition de deux allèles consécutifs (échange de deux allèles consécutifs dans un chromosome)

-Transposition de deux allèles quelconque (échange de deux allèles choisis d'une manière aléatoire dans le même chromosome)

-Inversion d'allèles (on tire deux positions dans un chromosome d'une manière aléatoire et on inverse l'ordre des allèles dans la zone sélectionnée) [31].

La mutation entrainant des inversions de bits de manière aléatoire permet de réintroduire des différences entre les individus et donc de nous extirper de cette situation.

Il est quand même utile de garder à l'esprit que ceci n'est pas une solution "miracle" et qu'il est bien entendu plus intelligent de ne pas utiliser de méthodes de sélection connues pour entrainer ce type de problème. On notera que la mutation règle donc le problème exposé à la fin du Section sur le croisement.

4. La sélection finale

Cette étape consiste à garder seulement les solutions les plus intéressantes, tout en maintenant une population assez grande et assez diversifiée. C'est pourquoi la taille de la population doit rester la même d'une génération à l'autre. La sélection revient à choisir les

meilleurs individus pour former la nouvelle génération, c'est à dire éliminer *N* individu parmi les 2*N* individus (*N* parents et *N* enfants) pour cela plusieurs méthodes sont proposées [29].

a) La sélection par descendance

Dans cette méthode, on garde toujours les enfants, quel que soit leur adaptation la population de la nouvelle génération est obtenue par descendance ; les enfants remplaçant automatiquement leurs parents.

L'inconvénient de cette sélection est que l'on risque de voir disparaître les caractéristiques génétiques des parents les mieux adaptés si elles n'ont pas été totalement transmises lors de la recombinaison génétique.

b) La sélection par compétition

Une compétition a lieu entre parents et enfants pour déterminer ceux qui feront partie de la génération suivante. Les enfants sont insérés dans la population si et seulement si leurs performances sont supérieures à celles de leurs parents, à rang équivalent.

c) La sélection de procréation sélective

On garde les *N* meilleurs individus parmi la population intermédiaire de parents et d'enfants.

I.3.4.5 Critère d'arrêt et convergence

La recherche se termine lorsque le critère de convergence est atteint. Ce dernier peut prendre différentes formes. On peut tout simplement fixer un nombre de générations maximales après quoi on interrompt le calcul. Si la valeur optimale de la fonction objectif *f(x)* est connue, il est également possible d'arrêter l'optimisation lorsque l'on obtient un résultat suffisamment près de l'optimum désiré. Une autre approche, fréquemment employée, consiste à fixer le critère de converge à un nombre minimal de générations consécutives sans amélioration du meilleur individu. En d'autres termes, l'algorithme génétique pour suit sa recherche tant que la température maximale demeure inchangée pendant plusieurs générations successives, ce qui constitue un critère de convergence. Cette dernière approche est employée au cours de cette étude. Le nombre minimal de générations sans changement choisi est 250. Toutefois, un nombre total de générations est aussi utilisé en guise de critère d'arrêt de sûreté [32].

Les deux étapes d'évaluation et d'évolution de la population sont répétées jusqu'à ce qu'un critère d'arrêt soit satisfait. Les critères d'arrêt envisageables pour les algorithmes génétiques sont :

- Un nombre maximal de génération : si l'algorithme ne converge pas vers une solution, la procédure est stoppée après un nombre maximal de calculs.
- La convergence vers l'optimum : si la fonction erreur ou la moyenne sur la fonction erreur converge vers la solution.

I.3.4.5.a Paramètre des conditions Critère d'arrêt

On peut imaginer une multitude de conditions d'arrêt ; nous en avons choisi trois. Chacune comportant un paramètre. L'arrêt de la recherche intervient dès qu'une des conditions, détaillées ci-dessous, intervient :

- **Stagnation de la valeur du cout du meilleur individu**

Si la valeur du coût du meilleur individu de la population stagne pendant un nombre important de générations, on peut raisonnablement penser que l'algorithme ne trouvera pas de meilleures solutions aux problèmes car on ne remarque plus de changement considérable du coût entre les différentes itérations. Le paramètre est donc le nombre de générations de stagnation de la meilleure valeur du cout à partir de laquelle on doit interrompre la recherche.

- **Coût acceptable**

Ce test d'arrêt intervient lorsque l'on est capable de dire à partir de quelle limite on sera satisfait par la solution. Cela se traduira par la définition d'une valeur du coût maximale de la solution (c'est le paramètre à donner). L'algorithme interrompra donc les recherches des que le cout atteindra ou passera au-dessous de cette valeur.

- **Nombre maximum de générations**

Il existe des cas pour lesquels on sait que l'on dispose d'un temps limité pour trouver une solution, on peut donc limiter l'exécution de l'algorithme à un nombre maximum de générations après lequel l'algorithme arrête sa recherche. [33]

Un niveau bas de mutation sert à empêcher tous les éléments dans le chromosome de rester fixe a une valeur constante au sein de la population entière, tandis qu'un niveau élevé de mutation va rendre les individus totalement aléatoires. Pour maintenir un certain équilibre entre les deux extrémités, il faut bien choisir la valeur de a_m. [34]

I.3.5 Paramètre de la population

Ces paramètres sont pris en compte au début de la résolution d'un problème pour fixer les paramètres propres à la population et à son évolution :

- Nombre d'individus de la population Fixe le nombre d'individus générés aléatoirement au début pour former la population initiale.

C'est aussi le nombre d'individus à la fin de chaque phase d'évolution. En effet, le nombre d'individus de la population est constant : il y a autant de morts que de naissances à chaque étape.

- Taux de reproduction

C'est la proportion de la population qui se reproduit après croisement à chaque étape de l'évolution. C'est donc, aussi la proportion des individus qui mourront au cours d'une étape.

- Probabilité de mutation

C'est la probabilité que chaque gène de chaque individu de la population courante subisse une mutation aléatoire au cours d'une phase d'évolution. [33]

I.3.6 Les paramètres d'un AG

1) Taille de la population

Le choix de la taille de la population est un des paramètres les plus importants d'un algorithme évolutionnaires [Goldberg, 1986]. Et par-là même, c'est l'un des problèmes majeurs. De nombreuses études ont été menées pour essayer de cerner la taille optimale de la population, mais sans obtenir de résultats vraiment généralisables [Alender, 1992, Reeves, 1993].Certaines approches proposent des tailles de population variables, qui évoluent au cours du temps [Arabas, 1994]. La principale originalité de ce modèle est d'être plus proche de la réalité. En lieu et place d'une population de taille fixe où la fonction de mérite sert à la sélection, il propose plutôt d'indexer l'espérance de vie d'un individu à sa fonction de mérite. Les meilleurs individus reçoivent bien sûr une espérance de vie supérieure à celle des moins bonnes solutions. Au fur et à mesure que la population évolue, les individus vieillissent et finissent par mourir lorsqu'ils arrivent au terme de leur espérance de vie. Dans ce modèle, La taille de la population varie donc de manière dynamique en fonction de l'espérance de vie des individus que la composent. Cette étude détaillée de la mise en place de population de très petites tailles. Cette manipulation a principalement pour but d'accéder la convergence de l'algorithme. Un des avantages supplémentaires est qu'elle permet également d'implémenter une version efficace d'un algorithme génétique parallèle [35].

2) Le taux de croisement

Il détermine la proportion des individus qui sont croisés parmi ceux qui remplacement l'ancienne génération. L'opérateur de croisement est appliqué avec une probabilité Pc, et plus cette valeur est élevée plus de nouvelles structures (individus) sont introduites dans la nouvelle génération, les structures performantes sont trop fréquemment détruites. Par contre,

si ce taux est trop bas, la population n'évolue pas assez vite. En général, Pc varie 0.25 et 0.90.

3) Le taux de mutation

L'opérateur de mutation est appliqué avec une probabilité Pm ; si ce taux est grand alors la recherche devient purement aléatoire, la population est diversifiée et l'AG perd de son efficacité. Si au contraire ce taux est faible, la population est moins diversifiée et en plus il y a risque de stagnation. Des études empiriques conseillent pour l'obtention de bons résultats une fréquence qui se situe autour d'une mutation tous les 100 bits. A noter ces paramètres dépend étroitement du type de problème à résoudre.

4) Le fossé des génétiques

K.DE JONG a proposé de ne pas remplacer l'ensemble de la population à chaque génération et pour, il introduit la notion d'écart entre générations, un nombre compris entre 0 et 1 qui indique la proportion de parents qui sont remplacés par leurs descendants. Si ce taux à 1, l'ensemble de la population est remplacé.

I.3.7 Les avantages et les inconvénients des AG :

- **Avantages**

-Aucune hypothèse à faire sur l'espace de recherche.

-Nombreuses méthodes disponibles.

-Solutions intermédiaires interprétables.

-Adaptation rapide à de nouveaux environnements.

-Coévolution (tournoi), parallélisme et distribution aisés.

-Les représentations facilitent la compréhension.

- **Inconvénients**

-Aucune garantis de solution optimale en un temps fini.

-Initialisation de plusieurs paramètres, choix des méthodes importantes.

-Coût d'exécution important [35].

-résoudre des problèmes assez complexes. La résolution de ces problèmes est obtenue grâce aux opérateurs de reproduction.

I.3.8 Conclusion

Dans ce chapitre, nous avons présenté des généralités sur les algorithmes évolutionnaires. En premier lieu, nous avons introduit un rappel sur l'évolution des algorithmes évolutionnaires. Puis nous avons fait un rappel sur les différents types de ces algorithmes et leurs domaines d'application. Ainsi que ce chapitre nous a permis d'avoir une vue générale sur les concepts

des AG, leurs applications et leurs mécanismes de fonctionnement. Une de ces applications est l'optimisation. Les résultats d'optimisation pour une commande PID feront l'objet du prochain chapitre.

Chapitre II

L'OPTIMISATION DE REGULATEUR PID PAR DES ALGORITHMES GENETIQUES

II.1 Introduction

Le régulateur standard le plus utilisé dans l'industrie est le régulateur *PID* (proportionnel, intégral, dérivé), car il permet de régler à l'aide de ses trois paramètres les performances (amortissement, temps de réponse) d'une régulation d'un processus modélisé par un deuxième ordre. Nombreux sont les systèmes physiques qui, même en étant complexes, ont un comportement voisin de celui d'un deuxième ordre, dans une certaine échelle de temps. Par conséquent, le régulateur *PID* est bien adapté à la plupart des processus de type industriel et est relativement robuste par rapport aux variations des paramètres du procédé, quand on n'est pas trop exigeant pour les performances de la boucle fermée par rapport à celles de la boucle ouverte (par exemple, accélération très importante de la réponse ou augmentation très importante de l'amortissement en boucle fermée).

La réalisation d'une boucle d'asservissement par *PID* est un problème très important, car elle influe sur :

- la qualité de la régulation sur un site industriel ;
- le temps de mise en œuvre de la commande ; et comporte deux aspects essentiels :
- le réglage du régulateur *PID*, pour lequel la connaissance d'un modèle dynamique du procédé d'une part et les performances désirées d'autre part déterminent le choix de la méthode de synthèse;
- l'implantation du régulateur dans une version analogique ou numérique et dans une configuration série, parallèle ou mixte.

L'objectif de ce chapitre est de proposer des méthodes optimisées de conception du régulateur PID des incertitudes paramétriques, qui soit facile à implanter dans le milieu industriel. Initialement, nous présentons la structure du correcteur PID adoptée. Ensuite, on présente la boucle utilisée pour la commande en boucle fermée du système. Par la suite, les trois coefficients du premier régulateur PID sont déterminés à l'aide d'un algorithme génétique (AG). L'AG est utilisée pour la détermination des paramètres optimaux du régulateur PID [36].

II.2 La structure du correcteur PID

La fonction de transfert du régulateur *PID* que nous utilisons comme base de notre étude d'optimisation est la suivante :

$$G_{PID} = K_P \left[1 + \frac{1}{T_{ip}} + \frac{T_{dp}}{1+aT_{dp}}\right] \qquad (II.1)$$

Où K_p est le gain proportionnel, T_i la constante d'intégration, T_d la constante de dérivation et a est la constante du filtre sur la dérivée, généralement sa valeur varie entre 0 et 1. La fonction de chaque paramètre peut être brièvement résumée :

– le gain proportionnel - assure une transmission instantanée du signal d'erreur (différence entre consigne de vitesse et vitesse mesurée) ;

– le terme intégral - le comportement intégrateur à basse fréquence permet d'annuler l'erreur en régime permanent ;

–le terme dérivée-le comportement différentiel à haute fréquence permet d'améliorer la rapidité de la réponse du régime transitoire et limite les dépassements (ou l'instabilité au sens général) ;

– le filtre - assure la causalité de la fonction de transfert et la limitation du gain en haute fréquence [36].

II.3 Notations et définitions

La structure du système de commande est décrite par la figure II.1:la fonction de transfert du régulateur y est notée $C(s)$ et celle du procédé (s) , où s est la variable de Laplace. La fonction de transfert de la boucle ouverte compensée sera donc :

$$H_{BO}(s) = C(s)G(s) \tag{II.2}$$

Et celle de la boucle fermée :

$$(s) = \frac{H_{BO}(s)}{1+H_{BO}(s)} \tag{II.3}$$

D'une façon générale, les transformées de Laplace seront notées avec des lettres majuscules. On définit ainsi le signal d'erreur par $E(s) = R(s) - Y(s)$ où la sortie du procédé est donnée par :

$$Y(s) = \frac{C(s)G(s)}{1+C(s)G(s)}R(s) + \frac{G(s)}{1+C(s)G(s)}P_1(s) + \frac{1}{1+C(s)G(s)}(V(s) + P_2(s) \tag{II.4}$$

Le régulateur PID classique relie directement **[37]** le signal de commande $u(t)$ au signal d'écart$e(t)$. Sa description temporelle est la suivante :

$$u(t) = K_p(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\left(\frac{de(t)}{dt}\right) \tag{II.5}$$

Avec l'écart défini comme suit :

$$e(t) = w(t) - y(t) \tag{II.6}$$

Sa fonction de transfert s'écrit :

$$G_s(s) = \frac{U(s)}{E(s)} = K_P(1 + \frac{1}{sT_i} + sT_d) \quad \text{(II.7)}$$

Cette combinaison du terme P, I et D est aussi désignée sous le nom de forme parallèle ou non-interactive. Le schéma fonctionnel d'un processus réglé à l'aide d'un tel régulateur est donné à la figure (II-1). En plus des signaux dénis plus haut, on y trouve la perturbation $v(t)$, le bruit de mesure $n(t)$ et le signal de sortie non bruité $x(t)$.

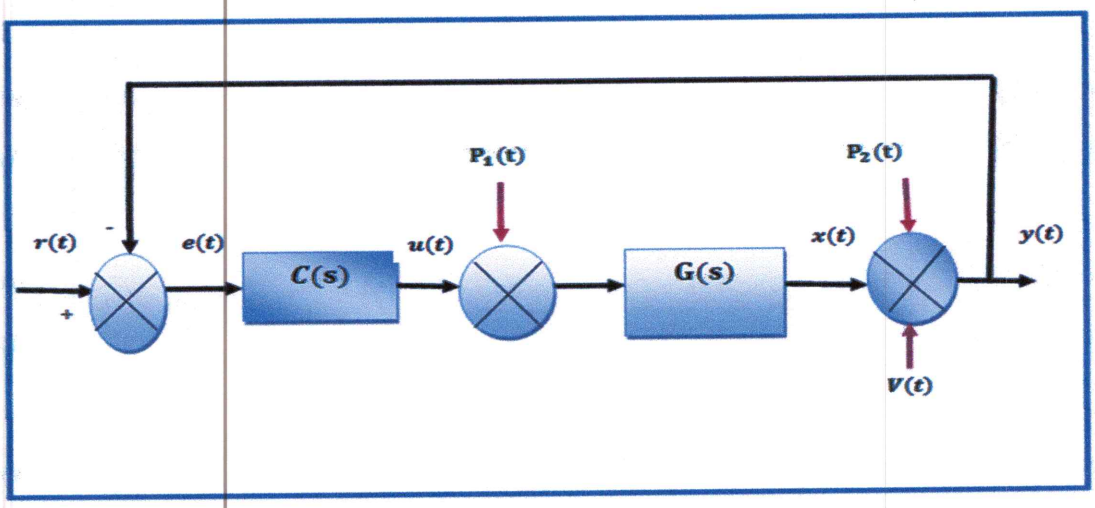


Figure II.1 : *Schéma fonctionnel d'un processus réglé par un PID classique*

Avec

$\boldsymbol{r}(\boldsymbol{t})$: Signal de référence ou consigne (entré par l'utilisateur ou un autre régulateur),

$\boldsymbol{e}(\boldsymbol{t})$: Erreur (entrée du régulateur).

$$\boldsymbol{e}(\boldsymbol{t}) = \boldsymbol{r}(\boldsymbol{t}) - \boldsymbol{y}(\boldsymbol{t}).$$

$\boldsymbol{u}(\boldsymbol{t})$: Commande (sortie du régulateur).

$\boldsymbol{p_1(t)}, \boldsymbol{p_2(t)}$: Perturbation à l'entrée et à la sortie du procédé respectivement.

$\boldsymbol{v}(\boldsymbol{t})$: Bruit à la sortie du procédé (par exemple bruit de mesure).

$\boldsymbol{y}(\boldsymbol{t})$: Sortie mesurée du procédé (variable à commander).

$\boldsymbol{C}(\boldsymbol{s})$: Fonction de transfert du régulateur.

$\boldsymbol{G}(\boldsymbol{s})$: Fonction de transfert du procédé.

La fonction de sensibilité de la boucle fermée (sensibilité perturbation sortie - sortie) est notée :

$$S_{YV}(s) = \frac{1}{1+H_{BO}(s)} \quad \text{(II.8)}$$

Son maximum étant défini par :

$$S_{YVmax} = max_w|S_{YV}(jw)| \quad \text{(II.9)}$$

avec w pulsation.

On appelle **zéros du régulateur** les racines du numérateur de la fonction de transfert C (s) de ce régulateur.

On parle de pôles du régulateur quand il s'agit des racines du dénominateur de la fonction de transfert.

Si ces racines sont réelles, on parle de **zéros** ou de **pôles réels.**

Si les modules de ces racines sont faibles (grandes constantes de temps), on parle de zéros ou de **pôles lents** (avec une valeur limite quand la racine est nulle donc pour une constante de temps infinie).

I.4 Paramètres

La commande $U\ (t\ )$donnée par le régulateur PID, dans sa forme Classique est décrite par :

$$u(t) = K_p(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d(\frac{de(t)}{dt}) \quad \text{(II.10)}$$

Elle est la somme de trois termes :

- **le terme proportionnel** $P = K_p\, e(\,t)$**(proportionnel à l'erreur).**
- le terme intégral $I = K_i\frac{1}{T_i}\int_0^t e(t)dt$(proportionnel à l'intégrale de l'erreur).
- le terme dérivatif $D = K_d T_d \frac{de(t)}{dt}$(proportionnel à la dérivée de l'erreur).

Les paramètres du régulateur PID **sont le gain proportionnel** Kp**, le temps intégral** T_i et le temps dérivatif T_d, les temps étant exprimés en secondes.

Dans les régulateurs PID existants sur le marché, les **[36]** actions proportionnelle et intégrale peuvent être exprimées comme suit :

L'action proportionnelle s'exprime soit par le gain proportionnel Kp**, soit par la bande** proportionnelle BP. Cette dernière est définie comme la variation, en pourcentage, de l'entrée du régulateur e nécessaire pour que la sortie varie de 100 %.

$$BP\% = \frac{100}{K_P}$$

La relation entre le gain Kp et la bande proportionnelle (BP), exprimée en % est :

L'action intégrale s'exprime soit par le temps, qui représente le temps nécessaire pour que la variation de la sortie (u) soit égale à celle de l'entrée (e), soit par l'inverse du temps, n, qui exprime le nombre de fois que la sortie u répète l'entrée e, dans l'unité de temps (mn, s) :$T_i = \frac{1}{n}$.

II.5 Formes des régulateurs PID

II.5.1 Forme standard

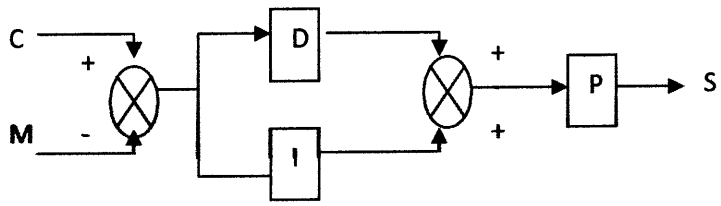
La forme standard de la fonction de transfert du régulateur *PID* est :

$$C(s) = K_P\left[1 + \frac{1}{sT_i} + \frac{sT_d}{1+s\frac{T_d}{N}}\right] \quad \text{(II.11)}$$

On note que l'action dérivative est implémentée avec un filtre du premier ordre, dont la constante de temps est N fois plus petite que Td. Ce filtre permet d'atténuer l'effet du bruit hautes fréquences dans la commande du *PID*, le choix typique de N étant compris entre 10 et 20. L'équation (II.10) représente la forme conventionnelle, commercialisée par la plupart des constructeurs de régulateurs *PID*.

Les deux zéros « lents » introduits par le régulateur peuvent être complexes, ce qui est utile quand on veut commander des systèmes avec un mode oscillatoire **[38].**

En a présenter dans le schéma suivant (PID classique standard sans filtre):



$$K_p(1 + \frac{1}{pT_i} + pT_d)$$

Figure II.2: structure *PID* standard

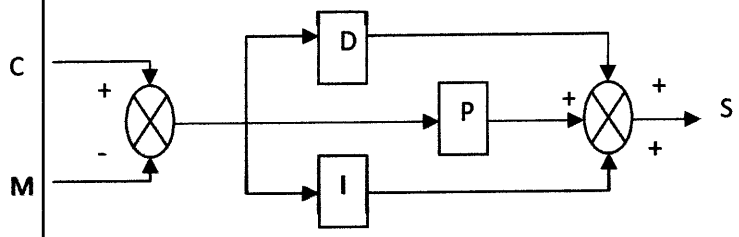
II.5.2 Forme parallèle

La forme « parallèle » de la fonction de transfert du régulateur *PID* est donnée par:

$$C(s) = K_p + \frac{K_i}{s} + \frac{K_d s}{1+(K_d+K)s} \quad \text{(II.12)}$$

Avec kp, ki et kd constantes.

Cette forme est équivalente à la forme standard, et la relation entre les paramètres de la forme parallèle et la forme standard est évidente. L'avantage de cette forme vient du fait qu'une action proportionnelle, intégrale ou dérivée pure peut être obtenue avec des paramètres finis du régulateur. Par contre, les paramètres n'ont pas une interprétation physique évidente. La figure II.3 montre un PID classique mixte sans filtre:



$$K_p + \frac{1}{pT_i} + pT_d$$

Figure II.3 : structure *PID* mixte

I.4.3 Forme série

La forme « série » de la fonction de transfert du régulateur *PID* est :

$$C(s) = \frac{(1+t_1 s)(1+t_2 s)}{t_i s(1+t_N s)} \qquad \text{(II-13)}$$

Avec $t1, t2, ti$ et tN constantes de temps.

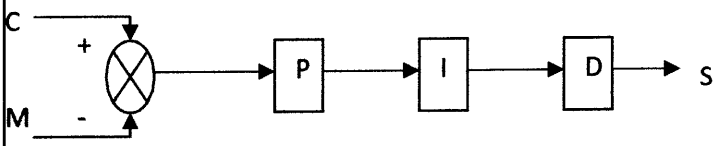
Cette forme permet de mettre plus facilement en relation les paramètres du régulateur avec les constantes de temps du procédé. Sous cette forme, le régulateur a deux zéros réels.

Le régulateur *PID* sous la forme série (II-13) peut toujours être représenté sous la forme parallèle (II-11), selon :

$$\begin{cases} K_P = \frac{t_1+t_2-t_N}{t_i} \\ T_i = t_1 + t_2 - t_N \\ T_d = \frac{t_1 t_2 - t_N t_1 - t_2 t_N + t_N^2}{t_1+t_2-t_N} \\ N = \frac{t_1 t_2 - t_N t_1 - t_2 t_N + t_N^2}{t_N(t_1+t_2-t_N)} \end{cases} \qquad \text{(II.14)}$$

L'inverse n'est possible que si $NT_i - T_d \geq 2N\sqrt{T_i T_d}$ (condition sous laquelle la forme standard donne des zéros réels) qui montre un PID classique série sans filtre.

$$\begin{cases} t_{1,2} = \frac{T_i + \frac{T_d}{N}}{2}\left[1 \pm \sqrt{1 - \frac{4T_iT_d(\frac{N+1}{N})}{(T_i+\frac{T_d}{N})^2}}\right] \\ t_i = \frac{T_i}{K_P} \\ t_N = \frac{T_d}{N} \end{cases} \quad \text{(II.15)}$$



$$K_p(\frac{T_i + T_d}{T_i} + \frac{1}{pT_i} + pT_d)$$

Figure II.4 : structure *PID* série

La forme standard est ainsi plus générale que la forme série. Son inconvénient majeur est le fait qu'une action intégrale ou proportionnelle pure ne peut pas être obtenue avec des paramètres finis du régulateur.

I.6 Les actions PID

En pratique, à une catégorie donnée de systèmes à asservir correspond un type de correcteur adopte. Pour effectuer un choix judicieux, il faut connaitre les effets des différentes actions : proportionnelle, intégrale et dérivée.

Un régulateur *PID* est obtenu par l'association de ces trois actions et il remplit essentiellement les trois fonctions suivantes :

- **Il fournit un signal de commande en tenant compte de l'évolution du signal de sortie** par rapport a la consigne
- Il éliminer l'erreur statique grâce au terme intégrateur
- **Il anticipe les variations de la sortie grâce au terme dérivateur.**

La commande $U(t)$ donnée par le régulateur *PID*, dans sa forme Classique est décrite par :

$$u(t) = K_p(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}) \quad \text{(II.16)}$$

Elle est la somme de trois termes :

- le terme proportionnel $P = K\,e(t)$

- le terme intégral $I = K_p \frac{1}{T_i} \int_0^t e(t)dt$
- le terme dérivatif $D = K_p T_d \frac{de(t)}{dt}$

I.6.1 L'action proportionnelle

La sortie $U(t)$ du regulateur proportionnel est donnee en fonction de son entrée $e(t)$ qui represente l'ecart entre la consigne et la mesure par la relation :

$$u(t) = K_p . e(t) \quad (II.17)$$

Pour le cas discret, cette relation reste la mémé telle que :

$$u(k) = K_p . e(k) \quad (II.18)$$

Le rôle de l'action proportionnelle est de minimiser l'écart e entre la consigne et la mesure et elle réduit le temps de monter et le temps de réponse. On constate qu'une augmentation du gain K_p du régulateur entraine une diminution de l'erreur statique et permet d'accélérer le comportement global de la boucle fermée. On serait tente de prendre des valeurs de gain élevées pour accélérer la réponse du procède mais on est limite par la stabilité de la boucle fermée. En effet, une valeur trop élevée du gain augmente l'instabilité du système et donne lieu à des oscillations **[39].**

I.6.2 L'action dérivée

C'est une action qui tient compte de la vitesse de variation de l'écart entre la consigne et la mesure, elle joue aussi un rôle stabilisateur, contrairement a l'action intégrale.
En effet, elle délivre une sortie variant proportionnellement à la vitesse de variation de l'écart e :

$$u(t) = T_d \frac{de(t)}{dt} \quad (II.19)$$

avec T_d le dosage de l'action dérivée, exprime en minutes ou en secondes.
L'action dérivée va ainsi intervenir uniquement sur la variation de l'erreur ce qui augmente la rapidité du système (diminution des temps de réponses). L'action dérivée permet aussi d'augmenter la stabilité du système par apport de phase $(+\frac{\pi}{2})$ ce qui augmente la marge de phase). L'annulation de cette action en régime statique impose donc de ne jamais l'utiliser seule : l'action dérivée n'exerce qu'un complément a l'action proportionnelle. En pratique, il est souhaitable de limiter l'action dérivée afin de ne pas amplifier les bruits haute fréquence et de limiter l'amplitude des impulsions dues aux discontinuités de l'écart. Lorsque la période

d'échantillonnage Te est petite, la différence vers l'arrière (Approximation d'Euler rétrograde) nous permet d'approcher la dérivée d'un signal a temps continu par :

$$\frac{df(t)}{dt} = \frac{f(t)+f(t-T_e)}{T_e} \qquad \text{(II.20)}$$

L'opération de dérivation se traduit par une multiplication par la variable de Laplace p en continu. Dans le cas discret, en appliquant la transformée en z.a l'équation (II.18) on obtient :

$$Z\left\{\frac{f(t)-f(t-T_e)}{T_e}\right\} = \frac{1-Z^{-1}}{T_e}F(z) \qquad \text{(II.21)}$$

Ceci conduit à établir l'équivalence linéaire entre la variable de Laplace p et la variable z:

$$p \longleftrightarrow \frac{1}{T_e}.(1-z^{-1})$$

Dans le cas discret, le terme dérivé peut être remplace par :

$$u(k) = \frac{T_d}{T_e}.\big(e(k)-e(k-1)\big) \qquad \text{(II.22)}$$

I.6.3 L'action intégrale

L'action intégrale agit proportionnellement à la surface de l'écart entre la consigne et la mesure, et elle poursuit son action tant que cet écart n'est pas nul. On dit que l'action intégrale donne la précision statique (Elle annule l'erreur statique). L'action intégrale est conditionnée par le temps d'intégrale Ti.

$$u(t) = \frac{1}{T_i}\int_0^t e(t).dt \qquad \text{(II.23)}$$

Comme dans le cas de l'action proportionnelle, un dosage trop important de l'action intégrale engendre une instabilité de la boucle de régulation. Pour son réglage, il faut la aussi trouver un compromis entre la stabilité et la rapidité.

L'ajout du terme intégral permet d'améliorer la précision mais en contrepartie, il introduit malheureusement un déphasage de $\frac{\pi}{2}$ ce qui risque de rendre le système instable du fait de la diminution de la marge de phase. Enfin, le correcteur intégral **[40]** présente le défaut de se saturer facilement si l'écart ne s'annule pas rapidement, ce qui est le cas des systèmes lents. En effet, tout actionneur est limite : un moteur est limite en vitesse, une vanne ne peut pas être

plus que totalement ouverte ou totalement fermée. Il se peut que la variable de commande amène l'actionneur a sa limite ce qui suspend la boucle de retour et le système aura une configuration assimilable a une boucle **[41]** ouverte puisque l'actionneur demeurera sature indépendamment de la sortie du système.

Quand l'erreur est réduite (action intégrale non saturée), il se peut qu'il faille un temps important pour que les valeurs des variables ne soient correctes de nouveau : on appelle ce phénomène l'emballement du terme intégral.

Pour l'éviter, on peut :

- soit suspendre l'action intégrale quand la commande est saturée ;
- soit appliquer une méthode d'anti-saturation, qui consiste à recalculer le terme intégral pour ne pas saturer.

Pour le cas discret, le terme intégral peut être remplace par la somme des écarts et la différentielle dtpar Te ce qui nous donne le résultat suivant :

$$u(n) = \frac{T_e}{T_i}\sum_{k=0}^{n} e(k) = u(n-1) + \frac{T_e}{T_i}e(n) \qquad \text{(II.24)}$$

Ti : Constante de temps intégrale et s'exprime en (s).

Le problème est que lorsque e redevient nul, le signal U peut avoir atteint une valeur trop élevée pour qu'il y ait équilibre. Autrement, elle permet d'éliminer l'erreur résiduelle en régime permanent.

L'action intégrale est utilisée lorsqu'on désire avoir en régime permanent, une précision parfaite, en outre, elle permet de filtrer la variable à régler d'ou l'utilité pour le réglage des variables bruites telles que la commande

II.7 Résumé des avantages et désavantage des actions *P, I, et D*

Tableau II-1 : *Résumé des effets respectifs des actions P, I, et D*

Action	Avantage	Désavantage
P	Dynamique	Ne permet pas d'annuler une erreur statique
I	Annulation d'erreur statique, amélioration de la robustesse	Action lente, ralentit le système (effet déstabilisant)
D	Action très dynamique, améliore la rapidité (effet stabilisant)	Sensibilité aux bruits forte sollicitation de l'organe de commande

II.8 L'approche basée sur l'Algorithme Génétique

Dans cette section, nous traitons le problème d'optimisation des trois paramètres du régulateur *PID* basé sur un Algorithme Génétique (AG). Ainsi, nous décrivons la façon dont laquelle l'AG est utilisé pour le calcul des paramètres du *PID*, en considérant les mêmes critères de robustesse et de performance qui ont été établit dans la section précédente.

II.8.2 Application des AG à l'optimisation du régulateur PID

Dans cette section, nous détaillons les aspects de la mise en œuvre de l'optimisation des paramètres du *PID* à partir de l'utilisation des AG. La structure du régulateur est la même que celle qui a été présentée dans la section précédente. L'objectif est toujours de trouver l'ensemble des paramètres optimisés de façon à ce que la réponse en vitesse du système en boucle fermée soit stable et la plus robuste possible pour toute variation. Avec l'utilisation de l'AG il est possible de réaliser la recherche de la solution optimale dans un espace tridimensionnel au détriment de la recherche quasi unidimensionnelle qui a été mise en œuvre à partir de l'approche itérative. Chaque dimension de l'espace de recherche tridimensionnel correspond à une variable de l'individu x que nous cherchons à optimiser. L'approche d'optimisation du régulateur *PID* par AG est définit dans les étapes suivantes :

a) Définition de la population initiale

La population initiale est choisie de façon aléatoire dans un espace de recherche des solutions. L'intervalle de variation de chaque variable de *D* est définit de la façon suivante :

1. Nous adoptons l'extrémité de la plage de variations de J_l comme référence pour le système :

$$J_l = J_{l\,min} \quad (1)$$
$$J_l = J_{l\,max} \quad (2) \qquad \text{(II.25)}$$

2. Pour chaque référence d'inertie nous calculons, à partir des simulations en boucle fermée, les valeurs maximales de chaque coefficientK_P, T_i et T_d en respectant le critère de robustesse et stabilité. Comme résultat nous avons deux ensembles de paramètres du régulateur :

Ensemble(1) : K_{p1} , T_{i1}, T_{d1}. (II.26)

Ensemble(2) : K_{p2}, T_{i2}, T_{d2} .

3. L'**espace de recherche *D* est définit par** :

Pour la définition des limites de l'espace de recherche D de façon nous garantissons que les valeurs optimales du régulateur PID sont incluses dans l'espace de recherche D.

$$D = [0 \ max(K_{p1}, K_{p2})] * [0 \ max(T_{d1}, T_{d2})] * [0 \ max(T_{i1}, T_{i2})] \qquad \text{(II.27)}$$

$$\begin{cases} x_{1opt} = K_{Popt} & \in & [0 \ max(K_{p1}, K_{p2})] \\ x_{2opt} = K_{dopt} & \in & [0 \ max(T_{d1}, T_{d2})] \\ x_{3opt} = K_{iopt} & \in & [0 \ max(T_{i1}, T_{i2})] \end{cases} \qquad \text{(II.28)}$$

b) La fonction d'adaptation

L'inverse de l'intégrale de l'erreur quadratique (ISE) est utilisée comme fonction d'adaptation à maximisé sur un intervalle T :

$$f_{ad}(x_{1i}, x_{2i}, x_{3i}) = \frac{1}{\frac{1}{T}\int_0^T [w_{lref} - w_l]^2 dt} \qquad \text{(II.29)}$$

L'adaptation de chaque individu d'une population doit être calculée sur toute la plage de variation. En sachant que le système en boucle fermée a un comportement linéaire sur la variation paramétrique, nous calculons la fonction d'adaptation de chaque individu selon l'équation suivante :

$$f_{ad}(x_{1i}, x_{2i}, x_{3i}) = \frac{1}{\sum_{n=J_{l\,min}}^{J_{l\,max}} \frac{1}{T}\int_0^T (w_{lref} - L^{-1}[H_{l-C}(p,n)] * C_{em})dt} \qquad \text{(II.30)}$$

ou L^{-1} Représente la transformation inverse de Laplace, et _ représente une convolution temporelle. La fonction de transfert est donnée par :

Chapitre III

SIMULATION ET INTERPRETATIONS

III.1 Introduction

Le problème de la commande PID par algorithme génétique (AG) est de trouver des gains de commandeKp, Ki et Kd qui optimisent une fonction fitness (indice ou critère de performance) qui satisfait les équations d'état du système et les contraintes. Il a d'importantes applications de types économiques, environnementaux, de l'engineering, etc. dans ce chapitre, on aborde quelques exemples pratiques et de voir l'influence des lois de commande sur le comportement des différents systèmes étudiés. La figure ci-dessous donne un organigramme général sur l'optimisation des paramètres à l'aide des AGs.

III.2 Organigramme générale d'un algorithme génétique.

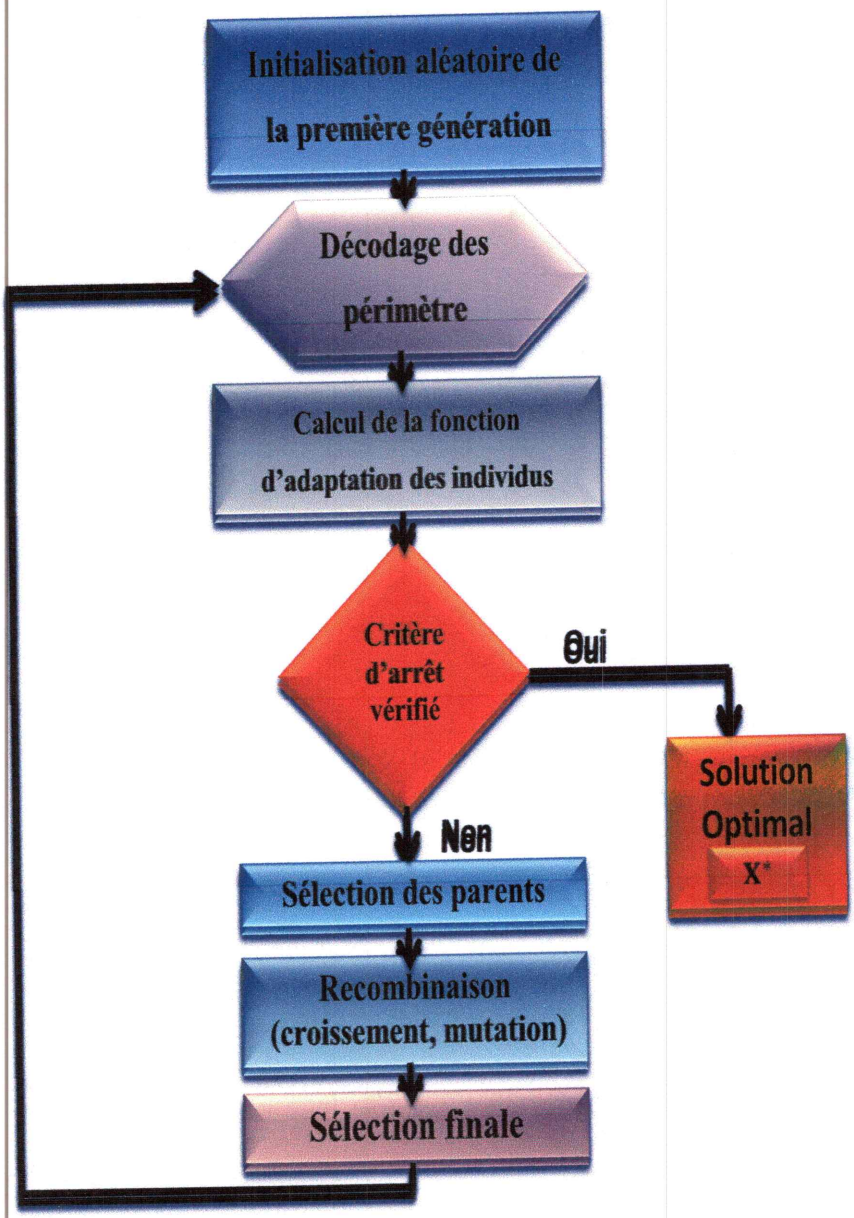


Figure III.1 : *Organigramme générale d'un algorithme génétique.*

Les différentes étapes de fonctionnement des **(AG)** se résument à qui suit :

1. **Initialisation : Une population initiale de *N* individus est générée aléatoirement.**
2. **Evaluation :** Chaque individus est décodé, puis évalué.
3. **Sélection:** Création d'une nouvelle population par l'utilisation d'une méthode de sélection appropriée.
4. **Recombinaison: croisement et mutation au sein de la nouvelle population.**
5. **Retour:** à la phase d'évaluation jusqu'à la vérification du critère d'arrêt de l'algorithme.

La mise en œuvre des algorithmes génétiques nécessite donc plusieurs étapes. L'idée **fondamentale est que: la population choisie contient potentiellement la solution, ou plutôt la** meilleure solution, à un problème donné. Cette solution n'est pas exprimée car la combinaison génétique sur laquelle elle repose est dispersée chez plusieurs individus. Ce n'est que par l'association de ces combinaisons génétiques au cours de la reproduction que la solution **pourra s'exprimer. Lors de la reproduction et de la recombinaison génétique associée,** un individu hérite, par hasard, d'un des gènes de chacun de ses parents. L'originalité des mécanismes repose en particulier sur le fait qu'il n'a pas considéré les seules mutations comme **source d'évolution mais aussi et surtout les phénomènes de croisement. C'est en croisant les** solutions potentielles existant que l'on peut se rapprocher de l'optimum.

III. 3 Optimisation des paramètres de régulateur PID par les algorithmes génétiques

Nous allons commencer par une présentation des algorithmes génétiques pour l'optimisation des paramètres de régulateur PID en vue de la commande des différents systèmes.

❖ Stratégie d'optimisation du régulateur PID

Pour décrire le processus d'optimisation des paramètres de régulateur PID utilisé ; on considère le schéma bloc fonctionnel donné dans la figure (***III.2***) en changeant les valeurs de PID. L'objectif global du système est de commander la pompe de réglage de surtension **par un régulateur PID avec optimisation des paramètres par AG, qui est réduite au minimum** l'erreur des paramètres (kp, ki, kd).

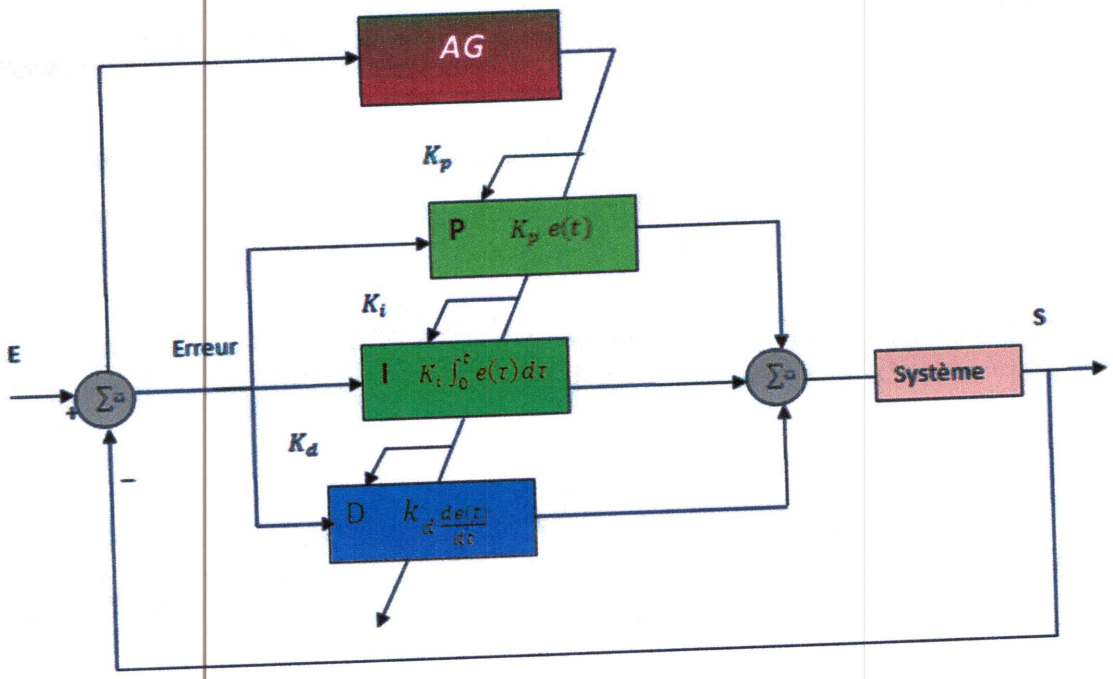


Figure III.2 : *Schéma bloc fonctionnel de la commande PID et optimisation par AG.*

III.3.1 Exemple 1 : Identification des paramètres des circuits électroniques par les *AGs*.

Un circuit RC est un circuit électrique, composé deux résistance en parallèle et d'une résistance et d'un condensateur montés en série, cette condensateur déchargera son énergie entreposée à travers la résistance. . Dans leur configuration série, les circuits RC permettent de réaliser des filtres électroniques passe-bas. La constante de temps \tau d'un circuit RC est donnée par le produit de la valeur de ces deux éléments qui composent le circuit.

On considère le circuit suivant :

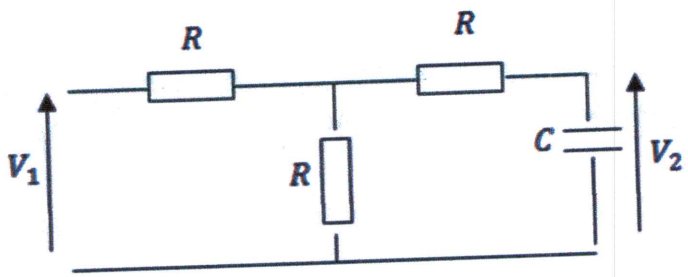


Figure III.3 : *Schéma électrique.*

III.3.1.1 Le modèle mathématique de circuit :

D'après la figure III.3, on a

$$V_1(t) = \left(\frac{R}{2} + R\right) i(t) + V_2(t) \quad \text{(III. 1)}$$

$$i(t) = c\frac{dV_2(t)}{dt} \quad \text{(III.2)}$$

$$V_1(t) = \left(\frac{3RC}{2}\right)\frac{dV_2(t)}{dt} + V_2(t) \quad \text{(III.3)}$$

On utilise la transformée de Laplace, on obtient:

$$TL\,[V_1(t)] = TL\left(\left(\frac{3RC}{2}\right)\frac{dV_2(t)}{dt}\right) + TL\,V_2(t) \quad \text{(III.4)}$$

$$V_1(p) = \frac{3}{2}RCp + V_2(p) \quad \text{(III.5)}$$

D'où:

$$\frac{V_2}{V_1} = H(p) = \frac{1/2}{1+\frac{3}{2}RCp} \quad \text{(III.6)}$$

La figure ci-dessous illustre le schéma bloc et le graphe de fluence correspondant :

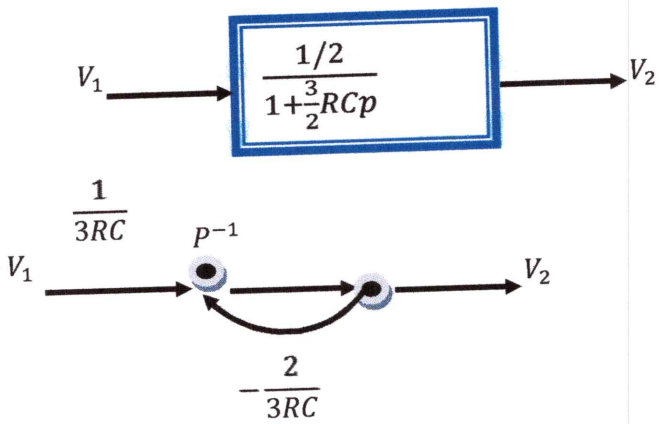


Figure III.4 : *Schéma bloque du système.*

Les valeurs des composantes R et C permettant au filtre d'avoir les caractéristiques désirées :

$$\begin{cases} Gain\ en\ tension = \left|\frac{V_2}{V_1}\right| = \frac{1}{2} \\ Fréquence\ de\ coupure\ f_c = 106.1\ Hz \end{cases}$$

On a:

$$\frac{V_2}{V_1} = H(j\omega) = \frac{\frac{1}{2}}{1+j\frac{3}{2}RC\omega} = \frac{\frac{1}{2}}{1+j\frac{\omega}{\omega_c}} \quad \text{(III.7)}$$

Avec :

$$\omega_c = \frac{2}{3RC}\ rad/s \quad \text{(III.8)}$$

Donc :

$$f_c = \frac{\omega}{2\pi} = \frac{1}{3\pi RC} = 106.1\ Hz, \quad \text{(III.9)}$$

D'où :

$$RC = \frac{1}{3\pi 106.1} = 0.001 \quad \text{(III.10)}$$

On choisit : $R = 1\ k\Omega$ ce qui donne$C = 1\mu F$. Afin d'optimiser le rendement du circuit électrique, l'objectif est de déterminer les valeurs R et C en utilisant un AG ayant les caractéristiques suivantes : Binaire, Sélection par roue de fortune, Croisement et mutation simples ; où les probabilités sont dynamiques)

III.3.1.2 Résultat de simulation :

Après simulation, on obtient les valeurs suivantes :

R = 97.6786 Ω ; C = 1.0259e-005 F ; fc =105.8835 Hz

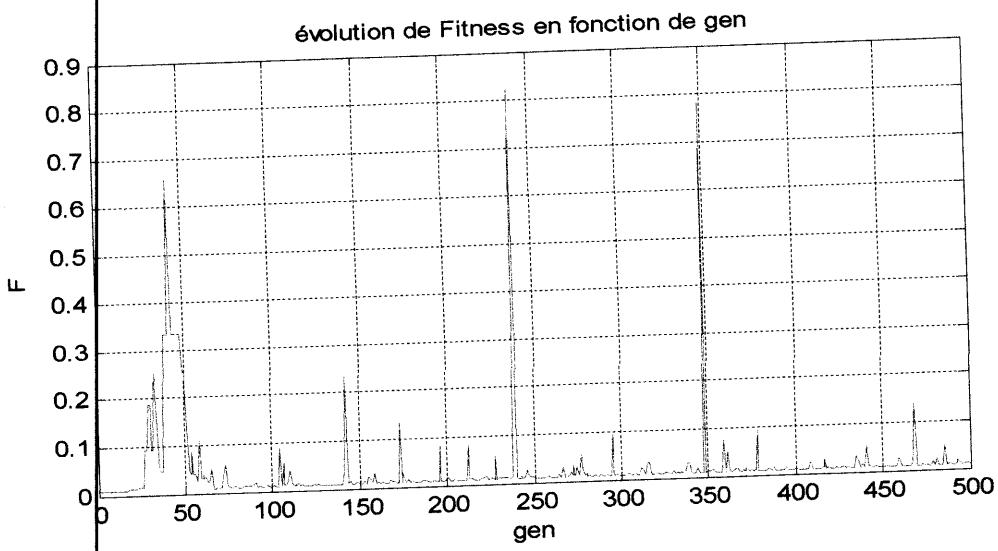


Figure III.5 : *L'évolution de fitness en fonction de génération*

Nous avons constaté que les valeurs ne sont pas normalisées, pour ça, il vaut mieux de fixer une variable (ex ; R=1KΩ) et déterminer l'autre. En effet, même si la deuxième n'est pas normalisée, on peut utiliser un ajustable (ex condensateur ajustable).

Résultat : Avec C=1μF une valeur fixée, ce qui donne un bon résultat

R = 1.0000e+003 Ω ; fc = 106.1021 Hz

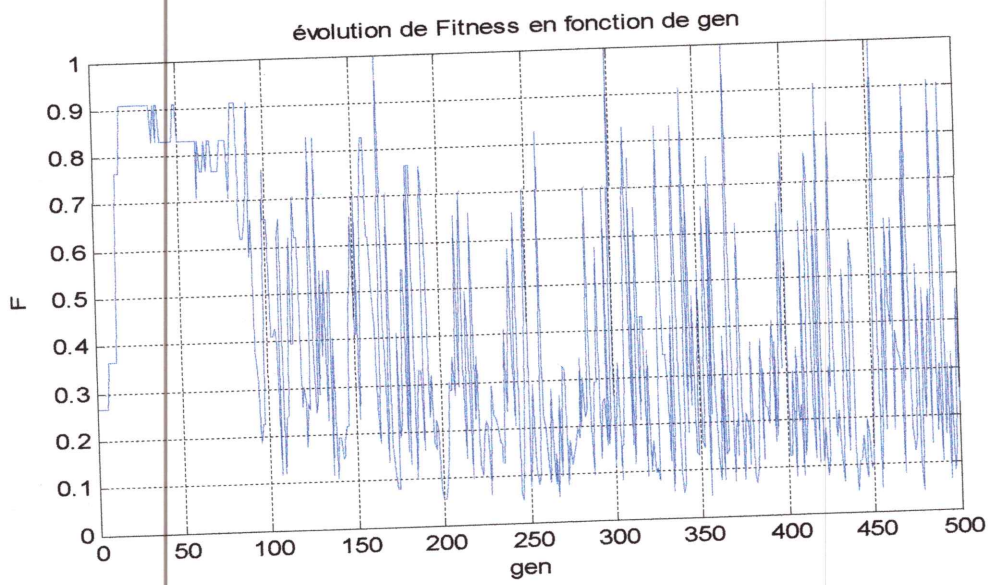


Figure III.6 : *L'évolution de fitness en fonction de génération*

III.3.2 Exemple 2 : Commande de la température d'un processus thermique.

Dans ce cas, le système de commande doit souvent assurer un élément de contrôle séquentiel. Ainsi, les facteurs qui influenceront les exigences de contrôle de séquence sont:

- Actionnement de la porte du four ;
- Mouvement de la sole ou de la hotte du four ;
- Mécanismes de transfert du four ;
- Transfert vers la trempe ;
- Contrôle de la trempe au gaz/ventilé ;
- Actionnement de la porte du blindage thermique ;
- Contrôle de l'eau de refroidissement.

Le système de commande est donné par la figure suivante [43] :

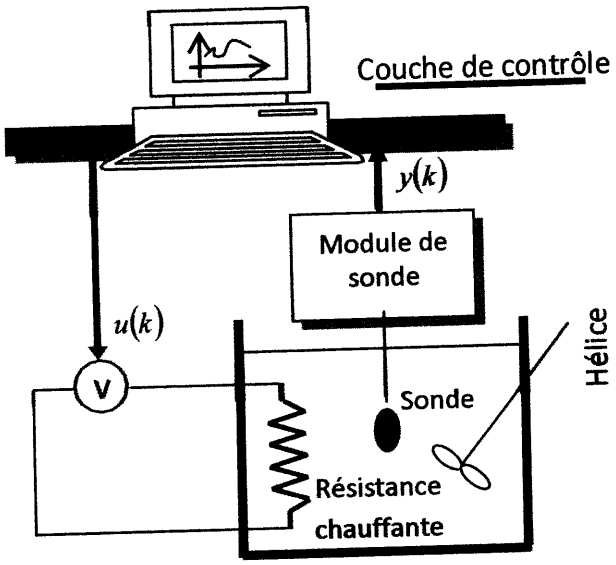


Figure III.7: *Structure du procédé.*

III.3.2.1 Le modèle mathématique du système

Le modèle du système est donné par [43] :

$$y(k+1)=Q_1(T_s)\cdot y(k)+\frac{Q_2(T_s)}{1+\exp(0.5y(k)-\gamma)}u(k)+(1-Q_1(T_s))\cdot Y_0+d_{ist}(k) \qquad \text{(III.11)}$$

Avec :

$$Q_1(T_s)=\exp(-\alpha T_s)$$
$$Q_2(T_s)=\frac{\beta}{\alpha}(1-\exp(-\alpha T_s)) \qquad \text{(III.12)}$$

où :

- *T(k)* : la température du système.
- *u(k)* : signal de commande à appliquer.
- y0 : La température ambiante.
- Ts : La période d'échantillonnage.
- α et β sont des constantes décrivant le système.

Les paramètres de système utilisés sont donnés par [43] :

Tableau III. 1 : Paramètres du système.

Coefficient	Valeur
α	$1.0015.10^{-4}$
β	$8.6793.10^{-3}$
γ	40
T_s [s]	30
Y_0 [°C]	25.0

L'objectif de commande est de permettre aux états du système de suivre une trajectoire de référence prédéterminée. Dans cette partie, l'objectif de contrôle est de générer une séquence d'actions $0[\text{v}] \le u(k) \le 5[\text{v}]$ permettant au système de suivre la trajectoire de référence donnée par :

$$y^q(k) = \begin{cases} 35[°C] & \text{si} \quad k \le 40 \\ 55[°C] & \text{si} \quad 40 < k \le 80 \\ 75[°C] & \text{si} \quad 80 < k \le 120 \end{cases} \qquad \text{(III.13)}$$

Un régulateur PID en utilisant un AG binaire, Sélection par roue de fortune, Croisement et mutation simples ; où les probabilités sont dynamiques.

III.3.2.2 Résultat de simulation:

Cette figure représente la minimisation de fitness en fonction de la génération ; on remarque que la solution optimale est donnée pour un nombre de génération égale à 50.

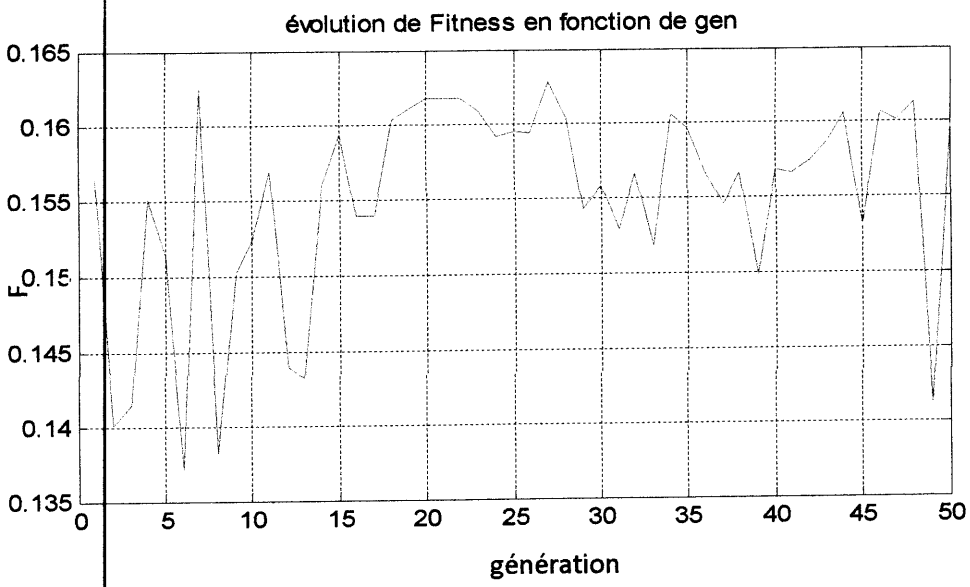


Figure III.8 : *L'évolution de fitness en fonction de génération.*

On obtient les valeurs des gains comme suit : $K_p = 60, K_d = 1.428, K_i = 5$

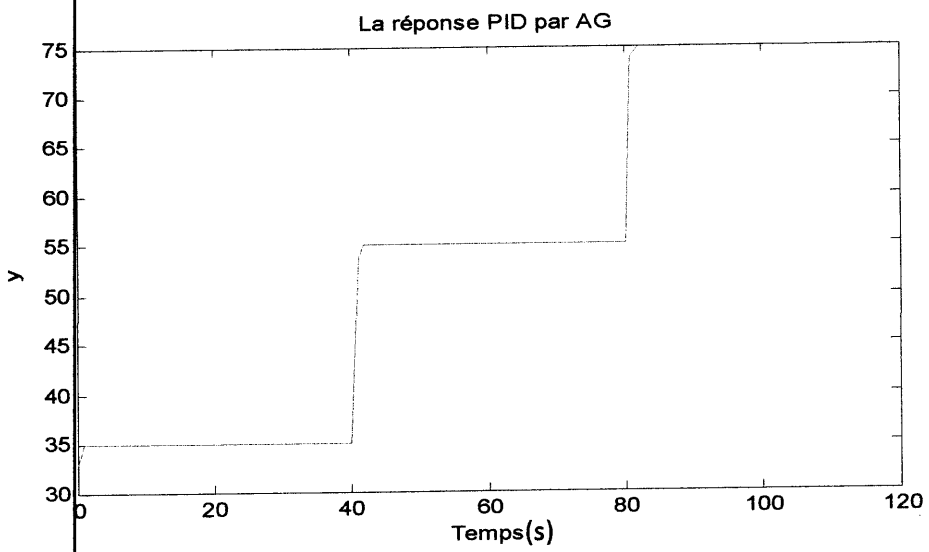


Figure III.9 : *La commande PID par AG.*

La figure III.9 illustre le changement proportionnel de la sortie y en fonction du temps pour les valeurs de PID données ci-avant.

III.4 Le système du réservoir surtension

Dans ce cas, on étudie un système utilisé pour la régulation des fluides dans les pompes de surtension agissant comme des réservoirs. Ces pompes régulatrices de surtension peuvent modifier le flux, la température de la composition ainsi que la pression des fluides. Afin de modéliser une pompe de réglage de surtension avec précision, il faut prendre en compte toutes les données physiques agissant sur le système pour pouvoir établir des relations entre les différentes caractéristiques. Le Schéma bloc fonctionnel du réservoir est donné par la figure suivante.

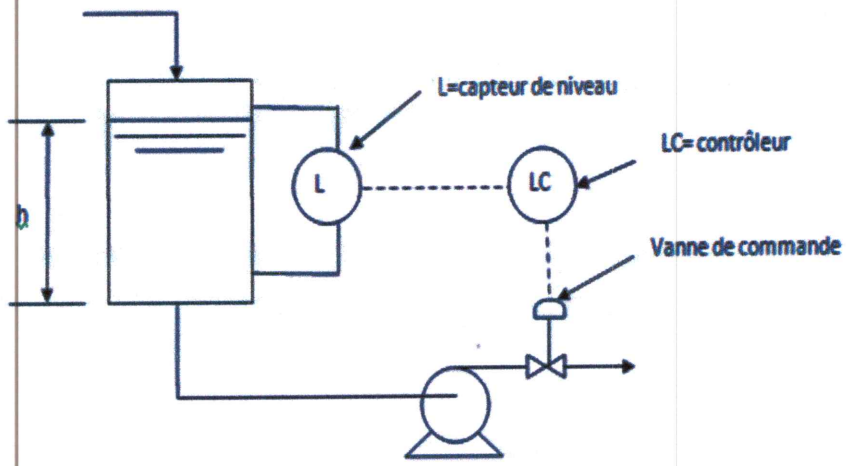


Figure III.10: *Schéma bloc fonctionnel du réservoir.*

III.4.1 Le modèle mathématique du réservoir dynamique

L'équation différentielle du système donnée par

$$\frac{dy(t)}{dt} = \frac{-c\sqrt{2gh(t)}}{A_r(h(k))} + \frac{T}{A_r(h(k))}\,u(t) \qquad (III.14)$$

Où :

- u(t) est le flux entrant qui peut être positif ou négatif.
- h(t) est le niveau du fluide étudie (grandeur de sortie).
- A_r(h(t)) est la coupe (section) du réservoir,
- g = 9.8m/sec^2est la l' accélération gravitationnelle
- C = 1 est la section du canal de sortie.

La section du réservoir est donnée par :

$$A_r(h(t))=\sqrt{ah(t)+b} \quad \text{(III.15)}$$

avec a=1 et b=3.

En utilisant l'algorithme d'Euler pour discrétise le système, on obtient :

$$h(k+1) = h(k) + T\left[\frac{-\sqrt{19.6h(k)}}{A_r(h(k))} + \frac{u(k)}{A_r(h(k))}\right] \quad \text{(III.16)}$$

où T=0.1, d=1,

$$f_0(x(k)) = h(k) - \frac{T\sqrt{19.6h(k)}}{A_r(h(k))} \quad \text{(III.17)}$$

Et

$$g_0(x(k)) = \frac{T}{A_r(h(k))} \quad \text{(III.18)}$$

Le système est testé pour h(t) > 0.Le teste de la qualité de commande du système non-linéaire choisi par le correcteur PID avec AG. Dans un premier temps, nous avons créé des modèles afin de simuler le comportement du système. Ensuite, nous avons implémenté notre retour d'état sur les modèles dans le but de faire un asservissement. Pour les simulations, on utilisé le modèles dynamique décrits par l'équation différentielle. La simulation inclue 1000 échantillonnages et a comme entrée de référence une impulsion carrée. Initialement, les paramètres de l'AG sont donnés comme pour les systèmes précédents comme indiqué dans le tableau III.2. Les probabilités de croisement et de mutation ont également un impact important sur le déroulement de la recherche de la solution optimale.

Tableau III.2 : *les paramètres initiaux d'AG*

paramètre	symbole	valeur
Taille de population	n_p	10
Probabilité de croisement	p_c	0.9
Probabilité de mutation	p_m	0.05

III.4.2 Résultats de simulation :

Les résultats de simulation pour l'optimisation du fonctionnement par AG sont donnés par les figures suivantes :

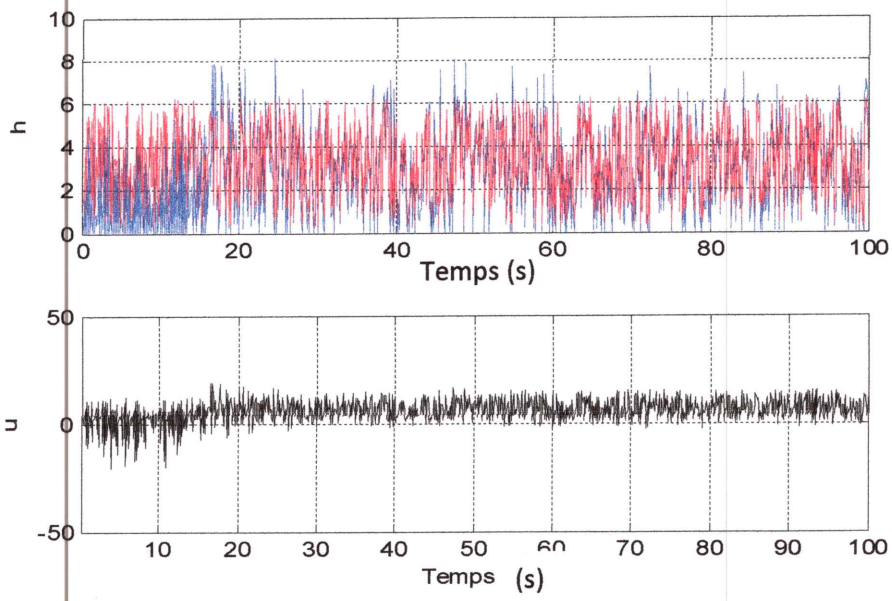


Figure III.11 : *Niveau du liquide et l'entrée de référence.*

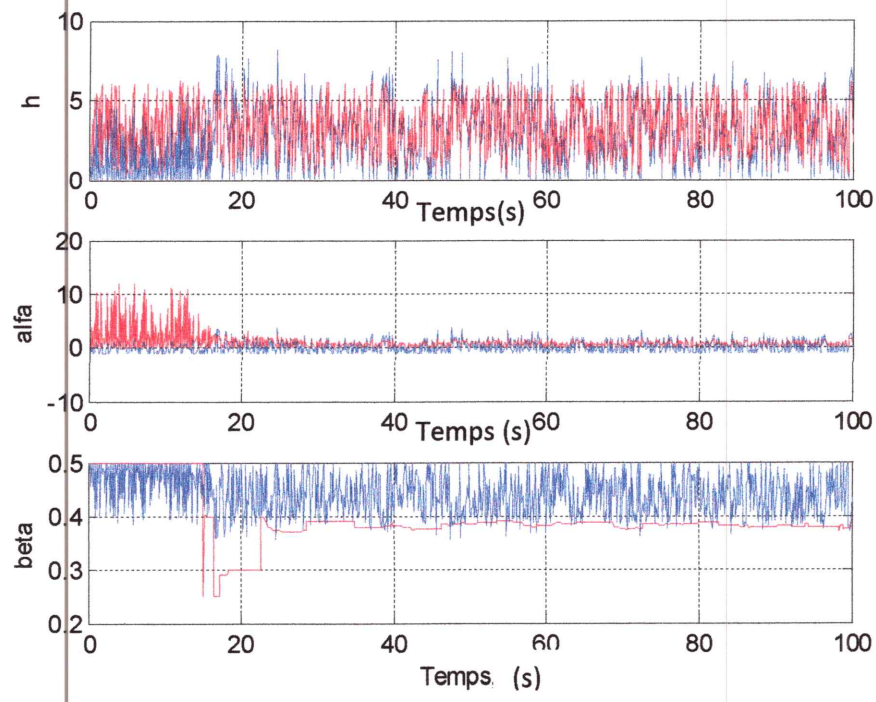


Figure III.12 : *Estimation des paramètres d'optimisation par AG.*

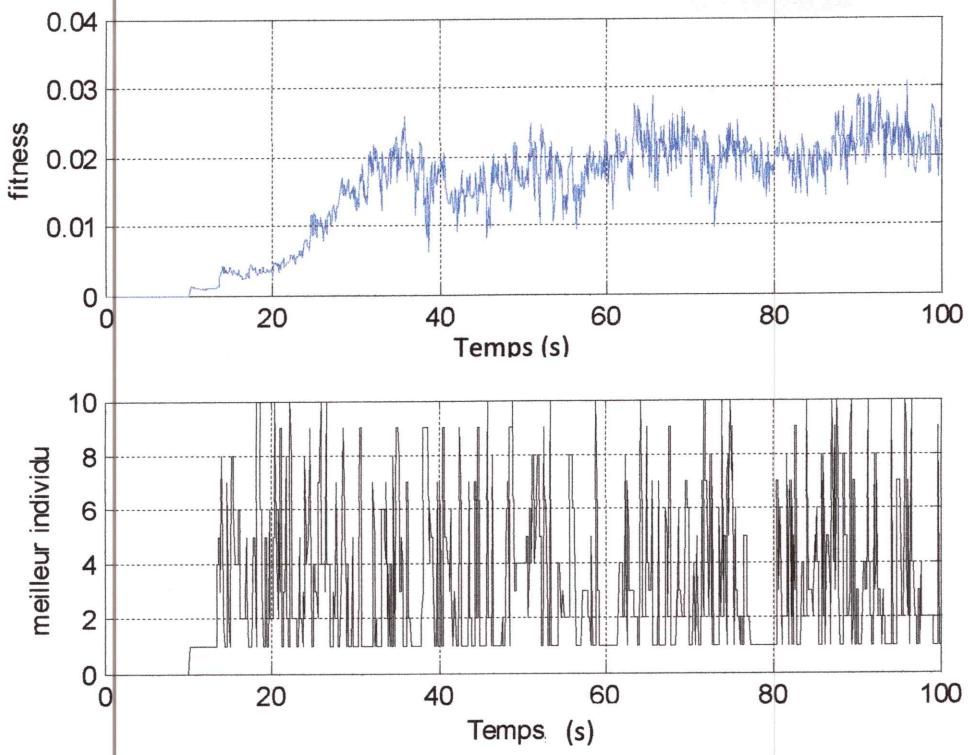


Figure III.13 : *La fonction fitness.*

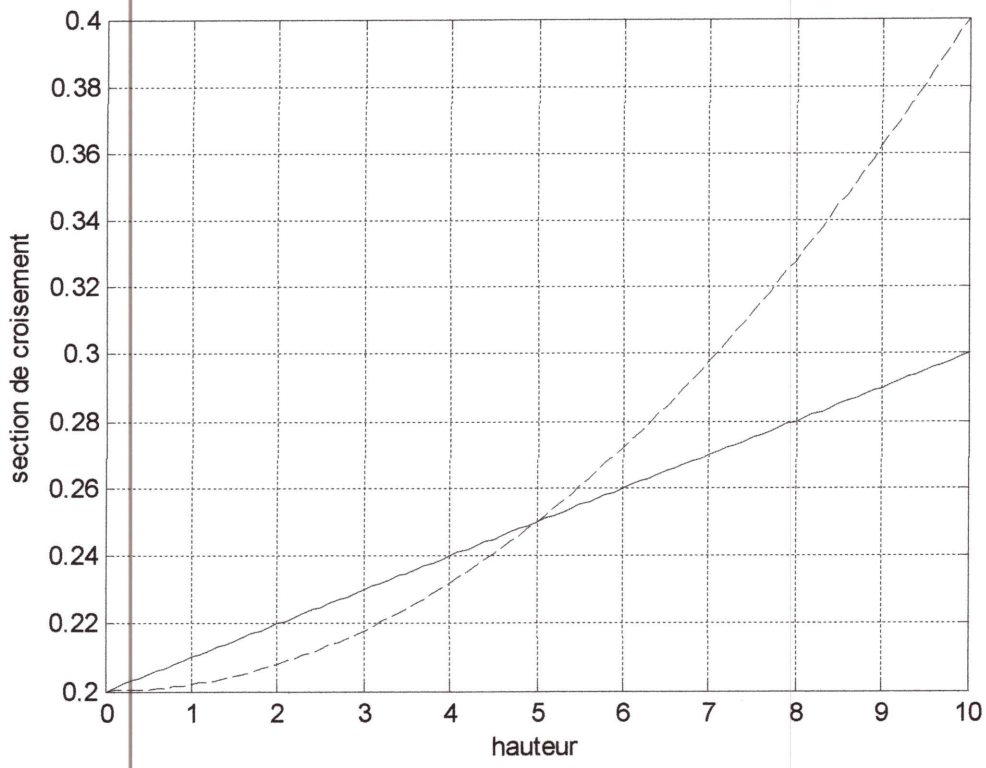


Figure III.14 : *Section de croisement.*

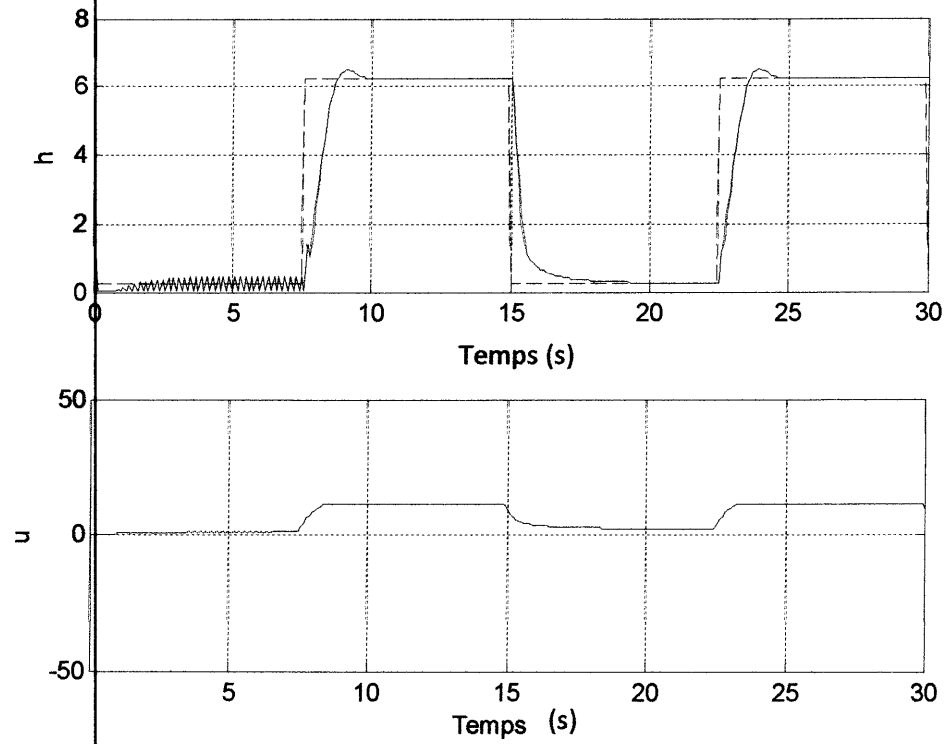


Figure III.15 : *Niveau du liquide et l'entrée de commande.*

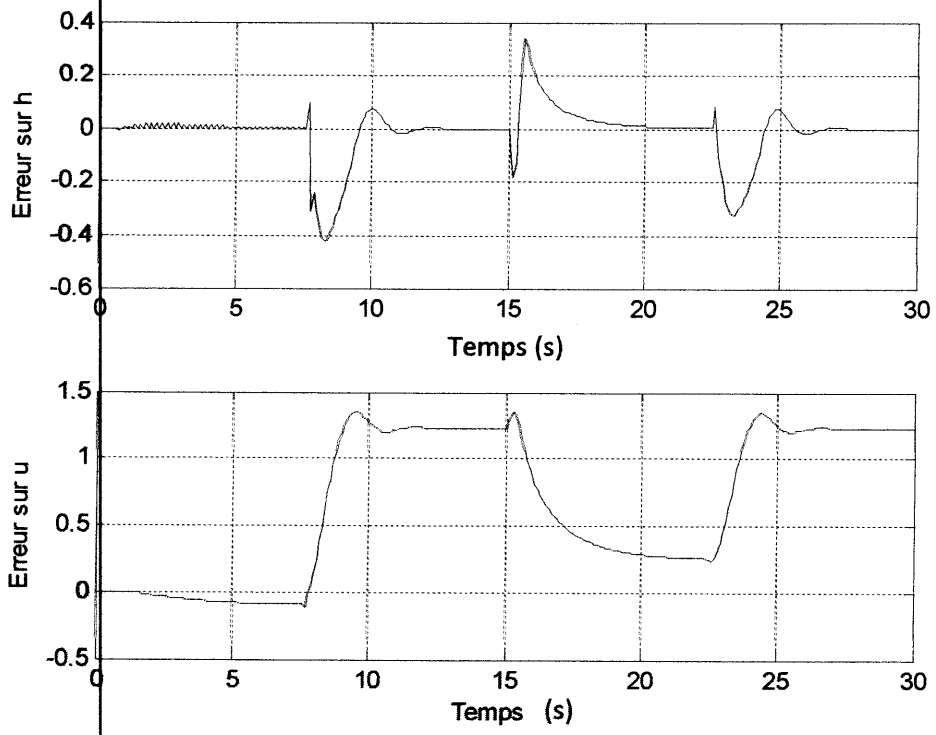


Figure III.16: *L'erreur sur le niveau du liquide et l'entrée de commande.*

III.4.3 Interprétation des résultats

Pour vérifier l'efficacité d'un régulateur PID basé sur l'AG, nous avons réalisé plusieurs simulations du système expérimental complet. Ceci a permis de donner des règles simples de synthèse et de paramétrisation de la méthode d'optimisation comme présenté sur la Figure III. 14.

D'après les résultats de simulation, on observe qu'avant l'application de la commande, le système (en BO) est toujours instable. Mais après l'introduction des gains du régulateur PID dans la chaîne de retour qui sont calculés par l'algorithme génétique, le système devient stable et il satisfera les exigences avec le minimum d'énergie. (Figure III.16). D'après la figure III.16 on remarque que le système passe par deux états : sur l'intervalle] 7, 25[le système est instable et sur l'intervalle] 25, 30[le système devient stable après l'introduction du PID. De plus, d'après les figures III.15 et III.16, on observe que le niveau du liquide se stabilise sur le niveau désiré tout en maintenant une erreur de commande sur des valeurs très petites. Ceci démontre l'efficacité de l'algorithme génétique pour le choix optimum des paramètres du régulateur PID.

III. 5 Conclusion

Dans ce chapitre, nous avons fait une étude détaillée sur trois cas pratiques, on a commencé par l'étude d'un circuit électrique simple, on a modélisé le système, et après sa linéarisation, on a appliqué des lois de commande pour minimiser un certain critère de performance afin d'obtenir une commande robuste. Les mêmes étapes sont réalisées pour la commande d'un processus thermique ainsi qu'un système réel de réglage de surtension dans un système de pompage. D'après les résultats de simulations, on peut confirmer que l'optimisation par AG permet de choisir les meilleures valeurs des paramètres des gains du régulateur PID au lieu de le faire par tâtonnement.

Conclusion générale

Le travail réalisé dans le cadre de ce mémoire a en grande partie porté sur l'étude et la commande PID avec optimisation par algorithme génétique de quelques systèmes dynamiques non linéaires. De ce fait, et de façon à améliorer les performances dynamiques, nous avons utilisé une compensation par retour d'état. La méthode consistant à optimiser les paramètres du contrôleur PID par l'utilisation d'une fonction fitness (cout) qui assure des réponses minimales de phase dans les cas étudiés.

On a commencé notre mémoire par l'introduction des notions fondamentales des algorithmes génétiques. Ensuite, on a abordé la commande PID par les algorithmes génétiques. On commence par donner un aperçu général sur les principaux critères et les domaines d'application de la commande PID ainsi que les critères de choix des paramètres PID. Enfin, on propose une étude détaillée sur des cas pratiques.

L'étude tout au long de ce mémoire a montré que la synthèse d'une commande PID par retour d'état nécessite la connaissance des paramètres du contrôleur. Or, ceci n'est généralement pas chose facile, soit par manque de connaissance sur le système étudié, soit parce que les paramètres choisies ne correspondent pas à une variable physique du système. Dans ce cas, il est nécessaire d'optimiser ces paramètres. On met donc en place, en parallèle sur le système considéré, une optimisation des paramètres par les algorithmes génétiques. Les résultats obtenus montrent l'efficacité de la démarche proposée.

BIBLIOGRAPHIE

[1] A. Boumaza, "Introduction de techniques d'évolution artificielle en vision tridimensionnelle et en robotique mobile", Thèse doctorat, Université Rene Descares– Paris 2004.

[2] R. Dupas, "Amélioration de performance des systèmes de production: apport des algorithmes évolutionnistes aux problèmes d'ordonnancement cycliques et flexibles", Université d'Artois 2004.

[3] Bàck, T, U. Hammel, and H, P. Schwefel. Evolutionary computation: Comments on the history and current state IEEE Transactions on Evolutionary Conputation 1(1), 3-17, 1997.

[4] S. Baudot-Roux. "Algorithmes évolutionnaires, hybridation" cf. PARCFD'99, Surveys on Mathematics 2000, Eurogen.

[5] P. Lucidarme. "Apprentissage et adaptation pour des ensembles de robots réactif coopérants". Proc. ICRA'02, Washington, 2002.

[6] A. Nabonne. "Algorithmes évolutionnaires et problèmes inverses", chapitre 8, juin 2004.

[7] Davis L., Genetic Algorithms Simulated Annealing. Pittman, London, 1987.

[8] Goldberg D. E., Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, inc., Reading, MA, January 1989.

[9] Michalewicz Z., Genetic Algorithms + Data Structures =Evolution Programs. Artificial Intelligence. Springer Verlag, NewYork, 1992.

[10] De Jong K. A., Analysis of the Behavior of a Class of GeneticAdaptiveSystems.PhD thesis, University of Michigan, 1975. 5140B.

[11] Baeck T., Schwefel H. P., An overview of evolutionnary algorithms for parameter optimisation. Technical report, University of Dortmund, 1992.

[12] Hoffmeister F., Bäck T., Genetic algorithms evolution strategies: similarities and differences. In H. P. Schwefel and R. Männer, editors, Parallel Problem Solving from Nature - Proceedings of 1st Workshop, PPSN 1, volume 496, 1991.Springer-Verlag, Berlin, Germany.

[13] Hoffmeister F., Baeck T., Genetic algorithms and evolution strategies: Similarities and differences. Technical Report SYS-1/92, University of Dortmund, February 1992.

[14] S. Bernard, "Algorithmes Evolutionnaires", Addison-Wesley, Paris 2003.

[15] E. Lutton, "Darwinisme artificiel ", INRIA - Rocquencourt - Equipe Complex.

[32] Site : //wwwsi.supelec.fr/yb/projets/algon/voydeCom/voydeCom.html.

[33] W.K. Lai, George G. Goghill, "Channel Assignment Thought Evolutionary Optimization '' IEEE transaction en vehicular technology, Vol. 45, N 01, 1996.

[34] Clerc et Siarry, « Une nouvelle métaheuristique pour l'optimisation difficile : la méthode des essaims particulaires »Vol. 3-7 France Télécom R&D; Université Paris 12, 2004.

[35] K. H. Ang, G. Chong and Y. Li, PID control system analysis, design, and technology.

[36] IEEE Transactions on Control Systems Technology, Vol.13, N.4, pp. 559-576, July 2005.

[37] Aström K.J. et Hägglund T. Automatic Tuning of PID Controllers. ISA Research Triangle Parc, 1988.

[38] Aström K.J. et Hägglund T – PID Controllers. ISA Research Triangle Parc, 1995.

[39] Aström K.J., Hang C.C., Persson P. et Ho W.K. Towards intelligent PID control Automatica. vol.28, p. 1-9, 1992.

[40] Belkacem Ould-Bouamama Regulation automatique , 1998 site : www.eudil.fr\eudil\belk\sc00a.htm

[41] Alina Besacon- Voda et Sylvianne Gentil, Régulateurs PID analogique et numérique Technique de l'ingénieur : R7416 (03/1999).

[42] D. E. Goldberg, Genetic Algorithms in search, optimization and machine learning. Reading, MA: Addison-Wesley, 1989.

[43] Ammar SOUKKOU, ''*Modélisation & Commande des Systèmes Industriels Complexes par les Techniques Intelligentes*'', Thèse de doctorat, Université de Sétif, Octobre 2008.