République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université 20 Août 1955 - Skikda

Faculté de Technologie
Département de Génie Electrique
Laboratoire d'Electrotechnique de Skikda

D012119009D

THÈSE

EN VUE DE L'OBTENTION DU DIPLOME DE DOCTORAT EN SCIENCES

Filière : Electrotechnique
Spécialité : Modélisation et Simulation des Installations Electriques Industrielles

Présentée par

LAOUAFI Farida

Thème

Optimisation du plan de tension et de la répartition de la puissance réactive par les techniques intelligentes hybrides améliorées

Soutenue publiquement le : 24 / 06 / 2019
Devant le Jury composé de :

M. MORDJAOUI Mourad	**Professeur**	**Université 20 Août 1955 - Skikda**	Président
M. BOUKADOUM Ahcene	**Professeur**	**Université 20 Août 1955 - Skikda**	Rapporteur
M. BOUDEBBOUZ Omar	**MCA**	**Université 20 Août 1955 - Skikda**	Examinateur
M. DIB Djalel	**Professeur**	**Université de Tébessa**	Examinateur
M. BOUZERARA Ramdane	**Professeur**	**Université de Constantine**	Examinateur
M. BAYADI Ablelhafid	**Professeur**	**Université de Sétif**	Examinateur

Année Universitaire : 2018 – 2019

Remerciements

Ce travail a été effectué au laboratoire d'électrotechnique de Skikda (LES) sous la direction de Messieurs : Salah LEULMI et Ahcene BOUKADOUM, Professeurs au département de génie électrique de l'université de Skikda.

Je remercie tout d'abord Dieu le tout Puissant qui, par sa grâce et sa guidance, a permis que cet effort aboutisse et réussisse, en me donnant la force, la patience et la volonté.

Ensuite, je tiens à remercier tout particulièrement mon directeur de thèse Pr. Salah LEULMI, le premier qui m'a donné la chance d'entrer dans le monde de la recherche. Je voudrais lui témoigner toute ma reconnaissance pour sa rigueur et ses conseils.

Mon codirecteur, Monsieur Ahcene BOUKADOUM, Professeur au département de génie électrique de l'université de Skikda, à qui je veux apporter mes remerciements tous particuliers pour sa disponibilité et son engagement scientifique. Il a su m'encadrer avec efficacité, clairvoyance et beaucoup de patience.

Je voudrais également exprimer mes remerciements aux membres du jury pour l'honneur qu'ils m'ont fait en acceptant de juger mon travail.

Mes remerciements s'adressent à Monsieur Mourad MORDJAOUI, Professeur au département de génie électrique de l'université de Skikda, de m'avoir fait l'honneur de présider le jury de ma soutenance.

Mes remerciements vont également à Monsieur Omar BOUDEBBOUZ, Maître de Conférence au département de génie électrique de l'université de Skikda; au Pr. Djalel DIB, Professeur à l'université de Tébessa; au Pr. Ramdane BOUZRARA, Professeur à l'université de Constantine; et au Pr. Ablelhafid BAYADI, Professeur à l'université de Sétif qui ont accepté d'évaluer ce travail et participer au jury de thèse comme examinateurs.

Dédicaces

Je dédie ce modeste effort, à mes très chers parents que Dieu les garde.

A mes sœurs, mes frères, mon mari et mes enfants Kaoutar et Ayoub.

À tous mes collègues de l'Université de Skikda.

ملخص

إن الهدف من هذا العمل هو حل مشكلة التوزيع الأمثل للاستطاعة الرد فعلية (ORPD) والذي يعتبر مشكلة أمثلية متعددة المتغيرات ، ذات قيود ودالة مراد تحسينها غير خطية . إن المبتغى من حل هذا المشكل يمكن تلخيصه بشكل رئيسي في التقليل من ضياع الاستطاعة الفعالة في الخطوط الكهربائية و كذا تحسين مستويات الجهد للشبكة الكهربائية باستعمال عدد من وسائل التحكم المتاحة مثل مولدات الجهد، منابع الاستطاعة الرد فعلية و المحولات القابلة لتغيير معامل تحويلها. إن الضبط الأمثل لقيم هذه الوسائل، من شأنه أن يضمن لنا توزيعا مثاليا للاستطاعة الرد فعلية و يقلل من ضياعات نقل الطاقة .

تعتبر خوارزميات الاستدلال الفوقي خوارزميات عشوائية قادرة على حل مجموعة واسعة من مشاكل التحسين التي لا توجد لحلها طرق تقليدية فعالة معروفة. غالباً ما تكون هذه التقنيات مستوحاة من علم الأحياء (خوارزميات تطورية، تطور تفاضلي)، الفيزياء (محاكاة الصلب، خوارزمية بحث الجاذبية) وعلم الأعراق (مستعمرات النمل، التحسين بأسراب الجسيمات). لقد أصبح الذكاء المستوحى من الطبيعة أكثر انتشارا، وتم تطوير عدد كبير من الطرق القائمة على مفاهيم من الطبيعة أو علم الأحياء. يتطلب أداء وكفاءة خوارزمية الاستدلال الفوقي توافقا بين استغلال واستكشاف مساحة البحث، ومع ذلك ، فمن النادر أن تتوفر الميزتان معا في نفس طريقة البحث ، و لهذا ساد الانتشار الحالي للطرق الهجينة.

في هذا العمل ، سنحاول حل مشكلة التحكم في الجهد/الاستطاعة الرد فعلية من خلال تطبيق العديد من خوارزميات الاستدلال الفوقي مثل: الخوارزميات الجينية المحسنة IGA، التحسين من خلال أسراب الجسيمات PSO، محاكاة الصلب SA ، التطور التفاضلي DE واستعمال تركيبتين هجينيتين. الأولى هي صيغة هجينة بين خوارزمية التحسين بسرب الجسيمات PSO و الخوارزميات الجينية المحسنة IGA ذات تشفير حقيقي. أما الثانية فهي صيغة هجينة بين خوارزمية التطور التفاضلي DE (القائمة على أساس إيجاد مجموعة من الحلول خلال كل تكرار) ومحاكاة الصلب SA (القائمة على أساس إيجاد حل واحد خلال كل تكرار).

بالنسبة لمتغيرات التحكم ، فقد تم اقتراح تمثيل مختلط (مستمر / متقطع). و قد تم اختبار كفاءة التقنيات المستخدمة على شبكتي اختبار قياسيتين هما Ward-Hale 6 bus و IEEE 30 bus .

الكلمات المفتاح : التوزيع الأمثل للاستطاعة الرد فعلية، تحسين مستوى الجهد، التهجين، الخوارزمية الجينية المحسنة ، التحسين بطريقة أسراب الجسيمات، التطور التفاضلي، محاكاة الصلب.

Résumé

Le but de ce travail est de résoudre le problème de la répartition optimale de la puissance réactive (ORPD) qui est considéré comme un problème d'optimisation multivariable, avec contraintes et à fonction objective non linéaire. Les objectifs peuvent être principalement résumés dans la minimisation des pertes de puissance active dans les lignes et l'amélioration du profil de tension dans un réseau électrique en utilisant un certain nombre de moyens de contrôle tels que les sources de puissance réactive, les gradins des transformateurs réglables en charge et les tensions des JdB des générateurs (PV JdB). L'ajustement optimal de ces moyens de contrôle peut mener à une répartition optimale de la puissance réactive et à réduire au minimum les pertes de transmission.

Les méta-heuristiques sont des algorithmes stochastiques capables de résoudre une large gamme de problèmes d'optimisation pour lesquels il n'existe pas de méthodes conventionnelles efficaces connues. Ces techniques sont souvent inspirées de la biologie (algorithmes évolutionnaires, évolution différentielle), de la physique (recuit simulé, algorithme de recherche gravitationnelle) et de l'ethnologie (colonies de fourmis, optimisation par essaim de particules). L'intelligence inspirée de la nature devient, de plus en plus, populaire et un nombre important de méthodes entraînées par des concepts issus de la nature ou de la biologie ont été développées. La performance d'une métaheuristique nécessite un compromis entre l'exploitation et l'exploration de l'espace de recherche. Cependant, il est rare d'avoir les deux caractéristiques dans la même méthode de recherche, où l'émergence actuelle des méthodes hybrides.

Dans ce travail, nous tenterons de résoudre le problème du contrôle de tension / puissance réactive en appliquant plusieurs métaheuristiques telles que: les algorithmes génétiques améliorés IGA, l'optimisation par essaims particulaires PSO, le recuit simulé SA, l'évolution différentielle DE et deux formulations hybrides. La première est une formulation hybride entre l'optimisation par essaims particulaires PSO et les algorithmes génétiques améliorés avec un codage réel IGA. La deuxième est une formulation hybride entre l'algorithme de l'évolution différentielle DE (basé sur une population de solution) et le recuit simulé SA (basé sur une solution unique).

Pour les variables de contrôle, une représentation mixte (continue / discrète) est proposée. La robustesse des techniques utilisées est testée sur deux réseaux standards de test : Ward-Hale 6 bus et IEEE 30 bus.

Mots clés : Répartition optimale de la puissance réactive, amélioration du plan de tension, hybridation, algorithme génétique amélioré, optimisation par essaims particulaires, évolution différentielle, recuit simulé.

Abstract

The aim of this work is to solve the optimal reactive power dispatch (ORPD) problem which is a multi constrained nonlinear multivariable optimization problem. The objectives are to minimize real power losses and improve the voltage profile of a given electric power system by using a number of control tools such as switching VAr sources, changing generator voltages and adjusting transformer settings. By an optimal adjustment of these controllers, the optimal distribution of reactive power would minimize transmission losses.

Metaheuristic algorithms have been extensively used to solve optimization problems in a reasonable time without requiring in-depth knowledge of the treated problem. These techniques are often inspired from biology (Evolutionary algorithms, Differential evolution), physics (Simulated annealing, Gravitational search algorithm) and ethnology (ant colonies, particle swarm optimization). The intelligence inspired from nature became more and more popular and a large number of methods driven by concepts from nature or biology have been developed. The performance of a metaheuristic requires a compromise between exploitation and exploration of the search space. However, it is rarely to have the two characteristics in the same search method, where the current emergence of hybrid methods.

In this work, we will try to solve the problem of voltage / reactive power control by applying several metaheuristics such as: improved genetic algorithms IGA, particle swarm optimization PSO, simulated annealing SA, differential evolution DE and two hybrid formulations. The first one is a hybrid formulation between particle swarm optimization PSO and improved genetic algorithms with real coding IGA. While the second is a hybrid formulation between two different metaheuristics: differential evolution DE (based on a population of solution) and simulated annealing SA (based on a unique solution).

For the control variables, a mixed representation (continuous/discrete), is proposed. The robustness of the used techniques is tested on the Ward-Hale 6 bus system and the IEEE 30 bus test system.

Key words: Reactive power dispatch optimization, voltage profile improvement, hybridization, improved genetic algorithms, particle swarm optimization, differential evolution, simulated annealing.

TABLE DES MATIERES

Liste des Abréviations

ABC	Colonie d'abeille artificielle.
ACO	Optimisation par colonies de fourmis.
AIS	Systèmes immunitaires artificiels.
AIT	Techniques intelligentes artificielles.
ANN	Réseaux de neurones artificiels.
BFA	Algorithme de la recherche de nourriture bactérienne.
BH	Trous noirs (Black Holes).
CEA	Algorithmes co-évolutionnaires.
CS	La recherche coucou (Cuckoo Search).
DE	Evolution différentielle.
DE-ACO	Optimisation par évolution différentielle-colonie de fourmis.
EA	Algorithme évolutionnaire.
EP	Programmation évolutionnaire.
ES	Stratégies d'évolution.
FACTS	Systèmes flexibles de transmission à courant alternatif.
FL	Logique floue.
GA	Algorithme génétique.
GP	Programmation génétique.
GS	Recherche gravitationnelle.
GWO	Optimisation du loup gris (Gray Wolf Optimizer).
HDESA	Hybridation entre l'évolution différentielle DE et le recuit simulé SA.
HEC	Calcul évolutionnaire hybride.
HMPSO	Algorithme hybride d'optimisation par essaim particulaire multi-essaim.
HPSO	Optimisation par essaim particulaire hybride.
HPSOIGA	Hybridation entre l'essaim particulaire PSO et l'algorithme génétique amélioré IGA.
HS	Algorithme de recherche d'harmonie.
IGA	Algorithme génétique amélioré.
IPSO	Optimisation par essaim particulaire amélioré.
JdB	Jeux de barres.

KHA	Algorithme du troupeau de krill (Krill Herd algorithm).
LP	Programmation linéaire.
OPF	Ecoulement optimal de puissance.
ORPD	Répartition optimale de la puissance réactive.
PSO	Optimisation par essaim particulaire.
QC	Calcul quantique (quantum computation).
QP	Programmation quadratique.
SA	Recuit simulé.
SVC	Compensateur statique.
AIT	Technique intelligente artificielle.

LISTE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION GENERALE

En raison de l'existence des limites sur la production et le transport de la puissance réactive dans les réseaux électriques et les besoins de la puissance réactive pour les consommateurs, la répartition optimale de la puissance réactive a attiré l'attention des chercheurs pendant les cinquante dernières années. De l'autre côté, les sous-tensions et les surtensions dans les lignes causent l'instabilité du réseau électrique, la décroissance de la qualité de l'énergie et la destruction de l'isolation d'équipement. Tout changement dans la demande ou dans la configuration du réseau peut causer une augmentation ou une diminution du niveau de tension.

Dans les réseaux électriques, la tension constitue, avec la fréquence, un des principaux paramètres de la sureté de fonctionnement du système. Si la fréquence a une grande dépendance avec les variations de la puissance active générée, transportée et consommée, les modules de la tension seront affectés, principalement, par les variations de la puissance réactive. Ainsi, l'optimisation du plan de tension d'un réseau électrique, suppose donc, de maîtriser les transits d'énergie réactive et de minimiser les pertes dans le réseau.

L'objectif principal de la répartition optimale de la puissance réactive est de réduire au minimum les pertes actives et d'améliorer le profil de tension dans les réseaux électriques en utilisant un nombre de moyens de contrôle (sources de puissance réactive, générateurs de tension et transformateurs réglables). Par un ajustement optimal de ces moyens de contrôle, on peut obtenir une distribution optimale de la puissance réactive. Alors résoudre ce problème revient à résoudre un problème d'optimisation (minimisation) avec contraintes.

De nombreuses approches ont été proposées pour résoudre ce problème par différentes méthodes mathématiques et par l'utilisation de certaines simplifications et traitements spéciaux : la méthode de Gradient, la programmation quadratique, la programmation linéaire, la méthode de Newton et la technique du point intérieur. Cependant, toutes ces techniques ont beaucoup de problèmes tels que:

- la convergence vers une solution locale,
- le grand nombre d'itérations,
- la sensibilité à un point de recherche initial,
- la capacité de modélisation limitée (dans la gestion des contraintes et fonctions non linéaires et discontinues).

Ces problèmes peuvent être examinés par l'introduction des techniques intelligentes telles que les réseaux de neurones, la logique floue et les algorithmes évolutionnaires.

Avec l'évolution de l'informatique au cours des dernières années, de nombreuses nouvelles méthodes de recherche stochastique ont été développées pour les problèmes d'optimisation globale.

Les méta-heuristiques sont des algorithmes stochastiques capables de résoudre une large gamme de problèmes pour lesquels il n'existe pas de méthodes conventionnelles efficaces connues.

Ces techniques sont souvent inspirées de la biologie (algorithmes évolutionnaires, évolution différentielle), de la physique (recuit simulé) et de l'ethnologie (colonies de fourmis, optimisation par essaim de particules).

L'intelligence inspirée de la nature devient, de plus en plus, populaire et un nombre important de méthodes entraînées par des concepts issus de la nature ou de la biologie ont été développées.

Pour l'amélioration des performances des algorithmes d'optimisation, les algorithmes hybrides sont émergés. L'hybridation consiste à combiner les caractéristiques de deux méthodes différentes (ou plus) pour en tirer les avantages des deux méthodes et compenser les inconvénients subis par les deux algorithmes. Une hybridation idéale engendre une méthode hybride qui cumule les bonnes propriétés de ses constituants tout en inhibant leurs points faibles.

Ce modeste travail présente des techniques intelligentes simples et/ou hybrides dans leur forme améliorée (adaptative) pour résoudre le problème du contrôle optimal de tension/puissance réactive.

A ce propos, le manuscrit est constitué de cinq chapitres. Dans le premier chapitre, nous avons survolé l'historique et la synthèse bibliographique, relatives aux différentes techniques et méthodes d'optimisation classiques et modernes (techniques intelligentes & métaheuristiques), relatives à la résolution du problème du contrôle optimal de tension/puissance réactive. La formulation du problème du contrôle optimal de tension/puissance réactive du point de vue physique et mathématique, les objectifs et les contributions souhaités du présent travail sont, eux aussi, évoqués.

Le deuxième chapitre présente un état de l'art de l'optimisation et une classification de ses méthodes de résolution. Il ne s'agit pas de décrire en profondeur chacun de ces modèles, mais simplement de faire une synthèse de recensement des méthodes les plus répandues.

Dans le troisième chapitre, on va étudier avec détail, les métaheuristiques les plus populaires, qui vont être utilisées par la suite pour résoudre le problème du contrôle optimal de tension/puissance réactive. Les techniques l'hybridation vont être aussi discutées.

Le quatrième chapitre est consacré à la présentation des résultats de simulation suite à l'application de nos algorithmes.

La thèse se termine par une suite de conclusions et un ensemble de perspectives pouvant être développées par la suite, ouvrant ainsi de nouvelles voies et propositions dans le but d'améliorer ce travail et d'entamer de futures recherches d'habilitations à diriger.

1 - SYNTHESE BIBLIOGRAPHIQUE, PROBLEMATIQUE ET OBJECTIFS

1.1 - INTRODUCTION

La répartition optimale de la puissance réactive (ORPD) joue un rôle principal pour un fonctionnement sûr et économique d'un système d'énergie. En réalité, le problème de la répartition optimale de la puissance réactive est un problème de l'écoulement optimal de puissance (OPF) dont le but est de :

- Contrôler les tensions dans des gammes acceptables ;
- Minimiser la perte totale de la puissance en considérant d'autres contraintes des variables ;
- Eviter l'utilisation non-économique des transformateurs réglables et des sources de puissance réactive ;

L'opération économique du système d'énergie comporte 2 aspects : la régulation de la puissance active et la répartition de la puissance réactive.

L'écoulement optimal de puissance (OPF) a été présenté et employé pour l'opération et la planification économique des systèmes d'énergie. C'est un problème d'optimisation non-linéaire des systèmes à grande échelle. Ce problème est considéré par convention en tant que 2 problèmes séparés : P- et Q-problèmes. Le P-problème doit régler les sorties en puissance active des générateurs pour réduire au minimum le coût total (le dispatching économique). Le Q-problème doit contrôler les tensions des JdB des générateurs (PV JdB), les gradins des transformateurs réglables en charge et les sources de puissance réactive pour réduire au minimum les pertes de la puissance active du réseau.

La solution de ces problèmes est sujette à un certain nombre de contraintes, telles que les limites sur les tensions des JdB, les rapports des transformateurs, les puissances active et réactive des générateurs de puissance et des lignes de transmission et un certain nombre de variables,…etc.

1.2 – ETUDE HISTORIQUE ET SYNTHESE BIBLIOGRAPHIQUE

Le développement d'une solution optimale, pour le fonctionnement et l'exploitation des réseaux électriques, a été lancé en 1958 par L. K. Kirchmayer [1], en vue de minimiser le coût

d'exploitation de la fourniture d'énergie électrique à une charge donnée. Ainsi, le problème a évolué comme étant un problème de dispatch.

A ce moment là, l'écoulement ordinaire de puissance a fait des progrès considérables et l'utilisation des ordinateurs a montré des aspects prometteurs. Par conséquent, les analystes ont essayé d'incorporer ce succès dans le domaine de l'OPF.

En 1962, une formulation généralisée de la programmation non linéaire a été proposée par Carpentier [2] pour traiter le problème du dispatching économique tenant en compte la tension et d'autres contraintes d'exploitation. Des progrès conceptuels et remarquables ont été accomplis par Dommel et Tinney en 1968 [3]. Ils ont formulé les conditions exactes d'optimalité pour un OPF qui a permis l'utilisation de la solution d'un écoulement ordinaire de puissance. Le domaine de l'OPF devient, alors, un secteur actif de recherche.

Traditionnellement, la minimisation des pertes de transmission a été considérée comme étant la formulation du problème de la répartition optimale de la puissance réactive (ORPD).

Le problème de la (ORPD) est le problème le plus difficile dans le domaine de l'OPF. Son but principal est d'améliorer le profil de tension et de minimiser les pertes de la puissance active dans les réseaux électriques par l'ajustement nécessaire d'un nombre de moyens de contrôle.

L'ORPD est un problème d'optimisation multivariable, avec contraintes et à fonction objective non linéaire. Il a été bien reconnu que c'est un problème à programmation mathématique très compliquée. De nombreuses approches ont été proposées pour résoudre ce problème par différentes méthodes mathématiques et par l'utilisation de certaines simplifications et traitements spéciaux [3-8] : la méthode du gradient, la programmation linéaire (LP), la programmation quadratique non linéaire (QP), la méthode de Newton et la technique des points intérieurs…

La ORPD est un problème d'optimisation globale, qui peut avoir plusieurs minimums locaux et l'utilisation des méthodes d'optimisation mathématique peut, facilement, mener à des optimums locaux et parfois à une divergence. Généralement, ces techniques peuvent souffrir de quelques inconvénients en raison de la complexité algorithmique, des problèmes de la convergence, de la sensibilité à un point initial de recherche, …etc. A titre d'exemple :

- Vu la non linéarité et la non continuité de la fonction objective de l'ORPD, la méthode du gradient ne peut pas résoudre ce problème d'une manière suffisante.
- La programmation linéaire (LP) a été connue comme un outil puissant pour traiter les grands problèmes d'optimisation avec des fonctions linéarisées. Dues aux approximations introduites par les modèles linéarisés, les résultats obtenus par la LP ne peuvent pas, toujours, représenter la solution optimale de la fonction objective non linéaire.

La QP est une bonne approximation au problème de l'ORPD. Mais, elle souffre de la difficulté de calcul et d'un problème du temps de convergence.

La méthode de Newton a des limitations sévères avec le traitement des fonctions non linéaires, discontinues et à contraintes.

Le tableau (1.1) présente les avantages et les inconvénients de quelques méthodes mathématiques d'optimisation.

Tab. 1.1 - Avantages et inconvénients de quelques méthodes classiques d'optimisation

Méthodes classiques	Avantages	Inconvénients
Gradient	- Simple.	- insuffisante pour résoudre les problèmes à fonction objective non linéarité et non continue ; - la méthode risque de converger vers un optimum local dépendant du point de départ (initial) choisi.
Programmation linéaire	- puissante pour traiter les grands problèmes d'optimisation avec fonctions linéarisées.	- moins précise pour les problèmes non linéaires.
Programmation quadratique	- peut résoudre les problèmes d'optimisation avec fonctions objectives linéaires ou non (une bonne approximation au problème de l'ORPD).	- difficulté de calcul ; - problème du temps de convergence.
Newton	- algorithme simple ; - rapide si la solution initiale est prés de l'optimum ; - peut résoudre les problèmes d'optimisation avec fonctions objectives linéaires ou non linéaires.	- limitations sévères avec le traitement des fonctions non linéaires, discontinues et à contraintes ; - la solution initiale doit être proche d'un minimum local ; - lente pour les grands systèmes ; - ne converge pas toujours à la solution.
Points intérieurs	- convergence rapide vers la solution quelque soit la taille du système ; - peut résoudre les problèmes d'optimisation avec fonctions objectives linéaires ou non linéaires.	- la solution peut ne pas satisfaire toutes les contraintes d'inégalités ; - nécessite la résolution d'un système plus grand (les paramètres de Lagrange).

Ces dernières années, le domaine des systèmes d'énergie a connu l'utilisation de nouvelles méthodes basées sur les techniques intelligentes artificielles (AIT) telles que les réseaux de neurones (ANN) **[**9], la logique floue (FL) [10-11] et les algorithmes génétiques (GA) [12-20].

Le développement de l'informatique et des algorithmes évolutionnaires, au cours de la dernière décennie, a poussé les chercheurs à mieux examiner ces problèmes. Les avantages des algorithmes évolutionnaires en termes de capacité de modélisation et de la puissance de recherche ont encouragé leur application au problème de l'ORPD dans les réseaux électriques.

Q. H. Wu et J. T. Ma [13] ont proposé une programmation évolutionnaire (EP) pour la répartition optimale de la puissance réactive et le contrôle de la tension dans les réseaux électriques. Comparés à ceux obtenus par la méthode du gradient, les résultats de simulation ont montré l'efficacité de la méthode proposée [13].

J. T. Ma et L. L. Lai [14] ont présenté un algorithme génétique amélioré (IGA) pour résoudre le problème de l'ORPD. Les paramètres de l'algorithme génétique tels que la probabilité de mutation, le croisement et la technique du codage, sont améliorés pour rendre le GA un algorithme pratique dans l'optimisation des réseaux réels à grande échelle. Pour un GA classique, la probabilité de mutation est fixe. Dans ce travail, la probabilité de mutation est adaptative (variable).

Un algorithme génétique adaptatif (AGA) est proposé dans la référence [15]. Pour ce type des GA, les probabilités de croisement et de mutation sont variables selon les valeurs d'adaptation des solutions obtenues de chaque population.

Dans l'article [16], l'utilisation des GA est considérée et proposée pour résoudre le problème de l'ORPD. L'approche des GA a généré avec succès des valeurs appropriées des condensateurs shunts et des transformateurs réglables qui devraient être placés sur les JdB et sur les lignes pour produire un minimum de perte de puissance.

Un GA amélioré est proposé dans la référence [17] pour résoudre le problème de la puissance réactive dans les systèmes d'énergie. Basé sur un GA standard, un codage mixte (entier/réel) est utilisé. Les tensions nodales des générateurs sont codées pour être des valeurs réelles. Les gradins des transformateurs et le nombre des batteries de condensateurs installées dans les JdB sont codées pour être des nombres entiers. Ces 3 types de variables sont codés et placés dans un seul chromosome.

Pour résoudre le problème de la vitesse de convergence dû à la taille vaste de la population et d'un grand nombre des individus dans les GA, l'utilisation d'un GA synthétique et rapide a été proposée pour l'appliquer au problème de contrôle de l'ORPD [18].

A. Simoes et E. Costa [20] ont introduit un opérateur inspiré de la recombinaison biologique, qui a lieu dans les colonies de bactérie. Ce mécanisme est appelé « transformation ». Il est responsable des variations génétiques, par conséquent, aux caractéristiques prépondérantes que certaines bactéries peuvent les imposer. Ce mécanisme est incorporé dans un GA standard et comparé avec celui qui utilise un opérateur de croisement traditionnel.

Des mécanismes de transformation et de transposition sont utilisés dans la référence [21] comme alternative à l'opérateur de croisement traditionnel. Ces variantes des GA ont été utilisées pour l'optimisation des réacteurs tubulaires.

Un algorithme génétique auto-adaptatif à codage réel est proposé dans la référence [22] pour résoudre le problème de l'ORPD. La formulation du problème contient des variables de décision

continues (tensions des générateurs), discrètes (rapports des transformateurs) et binaires (sources de puissance réactive).

Dans la dernière décennie, de nombreuses nouvelles méthodes de recherche stochastique ont été développés pour les problèmes d'optimisation globaux. Les métaheuristiques sont des algorithmes de type stochastique visant à résoudre une large gamme de problèmes, pour lesquels on ne connaît pas de méthodes classiques plus efficaces. Souvent inspirées d'analogies avec la réalité comme la physique (recuit simulé, diffusion simulée...), la biologie (algorithmes évolutionnaires, évolution différentielle, recherche tabou...) et l'éthologie (colonies de fourmis, essaims particulaires...). Elles sont, généralement, conçues au départ pour des problèmes discrets, mais peuvent s'adapter aux autres types de problèmes [23-28] tels que :

- la minimisation du coût de production d'énergie active d'un réseau électrique;
- la planification active / réactive des systèmes d'énergie;
- l'ORPD.

L'optimisation par colonies de fourmis (ACO : Ant Colony Optimization) a été mise au point par Dorigo au début des années 90. Elle résulte de l'observation des insectes sociaux, en particulier des fourmis, qui résolvent, naturellement, des problèmes complexes [23-28].

Le système de fourmis a été utilisé dans de nombreux problèmes. K. Socha et M. Dorigo [23] ont présenté une extension de l'optimisation par colonies de fourmis au domaine continu. Ils ont montré comment l'ACO, qui est, initialement, développé pour être une métaheuristique pour résoudre les problèmes d'optimisation combinatoire, peut être adapté aux problèmes d'optimisation continue.

Une optimisation à colonie de fourmis a été utilisée pour minimiser le coût de production d'énergie active d'un réseau électrique Irakien, tout en maintenant la sécurité du système [24]. K.Y. Lee et J. G. Vlachogiannis [25] proposent l'application de l'optimisation par colonie de fourmis (ACO) pour la planification active / réactive des systèmes d'énergie. Les résultats sont comparés à ceux obtenus par la méthode du recuit simulé. L'optimisation par colonie de fourmis est utilisée comme approche pour résoudre le problème de l'ORPD [26-28].

L'optimisation par essaim particulaire (PSO : particle swarm optimisation) a été proposée par Kennedy et Eberhart en 1995 [29]. Elle s'inspire du comportement social des animaux évoluant en essaim, tels que les nuées d'oiseaux et les bancs de poissons.

Pour augmenter la vitesse de convergence et éviter les minimums locaux, une optimisation par essaims particulaires avec mutation a été examinée par [30]. La notion de chaos et des cartes chaotiques est intégré dans [31] et [32] pour l'adaptation des paramètres de la PSO.

L'essaim de particules est une technique adaptable et applicable à une large classe de problèmes dans les systèmes d'énergie tels que le problème de l'ORPD [33-35], le dispatching

économique [36-41], l'emplacement optimal des systèmes flexibles de transmission à courant alternatif (FACTS) avec un coût minimal d'installation [42], la taille et l'emplacement optimal des batteries de condensateurs shunts pour améliorer la stabilité de tension dans un réseau électrique [43-44]. D'autres applications de la PSO dans les systèmes de contrôle et de communication sont indiquées dans les références [45] et [46].

Dans la référence [47], le paramètre appelé poids d'inertie, a été introduit dans l'algorithme original de l'optimisation par essaim de particules. Des simulations ont été réalisées pour illustrer l'impact de ce nouveau paramètre sur l'algorithme de l'optimisation par essaim de particules.

L'évolution différentielle (DE) est une technique d'optimisation continue proposée par Storn et Price en 1997 [48]. Elle appartient comme les GA, à la famille des algorithmes évolutionnaires. Elle est, aujourd'hui, utilisée dans différents domaines nécessitant une optimisation comme l'OPF et ses différentes applications [49-59]. Une étude détaillée sur la relation entre les valeurs des paramètres de contrôle de l'évolution différentielle et la convergence est faite par [60].

Le tableau (1.2) présente une comparaison générale entre les principales métaheuristiques les plus populaires.

Dans les dernières années, plusieurs processus biologiques et naturels ont nuancé beaucoup de chercheurs à établir de nouvelles méthodes d'optimisation d'une manière croissante [61-70]. L'intelligence inspirée de la nature devient, de plus en plus, populaire et un nombre important de méthodes entraînées par des concepts issus de la nature ou de la biologie ont été développées. Plusieurs techniques ont été adaptées pour résoudre les problèmes de type OPF.

- Récemment (spécialement en 2005), un certain nombre d'algorithmes d'intelligence d'essaim, basée sur le comportement des abeilles, ont été présentés. L'algorithme de colonie d'abeille artificielle (ABC : Artificiel Bee Colony) est un algorithme de recherche à base de population proposée par D. Karaboga en 2005 [61-62].

Un algorithme de colonie d'abeilles artificielles (ABC) est appliqué par Ali Ozturk et ses collègues [61] pour traiter le problème de l'ORPD comme un problème d'optimisation multi-objective. Ses 3 fonctions objectives utilisées sont : les pertes de puissance active, le profil de tension des JdB de charge et le coût des sources de puissance réactive. Le même type de problème est résolu par la technique ABC dans la référence [62], mais avec une optimisation multiobjective, qui utilise 2 fonctions objectives à optimiser dans la frontière de Pareto : les pertes des puissances actives et les déviations de la tension des JdB de charge. En outre, la théorie des ensembles flous est utilisée pour extraire la meilleure solution de compromis.

Tab. 1.2 - Comparaison générale entre les principales métaheuristiques les plus populaires

Techniques	Avantages	Inconvénients
GA	- capables de trouver de bonnes solutions sur des problèmes très complexes ; - capable de trouver des solutions pour les problèmes impliquant de multiples contraintes, objectifs et avec tous types de variables ; - ils permettent de traiter des problèmes où la fonction à optimiser ne présente aucune propriété de continuité ou de dérivabilité, par exemple.	- coûteux en temps de calcul, puisqu'ils manipulent plusieurs solutions simultanément ; - nécessité le réglage des paramètres de la méthode (la performance de la recherche dépend du choix de ces paramètres) ; - dans certains cas, le problème de la convergence prématurée peut être posé.
ACO	- il est possible de l'adapter à tous les grands problèmes combinatoires classiques ; - connue par sa flexibilité et sa capacité d'éviter d'atteindre un optimum local.	- coûteuse en temps de calcul ; - stagnation et convergence lente vers une solution optimale ; - cette technique a montré qu'elle n'est pas, suffisamment, efficace pour résoudre des problèmes complexes.
PSO	- simple et facile à mettre en œuvre ; - rapidité de convergence ; - capable de trouver des solutions pour les problèmes impliquant de multiples contraintes et objectifs ; - conçue pour résoudre les problèmes à variables continues mais peut être adaptée aux autres types de variables.	- l'algorithme original de PSO est susceptible de converger vers un optimum local ; - les contraintes d'inégalités dans la position suivante d'un individu produit par l'algorithme de PSO peuvent être perturbées.
DE	- c'est une version améliorée de l'algorithme génétique (GA) ; - simple, robuste et capable d'atteindre, rapidement la convergence ; - capable de trouver des solutions pour les problèmes impliquant de multiples contraintes et objectifs ; - capable de résoudre les problèmes de n'importe quel type de variables.	- la performance de la recherche dépend du choix des paramètres de la méthode (facteur de mutation, rapport de croisement et la taille de la population).

- Un algorithme bio-heuristique appelé algorithme amélioré de recherche de nourriture bactérienne est proposé par S. Jaganathan. et S. Palanisamy [63] pour résoudre le problème de l'OPF dans un système d'énergie déréglé et garantir un fonctionnement sûr et optimal, avec emplacement optimal des dispositifs FACTS, et étude de la stabilité.

Un algorithme évolutionnaire inspiré du calcul quantique (quantum computation) a été présenté en [64] pour une répartition optimale de la puissance active et réactive. Le problème du

contrôle de la tension et de la puissance réactive est résolu dans la référence [65] par un algorithme évolutionnaire amélioré basé sur la notion du calcul quantique.

- Les systèmes immunitaires artificiels (AIS) sont apparus dans les années 90. Ils imitent la défense d'un système immunitaire contre les bactéries, virus et autres organismes liés à la maladie [66-67]. Les AIS peuvent être utilisés pour résoudre des problèmes d'optimisation multivariables. Ils diffèrent des GA dans l'éducation de la mémoire et dans le système de production pour divers anticorps.

Un nouveau type d'algorithme génétique immunitaire (immune genetic algorithm) est proposé dans [66] pour l'optimisation de la puissance réactive d'un réseau électrique. Les probabilités de croisement et de mutation sont adaptées (variables) selon la valeur de la fonction d'adaptation (fitness) de l'individu.

Un algorithme immunitaire adaptatif et multi-objectif est proposé dans [67] pour traiter le problème de l'OPF réactive avec étude de la stabilité statique de la tension.

- L'algorithme de recherche gravitationnelle (GS) est un algorithme de recherche à base de la population traité par la loi de la gravité et de l'interaction entre masses. Cette technique inspirée de la nature est proposée par [68] pour étudier le problème de l'ORPD où différentes fonctions objectives sont testées.

- Les algorithmes co-évolutionnaires (CEA) [69-70] sont une nouvelle extension des algorithmes évolutionnaires. Un GA coévolutionnaire coopératif est appliqué dans la référence [69] pour l'optimisation de puissance réactive. Dans cette approche, les variables de contrôle sont partagées sur différentes populations.

L'algorithme génétique avec coévolution coopérative consiste à identifier une décomposition normale du problème en des sous-composants. Chaque sous-composant est assigné à une population. Les individus dans une population représentent les sous-composants potentiels (ou solutions partielles) de la solution globale du problème. Ainsi, chaque sous-composant est évolué simultanément et séparément des autres. Le seul moment où il y a collaboration c'est pendant la phase d'évaluation. Afin d'évaluer la fonction d'adaptation d'un individu donné, des collaborateurs seront choisis parmi les autres populations (le meilleur individu de chacune des autres populations), de sorte qu'une solution complète puisse être formée (solution globale au problème). Cette solution sera utilisée pour évaluer la fonction d'adaptation de l'individu [70].

- Le recuit simulé (SA) est une approche d'optimisation stochastique proposée par Kirkpatrick en 1983[71]. Elle est inspirée du processus naturel du recuit utilisé en métallurgie pour améliorer la qualité d'un solide. Cette technique est utilisée dans [72] pour résoudre le problème du dispatching économique.

- Un algorithme de recherche d'harmonie améliorée est utilisé dans la référence [73] pour résoudre le problème de l'écoulement optimale de la puissance.

- Une très récente métaheuristique appelée la recherche coucou (CS: Cuckoo Search) [74-76] est proposée pour la première fois en 2009 par Yang et Deb [74]. La recherche coucou s'inspire du comportement de reproduction d'une espèce spéciale d'oiseaux parasites de nids appelés «Coucous». Dans l'algorithme de la recherche coucou, une solution possible est appelée « nid » ou «coucou».

Un algorithme amélioré de recherche coucou est utilisé dans [75] pour résoudre le problème de la répartition optimale de la puissance réactive.

Un autre algorithme amélioré de recherche coucou pour les problèmes d'ordonnancement du flow shop hybrides est appliqué dans la référence [76]. L'ordonnancement consiste à organiser, dans le temps, la réalisation des tâches compte tenu des contraintes de temps et de ressources, afin d'optimiser un ou plusieurs objectifs.

- Une nouvelle métaheuristique inspirée de la nature est proposé dans [77] pour résoudre le problème de l'écoulement optimal de la puissance. Cet algorithme est inspiré du phénomène des trous noirs (BH : Black Holes).

- Afin de résoudre le problème de la répartition optimale de la puissance réactive, M. H. Sulaiman et ses collègues [78] ont proposé l'utilisation d'une nouvelle méta-heuristique basée sur l'intelligence en essaim appelé l'algorithme d'optimisation du loup gris (GWO : gray wolf optimizer).

L'algorithme d'optimisation du loup gris est une technique d'optimisation qui était introduite pour la première fois en 2014 par S. Mirjalili [79].

- Pour l'amélioration des performances des algorithmes d'optimisation, certains auteurs ont proposé des algorithmes hybrides. Nous pouvons, notamment, citer la référence [80], qui propose un algorithme hybride basé sur la colonie de fourmis (ACO) et l'algorithme génétique (GA) pour étudier l'impact des sources de production décentralisée sur l'optimisation du contrôle tension / puissance réactive dans un réseau de distribution électrique. Les puissances réactives générées de ces générateurs de production, compensateurs statiques (SVC), changeurs de prises en charge et des contrôleurs locaux de réglage sont choisis comme variables de contrôle.

Dans [81], un algorithme efficace basé sur l'hybridation entre l'évolution différentielle (DE) et l'algorithme génétique (GA) avec des fonctions multi-objectifs est considéré. Dans cette approche, les meilleures caractéristiques des 2 techniques sont combinées et développées. Dans cet article, les fonctions multi-objectives considérées sont la minimisation des pertes de puissance active, la minimisation de l'écart de tension et la minimisation de l'indice de stabilité de la tension. Les meilleurs facteurs de la sélection et de croisement tirés de l'algorithme GA et le meilleur

facteur de mutation pris de l'algorithme DE sont combinés ensemble pour donner la solution efficace. La sélection de la roulette (Wheel) et la technique de croisement en un seul point sont utilisées. La stratégie de mutation de l'ED appelée (DE / rand / 1) est appliquée dans cet algorithme hybride.

Dans [82], un algorithme de DE est hybridé avec un algorithme de l'ACO. La technique d'optimisation par ACO a gagné une énorme popularité depuis sa mise au point au début des années 90, en raison de sa flexibilité et de sa capacité d'éviter d'atteindre un optimum local. Malheureusement, cet algorithme attrayant souffre de plusieurs inconvénients, notamment, de la stagnation et de la convergence lente vers une solution optimale. En outre, cette technique a montré qu'elle n'est pas, suffisamment, efficace pour résoudre des problèmes complexes. Au contraire, la méthode DE est plus robuste. Elle est capable d'atteindre, rapidement, la convergence. Ainsi, un nouvel algorithme, appelé optimisation par évolution différentielle-colonie de fourmis (DE-ACO : différentiel evolution- ant colony optimization) a été modélisé pour compenser les inconvénients de l'ACO. La couche de phéromone dans la technique de l'ACO sera modifiée par les processus de la mutation, de croisement et de la sélection de l'évolution différentielle (DE). L'algorithme a été utilisé pour résoudre le problème du dispatching économique.

Un type d'hybridation combinant aussi la DE et l'ACO a été proposé dans le document [83]. L'algorithme évolutionnaire hybride proposée (HEC : hybrid evolutionary computation) combine l'algorithme de l'évolution différentielle DE classique et la règle probabilistique de transition d'état utilisées dans le système de fourmi pour traiter le problème de l'ORPD. Pour conserver la diversité de la population, s'échapper de solutions optimales locales et augmenter la capacité de la recherche globale, une règle probabilistique de transition d'état, qui est utilisée dans le système de fourmi, est utilisée pour remplacer l'opération de sélection dans l'algorithme de la DE classique.

Une hybridation entre un type de réseaux de neurones (ridgelet neural network) et une optimisation par essaim de particules améliorée (IPSO : improved particle swarm optimization) est proposée dans [84]. La technique de PSO a l'avantage que son algorithme est simple, facile à réaliser présentant une vitesse de recherche considérable. Cependant, l'algorithme original de PSO est susceptible de converger vers un optimum local. Afin de résoudre ce problème d'une manière efficace et améliorer la performance de la convergence, une optimisation par IPSO avec un facteur de mutation auto-adaptatif est présentée dans cet article. Ainsi, la position de la $i^{ème}$ particule est ajustée, dynamiquement, en fonction de la valeur de l'adaptation (fitness) à l'itération k. De cette manière, certaines particules peuvent explorer d'autres domaines de manière adéquate pour améliorer la capacité de chercher une solution globale. L'IPSO est utilisée pour former un réseau de neurones de type ridgelet qui est appliqué ensuite à un problème de classification de données. Le

vecteur de position d'une particule de l'IPSO est composé de l'ensemble de poids de connexion, de la valeur de seuil et des paramètres de la fonction ridgelet.

Un essaim particulaire flou et adaptatif est proposé dans la référence [85] où une nouvelle formulation multi-objective est présentée pour l'étude du problème de contrôle tension/puissance réactive dans un réseau électrique. Les objectifs sont les pertes de la puissance active, les déviations de tension et l'indice de la stabilité de tension du système.

Pour améliorer les performances de la PSO, une technique de PSO floue-adaptative est proposée. Le système flou est utilisé pour ajuster de manière adaptative les paramètres du PSO, tels que le poids d'inertie et les facteurs d'apprentissage, au cours du processus d'évolution.

Certains auteurs ont étudié l'hybridation entre les GA et le PSO [86-87]. Ces derniers sont connus par leur capacité à trouver des solutions pour les problèmes impliquant de multiples contraintes et objectifs.

Dans l'article [86], l'efficacité d'une approche hybride GA-PSO est étudiée pour trouver des solutions au problème de la planification de la production des générateurs thermiques. Le problème de la planification de la production d'énergie électrique peut contenir 2 aspects :

- définir le nombre d'unités de production d'électricité dédiées à répondre à la demande de charge ;
- le problème de la répartition économique de la puissance active.

La limite maximale de capacité de charge (MLL : maximum loadability limit) est une approche établie pour estimer la stabilité de tension [87]. La MLL est la marge entre le point d'exploitation du système et le point de chargement maximal. Le coût optimal de la génération pour la MLL d'un réseau électrique peut être formulé comme un problème d'optimisation. Ce document [76] propose une optimisation par essaim particulaire hybride (HPSO : Hybrid particle swarm optimization). Cette technique hybride utilise les processus de sélection et de sous-population de l'algorithme génétique et les intègre dans l'algorithme du PSO. Ceci permet la recherche :

- de s'échapper des optimums locaux ;
- d'éviter une convergence prématurée ;
- de rechercher dans les différentes zones de l'espace de recherche.

Dans la référence [88], le problème du dispatching économique a été résolu par 4 techniques différentes : les algorithmes génétiques GA, la recherche de nourriture bactérienne (BFA), l'optimisation par colonies de fourmis (ACO) et l'optimisation par essaim de particules (PSO). Les résultats de simulation présentés en termes de précision, de fiabilité et de temps d'exécution, ont permis de montrer la performance de chaque algorithme et de faire une étude comparative entre ces différentes techniques.

La technique du recuit simulé peut être hybridée avec différents techniques tels que l'optimisation par essaim de particules (PSO) [89-90] et la recherche gravitationnelle [91].

Une hybridation entre le recuit simulé et l'algorithme du troupeau de krill (Krill Herd algorithm) est proposée dans [92] pour réduire les pertes des puissances actives et améliorer le profil de tension d'un réseau électrique. Le krill est le nom générique de petites crevettes des eaux froides. L'algorithme du troupeau de krill est basé sur l'imitation du comportement d'un troupeau des individus de krill.

Un type d'hybridation entre le recuit simulé et un algorithme appelé Nelder-Mead est proposé dans [93] pour résoudre le problème de la répartition optimale de la puissance réactive. L'algorithme de Nelder-Mead est l'un des algorithmes d'optimisation non linéaire sans dérivés les plus populaires.

Dans la référence [94], un algorithme d'optimisation par essaim particulaire multi-essaim (HMPSO) a été proposé pour résoudre le problème de l'ORPD. Ce dernier est formulé comme un problème d'optimisation multi-objectif non linéaire avec contraintes d'égalité et d'inégalité. HMPSO est l'un des algorithmes de recherche basés sur la population récemment proposé, dans lequel l'essaim existant est divisé en plusieurs sous- essaims. L'optimisation par essaim particulaire est appliquée comme moteur de recherche pour chaque sous-essaim. En outre, pour explorer des régions plus prometteuses de l'espace de recherche, l'algorithme d'évolution différentielle (DE) est appliqué. Pour trouver l'ensemble optimal de Pareto pour le problème de l'ORPD, la méthode de la somme pondérée a été exécutée. Ensuite, pour trouver la solution préférée dans l'ensemble Pareto-optimal, la fonction d'appartenance floue a été utilisée.

Dans la référence [95], une approche de programmation à trois niveaux est utilisée pour résoudre le problème de l'ORPD à l'aide d'une programmation par objectif floue basée sur un algorithme génétique.

1.3 – FORMULATION DU PROBLEME DU CONTROLE OPTIMAL DE TENSION / PUISSANCE REACTIVE

1.3.1 - PREAMBULE

Dans les réseaux électriques, la tension et la fréquence sont les principaux indicateurs de la bonne exploitation technique et économique du système de production et de transport.

Un grand réseau de distribution d'énergie électrique délivre une tension dont la qualité dépend à la fois :

- du **producteur**, qui s'engage contractuellement à maintenir l'amplitude et la fréquence entre des limites, qu'il est cependant amené à dépasser lors de manœuvres (mise sous tension, élimination automatique de courts-circuits, délestage, etc.).
- des **phénomènes aléatoires**, d'origine atmosphérique ou accidentelle (foudre, rupture de ligne, etc.).
- des **abonnés** qui, en manœuvrant à leur gré leurs appareils électriques (marche, arrêt, variation de charge), génèrent des courants transitoires non sinusoïdaux et qui, en utilisant des dispositifs à commutation électronique, engendrent des harmoniques de courant.

En l'absence des moyens de contrôle, le régime normal admissible de la tension ne peut avoir lieu que pour les réseaux à faible puissance et courtes distances entre les points de productions et de consommation.

Il est nécessaire de mettre en œuvre des moyens de réglage pour le maintien de la tension dans les limites admissibles. Ce réglage est d'une importance capitale pour garantir un bon fonctionnement des différents équipements du réseau (lignes, transformateurs, systèmes de protection, charges, …etc). En effet, une tension plus élevée entraîne le vieillissement ou la destruction du matériel raccordé, alors qu'une tension plus basse par rapport à la plage spécifiée conduit à un mauvais fonctionnement des équipements et des charges.

Les variations de tensions sont inévitables à cause des variations permanentes des puissances actives et réactives demandées ainsi que des changements de topologie du réseau. Ces variations peuvent être courtes causées principalement par le démarrage de grosses charges ou des défauts dans le réseau, ou lentes dues essentiellement aux variations de la consommation de puissance active et surtout réactive sur une période longue.

Tout d'abord, nous allons voir la relation entre les chutes de tensions, la puissance réactive et la puissance active transmissible dans un réseau électrique.

1.3.2 - CHUTES DE TENSION, PUISSANCE REACTIVE ET PUISSANCE ACTIVE TRANSMISSIBLE

1.3.2.1 - Chute de tension dans une ligne

Considérons la figure (1.1) qui représente une ligne de transport d'impédance complexe $\bar{Z} = R + jX$, et que la tension n'est tenue qu'à l'extrémité 1 (coté générateur), l'extrémité 2 (coté charge) absorbant une puissance $\overline{S_2} = P_2 + jQ_2$.

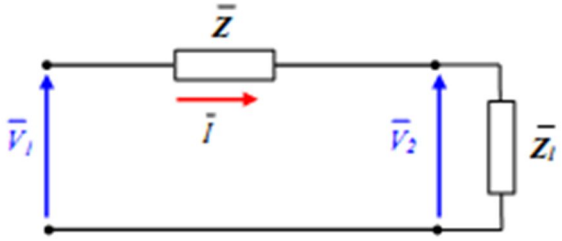


Fig. 1.1 - Schéma monophasé équivalent

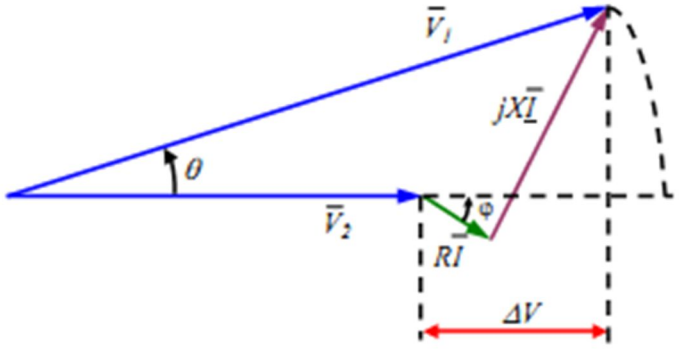


Fig. 1.2 - Diagramme vectoriel des tensions

Si le réseau n'est pas trop chargé, le diagramme de tension donné par la figure (1.2) conduit à assimiler la chute de tension ΔV à :

$$\Delta V \approx V_1 \cos\theta - V_2 \tag{1.1}$$

L'angle de transport θ étant petit (réseau peu chargé), si φ désigne le déphasage du courant par rapport à la tension à l'extrémité réceptrice 2, on peut écrire, pour un réseau monophasé :

$$\Delta V \approx RI\cos\varphi + XI\sin\varphi \tag{1.2}$$

$$\Delta V \approx RI\frac{V_2}{V_2}\cos\varphi + XI\frac{V_2}{V_2}\sin\varphi \tag{1.3}$$

$$\Delta V \approx \frac{RP_2+XQ_2}{V_2} \tag{1.4}$$

On a aussi :

$$V_1\sin\theta = XI\cos\varphi - RI\sin\varphi \tag{1.5}$$

$$\Rightarrow \sin\theta = \frac{XI\cos\varphi - RI\sin\varphi}{V_1}$$

$$= \frac{XI\,\mathrm{V}_2\cos\varphi - RI\mathrm{V}_2\sin\varphi}{V_1\mathrm{V}_2}$$

$$\Rightarrow \sin\theta \approx \frac{XP_2 - RQ_2}{V_1V_2} \quad (1.6)$$

L'hypothèse du réseau peu chargé permet d'écrire :

$$V_1 \approx V_2 = V \quad (1.7)$$

Soit, pour un réseau triphasé et en notant U la tension composée correspondant à V, P et Q les puissances de transit triphasé :

$$\frac{\Delta U}{U} \approx \frac{RP + XQ}{U^2} \quad (1.8)$$

$$\sin\theta \approx \frac{XP - RQ}{U^2} \quad (1.9)$$

On peut également noter que si $R \ll X$ ($R \approx 0$) :

$$\Delta U \approx \frac{XQ}{U} \quad (1.10)$$

$$\sin\theta \approx \frac{XP}{U^2} \quad (1.11)$$

Dans ces conditions, les relations (1.10) et (1.11) illustrent le fait que :

- Si P et Z sont fixées, toute variation de la puissance réactive entraîne une variation de tension et réciproquement.
- Principalement, la chute de tension dépend de l'énergie réactive consommée par l'extrémité réceptrice. C'est donc la circulation du flux réactif qui favorise l'augmentation des chutes de tension dans les lignes électriques.
- Tension et puissance sont, donc, des grandeurs très liées.
- La chute de tension est fonction de la valeur de la réactance X. Toute augmentation (respectivement diminution) de X entraîne une augmentation (respectivement diminution) de la chute de tension.
- l'angle de transport θ dépend principalement de la puissance active transmise.

1.3.2.2 - Puissance maximale transmissible

Par ailleurs, si l'on considère une charge variable de tension simple V_c, représentée par une impédance $\overline{Z} = Z\angle\varphi$, de déphasage φ, alimentée, à travers un réseau assimilable à une réactance X, par une source de tension efficace simple constante V_s, comme le montre la figure (1.3.a). L'application de la loi d'Ohm à un tel système donne les relations suivantes :

- Courant traversant le dipôle :

$$I = \frac{V_s}{\sqrt{X^2 + Z^2 + 2\ X\ Z\ \sin\varphi}} \tag{1.12}$$

- Tension aux bornes de la charge :

$$V_c = Z \times I = \frac{Z\ V_s}{\sqrt{X^2 + Z^2 + 2\ X\ Z\ \sin\varphi}} \tag{1.13}$$

- Puissance active absorbée par la charge :

$$P_c = V_c\ I\ \cos\varphi = \frac{V_s^2\ \cos\varphi}{\frac{X^2}{Z} + Z + 2\ X\ \sin\varphi} \tag{1.14}$$

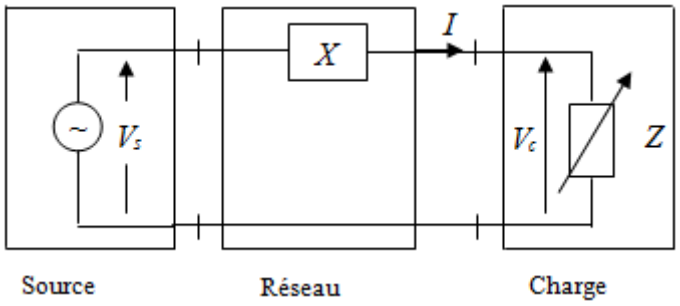
La puissance maximale P_{cmax} transmise par la source est obtenue, pour X fixe et Z variable, tout en conservant un déphasage φ constant, est obtenue par le principe de transfert maximal de puissance ou d'adaptation d'impédance, pour $Z = X$:

$$P_{c\max} = \frac{V_s^2\ \cos\varphi}{2\ X\ (1 + \sin\varphi)} \tag{1.15}$$

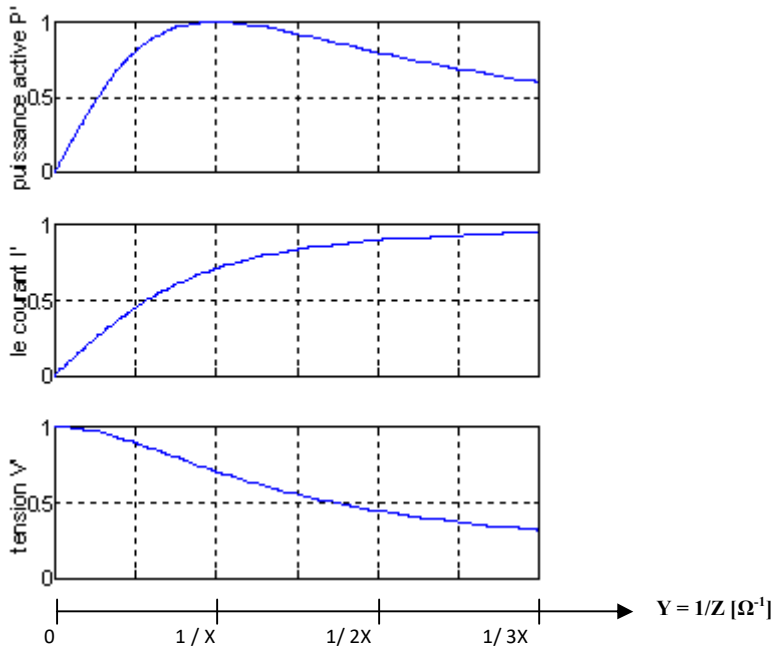
On met ainsi en évidence l'existence d'une limite de puissance transmissible par un réseau. Les courbes de la figure (1.3.b) montrent l'évolution des trois grandeurs (courant, tension, puissance active), exprimées en valeurs réduites, lorsque la charge croit (c'est à dire lorsque l'impédance Z diminue, ou autrement dit, lorsque l'admittance $Y = 1/Z$ augmente). On constate, au départ, la puissance active croit rapidement, avec le courant pour augmenter, ensuite, de moins en moins tout en passant par un maximum puis diminue graduellement. Au-delà du maximum, en effet, la chute de tension dans le réseau provoquée par l'appel de courant fort devient très importante et l'augmentation de I ne suffit pas à compenser la diminution de V_c dans l'expression (1.14) de P_c. Pour $Z = X$, la tension limite correspondante au transfert maximal de puissance $P_{c\max}$ sera :

$$V_{crit} = \frac{V_s}{\sqrt{2\ (1 + \sin\varphi)}} \tag{1.16}$$

V_{crit} est la tension critique ou limite, correspondante au point critique M, au-delà duquel il devient impossible de faire transiter plus de puissance vers la charge. Ces notions de puissance maximale transmissible et de tension critique peuvent, également, être mises en évidence par la courbe de la figure (1.4). Cette dernière représente l'évolution de la tension V_c aux bornes de la charge en fonction de la puissance absorbée.



a) – Une charge variable alimentée à travers un réseau par une source de tension



b) - Evolution des trois grandeurs (courant, tension, puissance active), exprimées en valeur réduite, lorsque la charge croit ($V' = \frac{V_c}{V_s}$, $I' = \frac{I}{I_0}$, $I_0 = \frac{V_s}{X}$, $P' = \frac{P_C}{P_{cmax}}$).

Fig. 1.3 – Variation des grandeurs électriques aux bornes de la charge en fonction de son admittance $Y = 1/Z$

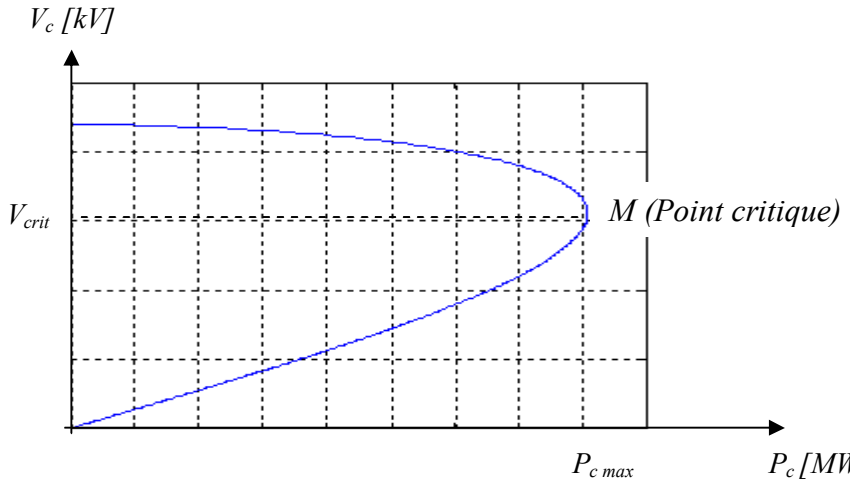


Fig. 1.4 – Evolution de la tension aux bornes de la charge en fonction de la puissance fournie

L'expression (1.15) de P_{cmax} montre, entre autres, que :

- Plus la tension d'exploitation V_s est haute, plus la puissance maximale transmissible P_{cmax} est grande. D'où l'intérêt d'exploiter avec un plan de tension le plus haut possible.
- Plus l'impédance X du réseau est faible, plus la puissance maximale transmissible P_{cmax} est grande. D'où l'intérêt d'avoir un réseau, suffisamment, dimensionné et d'exploiter avec le maximum de lignes disponibles.
- Plus φ diminue, c'est-à-dire, plus la compensation de la charge augmente (grâce à l'adjonction de condensateurs), plus la puissance transmissible croît. D'ou l'intérêt de compenser au maximum et au plus près des charges, l'énergie réactive qu'elles consomment.

Les transits d'énergie réactive sont, principalement, dus :

- A la consommation des charges, caractérisée par la tangente φ des récepteurs, très variables selon le type des récepteurs, très variables selon le type de charges.
- Aux éléments du réseau, principalement, les lignes qui peuvent fournir ou absorber de la puissance réactive selon que la puissance transitée est inférieure ou supérieure à la valeur caractéristique.

1.3.3 – FORMULATION MATHEMATIQUE DU PROBLEME

Comme il a été mentionné précédemment, les modules de la tension dans un réseau électrique sont affectés, principalement, par les variations de la puissance réactive. Ceci tient en fait de la puissance réactive Q qui intervient de manière importante dans l'expression de la chute de tension ΔU.

Dans un réseau électrique, le bilan global de la puissance réactive produite et consommée dans l'ensemble du système doit être équilibré. Toutefois, l'équilibre local n'est pas naturel. Il en

résulte des transits de la puissance réactive. Or, ces transits provoquent des chutes de tension et des pertes [96-98].

Le réseau en lui- même est une source non négligeable de puissance réactive. Ainsi, en dehors de la production de l'énergie réactive par les générateurs, le réseau doit faire appel à d'autres sources ou plutôt à d'autres moyens de compensation, qui, finalement, sont au moins aussi souvent consommateurs que fournisseurs d'énergie réactive.

L'analyse des variations de la demande de la puissance réactive montre que le problème de l'adaptation offre-demande présente 2 aspects qui nécessitent l'emploi de dispositifs aux caractéristiques très différentes:

- le premier consiste à suivre les fluctuations périodiques. Celles-ci sont connues, tout au moins pour les charges dans une large mesure prévisible. Une grande part de l'ajustement peut, donc, être réalisée à l'aide de moyen dont l'action est discontinue et le temps de réponse est, relativement, long. Cette catégorie comprend les batteries de condensateurs et les inductances installées sur les réseaux ;
- le second consiste à faire face aux variations brusques et aléatoires. Ceci nécessite la mise en œuvre de moyens dont le temps de réponse est très court. Cette catégorie comprend les groupes de production ainsi que les compensateurs synchrones et les compensateurs statiques…

L'optimisation du plan de tension d'un réseau électrique, suppose donc, de maîtriser les transits d'énergie réactive et de minimiser les pertes dans le réseau. Cela n'est possible que par un ajustement optimal d'un nombre de moyens de contrôle, tels que les sources de puissance réactive (switching VAr sources), les générateurs de tension (en agissant sur leurs tensions) et les transformateurs réglables.

Il est évident que le problème du contrôle tension/ puissance réactive dans les réseaux électriques est devenu un sujet stratégique.

Alors résoudre ce problème revient à résoudre un problème d'optimisation (minimisation) avec contraintes.

La première étape du processus d'optimisation est la phase de modélisation du système considéré. Cette première étape est très importante : de la qualité de la modélisation dépendra la qualité des résultats obtenus vis à vis du problème initial :

- Un modèle trop simpliste ne permettra pas d'obtenir de résultats significatifs par rapport à la réalité du problème ;
- tandis qu'un modèle trop complexe pourra être trop difficile à résoudre, impliquant par exemple des temps de traitement trop longs.

Pour chaque problème d'optimisation, la phase de modélisation nécessite 3 sous - étapes :

- la définition des variables de décision ;
- la définition des contraintes ;
- la définition de la fonction objective.

1.3.3.1 - Variables de décision

L'ensemble des variables de décision est l'ensemble des variables sur lesquelles il est possible d'agir pour modifier l'état du système. Ces variables peuvent être discrètes ou continues. Dans le cas du problème du contrôle optimal de tension/puissance réactive, les variables de décision seront, les tensions des générateurs, les rapports de transformation des transformateurs réglables et les puissances réactives injectées par les batteries de condensateurs shunts.

1.3.3.2 - Contraintes

Les contraintes sont les limites que le système ne peut pas franchir. Il existe des problèmes d'optimisation contraints et non contraints. Le réglage de tension est un problème contraint où on peut distinguer :

- **les contraintes physiques :** imposées par les lois de la physique et de l'électrotechnique en particulier (par exemple, les limites de fonctionnement d'une génératrice électrique) ;
- **les contraintes techniques :** imposées par exemple par la tenue mécanique ou thermique de certains éléments du réseau (par exemple, l'échauffement des câbles),
- **les contraintes réglementaires :** imposées par une réglementation (par exemple, les contraintes de tension) pour des raisons techniques et économiques.

Deux types principaux de contraintes peuvent être décrits : contraintes d'égalité et contraintes d'inégalités.

1.3.3.3 - Fonction objective

La dernière étape de la modélisation du système passe par l'élaboration d'une fonction objective appropriée. La fonction objective est la mesure du système considéré. C'est la fonction que l'algorithme cherche à minimiser. La façon de la définir est importante car, de sa nature (linéaire, non linéaire, simple variable, multi-variables, …etc.), liée à la nature des variables de décision et des contraintes, dépend le type d'algorithme d'optimisation utilisé.

1.3.3.4 - Formulation mathématique

L'objectif de la répartition optimale de la puissance réactive est de réduire au minimum les pertes de puissance active, en maintenant un plan de tension convenable dans le réseau. La fonction de perte totale peut être décrite comme suit :

$$f(x) = p = \sum_{k\in N_E} P_{kLoss} = \sum_{k\in N_E} g_{ij}\left(V_i^2 + V_j^2 - 2V_iV_jcos\theta_{ij}\right) \qquad \text{Avec } k = (i,j) \qquad (1.17)$$

Cette fonction est le sujet d'un nombre de contraintes d'égalité relatives à l'écoulement de puissance :

$$\Delta P_i = P_{gi} - P_{di} - V_i \sum_{j\in N_i} V_j(G_{ij}cos\theta_{ij} + B_{ij}sin\theta_{ij}) = 0, \qquad i \in N_0 \qquad (1.18)$$

$$\Delta Q_i = Q_{gi} - Q_{di} - V_i \sum_{j\in N_i} V_j(G_{ij}cos\theta_{ij} - B_{ij}sin\theta_{ij}) = 0, \qquad i \in N_{PQ} \qquad (1.19)$$

Sous contraintes d'inégalités relatives aux variables de contrôle:

$$T_{imin} \leq T_i \leq T_{imax}, \qquad i \in N_T \qquad (1.20)$$

$$Q_{g_{imin}} \leq Q_{g_i} \leq Q_{g_{imax}}, \qquad i \in N_{cap} \qquad (1.21)$$

$$V_{imin} \leq V_i \leq V_{imax}, \qquad i \in N_{PV} \qquad (1.22)$$

Et sous contraintes d'inégalités relatives aux variables dépendantes :

$$V_{imin} \leq V_i \leq V_{imax}, \qquad i \in N_{PQ} \qquad (1.23)$$

$$Q_{g_{imin}} \leq Q_{g_i} \leq Q_{g_{imax}}, \qquad i \in N_{PV} \qquad (1.24)$$

L'ensemble des variables de contrôle est composé des variables qui affectent les injections de la puissance réactive, c'est-à-dire, les tensions des générateurs, les rapports de transformation des transformateurs réglables et les puissances réactives injectées des batteries de condensateurs shunts.

Par contre, les variables dépendantes sont les tensions des JdB de charge et les injections de puissance réactive aux JdB des générateurs.

Les variables de contrôle sont auto-restreintes et les variables dépendantes sont limitées par leur addition sous forme de pénalité quadratique ajoutée à la fonction objective. Par conséquent, le problème peut être reformulé, comme suit :

$$Min\ F(x) = p + \sum_{i\in N_{PQ}} \lambda_{Vi}(V_i - V_i^{lim})^2 + \sum_{i\in N_{PV}} \lambda_{Qgi}(Q_{gi} - Q_{gi}^{lim})^2, \qquad (1.25)$$

Sous les contraintes d'égalité relatives à l'écoulement de puissance données par (1.18) - (1.19). Et sous les contraintes d'inégalités relatives aux variables de contrôle, représentées par (1.20) - (1.22).

Avec :

λ_{Vi} et λ_{Qgi} : facteurs de pénalité.

$$V_i^{lim} = \begin{cases} V_{imin} & si \quad V_i < V_{imin} \\ V_{imax} & si \quad V_i > V_{imax} \\ V_i & si \quad V_{imin} \leq V_i \leq V_{imax} \end{cases} \tag{1.26}$$

$$Q_{gi}^{lim} = \begin{cases} Q_{gimin} & si \quad Q_{gi} < Q_{gimin} \\ Q_{gimax} & si \quad Q_{gi} > Q_{gimax} \\ Q_{gi} & si \quad Q_{gimin} \leq Q_{gi} \leq Q_{gimax} \end{cases} \tag{1.27}$$

Où les différents symboles et indices sont définis comme suit :

V_i : l'amplitude de la tension au JdB *i*

θ_{ij} : le déphasage de tension entre le JdB *i* et le JdB *j*

g_{ij} : la conductance de la ligne entre le JdB *i* et le JdB *j*

G_{ij} : la conductance de transfert entre le JdB *i* et le JdB *j*

B_{ij} : la susceptance de transfert entre le JdB *i* et le JdB *j*

T_i : le rapport de transformation du transformateur *i*

N_E : le nombre des branches dans le réseau

N_{PQ} : le nombre des JdB du type *PQ*

N_{PV} : le nombre des JdB du type *PV* (contenant le JdB d'équilibre)

N_B : le nombre total des JdB

N_0 : le nombre total des JdB sauf le JdB d'équilibre

N_i : l'ensemble des JdB contenant le JdB i et les autres JdB adjacents

N_T : le nombre des branches contenants des transformateurs

N_{cap} : le nombre des JdB contenants des batteries de condensateurs shunts

P_{Gi} : la puissance active injectée au JdB *i*

Q_{Gi} : la puissance réactive injectée au JdB *i*

P_{Di} : la puissance active demandée au JdB *i*

Q_{Di} : la puissance réactive demandée au JdB *i*

P_{kLoss} : la perte de la puissance active dans la branche *k*

1.4 – OBJECTIFS ET CONTRIBUTIONS SOUHAITEES

Le problème de l'optimisation du plan de tension et de la répartition de la puissance réactive est un problème d'optimisation avec contraintes d'égalité et d'inégalité.

Il existe plusieurs méthodes pour résoudre les problèmes d'optimisation. Mais, depuis une dizaine d'années, les métaheuristiques ont révélé leur grande efficacité pour fournir des solutions approchées de bonne qualité pour un grand nombre de problèmes d'optimisation classiques et d'applications réelles de grande taille. C'est pourquoi l'étude de ces méthodes est, actuellement, en plein développement.

L'intelligence inspirée de la nature devient de plus en plus populaire et un nombre important de méthodes entraînées par des concepts issus de la nature / la biologie ont été développées.

Ce travail consiste à utiliser des techniques intelligentes et métaheuristiques simples et hybrides dans leur forme améliorée (adaptative) pour résoudre le problème du contrôle optimal de tension/puissance réactive.

1.5 – CONCLUSION

Dans ce chapitre, nous avons survolé l'historique et la synthèse bibliographique relative aux différentes techniques et méthodes d'optimisation classiques et modernes (techniques intelligentes artificielles & métaheuristiques), relatives à la résolution du problème du contrôle optimal de tension/puissance réactive. La formulation de ce problème du point de vu physique et mathématique, les objectifs et les contributions souhaitées du présent travail sont, eux aussi, évoqués.

Le chapitre suivant présente un état de l'art de l'optimisation et une classification de ses méthodes de résolution. Il ne s'agit pas de décrire en profondeur chacun de ces modèles, mais simplement de faire une synthèse de recensement des méthodes les plus répandues.

2 - OPTIMISATION : ETAT DE L'ART ET METHODES

2.1 - INTRODUCTION

De nombreux secteurs scientifiques et de l'industrie, comme dans le traitement des images, la conception de systèmes mécaniques, la planification et l'exploitation des réseaux électriques,… etc. sont concernés par les problèmes d'optimisation, dont lesquels on définit une fonction objective que l'on cherche à minimiser (ou maximiser) par rapport à tous les paramètres concernés. La définition du problème d'optimisation est souvent complétée par la donnée de contraintes : tous les paramètres (ou variables de décisions) de la solution proposée doivent respecter ces contraintes, faute de quoi la solution n'est pas réalisable.

Ce chapitre est consacré à donner un bref aperçu sur l'optimisation, ainsi que la classification des méthodes utilisées pour résoudre les problèmes de ce type.

2.2 - DEFINITION DE L'OPTIMISATION

Un problème d'optimisation se définit comme la recherche du minimum ou du maximum (de l'optimum) d'une fonction donnée. On peut aussi trouver des problèmes d'optimisation pour lesquelles les variables de la fonction à optimiser sont contraintes d'évoluer dans une certaine partie de l'espace de recherche. Dans ce cas, on a une forme particulière de ce que l'on appelle un problème d'optimisation sous contraintes.

2.3 - ELEMENTS CONTITUANT UN PROBLEME D'OPTIMISATION

Un problème d'optimisation est défini par :

- **un espace de recherche (de décision)** : ensemble de solutions ou de configurations constitué des différentes valeurs prises par les variables de décision. Elles sont regroupées dans le vecteur $\vec{x}$. C'est en faisant varier ce vecteur que l'on recherche un optimum de la fonction f.
- **une ou plusieurs fonction(s) dite(s) objective(s)**, à optimiser (minimiser ou maximiser). Une fonction objective est le nom donné à la fonction f (on l'appelle aussi fonction de coût ou critère d'optimisation). C'est cette fonction que l'algorithme d'optimisation va devoir optimiser (trouver un optimum).

- un ensemble de **contraintes** à respecter : Dans la plupart des problèmes d'optimisation, des restrictions sont imposées par les caractéristiques du problème. Ces restrictions doivent être satisfaites afin de considérer une solution acceptable. Cet ensemble de restrictions, appelées contraintes.

Dans la plupart des problèmes, l'espace d'état (décision) est fini ou dénombrable. Les variables du problème peuvent être de nature diverse (réelle, entier, booléenne, etc.) et peuvent exprimer des données qualitatives ou quantitatives. La fonction objectifve représente le but à atteindre pour le décideur. L'ensemble de contrainte définit des conditions sur l'espace d'état que les variables doivent satisfaire. Ces **contraintes** sont souvent des **contraintes d'inégalité** ou **d'égalité** et permettent en général de limiter l'espace de recherche (solutions réalisables). La résolution optimale du problème consiste à trouver le point ou un ensemble de points de l'espace de recherche qui satisfait au mieux la fonction objective. Le résultat est appelé valeur optimale ou optimum.

2.4 - FORMULATION MATHEMATIQUE GENERALE D'UN PROBLEME D'OPTIMISATION

Un problème d'optimisation peut être présenté en général sous la forme suivante :

Minimiser $f(\vec{x})$ (fonction à optimiser)

Sujet de : $\vec{g}(\vec{x}) \leq 0$ (m contraintes d'inégalité)

$\vec{h}(\vec{x}) = 0$ (p contraintes d'égalité)

Avec R^n , $\vec{g}(\vec{x}) \in R^m, \vec{h}(\vec{x}) \in R^p$ (2.1)

Les vecteurs $\vec{g}(\vec{x})$ et $\vec{h}(\vec{x})$ représentent respectivement m contraintes d'inégalité et p contraintes d'égalité.

2.5 - MINIMUM GLOBAL

On a la fonction : $f : \Omega \subseteq R^n \rightarrow R$, tel que $\Omega \neq \emptyset$. Pour $x^* \in \Omega$, on dit que x^* est un optimum global si et seulement si :

$\forall x \in \Omega : f(x^*) \leq f(x)$ (2.2)

Tel que :

x^* : l'optimum global ;

f : la fonction objective ;

Ω : la région faisable ($\Omega \in s$) ;

s : l'espace de recherche globale ;

Le minimum global est illustré par le point M_3 dans la Figure (2.1).

2.6 - MINIMUM LOCAL

Un point x^* est un minimum local de la fonction f si et seulement si :

$$f(x^*) \prec f(x)\ , \forall x \quad \in V(x^*) \text{ et } x^* \neq x \tag{2.3}$$

D'où $V(x^*)$ définit un voisinage de x^*

Deux minimums locaux sont illustrés dans la figure (2.1) sont les points M_1 et M_2.

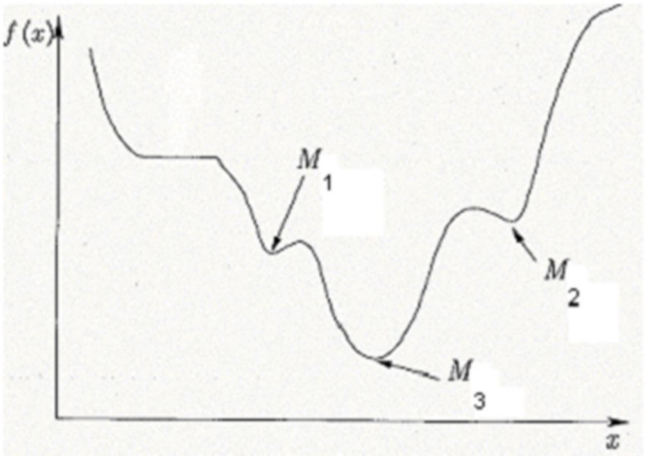


Fig. 2.1 - Illustration des différents minima d'une fonction objective

2.7 - TYPE DES PROBLEMES D'OPTIMISATION

Un problème d'optimisation est caractérisé par:

- le **domaine** des **variables de décision** : soit **continu** et on parle alors de problème **continu**, soit **discret** et on parle donc de problème **combinatoire** ;
- la **nature** de la **fonction objectif** à optimiser : soit **linéaire** et on parle alors de problème linéaire, soit **non linéaire** et on parle donc de problème non linéaire ;
- le **nombre** de **fonctions objectifs** à optimiser : soit une **fonction scalaire** et on parle alors de **problème mono-objectif**, soit une **fonction vectorielle** et on parle donc de problème **multi-objectif** ;
- la **présence** ou non des **contraintes** : on parle de problème **sans contrainte** ou **avec contrainte**
- sa **taille** : problème de **petite** ou de **grande taille** ;

- l'**environnement** : problème **statique** ou **dynamique** (la fonction objectif change dans le temps).

2.8 - OPTIMISATION AVEC CONTRAINTES

Les problèmes physiques enveloppent souvent des contraintes qui doivent être respectées par les solutions, l'ensemble des contraintes définissent le domaine de faisabilité, une solution qui viole une ou plusieurs contraintes est dite non faisable et elle ne peut pas être considérée comme une solution adéquate au problème, même si, elle optimise la fonction objective.
Les contraintes peuvent être traitées par [99-100] :

- L'élimination de toutes les solutions qui ne respectent pas les contraintes ou imposer aux opérateurs de ne créer que des solutions faisables.
- L'utilisation des fonctions de pénalités qui pénalisent les solutions qui violent les contraintes.
- La transformation du problème en un problème d'optimisation multi-objective, en reformulant les contraintes en des fonctions objectives supplémentaires

2.8.1 – FONCTIONS DE PENALITES

Les fonctions de pénalité introduisent une mesure du degré de satisfaction des contraintes. L'introduction des pénalités permet de transformer le problème d'optimisation avec contraintes en un problème d'optimisation sans contraintes (ou avec minimum de contraintes), plus facile à traiter. La formulation générale d'un problème d'optimisation avec contraintes est comme suit [99-100] :

Minimiser $f(x)$
Sujet de : $x \in A$ (2.4)
$x \in B$
Où : x est le vecteur objectif.

Les contraintes $x \in A$ sont considérées comme des contraintes paramétriques qui définissent le domaine de recherche.

Les contraintes $x \in B$ sont des contraintes dures, difficiles à satisfaire, elles déterminent la faisabilité de chaque solution.

En introduisant une fonction de pénalité le problème peut être reformulé comme suit [83]:

Minimiser $f(x) + p(d(x,B))$
Sujet de : $x \in A$ (2.5)

Avec :

d(x,B) est une fonction décrivant la distance de la solution *x* du domaine de faisabilité *B*.

p( ) est la fonction de pénalité monotone non décroissante avec *p(0) = 0.*

Pratiquement, les contraintes $x \in B$ sont exprimées en contraintes d'égalités et contraintes d'inégalités sous la forme :

$g_i(x) \leq 0$ pour $i = 1, \ldots, q$ (2.6)

$h_i(x) = 0$ pour $i = q + 1, \ldots, m$ (2.7)

Où :

q : le nombre de contraintes d'inégalité.

$(m - q)$: le nombre de contraintes d'égalité.

Il existe plusieurs formes possibles de fonction de pénalités, le choix d'une forme particulière dépend de la nature et de la complexité du problème à traiter.

2.8.1.1 – Fonctions de pénalités statiques

Une fonction de pénalité statique impose une sévérité de pénalité constante sur les solutions qui dépassent le domaine de faisabilité.

Une fonction de pénalité statique en fonction du nombre de contraintes violées peut être formulée comme suit [99] :

$$p(X) = \sum_{i=1}^{m} C_i \delta_i \quad \text{où} \quad \begin{cases} \delta_i = 1 & \text{si la contrainte i est violée} \\ \delta_i = 0 & \text{si la contrainte i est satisfaite} \end{cases}$$

(2.8)

Où : C_i est une constante déterminant la sévérité de la pénalité imposée.

La forme de pénalité statique la plus utilisée introduit une pénalité en fonction de la distance de la région de faisabilité [99] :

$$p(X) = \sum_{i=1}^{m} C_i d_i^k \quad \text{où} \quad d_i = \begin{cases} \delta_i g_i(x) & \text{pour} \quad i = 1, \ldots, q \\ |h_i(x)| & \text{pour} \quad i = q+1, \ldots, m \end{cases} \tag{2.9}$$

Où :

C_i est un paramètre déterminant la sévérité de la pénalité.

d( ) est la distance euclidienne entre la solution *x* et de la contrainte *i*, $k = 1 \div 2$.

Les fonctions de pénalités statiques présentent un inconvénient majeur : la sévérité de la pénalité est introduite par un coefficient constant C_i, qui doit être déterminé pour chaque problème. La sévérité de la pénalité doit être finement posée car :

- Si C_i est trop élevé les solutions finales seront faisables mais pas optimales.
- Si C_i est trop faibles, les solutions finales seront optimales mais pas faisables.

2.8.1.2 – Fonctions de pénalités dynamiques

Dans les problèmes d'optimisation avec contraintes, il existe souvent un conflit entre permettre l'exploration des régions infaisables et l'exigence d'une solution finale faisable. Les fonctions de pénalités dynamiques permettent d'augmenter la sévérité de la pénalité imposée avec la progression de la recherche. Imposer une faible sévérité au début de la recherche permet une bonne exploration de l'espace. L'augmentation de la sévérité au cours de la recherche permet de la diriger vers les régions faisables et garantir, ainsi, une pleine satisfaction des contraintes par les solutions finales.

La forme générale d'une fonction de pénalité dynamique basée sur la mesure de la distance de la région de faisabilité peut être décrite comme suit [99] :

$$p(X,t) = \sum_{i=1}^{m} s_i(t) d_i^k \qquad \text{où} \quad d_i = \begin{cases} \delta_i g_i(x) & \text{pour} \quad i = 1, \dots, q \\ |h_i(x)| & \text{pour} \quad i = q+1, \dots, m \end{cases} \tag{2.10}$$

Où : $s_i(t)$ est une fonction monotone non décroissante, *t* peut être considéré comme le nombre de génération. $s_i(\,)$ peut être comme suit : $s_i(t) = (C_i t)^{\alpha}$ avec $\alpha = 1 \div 2$.

2.8.1.3 – Fonctions de pénalités adaptatives

Les fonctions de pénalités dynamiques sont généralement efficaces, mais la recherche peut être mieux affinée en incorporant un aspect adaptatif à la fonction de la pénalité. Les fonctions de pénalités adaptatives, permettent d'ajuster la sévérité de la pénalité imposée en fonction de la faisabilité des solutions des générations précédentes. Un exemple d'une fonction de pénalité adaptative est [99] :

$$p(X,t) = \sum_{i=1}^{m} \tau_t d_i^k \tag{2.11}$$

Avec :

$$\tau_{t+1} = \begin{cases} \tau_t \beta_1 & \text{si la meilleure solution des } Nf \text{ générations précédentes est infaisable} \\ \tau_t / \beta_2 & \text{si la meilleure solution des } Nf \text{ générations précédentes est faisable} \\ \tau_t & \text{sinon} \end{cases} \quad (2.12)$$

Cette méthode utilise deux constantes β_1 et β_2 avec $\beta_1 > \beta_2 > 1$ pour adapter la pénalité appliquée, et utilise l'évaluation de la faisabilité des meilleures solutions trouvées dans un intervalle de *Nf* générations. La pénalité est mise à jour chaque *Nf* génération.

2.9 - METHODES DE RESOLUTION D'UN PROBLEME D'OPTIMISATION

Pour résoudre un problème d'optimisation il faut :

- élaborer un modèle (mathématique) : l'expression de l'objectif à optimiser et les contraintes à respecter.
- développer un algorithme de résolution.
- évaluer la qualité des solutions produites.

En mathématiques, *l'optimisation* recouvre toutes les méthodes qui permettent de déterminer l'optimum d'une fonction, avec ou sans contraintes.

La figure (2.2) donne une classification générale des méthodes d'optimisation.

Avant de discuter les méthodes de résolution, il est préférable de donner quelques terminologies :

Méthode complète : trouve toujours une solution.

Méthode optimale : trouve toujours la meilleure solution (optimum global).

Méthode exacte : explore l'espace de recherche dans sa totalité (énumération intelligente) pour obtenir la solution optimale.

Méthode approchée (approximative) : explore une sous-partie de l'espace de recherche.

Méthode déterministe : exécute toujours la même suite d'opérations.

Méthode probabiliste (ou **stochastique**) : fait des choix probabilistes guidés par des tirages aléatoires.

Recherche locale (ou voisinage) : une solution initiale modifiée itérativement.

Heuristique : (du grec *heuriskêin*, trouver), qui consiste ou qui tend à trouver. Une heuristique est une méthode approchée conçue pour un problème d'optimisation particulier permettant de trouver des solutions avec un temps de calcul raisonnable (pas de garantie d'optimalité).

Métaheuristique : est dérivé de la composition de deux mots grecs « heuriskein » qui signifie « trouver », « méta » qui signifie « au-delà » : trouver avec un plus haut niveau d'abstraction. Une

méta-heuristique est une heuristique généraliste, pouvant s'appliquer à plusieurs problèmes d'optimisation.

Hybridation : mélange des approches.

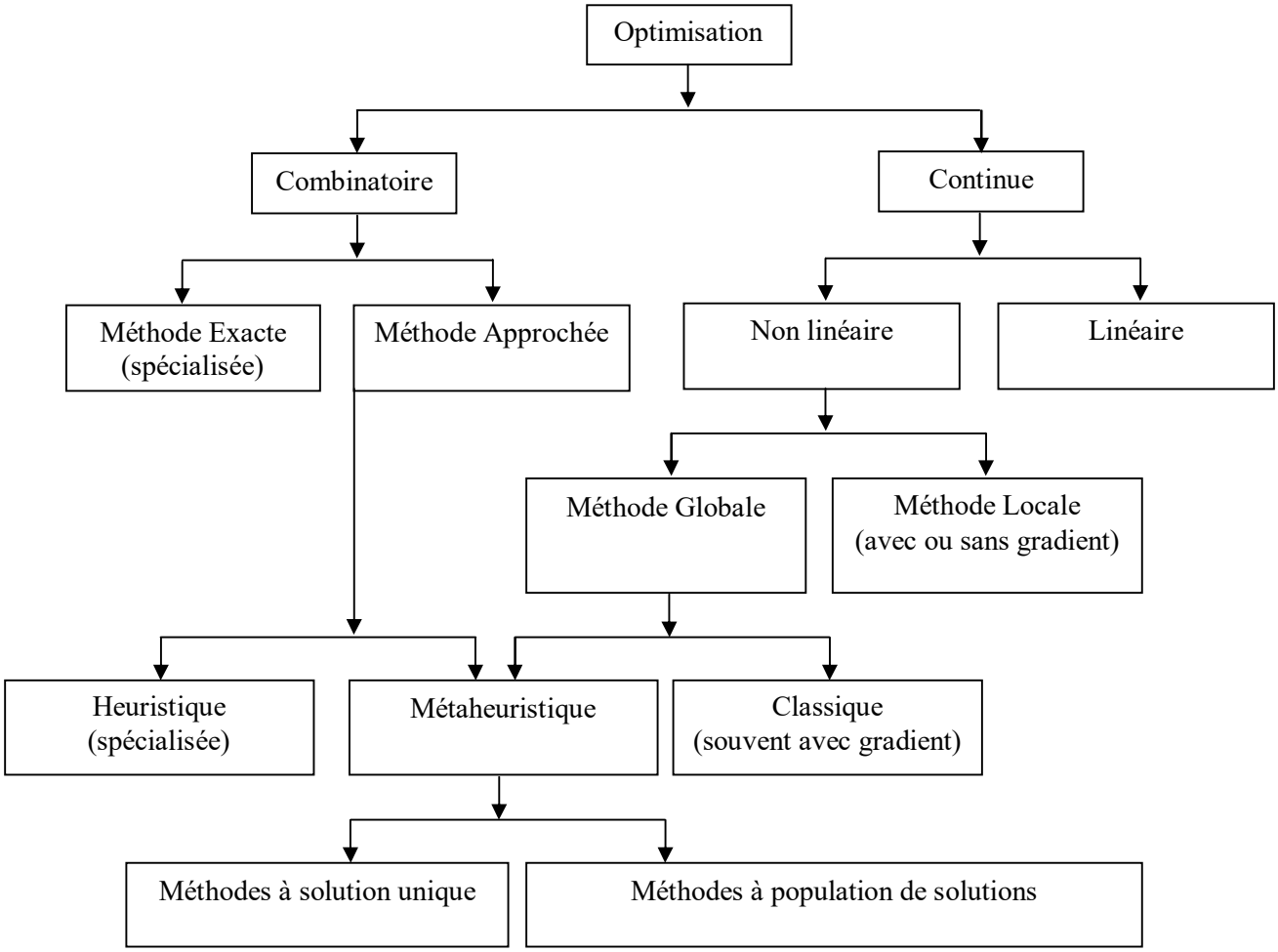


Fig. 2.2 - Classification générale des méthodes d'optimisation

- Nombreuses sont les méthodes d'optimisation. On peut cependant les classer selon le mode de recherche de l'optimum, en deux grands groupes : les méthodes **déterministes** et les méthodes **stochastiques** [101].
- Pour les problèmes d'optimisation combinatoires, on peut distinguer deux nominations de méthodes de résolution :
  - les méthodes **exactes** dédiées à résoudre optimalement les petites instances.
  - les méthodes **approchées** : les heuristiques et en particulier les métaheuristiques (génériques) permettant d'approximer les meilleures solutions sur les plus grandes instances.
- Une façon d'améliorer les performances d'un algorithme ou de combler certaines de ses lacunes consiste à le combiner avec une autre méthode (**hybridation**). L'intérêt de ces

approches coopératives est de permettre à différentes méthodes d'optimisation de combiner leurs avantages dans le but d'améliorer les performances globales afin d'obtenir de bon résultats. Actuellement, les meilleurs résultats obtenus sont issus de ce type d'approche.

2.10 - METHODES DETERMINISTES

Lorsque l'évolution de la méthode de résolution est prévisible et ne laisse aucune place au hasard, celle-ci est qualifiée de **déterministe**. Dans cette première classe, on rencontre toutes les méthodes qui cherchent le minimum d'une fonction en se basant sur la connaissance d'une direction de recherche, souvent donnée par le gradient de cette fonction. Dans le cas d'optima multiples, elles s'arrêtent sur le premier rencontré. Parmi ces méthodes : les méthodes statistiques, les méthodes mathématiques, ….etc.

2.11 - METHODES STOCHASTIQUES

Les méthodes d'optimisation stochastiques s'appuient sur des mécanismes de transition probabilistes et aléatoires. Cette caractéristique indique que plusieurs exécutions successives de ces méthodes peuvent conduire à des résultats différents pour une même configuration initiale d'un problème d'optimisation.

Ces méthodes ont une grande capacité à trouver l'optimum global du problème. Contrairement à la plupart des méthodes déterministes, elles ne nécessitent ni point de départ, ni la connaissance du gradient de la fonction objectif pour atteindre la solution optimale. Cependant, elles demandent un nombre important d'évaluations de la fonction objectif.

2.12 - METHODES EXACTES

Les méthodes exactes cherchent à trouver de manière certaine la solution optimale en examinant de manière explicite ou implicite la totalité de l'espace de recherche. Elles ont l'avantage de garantir la solution optimale, néanmoins le temps de calcul nécessaire pour atteindre cette solution peut devenir très excessif en fonction de la taille du problème (explosion combinatoire) et le nombre d'objectifs à optimiser. Ce qui limite l'utilisation de ce type de méthode aux problèmes bi-objectifs de petites tailles [102].

2.13 - METHODES APPROCHEES

Méthodes souvent inspirées de mécanismes d'optimisation rencontrés dans la nature. Elles sont utilisées pour les problèmes où on ne connaît pas d'algorithmes de résolution en temps polynomial et pour lesquels on espère trouver une solution approchée de l'optimum global. Elles cherchent à produire une solution de meilleure qualité possible dictée par des heuristiques avec un temps de calcul raisonnable en examinant seulement une partie de l'espace de recherche. Dans ce cas l'optimalité de la solution n'est pas garantie ni l'écart avec la valeur optimale.

Les méthodes approchées sont divisées généralement en deux : **heuristiques** et **métaheuristiques** [103]:

- Les **heuristiques** sont des méthodes basées sur le problème lui-même.
- Les **métaheuristiques** sont des méthodes générales qui reposent sur un concept non lié au problème.

2.14 - METAHEURISTIQUES

Les métaheuristiques sont des algorithmes génériques d'optimisation : leur but est de permettre la résolution d'une large gamme de problèmes différents, sans nécessiter de changements profonds dans l'algorithme. Elles forment une famille d'algorithmes visant à résoudre des problèmes d'optimisation difficile, pour lesquels on ne connaît pas de méthode classique plus efficace. Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global. Ils s'inspirent généralement d'analogies avec la physique (recuit simulé), avec la biologie (algorithmes évolutionnaires) ou encore l'éthologie (colonies de fourmis, essaims particulaires).

Plusieurs classifications des métaheuristiques ont été proposées, la plupart distinguent globalement deux catégories : celles se basant sur une **solution unique (méthodes de trajectoire)** et celles se basant sur une **population de solution** [104].

2.14.1 - METAHEURISTIQUES A BASE DE SOLUTION UNIQUE

Elles manipulent une seule solution à la fois et tentent itérativement d'améliorer cette solution (méthode du recuit simulé, recherche tabou). Elles construisent une trajectoire dans l'espace des solutions en tentant de se diriger vers des solutions optimales.

2.14.2 - METAHEURISTIQUES A BASE DE POPULATION DE SOLUTIONS

Travaillent sur un ensemble de points de l'espace de recherche en commençant avec une population de solution initiale puis de l'améliorer au fur et à mesure des itérations (algorithmes évolutionnaires, colonies de fourmis, essaims particulaires…). L'intérêt de ces méthodes est d'explorer un très vaste espace de recherche et d'utiliser la population comme facteur «de diversité».

Dans la section qui suit, on essayera de donner un bref aperçu sur les méthodes métaheuristiques de base les plus populaires. Les méthodes des algorithmes génétiques, l'évolution différentielle, l'optimisation par essaims particulaires et le recuit simulé vont être décrites en détail dans le chapitre suivant.

2.15 - RECUIT SIMULÉ (SIMULATED ANNEALING)

La technique du recuit simulé a été proposée en 1983 par Kirkpatrick [105]. Elle s'appuie sur des travaux réalisés par Metropolis [106], qui ont pu décrire l'évolution d'un système en thermodynamique. Cette métaheuristique est basée sur une technique utilisée depuis longtemps par les métallurgistes qui, pour obtenir un alliage sans défaut, faisant alterner les cycles de réchauffage (ou de recuit) et de refroidissement lent des métaux.

Un métal est chauffé à une température très élevée, il devient liquide et peut occuper toute configuration. Quand la température décroît, le métal va se figer peu à peu dans une configuration qu'il est de plus en plus difficile à déformer, il est refroidi. En le réchauffant (recuit), le métal peut être retravaillé de nouveau pour lui donner la forme désirée. Il faut baisser lentement la température en marquant des paliers suffisamment longs pour que le corps atteigne l'équilibre thermodynamique à chaque palier de la température, ce qui permet d'obtenir à la fin du processus un matériau dans un état cristallin bien ordonné correspondant à un état d'énergie minimum. Par contre, si la baisse de température se fait de manière trop brutale, le matériau est amorphe et ses atomes sont figés dans un état désordonné traduisant un minimum local d'énergie [101], [107].

2.16 - RECHERCHE TABOUE (TABU SEARCH)

La méthode de recherche Tabou est une métaheuristique originalement développée par Glover en 1986 spécifiquement pour des problèmes d'optimisation combinatoire et qui permet de trouver d'une manière flexible un compromis entre la qualité de la solution et le temps de calcul. Le principe de cette méthode est d'ajouter au processus de recherche une mémoire flexible qui permet de mener une recherche plus "intelligente" dans l'espace des solutions [108]. Comme la méthode du

recuit simulé, la méthode de recherche Tabou fonctionne avec une seule configuration courante, qui est actualisée au cours des itérations successives. La nouveauté ici est que, pour éviter le risque de retour à une configuration déjà visitée, on tient à jour une liste de mouvements interdits, appelée liste tabou.

La figure (2.3) donne un algorithme de la recherche par liste de taboue. En effet, à partir d'une solution initiale *s* dans un ensemble de solutions local *S*, des sous-ensembles de solution *N(s)* appartenant au voisinage *S* sont générés. Par l'intermédiaire de la fonction d'évaluation nous retenons la solution qui améliore la valeur de *f*, choisie parmi l'ensemble de solutions voisines *N(s).* L'algorithme accepte parfois des solutions qui n'améliorent pas toujours la solution courante. Nous mettons en oeuvre une liste tabou (tabu list) T de longueur k contenant les k dernières solutions visitées, ce qui ne donne pas la possibilité à une solution déjà trouvée d'être acceptée et stockée dans la liste tabou. Alors le choix de la prochaine solution est effectué sur un ensemble des solutions voisines en dehors des éléments de cette liste tabou.

Algorithme 2 : Recherche tabou

1- Initialisation :

s_0 une solution initiale

$s \leftarrow s_0$, $s^* \leftarrow s_0$, $c^* \leftarrow f(s_0)$

$T = \emptyset$

2 - Générer un sous-ensemble de solution au voisinage de *s*

$s' \in N(s)$ tel que $\forall\, x \in N(s)$, $f(x) \geq f(s')$ et $s' \notin T$

Si $f(s') < c^*$ alors $s^* \leftarrow s'$ et $c^* \leftarrow f(s')$

Mise-à-jour de T

3 - Si la condition d'arrêt n'est pas satisfaite retour à l'étape 2

Fig. 2.3 – Algorithme de recherche par liste Tabou

Quand le nombre k est atteint, chaque nouvelle solution sélectionnée remplace la plus ancienne dans la liste. Comme critère d'arrêt on peut par exemple fixer un nombre maximum d'itérations sans amélioration de s^*, ou bien fixer un temps limite après lequel la recherche doit s'arrêter.

La méthode modélise ainsi une forme primaire de mémoire à court terme. Dans sa forme de base, la méthode de recherche tabou présente l'avantage de comporter moins de paramètres que la méthode de recuit simulé. Dans la recherche Tabou avec des variables continues, deux concepts

sont mis en jeu : le voisinage d'un point donné et un mouvement aléatoire dans le voisinage. Pour un point X et un pas donné p, le voisinage $V(X, p)$ est défini comme suit [101], [108] :

$$V(X, p) = Y : |X - Y| \leq p \tag{2.14}$$

où le point Y est généré aléatoirement dans le voisinage $V(X, p)$ de X.
La taille du voisinage est le nombre de points Y générés dans le voisinage V de X avec un pas donné p. La performance de la recherche tabou dépend de la taille du voisinage et la longueur de la liste tabou.

En partant d'une solution quelconque X appartenant à l'ensemble des solutions S, la recherche se déplace vers une solution Y située dans le voisinage V par exploration itérativement de l'espace des solutions. Au début, le vecteur pas couvre la totalité de l'espace de recherche puis il diminue à chaque itération. Parmi tous les points générés dans le voisinage V de X, le meilleur est retenu et sera le centre du prochain voisinage.

L'algorithme a besoin d'une mémoire qui conserve la trace des dernières solutions visitées pendant un moment donné. Ces solutions sont déclarées taboues, d'où le nom de la méthode. Elles sont stockées dans une liste de longueur k fixe appelée liste taboue. Une nouvelle solution n'est acceptée que si elle n'appartient pas à la liste taboue. Le retour vers des solutions déjà explorées dépendant donc de la longueur de liste taboue. Elle sera d'autant plus difficile que cette dernière est longue. En conséquence, la recherche sera dirigée vers des régions non explorées.

La recherche par liste tabou est une méthode de recherche locale, et la structure de son algorithme de base est proche de celle du recuit simulé, avec l'avantage d'avoir un paramétrage simplifié. En revanche, la méthode tabou exige une gestion de la mémoire de plus en plus lourde en mettant des stratégies de mémorisation complexe.

2.17 - ALGORITHMES EVOLUTIONNAIRES (EA)

Les algorithmes évolutionnaires EA, élaborés au cours des années 1950, forment une famille d'algorithmes d'optimisation stochastiques inspirés de l'évolution biologique des espèces. Ils s'inspirent de la théorie Darwinienne de la sélection naturelle des espèces : les individus qui ont hérité de caractères bien adaptés à leur milieu ont tendance à vivre assez longtemps pour se reproduire, alors que les plus faibles ont tendance à disparaître.

Un algorithme évolutionnaire est typiquement composé de trois éléments fondamentaux :

- une population constituée de plusieurs individus représentant des solutions potentielles (configurations) du problème donné, permettant de mémoriser les résultats à chaque étape du processus de recherche.

- un mécanisme d'évaluation (fitness) des individus permettant de mesurer la qualité de l'individu.
- un mécanisme d'évolution de la population permettant, grâce à des opérateurs prédéfinis (tels que la sélection, la mutation et le croisement), d'éliminer certains individus et d'en créer de nouveaux.

Au cours des années 1960 et 1970, avec l'avènement des calculateurs, de nombreuses tentatives de modélisation de l'évolution ont été entreprises. Plusieurs approches ont ainsi émergé. On peut distinguer cinq grandes classes d'algorithmes évolutionnaires qui se différencient par leur manière de représenter l'information et par leur façon de faire évoluer la population d'une génération à l'autre :

– **Algorithmes génétiques (GA)**

En 1809 J.B Lamarck émit l'hypothèse de l'évolution (les organes d'un animal évolue en fonction de ses besoins). En 1859 C. Darwin émit l'idée de la sélection naturelle (dans tout espèce les meilleurs sont sélectionnés). Les bases de l'évolution étaient posées. En 1901 De-Vries exposa sa théorie du mutationnisme (les variations responsables de l'évolution ne se faisaient pas dans le temps mais de façon soudaine et se produisaient dans l'œuf). Les bases de la génétique étaient posées [102]. Ces concepts vont être utilisées en informatique : En 1975 Jhon Holland proposa l'Algorithme Génétique (GA). En 1989 Goldberg exposa les fondements mathématiques des GA [12].

Les GA sont utilisés dans le but de découvrir une solution à un problème donné, sans information ou peu d'informations a priori sur l'espace de recherche. Le mécanisme des GA consiste à faire évoluer, à partir d'un tirage initial, un ensemble de points de l'espace de recherche vers l'optimum du problème. Au cours de l'évolution, on utilise des opérateurs inspirés de l'évolution naturelle (croisement, mutation et sélection) pour en former de nouvelles solutions en essayant d'hériter des bonnes caractéristiques des solutions parents. Un critère de qualité est nécessaire pour discriminer différentes solutions, cette fonction s'appelle fitness ou fonction objective ou d'adéquation.

– **Statégies d'évolution (ES)**

Ces algorithmes forment une famille de métaheuristiques d'optimisation. Elles sont inspirées de la théorie de l'évolution. La méthode fut initialement proposée par Ingo Rencherberg, en 1965, à l'université technique de Berlin, en Allemagne. Elle était initialement conçue pour résoudre des problèmes à variables continues.

Dans sa version de base, l'algorithme manipule itérativement un ensemble de vecteurs de variables réelles, à l'aide d'opérateurs de mutation et de sélection. L'étape de mutation est classiquement effectuée par l'ajout d'une valeur aléatoire, tirée au sein d'une distribution normale. La sélection s'effectue par un choix déterministe des meilleurs individus, selon l'échelle de valeur de la fonction objectif.

Les stratégies d'évolution ES sont souvent nommées sous la forme (μ,λ)-ES ou (μ+λ)-ES qui désigne deux différentes stratégies, où μ désigne la dimension de l'ensemble de parents pour produire un ensemble de descendants de dimension λ(>= μ). (μ,λ)-ES indique que μ vecteurs sont choisis pour former la nouvelle génération parmi les meilleurs vecteurs λ. Pour la stratégie (μ+λ)-ES, les μ vecteurs de la nouvelle génération sont sélectionnés parmi le meilleurs entre les μ parents et leurs λ descendants [109].

– **Programmation évolutionnaire (EP)**

La programmation évolutionnaire EP à été introduite par Fogel [110], et a été initialement conçue pour faire évoluer les machines à états finis et a par la suite été étendue aux problèmes d'optimisation de paramètres. À la différence des autres branches de la famille des algorithmes évolutionnaires, la EP n'utilise pas une représentation spécifique. Elle se focalise sur l'opérateur de mutation approprié à un problème spécifique et n'utilise pas dans sa version originale la recombinaison des individus par croisement. Pour résoudre un problème, on génère aléatoirement une population de taille μ, et chaque individu i va générer λ descendants suite à l'application de l'opérateur de mutation. Une opération de sélection naturelle permet par la suite de former une nouvelle génération de taille μ à partir des parents et des descendants [109].

– **Programmation génétique (GP)**

John Koza [111] a été le premier à exprimer formellement le concept de la programmation génétique GP au début des années 1990 en s'appuyant sur les techniques des algorithmes génétiques. Cette technique est une extension du modèle d'apprentissage des GA à l'espace des programmes informatiques. 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 satisfait la résolution du problème choisi.

– **Evolution différentielle (DE)**

L'évolution différentielle (DE) est une technique d'optimisation continue proposée par Storn et Price en 1997 [112]. C'est une version améliorée des algorithmes génétiques, initialement conçue pour résoudre des problèmes à variables continues. Sa stratégie consiste à biaiser un opérateur de mutation, appliqué à un individu, en fonction des différences calculées entre d'autres individus sélectionnés aléatoirement. Les approches évolutionnaires s'appuient sur un modèle commun présenté par l'algorithme de la figure (2.4) [113].

Algorithme 3 : Modèle commun des algorithmes évolutionnaires

1 - **Initialisation** de la population de μ individus
2 – **Evaluation** des μ individus
3 – **tant que** le critère d'arrêt n'est pas satisfait **faire**
4 – | **Sélection** de ρ individus en vue de la phase de reproduction
5 – | **Croisement** des ρ individus sélectionnés
6 – | **Mutatio**n des λ enfants obtenus
7 – | **Evaluation** des λ enfants obtenus
8 – | **Sélection** pour le remplacement
9 – **fin**

Fig. 2.4 - Algorithme évolutionnaire générique

Les individus soumis à l'évolution sont des solutions possibles du problème posé. L'ensemble de ces individus constitue une population. Cette population évolue durant une succession d'itérations, appelées générations. Au cours de chaque génération, une série d'opérateurs est appliquée aux individus, pour créer la population de la génération suivante. Chaque opérateur utilise un ou plusieurs individus, appelés parents, pour engendrer de nouveaux individus, appelés enfants. A la fin de chaque génération, une sélection d'enfants créés durant la génération remplace un sous-ensemble d'individus de la population.

Un algorithme évolutionnaire dispose de trois opérateurs principaux :

1. **un opérateur de sélection**, qui favorise la propagation des meilleures solutions dans la population, tout en maintenant une certaine diversité génétique au sein de celle-ci.
2. **un opérateur de croisement**, mis en œuvre lors de la phase de création des enfants. Son but est d'échanger les gènes des différents parents pour créer les enfants. Un exemple de

croisement simple, pour des individus codés en représentation binaire, est présenté à la figure (2.5).

3. **un opérateur de mutation**, qui consiste à tirer aléatoirement une composante de l'individu parent et à la remplacer par une valeur aléatoire. L'apport d'un caractère aléatoire à la création de la descendance permet ainsi de maintenir une certaine diversité dans la population. La figure (2.6) montre un exemple de mutation, pour un individu codé en représentation binaire.

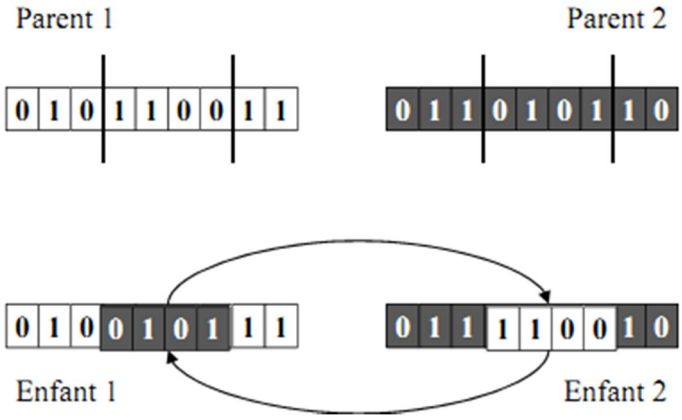


Fig. 2.5 - Exemple d'opérateur de croisement en représentation binaire

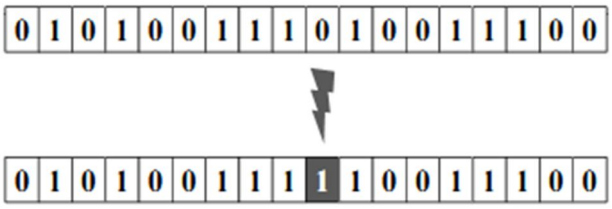


Fig. 2.6 - Exemple d'opérateur de mutation en représentation binaire

2.18 - OPTIMISATION PAR COLONIES DE FOURMIS (ACO)

La famille des algorithmes ACO (Ant Colony Optimization) a été mise au point par Marco Dorigo puis par d'autres depuis le début des années 90 [23-28], [115]. C'est une approche utilisant l'intelligence en essaim. Elle résulte de l'observation des insectes sociaux, en particulier des fourmis, qui résolvent naturellement des problèmes complexes. Une telle aptitude s'avère possible

en raison de la capacité des fourmis à communiquer entre elles indirectement, par le dépôt sur le sol de substances chimiques, appelées phéromones [113].

En effet, si un obstacle est introduit sur le chemin des fourmis (Fig. 2.7), ces dernières vont, après une phase de recherche, avoir tendance toutes à prendre le plus court chemin entre le nid et l'obstacle. Plus le taux de phéromone à un endroit donné est important, plus une fourmi va avoir une grande probabilité à être attirée par cette zone. Les fourmis qui sont arrivées le plus rapidement au nid en passant par la source de nourriture sont celles qui ont pris la branche la plus courte du trajet. Il en découle donc que la quantité de phéromones sur ce trajet est plus importante que sur le trajet le plus long. De ce fait, le plus court chemin a une probabilité plus grande d'être pris par les fourmis que les autres chemins, et il sera donc pris par toutes les fourmis. En considérant que la phéromone s'évapore, les chemins les moins renforcés finissent par disparaître, ce qui amène toutes les fourmis à suivre ce chemin le plus court.

L'algorithme de colonies de fourmis a été à l'origine principalement utilisé pour produire des solutions quasi-optimales au problème du voyageur de commerce, puis, plus généralement, aux problèmes d'optimisation combinatoire. Ensuite, son emploi se généralise à plusieurs domaines [23-28], [106], [113-115].

Les fourmis partent initialement de la source et parcourent les différents chemins possibles (Fig. 2.7) jusqu'à la fin. Les quantités de phéromone sont initialement égales. Sur leur chemin de retour, les fourmis déposent de la phéromone sur les différents chemins.

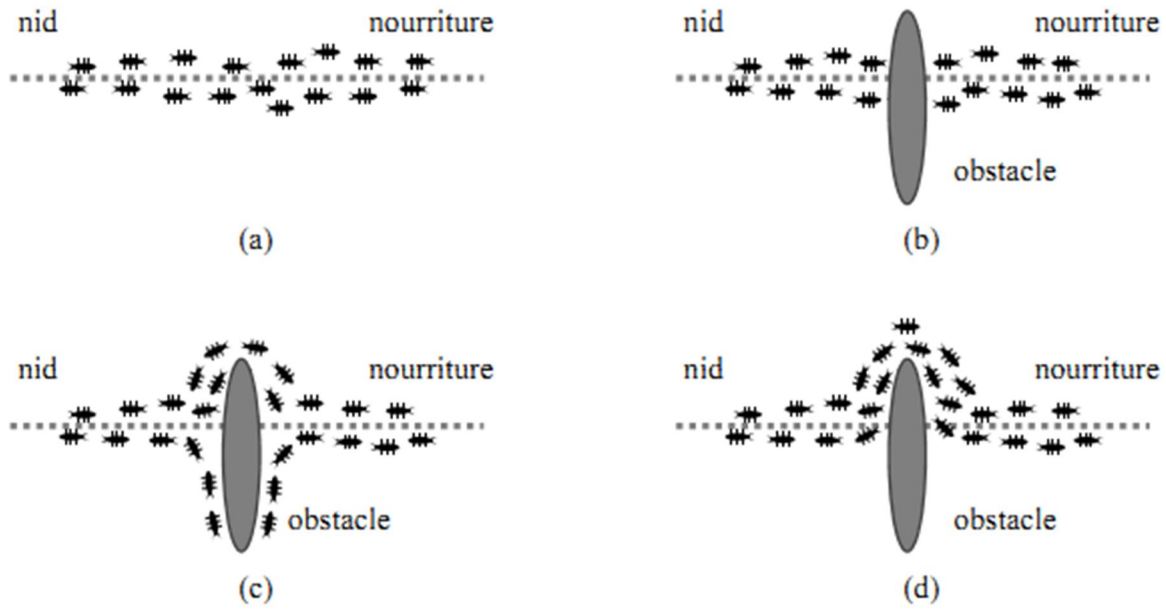


Fig. 2.7 - Illustration de la capacité des fourmis à chercher de la nourriture en minimisant leur parcours. (a) Recherche sans obstacle, (b) Apparition d'un obstacle, (c) Recherche du chemin optimal, (d) Chemin optimal trouvé.

Les valeurs des quantités de phéromone sont renouvelées selon l'équation :

$$\tau_i = \tau_i + \frac{Q}{l_i} \quad (2.15)$$

avec Q une constante et l_i la longueur du chemin parcouru. Donc la nouvelle valeur de la quantité de phéromone est inversement proportionnelle à la longueur du chemin.

Un aller-retour d'une seule fourmi constitue une construction complète de la solution. Le processus est réitéré, et les fourmis repartent à nouveau de la source. Une fourmi donnée va choisir entre les chemins selon la probabilité :

$$P_{e_i} = \frac{\tau_i}{\sum_{i=1}^{n} \tau_i} \quad (2.16)$$

D'autres changements affectent la phéromone, l'évaporation. Ce dernier changement a pour objectif d'empêcher une augmentation continue d'une seule solution, et de réduire l'importance d'autres solutions :

$$\tau_i = \tau_i.(1-\rho) \quad \text{avec} \quad \rho \in [0,1] \quad (2.17)$$

ρ est le paramètre qui va régler la quantité de phéromone déposée, donc c'est le paramètre qui va influencer la vitesse de convergence de l'algorithme vers une solution donnée.

2.19 - OPTIMISATION PAR ESSAIM PARTICULAIRE

L'optimisation par essaim particulaire (PSO) est une métaheuristique d'optimisation qui a été à l'origine développée par un psychologue social (James Kennedy) et un ingénieur électrique (Russell Eberhart) en 1995 [29], [116-118]. La méthode a été développée par une simulation des modèles sociaux simplifiés. Elle s'inspire du comportement social des animaux évoluant en essaim, tels que les nuées d'oiseaux et les bancs de poissons. En effet, tout comme ces animaux se déplacent en groupe pour trouver la source de nourriture ou éviter les prédateurs, les algorithmes à essaim de particules recherchent des solutions pour un problème d'optimisation.

Les individus de l'algorithme sont appelés particules et la population est appelée essaim. Dans cet algorithme, une particule décide de son prochain mouvement en fonction de sa propre expérience, qui est dans ce cas la mémoire de la meilleure position qu'elle a rencontrée, et en fonction de son meilleur voisin.

Cette méthode d'optimisation se base sur la collaboration des individus entre eux. Elle a d'ailleurs des similarités avec les algorithmes de colonies de fourmis, qui s'appuient eux aussi sur le concept d'auto-organisation. Le but d'un algorithme de PSO est de réduire au minimum une fonction fitness f qui dépend d'un ensemble de variables inconnues $(x_1, x_2, x_3, \ldots, x_n)$. Une solution

candidat pour ce problème est traduite dans le formalisme du PSO comme une position d'une particule *i* de l'essaim; $S_i = (x_{1i}, x_{2i}, x_{3i}\ldots,x_{ni})$.

2.20 - ALGORITHME DE COLONIE D'ABEILLE ARTIFICIELLE

Plusieurs processus biologiques et naturels ont influencé beaucoup de chercheurs à établir de nouvelles méthodes d'optimisation d'une manière croissante. Un certain nombre d'algorithmes d'intelligence d'essaim, basée sur le comportement des abeilles ont été présentés. Ces algorithmes sont divisés, principalement, en deux catégories en fonction de leur comportement dans la nature : le comportement d'accouplement et le comportement de la recherche de nourriture.

L'algorithme de colonie d'abeilles artificielle (ABC : Artificiel Bee Colony) est un algorithme de recherche à base de population proposée par D. Karaboga en 2005 [61-62], [119] pour l'optimisation de fonction. C'est un algorithme d'intelligence d'essaim, basée sur le comportement des abeilles.

Cette technique est développée en inspectant le comportement des abeilles réelles pour trouver la source de nourriture (solution possible), qui s'appelle le nectar, et partager l'information des sources de nourriture aux autres abeilles dans le nid. Chaque solution représente une position de nourriture potentielle dans l'espace de recherche et la qualité de la solution correspond à la qualité de la position alimentaire. Dans cet algorithme, les abeilles artificielles sont définies et classifiées en trois groupes :

- Abeilles employeuses qui recherchent la source de nourriture (solutions possibles) à partir d'un ensemble prédéfini des sources de nourriture et de partager cette information (danse frétillante de communication des abeilles) avec les autres abeilles dans la ruche.
- Abeilles spectatrices (abeilles d'observation) qui en fonction des informations qu'elles prennent des abeilles employeuses, elles recherchent une meilleure source de nourriture dans le voisinage des sources de nourriture mémorisées.
- Abeilles éclaireuses (scouts) qui sont chargées de trouver de nouvelles nourritures (le nectar de nouvelles sources).

2.21 - SYSTEMES IMMUNITAIRES ARTIFICIELS (AIS)

Les systèmes immunitaires artificiels (AIS) sont apparus dans les années 90 et sont inspirés du fonctionnement du système immunitaire naturel qui est un mécanisme de défense capable d'apprendre [66-67], [113], [120-123].

Ce système immunitaire naturel, composé d'organes, de cellules et de molécules, assure le maintien de l'intégrité de l'organisme qu'il défend. Il est responsable de la protection de

l'organisme contre les « agressions » d'organismes extérieurs. Ainsi, il peut être défini comme étant, l'ensemble des mécanismes biologiques permettant à l'organisme de reconnaître et de tolérer ce qui lui appartient en propre (le soi), et de reconnaître et de rejeter ce qui lui est étranger (le non soi) : les substances étrangères ou les agents infectieux auxquels il est exposé, mais aussi, ses propres constituants altérés (comme les cellules tumorales). Ce mécanisme de reconnaissance intelligent a été modélisé pour donner naissance au système immunitaire artificiel AIS.

L'immunité a deux composantes : naturelle (innée) et acquise (adaptative). Seule l'immunité acquise présente un réel intérêt informatiquement parlant vu ses facultés d'adaptation et d'évolution.

Les algorithmes des (AIS) peuvent être divisés en plusieurs grandes parties, chacune d'elles s'inspirant d'un comportement ou d'une théorie du système immunitaire biologique : la sélection clonale, la sélection négative et positive, les réseaux immunitaires (ou idiotypiques), la moelle osseuse et la théorie du danger.

La métaphore dont sont issus les algorithmes AIS met l'accent sur les aspects d'apprentissage et de mémoire du système immunitaire dit adaptatif (par opposition au système dit inné), notamment via la discrimination entre le soi et le non-soi [113]. En effet, les cellules vivantes disposent sur leurs membranes de molécules spécifiques dites antigènes. Chaque organisme possède ainsi une identité unique, déterminée par l'ensemble des antigènes présents sur ses cellules. Certaines cellules du système immunitaire, appelées lymphocytes (un type de globule blanc), possèdent des récepteurs capables de se lier spécifiquement à un antigène donné, permettant ainsi de reconnaître une cellule étrangère à l'organisme. Un lymphocyte ayant reconnu une cellule du non-soi va être incité à proliférer (en produisant des clones de lui-même) et à se différencier en cellule permettant de garder en mémoire l'antigène, ou en cellule permettant de combattre les agressions. Dans le premier cas, son clonage permettra à l'organisme de réagir plus vite à une nouvelle exposition à l'antigène (c'est le principe de la vaccination). A noter qu'un mécanisme d'hyper-mutation des cellules clonées assure la diversité des récepteurs dans l'ensemble de la population des lymphocytes. Dans le second cas, le combat contre les agressions est possible grâce à la production d'anticorps. Ces étapes sont résumées dans la figure (2.8) [113], [120].

Le AIS s'inspire essentiellement des sélections opérées sur les lymphocytes, et des processus permettant la multiplication et la mémoire du système. En effet, ces caractéristiques sont capitales pour maintenir les propriétés auto-organisées du système. Dans le cadre de l'optimisation, on peut considérer les AIS comme une forme d'algorithme évolutionnaire présentant des opérateurs particuliers. Par exemple, l'opération de sélection est basée sur une mesure d'affinité (i.e. entre le récepteur d'un lymphocyte et un antigène). La mutation s'opère, quant à elle, via un opérateur d'hyper-mutation directement issu de la métaphore. Un exemple d'un AIS basique est présenté dans l'algorithme de la figure (2.9), où *t* désigne l'itération courante [113].

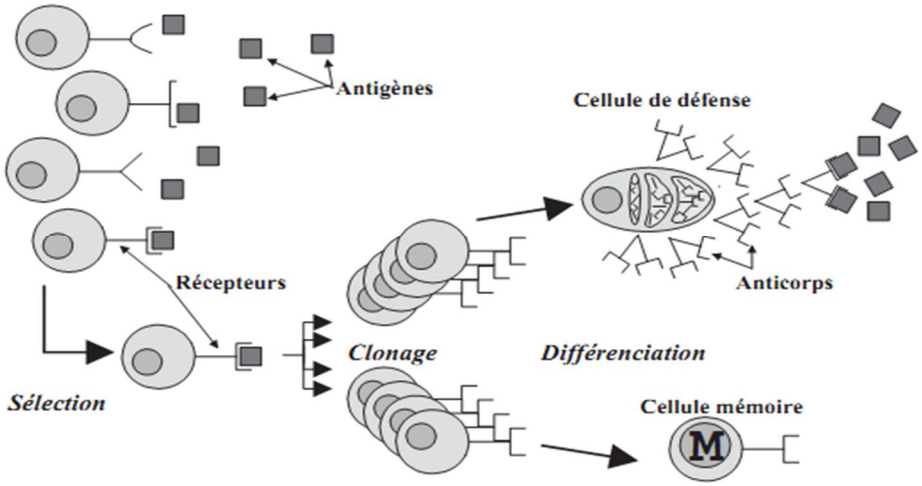


Fig. 2.8 – Sélection par clonage : des lymphocytes, présentant des récepteurs spécifiques d'un antigène, se différencient en cellule mémoire ou en cellule participant à la défense active de l'organisme par le biais d'anticorps.

Algorithme 4 : AIS

1 – t ← 1
2 – **Initialiser** aléatoirement une population $P(t)$ d'individus, composée de m cellules mémoires $P_m(t)$ et de r autres cellules $P_r(t)$: $P(t) = P_m(t) \cup P_r(t)$
3 – **Evaluer** l'affinité de chaque individu de $P(t)$
4 – **tant que** le critère d'arrêt n'est pas satisfait **faire**
5 – | **Sélectionner** un sous-ensemble $S(t)$ d'individus de plus haute affinité dans $P(t)$
6 – | **Cloner** chaque individu de $S(t)$ proportionnellement à son affinité, pour former une population $C(t)$
7 – | **Muter** chaque individu de $C(t)$ avec un taux inversement proportionnel à son affinité, pour former une population $C^*(t)$
8 – | **Evaluer** l'affinité de chaque individu de $C^*(t)$
9 – | **Sélectionner** les m individus de plus haute affinité dans $C^*(t) \cup P_m(t)$ pour former $P_m(t+1)$
10 – | **Remplacer** un sous-ensemble d'individus de plus faible affinité dans $P_r(t)$ par des nouveaux individus, tirés aléatoirement, pour former $P_r(t+1)$
11 – | **Evaluer** l'affinité de ces nouveaux individus
12 – t ←t+1
13 – **fin**

Fig. 2.9 - Exemple d'algorithme AIS basique

2.22 - ALGORITHMES INSPIRES DE L'INFORMATIQUE QUANTIQUE (QUANTUM COMPUTATION)

L'information quantique est la théorie de l'utilisation des spécificités de la physique quantique pour le traitement et la transmission de l'information. L'informatique quantique a été introduite au début des années 1980 par Feynmann et Beinoff [64-65], [124-125]. C'est le sous-domaine de l'informatique qui traite des calculateurs quantiques utilisant des phénomènes de la mécanique quantique, par opposition à ceux de l'électricité exclusivement, pour l'informatique dite « classique ». Les phénomènes quantiques utilisés sont l'intrication quantique et la superposition.

2.23 – ALGORITHMES DE RECHERCHE GRAVITATIONNELLE

L'algorithme de recherche gravitationnelle est un algorithme de recherche à base de population basé sur la loi de la gravité et de l'interaction entre masses. L'algorithme considère les agents comme des objets constitués de différentes masses. Les agents se déplacent ensemble à cause de la force d'attraction gravitationnelle agissant entre eux. Au cours du temps l'algorithme dirige les mouvements de tous les agents globalement vers les agents ayant des masses plus lourdes. Chaque agent dans l'algorithme est spécifié par quatre paramètres [68]: position de la masse, masse d'inertie, la masse gravitationnelle active et la masse gravitationnelle passive. La position de la masse d'un agent représente une solution du problème et la masse d'inertie d'un agent reflète sa résistance à faire son mouvement lent.

2.24 – ALGORITHME D'OPTIMISATION DE LA RECHERCHE DE NOURRITURE BACTERIENNE

L'optimisation de la recherche de nourriture bactérienne (BFA : Bacterial Foraging Algorithm) est une technique (développée en 2002 par Dr. Kevin Passino) [63] qui peut être utilisée pour trouver des solutions aux problèmes extrêmement difficiles, voire impossibles de maximisation ou minimisation numériques approximatives. BFO est une technique probabiliste qui modélise le comportement de recherche de nourriture et de reproduction des bactéries courantes telles que l'E.coli afin de résoudre les problèmes d'optimisation numérique où il n'y a aucune approche déterministe efficace.

L'optimisation par algorithme de BFA comprend les processus suivants : la chimiotactisme, l'essaimage, la reproduction, l'élimination et la dispersion. Un chimiotactisme est l'activité des bactéries qui se collecte dans les zones riches en nutriments spontanément. Un mécanisme de communication d'une cellule à cellule est créé pour simuler le comportement biologique de

l'essaimage des bactéries. La reproduction vient du concept de la sélection naturelle et seulement les bactéries mieux adaptées à leur environnement ont tendance à survivre et transmettre leurs caractères génétiques pour les générations futures, ceux qui sont moins adaptées ont tendance à être éliminées et dispersées [63].

2.25 - ALGORITHMES CO-EVOLUTIONNAIRES (CEA)

Les algorithmes Co-évolutionnaires (CEA) [69-70] constituent une nouvelle extension des algorithmes évolutionnaires. Ils sont basés sur le concept de la Co-évolution. La co-évolution consiste en plusieurs populations qui évoluent simultanément avec une interaction entre elles de sorte que l'opération d'évaluation de l'adéquation d'une population dépend de l'état de l'évolution des autres populations. Il existe deux types de co-évolution :

- la co-évolution compétitive, où les populations co-évoluées sont guidées par des objectifs opposés, la mesure d'adéquation des individus d'une population étant inversement proportionnelle à la performance des individus des autres populations (les individus bénéficient de l'échec de leurs pairs).
- la co-évolution coopérative, où les populations évoluent ensemble afin de résoudre différents éléments d'un même problème (les individus réussissent ou échouent ensemble dans la collaboration) [70].

2.26 - ALGORITHME DE RECHERCHE D'HARMONIE

L'algorithme de recherche d'harmonie (HS : Harmony Search) [73] est une méta-heuristique basée sur le processus de performance musicale, qui consiste à trouver l'harmonie parfaite dans un orchestre, où chaque musicien joue une note pour trouver une meilleure harmonie.

2.27 - ALGORITHMES DE LA RECHERCHE COUCOU

Ils sont inspirés du comportement de reproduction et de vie d'une espèce spéciale d'oiseaux nommés Coucou [74-76]. Ces derniers ont un caractère parasitaire dans la ponte, l'incubation et la nourriture de leurs poussins. En fait, certains coucous ne construisent jamais leurs propres nids pour pondre leurs propres œufs. Ils parasitent des nids d'autres oiseaux d'autres espèces. Après la ponte, les coucous se sauvent en s'affranchissant de leurs responsabilités parentales envers leurs œufs. Ils confient leurs œufs aux oiseaux hôtes qui seront leurs parents adoptifs. Les œufs coucous risquent d'être tués par leurs parents adoptifs si leur degré de similitude avec le modèle des œufs de l'oiseau

hôte est faible. La première métaheuristique inspirée coucou (CS : Cuckoo Search) a été proposée en 2009 par Xin-She Yang et Suash Deb [74].

2.28 – ALGORITHME DU TROU NOIR

C'est une nouvelle métaheuristique inspirée du phénomène des trous noirs [77]. Un trou noir est un objet céleste si compact que l'intensité de son champ gravitationnel empêche toute forme de matière ou de rayonnement de s'en échapper. De tels objets ne peuvent ni émettre, ni réfléchir la lumière et sont donc noirs, ce qui en astronomie revient à dire qu'ils sont invisibles. Par rapport à d'autres algorithmes d'optimisation bien connus, l'algorithme d'optimisation inspiré de ce phénomène est simple et caractérisé par un nombre moins de paramètres à régler.

2.29 - ALGORITHME D'OPTIMISATION DU LOUP GRIS

L'algorithme d'optimisation du loup gris (GWO : gray wolf optimizer) [78-79] est une nouvelle méta-heuristique basée sur l'intelligence en essaim. Elle était introduite pour la première fois en 2014 par S. Mirjalili [79]. Elle imite le comportement de chasse chez les loups gris.

2.30 – ALGORITHMES HYBRIDES

Afin d'améliorer les performances des algorithmes d'optimisation ou de combler certaines de ses lacunes, certains auteurs ont proposé des algorithmes hybrides. L'hybridation consiste à combiner les caractéristiques de deux méthodes différentes pour en tirer ses avantages. Une hybridation idéale engendre une méthode hybride qui cumule les bonnes propriétés de ses constituants tout en inhibant leurs points faibles. Actuellement les meilleurs résultats obtenus sont issus de ce type d'approche.

2.31 – CONCLUSION

La résolution d'un problème d'optimisation consiste à trouver l'état d'un système jugé le plus favorable au regard des circonstances données.

Nombreuses sont les méthodes d'optimisation, mais on peut généralement les classer selon le mode de recherche de l'optimum, en deux grands groupes : les méthodes déterministes et les méthodes stochastiques.

Depuis une dizaine d'années, des progrès importants ont été réalisés avec l'apparition d'une nouvelle génération de méthodes approchées puissantes et générales, souvent appelées métaheuristiques.

Les métaheuristiques constituent une classe de méthodes approchées adaptables à un très grand nombre de problèmes d'optimisation. Elles sont généralement des algorithmes stochastiques itératifs qui ont révélé leur grande efficacité pour fournir des solutions approchées de bonne qualité pour un grand nombre de problèmes d'optimisation classiques et d'applications réelles de grande taille. C'est pourquoi l'étude de ces méthodes est actuellement en plein développement.

Durant les dernières années, plusieurs processus biologiques et naturels ont influencé beaucoup de chercheurs à établir de nouvelles méthodes d'optimisation d'une manière croissante. L'intelligence inspirée de la nature devient de plus en plus populaire et un nombre important de méthodes entraînées par des concepts issus de la nature / la biologie ont été développées.

Afin de résoudre des instances de taille et de difficulté croissantes, il faut mettre au point des méthodes toujours plus puissantes. L'hybridation de plusieurs techniques différentes est une approche dont l'objectif est de combiner les avantages de chacun au sein d'un même algorithme.

3 – METHODES PROPOSEES ET HYBRIDATION

3.1 – INTRODUCTION

L'utilisation des métaheuristiques a attiré l'attention de la communauté de recherche depuis 30 ans. Les deux premières décennies ont été marquées par l'application des métaheuristiques standards. Toutefois, ces dernières années, il est devenu évident qu'une concentration sur une seule métaheuristique pour résoudre un problème complexe est plutôt restrictif, une combinaison avec d'autres techniques d'optimisation peut fournir un comportement efficace de la méthode, ainsi une grande flexibilité, surtout lorsqu'il s'agit des problèmes du monde réel à grande échelle. Il peut être extrêmement bénéfique d'associer une méthode de recherche dont les caractéristiques d'exploration sont très élevées à une méthode de recherche dont le point fort est l'exploitation. D'où l'idée de méthodes hybrides.

Dans ce chapitre, les métaheuristiques de base les plus populaires inspirées de la nature et de la biologie vont être décrites en détail : algorithmes génétiques, évolution différentielle, optimisation par essaims particulaires et recuit simulé. Nous nous intéressons aussi aux différentes techniques et stratégies d'hybridation.

3.2 - ALGORITHMES GENETIQUES

3.2.1 – PRESENTATION ET PRINCIPE DE FONCTIONNEMENT

Les algorithmes génétiques (GA) font partie de la famille des algorithmes évolutionnaires [12], [126-127]. Ils s'inspirent de l'évolution naturelle des espèces. Les algorithmes génétiques s'inspirent de l'évolution darwinienne des populations biologiques, pour définir leurs mécanismes. Les principes de cette évolution sont tels que, dans une population d'individus ce sont les plus forts, c'est-à-dire les mieux adaptés au milieu, qui survivront et pourront donner une descendance ou une bonne progéniture.

C'est en 1950 que les algorithmes génétiques ont vu le jour par des biologistes utilisant des ordinateurs pour simuler l'évolution des organismes. Vers 1960, John Holland et son équipe adaptaient ces algorithmes pour la recherche de solutions à des problèmes d'optimisation [127], en développant une analogie entre un individu dans une population et une solution d'un problème dans

un ensemble de solutions. Egalement les travaux de David Goldberg ont largement contribué à les enrichir [12].

Dans un GA, un individu ou une solution est caractérisée par une structure de données qui représente son empreinte génétique. La force d'un individu, encore appelée son "fitness", peut être mesurée par la valeur de la fonction d'évaluation correspondante. Les opérateurs génétiques de croisement et de mutation agissent sur les structures de données associées aux individus et permettent de parcourir l'espace des solutions du problème. Le renouvellement de la population c'est-à-dire de l'ensemble de solutions courantes, autrement dit la création d'une nouvelle génération, est obtenu par itération de l'algorithme GA qui va créer de nouveaux individus et en déduire d'autres, c'est équivalent au mécanisme de sélection naturelle. En résumé, pour mettre en œuvre un GA, il est nécessaire de disposer [126] :

- D'une représentation génétique du problème, c'est-à-dire d'un codage approprié des solutions sous la forme de chromosomes.
- D'un moyen de créer la population initiale.
- D'une fonction d'évaluation ou fonction objective pour mesurer la force de chaque chromosome.
- D'un mode de sélection des chromosomes à reproduire.
- D'opérateurs génétiques adaptés au problème.
- De valeurs pour les paramètres qu'utilisent l'algorithme : la taille de la population, la probabilité d'appliquer tel ou tel opérateur.

Le vocabulaire utilisé pour un algorithme génétique est le même que celui de la théorie de l'évolution et de la génétique, on emploie le terme individu (solution potentielle), population (ensemble de solutions), génotype (une représentation de la solution), gène (une partie du génotype), parent, enfant, reproduction, croisement, mutation, génération, etc.

Le fonctionnement des algorithmes génétiques est extrêmement simple, on part d'une population de solutions potentielles (chromosomes) initiales, arbitrairement choisies. On évalue leur performance (Fitness) relative. Sur la base de ces performances on crée une nouvelle population de solutions potentielles en utilisant des opérateurs évolutionnaires simples : la sélection, le croisement et la mutation. Quelques individus se reproduisent, d'autres disparaissent et seuls les individus les mieux adaptés sont supposés survivre. On recommence ce cycle jusqu'à ce qu'on trouve une solution satisfaisante. En effet, l'héritage génétique à travers les générations permet à la population d'être adaptée et donc répondre au critère d'optimisation. La figure (3.1) illustre les principales étapes d'un algorithme génétique. Un algorithme génétique recherche le ou les extrema d'une fonction définie sur un espace de données.

Les GA sont alors basés sur les phases suivantes :

1. **Initialisation**. Une population initiale de *N* chromosomes est tirée aléatoirement.
2. **Évaluation**. Chaque chromosome est décodé, puis évalué.
3. **Sélection**. Création d'une nouvelle population de *N* chromosomes par l'utilisation d'une méthode de sélection appropriée.
4. **Reproduction**. Possibilité de croisement et mutation au sein de la nouvelle population.
5. **Retour** à la phase d'évaluation jusqu'à l'arrêt de l'algorithme.

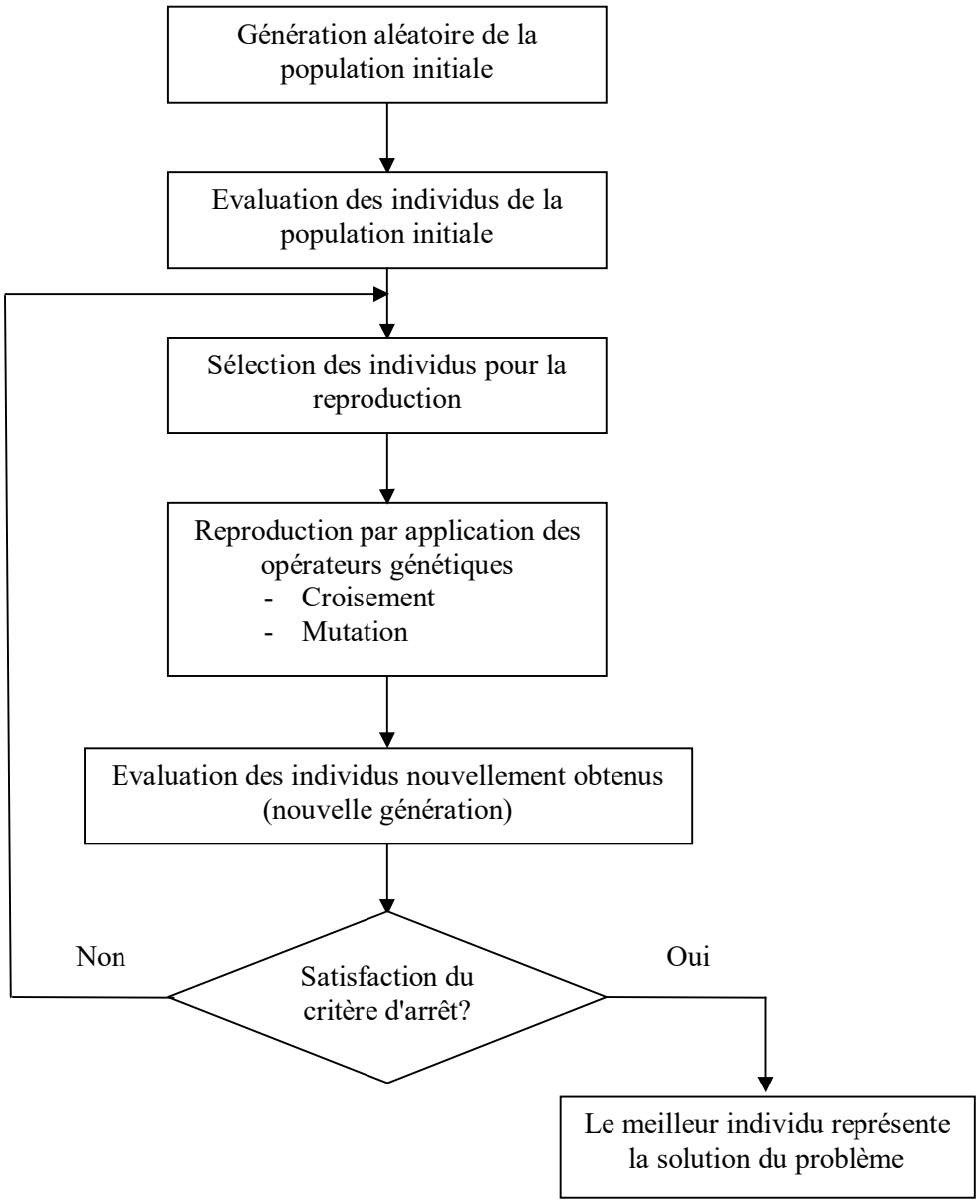


Fig. 3.1 - Organigramme d'un GA standard

3.2.2 - CODAGE ET POPULATION INITIALE

3.2.2.1 - Notion du codage [101]

La première étape dans le fonctionnement de l'algorithme génétique est le codage de l'ensemble des paramètres à optimiser en une chaîne. Pour une fonction (modèle d'un système) multivariable $f(x_1, x_2, \ldots, x_n)$, le codage se déroule de la façon suivante:

- Coder chaque variable x_i $(i = 1, \ldots, n)$.
- Construire les chaînes en concaténant les différents codes, c'est-à-dire, juxtaposer ces derniers, l'un après l'autre pour construire les chaînes (chromosome). Par exemple :

$$Chaîne = \underset{x_1}{(code_1)}\underset{x_2}{(code_2)} \ldots \underset{x_n}{(code_n)}$$

Il existe trois principaux type de codage : binaire, gray ou réel. Nous pouvons facilement passer d'un codage à l'autre. Certains auteurs n'hésitent pas à faire le parallèle avec la biologie et parlent de génotype en ce qui concerne la représentation binaire d'un individu (codage binaire) et de phénotype pour ce qui est de sa valeur réelle correspondante dans l'espace de recherche.

Un gène correspond de fait à un paramètre et un seul chromosome est constitué par un ensemble de gènes et décrit complètement un individu. L'ensemble des individus est appelé population. On aboutit ainsi à une structure présentant quatre niveaux d'organisation (Fig. 3.2). Un chromosome est une concaténation des gènes.

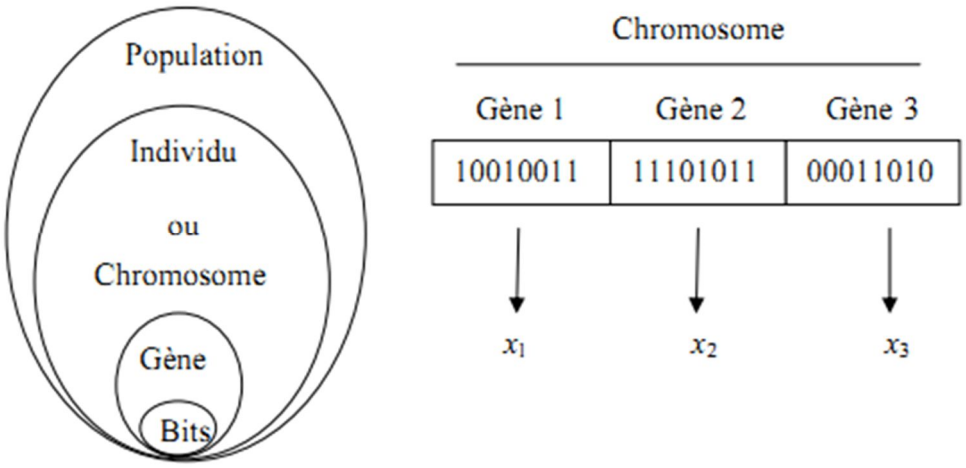


Fig. 3.2 - Les quatre niveaux d'organisation et notion du codage des variables d'un GA

3.2.2.2 - Génération de la population initiale

L'algorithme génétique démarre avec une population initiale d'individus dans le codage retenu. Le choix des individus conditionne fortement la rapidité de l'algorithme. Néanmoins, une initialisation aléatoire est plus simple à réaliser : les valeurs des gènes sont tirées au hasard selon une distribution uniforme. Toutefois, il peut être utile de guider la génération initiale vers des sous domaines intéressants de l'espace de recherche.

Par exemple lors d'une recherche d'optima dans un problème d'optimisation sous contraintes, il est préférable de produire des éléments satisfaisant les contraintes. La population initiale doit être suffisamment diversifiée et de taille assez importante pour que la recherche puisse parcourir l'espace d'état dans un temps limité.

3.2.3 - EVALUATION DE LA POPULATION

Cette étape consiste à évaluer chaque solution contenue dans la population. La mesure de performance des solutions s'appuie sur la valeur de la fonction objective. Cette étape permet de classer les solutions, afin de déterminer les solutions qui seront sélectionnées pour construire une nouvelle population de solutions.

En raison de son analogie avec le processus de l'évaluation naturelle, l'algorithme GA est naturellement formulé en terme de maximisation d'une fonction positive. Or dans beaucoup de problèmes, le but est de minimiser une fonction objective *f(x)*. Et même, dans un problème de maximisation rien ne garantit que la fonction objective reste positive pour tout individu *x*. Ainsi, il est souvent nécessaire de transformer la fonction objective en une fonction appelée fonction d'adaptation ou d'aptitude *g(x)* qui doit être maximisée.

Etant donnée une fonction *g* réelle à une ou plusieurs variables, le problème d'optimisation sur l'espace de recherche *E* s'écrit comme suit :

$$\max_{x \in E} \{g(x)\} \tag{3.1}$$

Pour éviter de manipuler avec les variables négatives de la fonction à optimiser, il suffit de transformer la fonction d'aptitude de la façon suivante [108] :

$$\max_{x \in E} \{g(x)\} + G_{\min} \tag{3.2}$$

où G_{min} est une constante permettant de réaliser cet objectif.

La fonction *g* peut être une transformation linéaire de *f* [128] :

$$g(x) = a \,.\, f(x) + b \tag{3.3}$$

avec *a* positif si on cherche le maximum de la fonction *f(x)* et *a* négatif si on cherche le minimum, *b* assurant que *g(x)* est toujours positif.

Une autre possibilité, lorsque l'on cherche à maximiser *f*, est d'utiliser une transformation par élévation à une puissance donnée:

$$g(x) = (a \,.\, f(x) + b)^k \tag{3.4}$$

En fonction du problème à résoudre, il peut être intéressant de faire une transformation logarithmique de la fonction d'adaptation de la forme suivante :

$$g(x) = b - \log(f(x)) \tag{3.5}$$

avec, $\forall x, b > \log(f(x))$.

Dans beaucoup de situations, l'objectif est exprimé sous la forme de minimisation d'une fonction objective ou d'un critère de performances *f(x)* :

$$\min_{x \in E} \{f(x)\} \tag{3.6}$$

Pour passer d'un problème de minimisation à un problème de maximisation bien adapté à la nature de l'AG, on peut utiliser la transformation $\max_{x \in E} \{g(x)\}$ avec [12] :

$$g(x) = F_{max} - f(x) \tag{3.7}$$

Avec $F_{max} = max(f(x))$ dans la génération courante.

Dans le cas où la fonction objective prendrait des valeurs négatives, elle peut être transformée de la façon suivante :

$$g(x) = f(x) - F_{min} \tag{3.8}$$

Où F_{min} est la valeur minimale de *f(x)* .

On peut aussi passer d'un problème de minimisation à un problème de maximisation en utilisant la transformation :

$$g(x) = \frac{1}{1 + f(x)} \tag{3.9}$$

3.2.4 - SELECTION

Cet opérateur, très important, permet aux individus d'une population de survivre, de se reproduire ou de mourir. En règle générale, la probabilité de survie d'un individu sera directement reliée à son efficacité relative au sein de la population. La sélection est fondée sur la qualité des individus, estimée à l'aide de fonction d'adaptation. Il existe plusieurs méthodes pour la sélection, on citera à titre d'exemple:

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

3.2.4.1 - Selection par roulette

La méthode de la sélection par la roue de fortune biaisée ou roulette de Goldberg (1989) [12] est la méthode de sélection la plus connue et utilisée. Selon cette méthode, chaque chromosome sera dupliqué dans une nouvelle population proportionnellement à sa valeur d'adaptation. L'application de cette technique de sélection vient après l'opération suivante:

On commence par le décodage des chromosomes pour obtenir les valeurs décimales qui seront remplacées dans la fonction d'évaluation (fonction coût) pour pouvoir calculer la valeur d'aptitude de l'individu. D'après ces valeurs, on peut réorganiser les individus selon chaque valeur de fitness calculée dans une roue appelée roue de fortune biaisée. Pour chaque individu, la force (valeur d'aptitude) relative est calculée par le rapport de la force de l'individu sur la somme des forces de l'ensemble de la population. On effectue ensuite N tirages aléatoires pour sélectionner la population des individus qui vont se produire. Si la force relative de l'individu est supérieure au nombre aléatoire tiré, il est sélectionné.

Soient F_i la valeur de la fonction d'aptitude associée au $i^{ème}$ individu i et n la taille de la population.

$$F_s = \sum_{i=1}^{n} F_i \tag{3.10}$$

La sélection peut être faite en utilisant le rapport $p_i = \frac{F_i}{F_s}$ caractérisant la probabilité pour que le $i^{ème}$ individu soit choisi pour participer à la prochaine population (recombinaison génétique). Sa

probabilité commulée sera donc : $C_i = \sum_{j=1}^{i} p_j$. Le $i^{ème}$ individu est sélectionné si la probabilité commulée C_i est plus grande qu'une certaine valeur comprise entre 0 et 1 générée aléatoirement. Une fois que l'opération de la sélection est terminée, la reproduction entre en jeu.

3.2.4.2 - Sélection par tournoi

Un nombre p d'individus est sélectionné parmi les N individus de la population. Le meilleur, répété jusqu'à obtenir N individu. Il est tout à fait possible que certains individus participent à plusieurs tournois. Ils ont donc droit d'être copiés plusieurs fois, ce qui favorise la pérennité de leurs gènes. En générale quatre individus sont sélectionnés pour chaque tournoi, $p = 4$ [101].

3.2.4.3 - Sélection par le rang

Il s'agit de classer la population suivant la fonction d'adaptation, chaque individu de la population se voit accorder un rang. Plus l'individu est bon, plus son rang est élevé. Le principe de la sélection par rang est similaire à la sélection par roulette, la différence est que la proportion est calculée sur les rangs et non sur la valeur de la fonction d'adaptation [108].

3.2.4.4 - Sélection uniforme

C'est une technique très simple qui consiste à sélectionner un individu aléatoirement de la population. La probabilité p_i pour qu'un individu soit sélectionné est définie par:

$$p_i = \frac{1}{taille\ pop} \tag{3.11}$$

3.2.4.5 - Décimation

Seuls, les p meilleurs individus de la population sont sélectionnés et autorisés à se reproduire par croisement et mutation jusqu'à obtenir une population de même taille N que la population initiale [101].

3.2.4.6 - Sélection par Elitisme

A la création d'une nouvelle population, il y a de grandes chances que les meilleurs chromosomes soient perdus après les opérations de croisement et de mutation. Pour éviter cela, on utilise la méthode d'élitisme. Elle consiste à copier un ou plusieurs des meilleurs chromosomes dans la nouvelle génération. Ensuite, on génère le reste de la population selon l'algorithme de

reproduction usuel. Cette méthode améliore considérablement les algorithmes génétiques, car elle permet de ne pas perdre les meilleures solutions [107].

3.2.4.7 - Proportionnelle à reste stochastique

Avec la sélection proportionnelle à reste stochastique, le nombre de copies *N (i)* d'un individu *i* est directement fixé par le rapport de sa performance avec la performance moyenne de la population :

$$N(i) = int\left(\frac{f_a(i)}{\overline{f_a}}\right) \quad (3.12)$$

Le nombre d'individus sélectionnés est donc $N_l = \sum_{i=1}^{N} N(i)$, mais, il est fort possible que $N_l < N$. La population est alors complétée par une sélection par roue de loterie où chaque individu a une probabilité de sélection égale à [101] :

$$p_s(i) = \frac{1}{N - N_l}\left(\frac{f_a(i)}{\overline{f_a}} - N(i)\right) \quad (3.13)$$

3.2.5 – REPRODUCTION

Cette opération permet la construction d'une nouvelle génération. Afin de donner naissance à un individu nouveau, il ne suffit pas de copier le génome de l'un des parents, mais au contraire, de prendre aléatoirement quelques gènes de chacun des parents. Ce phénomène est présent dans la nature (recombinaison). Il s'agit de phénomène essentiel qui permet d'explorer des solutions possibles:

- mutation
- croisement

3.2.6 - CROISEMENT

Après la sélection, un individu échange une partie des gènes de son chromosome avec un autre individu survivant pour donner naissance à deux nouveaux individus. La figure (3.3) représente ce processus.

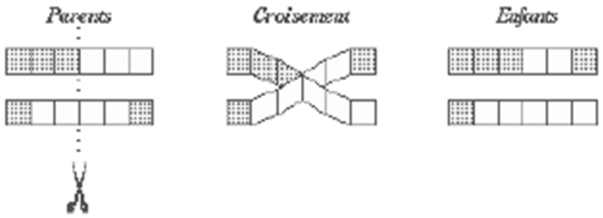


Fig. 3.3 - Croisement en codage binaire

Cet opérateur permet la création de deux nouveaux individus. Toutefois, un individu sélectionné lors de la reproduction ne subit pas nécessairement l'action d'un croisement. Ce dernier ne s'effectue qu'avec une certaine probabilité appelée probabilité de croisement P_c. Plus cette probabilité est élevée et plus la population subira de changement.

3.2.7 - MUTATION

La mutation est le changement aléatoire dans les informations du chromosome. Selon une probabilité p_m de mutation, cette opération permet de définir la variation dans le chromosome. Cette variation peut être locale (un seul site) ou globale (plusieurs sites). Cet opérateur permet d'avoir une large diversité de solutions dans la population comme le montre la figure (3.4).

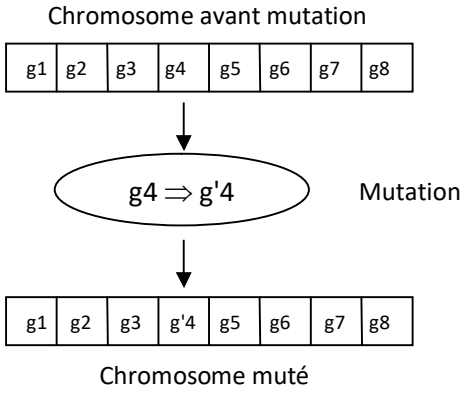


Fig. 3.4 – Principe de l'opérateur de mutation

3.2.8 – CODAGE REEL

La représentation binaire traditionnelle utilisée dans les algorithmes génétiques présente des insuffisances ou difficultés pour les problèmes d'optimisation de grandes dimensions à haute précision numérique.

En utilisant le codage réel, notre individu n'est alors plus qu'un chiffre à valeurs réelles dans l'espace des valeurs permises : $A = a$, avec $a \in D \subset R$. L'opérateur de sélection reste identique à celui de la roue de loterie biaisée. En revanche, il existe d'autres opérateurs de croisement et de mutation.

3.2.8.1 - Croisement

a) Croisement simple

L'opération de croisement simple tel que décrit dans le cas binaire ne peut pas s'effectuer dans le cas de recherche d'un point unique. Toutefois, pour une recherche de plus grande dimension,

nous pouvons utiliser de façon analogique cet opérateur. Ainsi, soient $Y = (y_1,y_2,y_3)$ et $X = (x_1,x_2,x_3)$ deux membres (vecteur de dimension trois) de la population initiale. Nous recherchons donc trois points dans un espace de recherche de dimension trois.

L'opération de croisement simple est identique dans le principe à celle décrite auparavant. Pour ce faire, nous générons un nombre aléatoire r à partir d'une distribution uniforme sur l'ensemble *{1,2,3}* et deux nouveaux individus, $\tilde{X}$ et $\tilde{Y}$, sont créés selon la règle suivante :

$$\tilde{x}_i = \begin{cases} x_i & si \quad i<r \\ y_i & si \quad non \end{cases} \tag{3.14}$$

$$\tilde{y}_i = \begin{cases} y_i & si \quad i<r \\ x_i & si \quad non \end{cases} \tag{3.15}$$

b) Croisement arithmétique

Un autre opérateur est le croisement arithmétique (valable même pour une recherche de dimension un). Ce croisement effectue une simple combinaison linéaire entre les parents. Soit, après avoir généré un chiffre aléatoire, $\alpha = U(0,1)$, les nouveaux parents sont [128] :

$$\tilde{X} = \alpha.X + (1-\alpha).Y \tag{3.16}$$

$$\tilde{Y} = (1-\alpha).X + \alpha.Y \tag{3.17}$$

c) Croisement heuristique

Enfin, il existe aussi le croisement heuristique. Cet opérateur effectue une extrapolation linéaire des deux individus. Un nouvel individu $\tilde{X}$ est créé selon le processus suivant (sous l'hypothèse que $X > Y$ en terme de fitness, sinon nous inversons X et Y dans les équations) [128] :

$$\tilde{X} = X + r.(X - Y) \tag{3.18}$$

$$\tilde{Y} = X$$

Avec la faisabilité F définie par :

$$F = \begin{cases} 1 & si \quad b_1^i < \tilde{x}_i < b_2^i, \quad \forall i \\ 0 & si \quad non \end{cases} \tag{3.19}$$

Où $b_1^i \; et \; b_2^i$ sont les bornes autorisées pour x_i et avec r un nombre aléatoire tirée dans $U(0,1)$. Nous devons donc avoir tout le temps $x_i \in [b_1^i, b_2^i]$. Si $\tilde{X}$ n'est pas faisable (i.e. faisabilité nulle) alors un nombre r est retiré et la procédure est recommencée jusqu'à ce que la solution soit faisable où qu'un

certain nombre d'essais ait été effectué. Dans le cas où $f(X) = f(Y)$ (même fitness), on reproduit simplement X et Y.

3.2.8.2 – Mutation

a) Mutation uniforme

La mutation uniforme est identique à celle du codage binaire. Ainsi, chaque variable $x_i \in X$ est changée (selon une certaine probabilité) en un nombre aléatoire tiré dans une distribution uniforme sur l'intervalle $\left[b_1^i, b_2^i\right]$ avec b_1^i et b_2^i les bornes inférieures et supérieures pour x_i.

b) Mutation non uniforme

La mutation non uniforme revient à changer la variable x_i en un nombre tiré dans une distribution non uniforme. Cette nouvelle variable $\widetilde{x}_i$ est telle que [128] :

$$\widetilde{x}_i = \begin{cases} x_i + \left(b_2^i - x_i\right) f(G) & si \quad \alpha < 0.5 \\ x_i - \left(x_i + b_1^i\right) f(G) & si \quad \alpha \geq 0.5 \end{cases} \tag{3.20}$$

Avec :

$$f(G) = \left(\widetilde{\alpha} \left(1 - \frac{G}{G_{\max}} \right) \right)^b \tag{3.21}$$

α , $\widetilde{\alpha}$ = Nombres aléatoires $\in (0,1)$.

G = La génération courante.

G_{max} = Le nombre maximal de génération.

b = Un paramètre déterminant le degré de non uniformité.

On trouve aussi une autre forme de la mutation non uniforme, où la nouvelle variable $\widetilde{x}_i$ peut prendre la valeur [128] :

$$\widetilde{x}_i = \begin{cases} x_i + \left(b_2^i - x_i\right) f(G) & si \quad la \ mutaion \ a \ lieu \\ x_i - \left(x_i + b_1^i\right) f(G) & si \quad non \end{cases} \tag{3.22}$$

Avec :

$$f(G) = \left(\left(1 - \tau^{\,1 - \frac{t}{T}} \right) \right)^b \tag{3.23}$$

Où :

τ = Nombre aléatoire $\in$ [0,1].

T = Nombre maximal de génération.

b = Entier positif.

c) Mutation dans les bornes

Avec cet opérateur, chaque variable $x_i \in X$ choisie pour muter prend pour valeur l'une des deux bornes b_1^i ou b_2^i avec équiprobabilité. A l'évidence, cet opérateur n'a d'intérêt et d'efficacité que si la solution est proche des bornes de l'espace de recherche. Notons qu'il est possible de combiner plusieurs opérateurs en même temps.

3.2.9 - AVANTAGES ET INCONVENIENTS

- Les algorithmes génétiques ont les caractéristiques suivantes :
  - Ils sont des outils efficaces pour une classe de problèmes très large.
  - Ils permettent de traiter des problèmes où la fonction à optimiser ne présente aucune propriété de continuité ou de dérivabilité, par exemple.
- Néanmoins, ils présentent, aussi, un certain nombre de limitations :
  - Ils sont moins efficaces qu'un algorithme déterministe spécifique (lorsqu'il en existe un).
  - Les nombreux paramètres à régler (probabilités de croisement et de mutation notamment, ainsi que le codage des chromosomes qui peut faire varier radicalement la vitesse de convergence).
  - La nécessité de calculer un très grand nombre de fitness avant l'obtention d'une bonne solution. Ce nombre de calcul important peut s'avérer problématique lorsque le coût de calcul (ressources systèmes ou temporelles) de la fitness est important, lorsqu'on travaille en grande dimension sur des fonctions à complexité importante par exemple.
  - Ils peuvent éprouver des difficultés à gérer des contraintes nombreuses et complexes car la technique des pénalités intégrées à la fonction de coût utilise des contraintes a posteriori de manière passive.

3.3 - EVOLUTION DIFFERENTIELLE

L'évolution différentielle (Differential Evolution DE) est une technique d'optimisation stochastique à base de population de solutions, introduite par Storn et Price en 1997 [112], dans le but d'optimiser les fonctions à paramètres réels. Cette technique de type métaheuristique est

caractérisée par sa simplicité et sa puissance en même temps. Elle appartient à la famille des algorithmes évolutionnaires. Elle utilise les mêmes opérateurs des algorithmes génétiques : croisement, mutation et sélection, mais elle se distingue par la manière qui permet de créer de nouveaux individus. La différence principale en construisant de meilleures solutions est que les algorithmes génétiques se fondent sur le croisement tandis que l'algorithme de DE se fonde sur l'opération de mutation, cette opération principale est basée sur la différence des paires de solutions aléatoirement prélevées dans la population. L'algorithme utilise l'opération de mutation comme un mécanisme de recherche et l'opérateur de sélection pour diriger la convergence vers les régions éventuelles dans l'espace de recherche [109].

3.3.1 - ALGORITHME A EVOLUTION DIFFERENTIELLE

Même si, à l'origine, la méthode de l'évolution différentielle était conçue pour les problèmes d'optimisation continus et sans contraintes, ses extensions actuelles peuvent permettre de traiter les problèmes à variables mixtes et gèrent les contraintes non linéaires. La figure (3.5) représente un algorithme d'évolution différentielle.

Dans la méthode DE, la population initiale est générée par tirage aléatoire uniforme sur l'ensemble des valeurs possibles de chaque variable. Les bornes inférieures et supérieures des variables sont spécifiées par l'utilisateur selon la nature du problème. Après l'initialisation, l'algorithme effectue une série de transformations sur les individus, dans un processus appelé évolution [129].

La population contient N individus. Chaque individu $x_{i,G}$ est un vecteur de dimension D, *où* G désigne la génération :

$$x_{i,G} = \left(x_{1i,G}, x_{2i,G}, \dots, x_{Di,G}\right) \qquad \text{avec} \quad i = 1, 2, \dots, N \tag{3.24}$$

Le standard DE utilise trois techniques (mutation, croisement et sélection) comme les algorithmes génétiques. A chaque génération, l'algorithme applique successivement ces trois opérations sur chaque vecteur pour produire un vecteur d'essai (*trial vector*) :

$$u_{i,G+1} = \left(u_{1i,G+1}, u_{2i,G+1}, \dots, u_{Di,G+1}\right) \qquad \text{avec} \quad i = 1, 2, \dots, N \tag{3.25}$$

Une opération de sélection permet de choisir les individus à conserver pour la nouvelle génération (G + 1).

Algorithme d'évolution différentielle (DE)

Entrées : N : nombre d'individus de la population, N_{iter} : nombre d'itérations, $F_{coût}$: fonction de coût, F : facteur de différenciation, CR : constante de recombinaison.

Sorties : x_{opt}, minimisant $F_{coût}$.

Initialisation : X aléatoire uniforme dans $(D)^N$, x_{opt}.

pour k de 0 à $(N_{iter} - 1)$ **faire**

1) Différentiation :

le $r^{ème}$ nouveau vecteur de paramètre potentiel, $X^r_{k,trial}$, est généré en additionnant $X^{r_1}_k$, tiré uniformément parmi les individus de la $k^{ème}$ génération, et la différence pondérée entre deux autres individus de la population, $X^{r_2}_k$ et $X^{r_3}_k$, avec $r_1 \neq r_2 \neq r_3$, ainsi :

$$\forall\, r = 1, \dots, N, \qquad X^r_{k,trial} = \min\left(\max\left(X^{r_1}_k + F.\left(X^{r_2}_k - X^{r_3}_k\right), X_{min}\right), \mathrm{X_{max}}\right)$$

2) Recombinaison :

Le $r^{ème}$ individu "mutant" de la $k^{ème}$ génération, $X^r_{k,mut}$, hérite des gènes de $X^r_{k,trial}$ avec une probabilité CR propre à chaque gène, ie on génère $u = \mathcal{U}(0, 1)$ et :

$$\forall\, i = 1, \dots, D \quad , \forall\, r = 1, \dots, N :$$

$$X^r_{k,mut}(i) = \begin{cases} X^r_{k,trial}(i) & si\ u < CR \\ X^r_k(i) & sinon \end{cases}$$

3) Sélection :

$$X^r_{k+1} = \begin{cases} X^r_{k,mut} & si \quad F_{coût}\left(X^r_{k,mut}\right) \leq F_{coût}(X^r_k) \\ X^r_k & sinon \end{cases}$$

$$x_{opt} = x \in \{X^1_k, X^2_k, \dots, X^N_k\}\ t.q. \forall\, X^i_k \in \{X^1_k, X^2_k, \dots, X^N_k\}, F_{coût}(x) \leq F\left(X^i_k\right)$$

retourner x_{opt}

Fig. 3.5 - Algorithme d'évolution différentielle

a - Mutation

Pour chaque vecteur courant $x_{i,G}$, on génère un vecteur mutant $v_{i,G+1}$ qui peut être créé en utilisant une des stratégies de mutation suivantes :

- **DE/Rand/1 :**

Cette notation indique que le vecteur à perturber est aléatoirement choisi et que la perturbation se compose sur une seule différence. L'individu muté est généré par :

$$v_{i,G+1} = x_{r_1,G} + F\left(x_{r_2,G} - x_{r_3,G}\right) \tag{3.26}$$

- **DE/Best/1 :**

Comme la stratégie précédente compte que l'individu de la prochaine génération est produit par le meilleur membre de la population :

$$v_{i,G+1} = x_{best,G} + F.\left(x_{r_1,G} - x_{r_2,G}\right) \quad (3.27)$$

- **DE/Current to best/1 :**

$$v_{i,G+1} = x_{i,G} + F.\left(x_{r_1,G} - x_{r_2,G}\right) + F.\left(x_{best,G} - x_{i,G}\right)$$

- **DE/Best/2 :**

Cette stratégie emploie deux vecteurs de différence comme perturbation :

$$v_{i,G+1} = x_{best,G} + F.\left(x_{r_1,G} - x_{r_2,G}\right) + F.\left(x_{r_3,G} - x_{r_4,G}\right) \quad (3.28)$$

- **DE/Rand/2 :**

$$v_{i,G+1} = x_{r_1,G} + F.\left(x_{r_2,G} - x_{r_3,G}\right) + F.\left(x_{r_4,G} - x_{r_5,G}\right) \quad (3.29)$$

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

b - Croisement

Après la mutation, une opération de croisement binaire forme le vecteur d'essai final $u_{i,G+1}$, selon le vecteur $x_{i,G}$ et le vecteur mutant correspondant $v_{i,G+1}$. L'opération de croisement est introduite pour augmenter la diversité des vecteurs de paramètres perturbés. Le nouveau vecteur $u_{i,G+1}$ est donné par la formule suivante :

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & si\ (randb(j) \le CR) \quad ou\ \ j = rnbr(i) \\ x_{ji,G} & si\ (randb(j) > CR) \quad et\ \ j \ne rnbr(i) \end{cases} \text{ pour tout } j \in \{1,2,\ldots,D\} \quad (3.30)$$

où *randb(j)* est la $j^{ème}$ valeur procurée par un générateur de nombre aléatoire uniforme appartenant à l'intervalle *[0,1]*. *CR* est le coefficient de croisement qui appartient à l'intervalle *[0,1]* et est déterminé par l'utilisateur. *rnbr(i)* est un indice choisi au hasard dans l'ensemble *{1,2,…,N}*.

c - Sélection

Pour décider quel vecteur, parmi $u_{i,G+1}$ ou $x_{i,G}$, doit être choisi dans la génération $G + 1$, on doit comparer les valeurs de fonction du coût de ces deux vecteurs. En effet, on garde le vecteur ayant la plus petite valeur de fonction du coût en cas de minimisation. Le nouveau vecteur $x_{i,G+1}$ est choisi selon l'expression suivante :

$$x_{i,G+1} = \begin{cases} u_{i,G+1} & si\ f(u_{i,G+1}) < f(x_{i,G}) \\ x_{i,G} & sinon \end{cases} \quad (3.31)$$

Il est clair qu'un bon réglage des principaux paramètres de l'algorithme (taille de la population *N*, facteur de mutation *F* et facteur de croisement *CR*) contribue de façon importante à l'efficacité de la méthode. L'auto-adaptation de ces paramètres paraît donc intéressante pour l'amélioration de l'algorithme.

3.3.2 – AVANTAGES ET INCONVENIENTS

- L'évolution différentielle est une technique d'optimisation globale puissante, simple et facile à utiliser. Elle est caractérisée par son aptitude à la parallélisation.
- Les inconvénients du DE peuvent être résumés dans le risque de la convergence prématurée : lorsque la population perdra complètement sa diversité, elle contiendra des éléments identiques, et elle reste inchangée par la perturbation créée par l'algorithme de l'évolution différentielle.

3.4 - OPTIMISATION PAR ESSAIMS PARTICULAIRES

3.4.1 - PRINCIPE GENERAL

L'optimisation par essaim particulaire est une métaheuristique d'optimisation née en 1995 aux États-Unis sous le nom de Particle Swarm Optimization (PSO). Elle a été développée par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue).

L'optimisation par essaim particulaire est inspirée du comportement social des animaux évoluant en essaim. L'exemple le plus souvent utilisé est le comportement des bancs de poissons et les nuées d'oiseaux. En effet, on peut observer chez ces animaux des dynamiques de déplacement relativement complexes, alors qu'individuellement chaque individu a une intelligence limitée et une connaissance seulement locale de sa situation dans l'essaim. Un individu de l'essaim n'a pour connaissance que la position et la vitesse de ses plus proches voisins. Chaque individu utilise donc, non seulement, sa propre mémoire, mais aussi l'information locale sur ses plus proches voisins pour décider de son propre déplacement. Des règles simples, telles que « aller à la même vitesse que les autres », « se déplacer dans la même direction » ou encore « rester proche de ses voisins » sont des exemples de comportements qui suffisent à maintenir la cohésion de l'essaim, et qui permettent la mise en œuvre de comportements collectifs complexes et adaptatifs. L'« intelligence globale » de l'essaim est donc la conséquence directe des interactions locales entre les différentes particules de

l'essaim. La performance du système entier est supérieure à la somme des performances de ses parties.

Kennedy et Eberhart se sont inspirés de ces comportements socio-psychologiques pour créer le PSO. Un essaim de particules, qui sont des solutions potentielles au problème d'optimisation, « survole » l'espace de recherche, en quête de l'optimum global. Le déplacement d'une particule est influencé par les trois composantes suivantes :

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

Dans le cas d'un problème d'optimisation, la qualité d'un site de l'espace de recherche est déterminée par la valeur de la fonction objective en ce point. La Figure (3.6) illustre la stratégie de déplacement d'une particule.

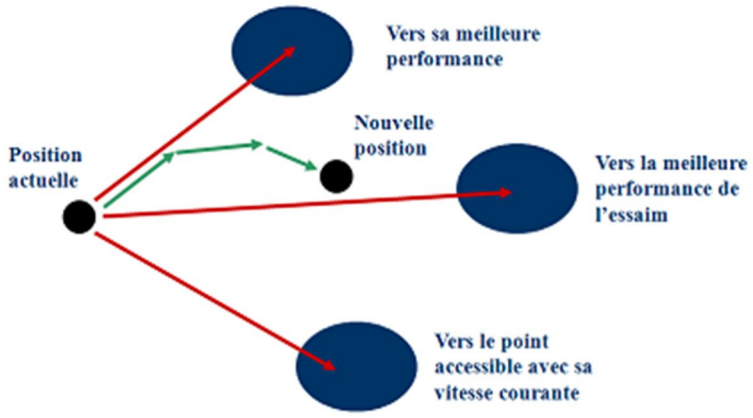


Fig. 3.6 - Déplacement d'une particule

3.4.2 - FORMALISATION

Dans un espace de recherche de dimension *D*, la particule *i* de l'essaim est modélisée par son vecteur position $\overrightarrow{X_i} = (x_{i1}, x_{i2}, \dots, x_{iD})$ et par son vecteur vitesse $\overrightarrow{V_i} = (v_{i1}, v_{i2}, \dots, v_{iD})$. Cette particule garde en mémoire la meilleure position par laquelle elle est déjà passée, que l'on note $\overrightarrow{p_i} = (p_{i1}, p_{i2}, \dots, p_{iD})$. La meilleure position atteinte par toutes les particules de l'essaim est notée $\vec{g} = (g_1, g_2, \dots, g_D)$. Au temps *t*, le vecteur vitesse est calculé à partir de la formule (3.32).

$$v_{ij}(t) = w.v_{ij}(t-1) + c_1.r_1.\left(p_{ij}(t-1) - x_{ij}(t-1)\right) + c_2.r_2.\left(g_j(t-1) - x_{ij}(t-1)\right), \quad j \in \{1, \dots, D\} \tag{3.32}$$

Où :

- *w* est appelée coefficient d'inertie,
- c_1 et c_2 sont deux constantes, appelées coefficients d'accélération,
- r_1 et r_2 sont deux nombres aléatoires tirés uniformément dans [0,1] à chaque itération et pour chaque dimension.
- $w.v_{ij}(t-1)$ correspond à la composante physique du déplacement.
- $c_1.r_1.\left(p_{ij}(t-1) - x_{ij}(t-1)\right)$ correspond à la composante cognitive du déplacement. c_1 contrôle le comportement cognitif de la particule.
- $c_2.r_2.\left(g_j(t-1) - x_{ij}(t-1)\right)$ correspond à la composante sociale du déplacement. c_2 contrôle l'aptitude sociale de la particule.

La combinaison des paramètres *w*, c_1 et c_2 permet de régler la balance entre les phases de diversification et intensification du processus de recherche [29], [47].

Le paramètre *w* contrôle l'influence de la direction de déplacement sur le déplacement futur. Il peut être une constante positive ou même une fonction linéaire ou non linéaire positive du temps [36], [47], [130]. Ce paramètre peut fournir un équilibre entre les explorations globales et locales. Le coefficient d'inertie *w* peut varier linéairement avec le temps selon la formule suivante [36] :

$$w = w_{max} - \left(\frac{w_{max} - w_{min}}{max_{iter}}\right).iter \tag{3.33}$$

Où :

- *iter* est l'itération courante.
- max_{iter} est le nombre maximal d'itérations.
- w_{max} *et* w_{min} désignent respectivement les valeurs maximum et minimum du coefficient *w*.

Cette diminution linéaire du poids d'inertie au cours des itérations, permet à l'algorithme du PSO d'avoir une grande capacité de recherche globale au début des itérations et une grande capacité de recherche locale à la fin [130].

Il est à noter que le terme « vitesse » est ici abusif car les vecteurs $\vec{V_i}$ ne sont pas homogènes à une vitesse. Il serait plus approprié de parler de « direction de déplacement ». Cependant, pour respecter l'analogie avec le monde animalier, les auteurs ont préféré utiliser le terme « vitesse » [129]. La position au temps *t* de la particule *i* est alors définie par (3.34) :

$$x_{ij}(t) = x_{ij}(t-1) + v_{ij}(t),\ j \in \{1, \dots, D\} \quad (3.34)$$

3.4.3 - ALGORITHME DU PSO

Le PSO est un algorithme à population. Il commence par une initialisation aléatoire de l'essaim dans l'espace de recherche. A chaque itération de l'algorithme, chaque particule est déplacée suivant (3.32) et (3.34). Une fois le déplacement des particules effectué, les nouvelles positions sont évaluées. Les $\overrightarrow{p_i}$ ainsi que $\vec{g}$ sont alors mis à jour. Cette procédure est résumée par l'algorithme de la figure (3.7). *N* est le nombre de particules de l'essaim.

Algorithme d'optimisation par essaim particulaire

Initialisation aléatoire des positions et des vitesses de chaque particule
Pour chaque particule *i*, $\overrightarrow{p_i} = \overrightarrow{X_i}$
Tant que le critère d'arrêt n'est pas atteint faire
  Pour $i = 1$ à N **faire**
    Déplacement de la particule à l'aide de (3.32) et (3.34)
    Évaluation des positions
    Si $f(\overrightarrow{X_i}) < f(\overrightarrow{p_i})$
      $\overrightarrow{p_i} = \overrightarrow{X_i}$
    Fin Si
    Si $f(\overrightarrow{p_i}) < f(\vec{g})$
      $\vec{g} = \overrightarrow{p_i}$
  Fin Pour
Fin Tant que

Fig. 3.7 - Algorithme d'optimisation par essaim particulaire

Le critère d'arrêt peut être différent suivant le problème posé. Si l'optimum global est connu à priori, on peut définir une « erreur acceptable » ε comme critère d'arrêt. Sinon, il est commun de fixer un nombre maximum d'évaluations de la fonction objectif ou un nombre maximum d'itérations comme critère d'arrêt. Cependant, au regard du problème posé et des exigences de l'utilisateur, d'autres critères d'arrêt peuvent être utilisés.

L'algorithme de cette méthode peut être décrit comme suit :

1ère étape : Initialisation des coefficients *($\overrightarrow{c_1}$, $\overrightarrow{c_2}$, $\vec{w}$, ...)*.

2ème étape : La création de la population initiale aléatoirement et le calcul de la fitness de chaque particule, la meilleure position de la particule *i* dans la population actuelle, la meilleure position dans toute les populations (la meilleure de meilleures).

3ème étape : Le calcul de la nouvelle vitesse et nouvelle position de chaque particule par l'utilisation des formules (3.32) et (3.33).

4ème étape : Le calcul de la meilleure fitness de la population initiale et comparer par la précédente pour trouver la meilleure de toute les populations.

5ème étape : Incrémentation du nombre d'itération $t = t+1$.

6ème étape : Si un critère d'arrêt est satisfait alors passer à la 7ème étape. Autrement aller à la 3ème étape.

7ème étape : Enregistrement de la solution optimale.

La figure (3.8) présente l'organigramme de la méthode PSO.

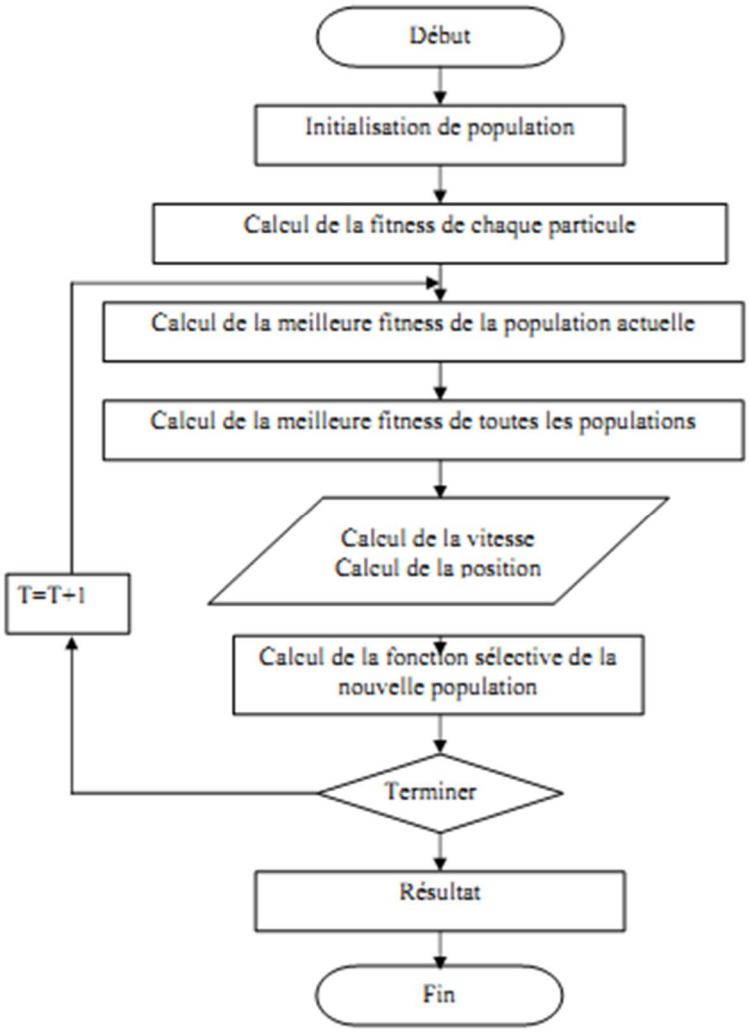


Fig. 3.8 - Organigramme de la méthode PSO

3.4.4 - CONFINEMENT DES PARTICULES

Il est possible que le déplacement d'une particule la conduise à sortir de l'espace de recherche. Pour s'affranchir de ce problème, on peut introduire un nouveau paramètre V_{max}, qui va permettre de contrôler l'explosion du système [130] et limiter la distance maximale qu'une particule va parcourir au cours d'une itération.

De plus, une stratégie de confinement des particules peut être introduite. Une telle stratégie permet de ramener une particule sortie de l'espace de recherche à l'intérieur de celui-ci.

3.4.5 - FACTEUR DE CONSTRICTION

Des améliorations ont été apportées à l'algorithme de base, notamment du point de vue du contrôle de la divergence. L'étude de la dynamique des particules au sein de l'essaim a conduit à la recherche de solutions pour éviter la divergence de l'algorithme. Dans les paragraphes précédents, on a vu en quoi l'introduction du paramètre V_{max} peut limiter la divergence des particules.

Clerc et Kennedy ont démontré qu'une bonne convergence peut être assurée en rendant dépendants les paramètres w, c_1 et c_2 [131]. L'utilisation d'un facteur de constriction $\emptyset$ permet de prévenir l'explosion de l'essaim, d'assurer la convergence, mais aussi de s'affranchir de la définition arbitraire d'un paramètre V_{max}. L'équation (3.32) devient alors [131-132] :

$$v_{i,j}(t) = \chi.\Big(v_{i,j}(t-1) + \emptyset_1.r_1.\Big(p_{i,j} - x_{i,j}(t-1)\Big) + \emptyset_2.r_2.\Big(g_j - x_{i,j}(t-1)\Big)\Big), j \in \{1, \dots, D\} \quad (3.35)$$

Avec

$$\chi = \frac{2}{\emptyset - 2 + \sqrt{\emptyset^2 - 4\emptyset}}, \emptyset = \emptyset_1 + \emptyset_2, \emptyset > 4 \quad (3.36)$$

Dans [131], de nombreux tests sont menés pour déterminer les valeurs optimales de $\emptyset_1$ et $\emptyset_2$. Dans la majorité des cas, on utilise $\emptyset = 4.1$ et $\emptyset_1 = \emptyset_2$, ce qui donne un coefficient multiplicatif approximativement égal à 0.7298.

3.4.6 – AVANTAGES ET INCONVENIONS

- Les principales caractéristiques et avantages du PSO sont :
  - Facilité à mettre en application et peu de paramètres à ajuster.
  - Méthode basée sur un concept simple qui exige peu de mémoire et un temps de calcul court.
  - Elle a été développée à l'origine pour des problèmes d'optimisation non-linéaire avec des

variables continues, mais il est facile à l'utiliser pour traiter des problèmes avec variables discrètes. Par conséquent, elle est applicable aux problèmes d'optimisation non-linéaire avec un mélange de nombres entiers des variables continues et discrètes.

- À la différence des algorithmes génétiques GA, le PSO n'a aucun opérateur d'évolution tel que le croisement et la mutation.
- Pour les techniques des GA, les chromosomes partage l'information de sorte que la population entière se déplace comme un groupe, mais dans le PSO, seulement la meilleure particule globale *(g_{best})* donne l'information aux autres.

- Les inconvénients du PSO peuvent être résumées dans :
  - La convergence prématurée.
  - La rapidité du flux d'information entre les particules, entraîne la création de particules similaires ce qui augmente la possibilité d'être piégé dans un optimum local.
  - Le réglage des paramètres qui influe totalement sur la performance de l'algorithme. L'augmentation du paramètre d'inertie w va augmenter la vitesse des particules en favorisant en plus l'exploration (recherche globale) et moins l'exploitation (Recherche locale). D'autre part, la réduction de w diminue la vitesse des particules et favorise en plus l'exploitation et moins l'exploration. Ainsi trouver la meilleure valeur pour le paramètre n'est pas une tâche facile et elle peut différer d'un problème à l'autre.

3.5 - RECUIT SIMULÉ (SIMULATED ANNEALING)

3.5.1 - PRINCIPE

Le recuit simulé (SA) est une approche d'optimisation stochastique proposée par Kirkpatrick en 1983 [105]. Elle est inspirée du processus naturel de recuit lié à la thermodynamique.

Le procédé de recuit est utilisé en métallurgie pour améliorer la qualité d'un solide. Un métal est chauffé à très haute température T_{max} et lentement refroidi à une température basse T_{min} qui peut cristalliser. La procédure de chauffage permet aux atomes de se déplacer de manière arbitraire. Si le refroidissement est assez lent, les atomes ont suffisamment de temps pour se réguler afin d'atteindre un état d'énergie minimale. Cette similitude peut être appliquée dans des problèmes d'optimisation, où l'état du métal correspond à une solution possible et l'état d'énergie minimale représente la meilleure solution finale.

Le principe du recuit simulé est de parcourir de manière itérative l'espace des solutions. On part avec une solution notée S_0 initialement générée de manière aléatoire dont correspond une énergie initiale E_0, et une température initiale T_{max} généralement élevée. A chaque itération de

l'algorithme, un changement élémentaire est effectué sur la solution, cette modification fait varier l'énergie du système ΔE. Si cette variation est négative (la nouvelle solution s' améliore la fonction objective $f(s')$, et permet de diminuer l'énergie du système $\Delta E = f(s') - f(s) < 0$), elle est acceptée. Si la solution trouvée s' est moins bonne que la précédente s $(\Delta E = f(s') - f(s) > 0)$, alors elle sera acceptée avec une probabilité p calculée suivant la distribution de Boltzmann suivante :

$$p(E,T) = exp^{-\frac{\Delta E}{T}} \tag{3.37}$$

En fonction du critère de Metropolis, un nombre $\epsilon \in [0,1]$ est comparé à la probabilité $p = exp^{-\frac{\Delta E}{T}}$. Si $p \geq \epsilon$ la nouvelle solution est acceptée. Le fonctionnement du critère de Metropolis est interprété par :

- Si $\Delta E = f(s') - f(s) \leq 0$ alors $e^{-\frac{\Delta E}{T}} > 1$, donc ϵ est tojours inférieur à cette valeur, et on accepte la solution s'
- Si $\Delta E > 0$ et T est très grande, alors $e^{-\frac{\Delta E}{T}} \cong 1$, tout voisin est systématiquement accepté.
- Si $\Delta E > 0$ et T est très petite, alors $e^{-\frac{\Delta E}{T}} \cong 0$, une dégradation a peu de chances d'être acceptée.

Deux approches sont possibles quant à la variation de la température :

1 - Pour la première, on itère en gardant la température constante. Lorsque le système a atteint un équilibre thermodynamique (au bout d'un certain nombre de changements), on diminue la température du système. On parle alors de paliers de température.

2 - La seconde approche fait baisser la température de façon continue. On peut alors imaginer toute sorte de loi de décroissance. La plus courante étant $T_{i+1} = \alpha.T_i$ avec $\alpha \in [0,1]$ (assez couramment $\alpha = 0.99$) un paramètre qui exprime la diminution de la température de l'itération i à $i + 1$.

Dans les deux cas, si la température a atteint un seuil assez bas, fixé au préalable, ou que le système devient figé, l'algorithme s'arrête [107].

La figure (3.9) présente l'organigramme de la méthode du recuit simulé. Dans la figure (3.10) un algorithme d'un recuit simulé est proposé où :

- *Trymax* est le maximum d'essais effectués à chaque valeur de température.
- Le critère d'arrêt est satisfait lorsque la température atteint T_{min}.
- la décroissance de la température à l'itération (t) peut être faite en utilisant : $T(t) = \alpha.T(t-1)$, Où α est un coefficient proche de 1.

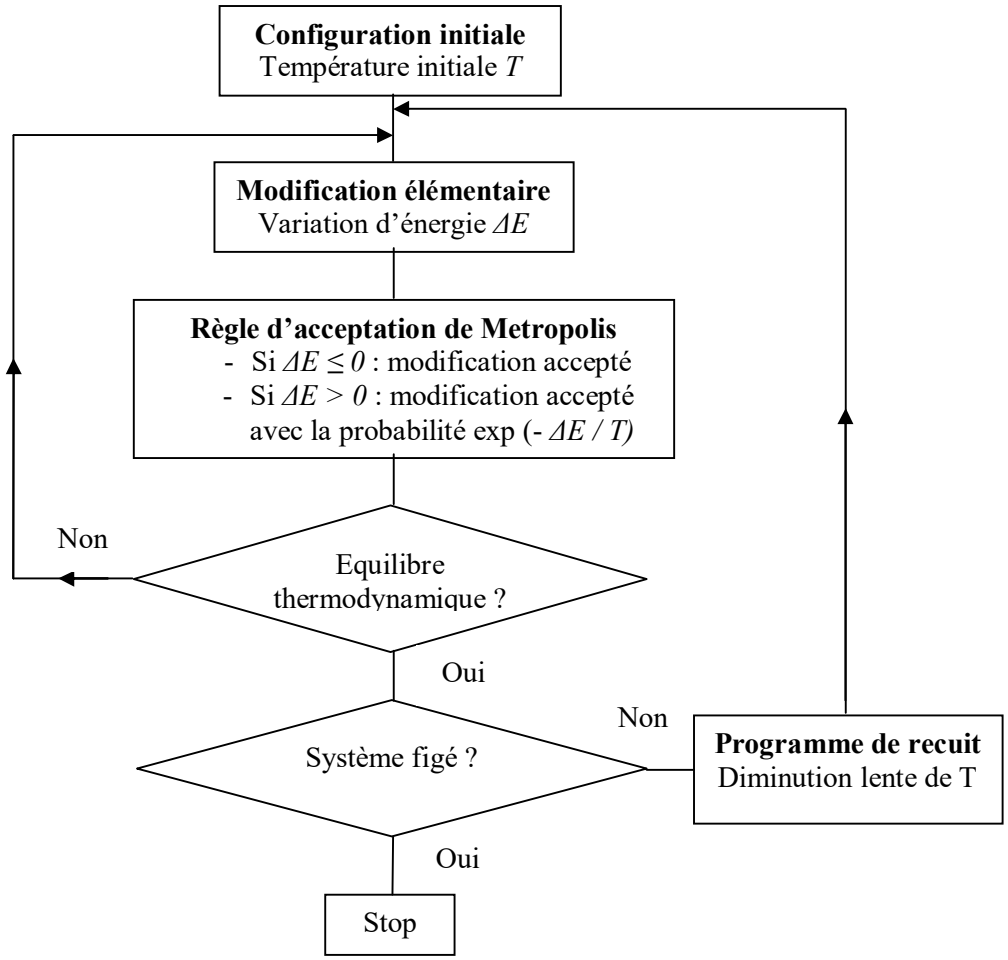


Fig. 3.9 - Fonctionnement de l'algorithme de recuit simulé

3.5.2 – AVANTAGES ET INCONVENIENTS

- Un algorithme du recuit simulé a l'avantage d'être souple, facile et rapidement implémentable lorsqu'on veut résoudre des problèmes d'optimisation complexes, le plus souvent de grande taille.
- Les principaux inconvénients de recuit simulé résident dans :
  - le choix des nombreux paramètres, tels que la température initiale, la loi de décroissance de la température, les critères d'arrêt ou la longueur des paliers de température. Ces paramètres sont souvent choisis de manière empirique.
  - la propriété d'exploration aléatoire locale, au voisinage d'un point donné. Il est utile pour une recherche locale rapide mais moins adapté pour une recherche globale.

Algorithme d'un recuit simulé

Initialisation

Choisir T_{max}, T_{min}, α, *Trymax* et initialiser la température à $T = T_{max}$.

Générer une solution initiale aléatoire *s* et calculer sa fonction objective $f(s)$.

Tand que $T \geq T_{min}$

Tand que le nombre des essais ≤ *Trymax*

Générer aléatoirement une solution adjacente s' et calculer sa fonction objective $f(s')$.

si $f(s') \leq f(s)$

accepter la nouvelle solution et replacer l'ancienne solution par la nouvelle.

Sinon si $f(s') > f(s)$

Mettre $\Delta E = f(s') - f(s)$, générer un nombre aléatoire $r \in [0,1]$,

Accepter la nouvelle solution si $exp^{-\frac{\Delta E}{T}} \geq r$ et remplacer l'ancienne solution par la nouvelle.

Fin de si

Fin de tand que

Faire baisser la température $T(t) = \alpha . T(t-1)$

Fin de tand que

Fig. 3.10 – Algorithme d'un recuit simulé

3.6 - HYBRIDATION DE MÉTHODES

3.6.1 - DEFINITION

Une méthode hybride est une méthode de recherche constituée d'au moins de deux méthodes de recherche distinctes. Elle consiste à exploiter les avantages respectifs de plusieurs méthodes en combinant leurs algorithmes suivant une approche synergétique.

Une méthode hybride peut être mauvaise ou bonne selon le choix et les rôles de ses composants. Pour définir une méthode hybride efficace, il faut savoir caractériser les avantages et les limites de chaque méthode. Par exemple, les algorithmes génétiques sont très performants lorsqu'il s'agit d'explorer l'espace de recherche, mais ils s'avèrent ensuite incapable d'exploiter efficacement la zone vers laquelle la population des solutions converge. Il est alors plus intéressant d'utiliser dans ce stade une autre méthode permettant une bonne exploitation comme par exemple le

recuit simulé ou une autre heuristique d'amélioration. Il faut souligner qu'il faut être prudent sur le choix des méthodes à hybrider ainsi sur le problème de multiplication des paramètres [133].

Actuellement, les approches hybrides gagnent en popularité car ce type d'algorithme produit généralement les meilleurs résultats pour plusieurs problèmes d'optimisation.

3.6.2 - MOTIVATION DE L'HYBRIDATION

Malgré le succès des métaheuristiques pour résoudre les problèmes complexes, plusieurs problèmes ont été rencontrés, à savoir :

- la présence de bruit lors de l'évaluation des solutions.
- la distinction entre une solution optimale locale et une autre globale est parfois difficile à percevoir.
- le problème de la convergence prématurée.
- le problème d'ajustement des paramètres.

L'hybridation de méthodes bio-inspirées a connu une grande popularité et a permis de bénéficier des points forts de chacune de ces méthodes et de surmonter leurs limites. L'hybridation a permis d'avoir un compromis entre l'exploration et l'exploitation de l'espace de recherche des solutions. Pour avoir une bonne exploitation, un algorithme est utilisé pour localiser les meilleures régions de l'espace de recherche, un autre est utilisé pour converger vers l'optimum global. L'hybridation est utilisée aussi pour optimiser les paramètres généraux [133]. On peut conclure que l'hybridation de métaheuristiques est la voie la plus prometteuse pour l'amélioration de la qualité des solutions dans beaucoup d'applications réelles. Ainsi, le choix d'une approche hybride devient aujourd'hui déterminant pour obtenir de meilleures performances lors de la résolution des problèmes complexes.

3.6.3 - CLASSIFICATION DES STRATEGIES D'HYBRIDATION

L'hybridation peut prendre place à deux niveaux : soit plusieurs méthodes de recherche coopèrent au sein d'une méthode hybride, soit une heuristique (ou une méthode de recherche) est insérée à l'intérieur d'une autre méthode de recherche. Dans ce dernier cas, il est nécessaire de définir à quel niveau, c'est-à-dire dans quel composant de la méthode de recherche va prendre place l'hybridation. Plusieurs classifications ont été proposées dans la littérature, les plus importantes sont décrites ci-dessous.

3.6.3.1 - Classification selon l'ordre d'exécution

Les méthodes hybrides peuvent être classifiées en trois catégories, selon leurs architectures [102], [133], [134] :

- hybridation séquentielle
- hybridation parallèle synchrone
- hybridation parallèle asynchrone

Hybridation séquentielle **:** C'est le type d'hybridation le plus populaire. Elle consiste à exécuter séquentiellement différentes méthodes de recherche de telle façon que le (ou les) résultat(s) d'une méthode serve(nt) de solution(s) initiale(s) à la suivante. Par exemple utiliser l'algorithme de recuit simulé pour améliorer les solutions obtenues par un algorithme génétique. C'est la technique d'hybridation la plus simple (figure 3.11), elle ne nécessite pas de modification des méthodes de résolution utilisées : il suffit d'initialiser chaque méthode à partir de solution pré-calculées.

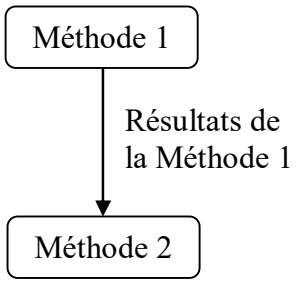


Fig. 3.11 – Hybridation séquentielle

Hybridation parallèle synchrone **:** Cette hybridation est obtenue en incorporant une méthode de résolution dans une autre. C'est une technique plus fine et plus complexe à mettre en œuvre que la précédente.

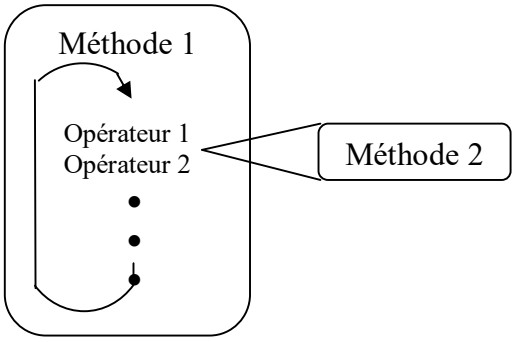


Fig. 3.12 - Hybridation parallèle synchrone

Hybridation parallèle asynchrone (coopérative) **:** Elle consiste à faire évoluer en parallèle différentes méthodes de résolution. Cette co-évolution permet une bonne coopération des méthodes

de résolution au travers d'un coordinateur chargé d'assurer le transfert d'informations entre les méthodes de résolution. Cette technique exige la modification de méthodes de résolution pour assurer la communication avec le coordinateur [107].

Les méthodes hybrides appartenant à cette classe sont caractérisées par une architecture telle que deux algorithmes A et B sont impliqués simultanément et chacun ajuste l'autre. Les Algorithmes A et B partagent et échangent de l'information tout au long du processus de recherche (Fig. 3.13) [133].

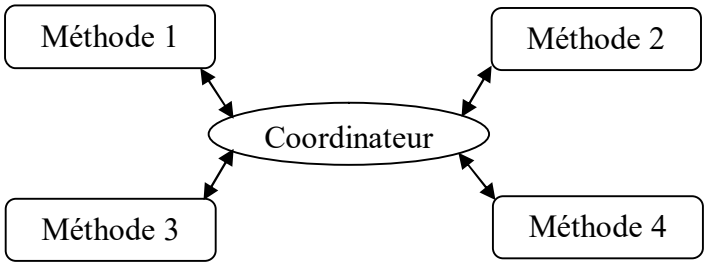


Fig. 3.13 - Hybridation parallèle asynchrone (coopérative)

3.6.3.2 - Classification hiérarchique

Cette classification est basée sur le niveau d'hybridation. On distingue deux niveaux : hybridation de **haut niveau** et hybridation de **bas niveau** (Fig.3.14).

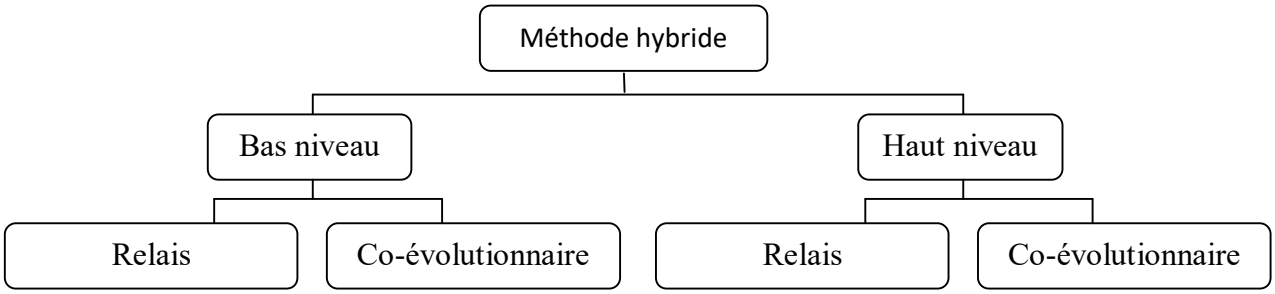


Fig. 3.14 - Classification hiérarchique

On parle d'une hybridation de bas niveau lorsqu'une fonction d'une métaheuristique est remplacée par une autre métaheuristique. Par contre, une hybridation de haut niveau est obtenue lorsque deux métaheuristiques sont hybridées sans que leur fonctionnement interne ne soit modifié.

Chacune des deux classes d'hybridation précédentes se subdivise en deux sous classes : **à relais et co-évolutionnaire**.

Lorsque les métaheuristiques sont exécutées de façon **séquentiell**e, l'une utilisant le résultat de la précédente comme entrée, on a une hybridation **à relais**. L'hybridation **co-évolutionnaire** se

fait lorsque des agents **coopèrent en parallèle** pour explorer l'espace de solutions. La combinaison des classes citées précédemment permet d'avoir quatre classes différentes d'hybridation [133]:

1. hybridation de bas niveau à relais **:** représente les algorithmes dans lesquels une métaheuristique est incorporée dans une autre métaheuristique à solution unique, par exemple, une recherche locale est incorporée dans un algorithme de recuit simulé pour résoudre le problème du voyageur de commerce.

2. hybridation de bas niveau co-évolutionnaire **:** consiste à incorporer un algorithme de recherche locale basé sur l'exploitation avec une métaheuristique à population de solution axée sur l'exploration. Par exemple, le remplacement de l'opérateur de mutation d'un algorithme génétique par une recherche locale pour résoudre le problème de partition des graphes (Fig. 3.15).

3. hybridation de haut niveau à relais **:** dans ce cas, des métaheuristiques complètes sont exécutées séquentiellement. Par exemple prendre les meilleures solutions trouvées par un algorithme d'optimisation par colonie de fourmis comme solution initiale d'une recherche tabou (Fig. 3.16).

4. hybridation de haut niveau co-évolutionnaire **:** implique un ensemble de métaheuristiques complètes qui travaillent en parallèle et coopèrent pour trouver la solution optimale du problème. Par exemple coopérer entre une recherche locale et une recherche tabou, de sorte que, à intervalles réguliers, les deux méthodes échangent des informations (Fig. 3.17).

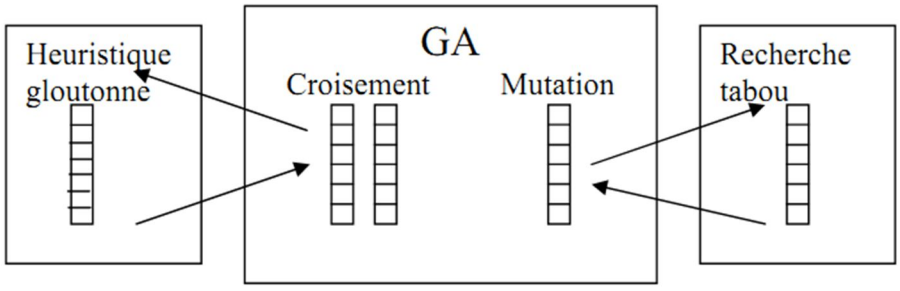


Fig. 3.15 - Exemple d'hybridation de bas niveau co-évolutionnaire

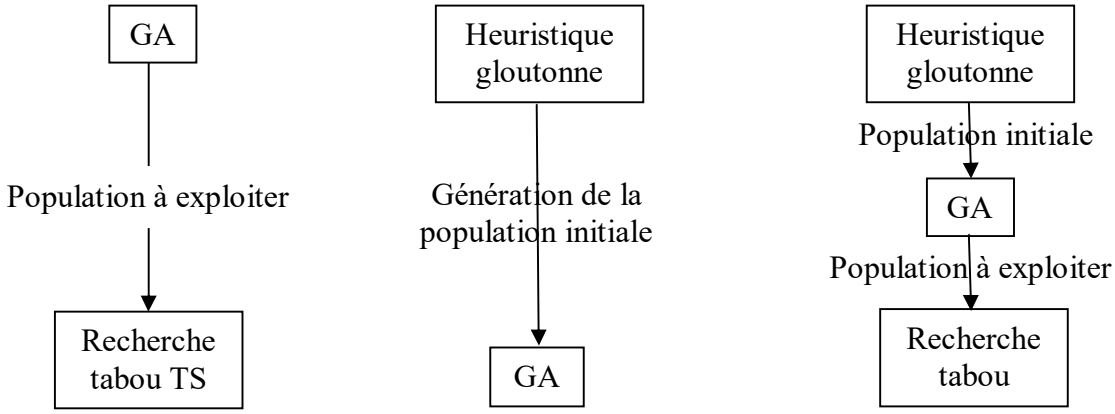


Fig. 3.16 - Exemple d'hybridation de haut niveau à relais

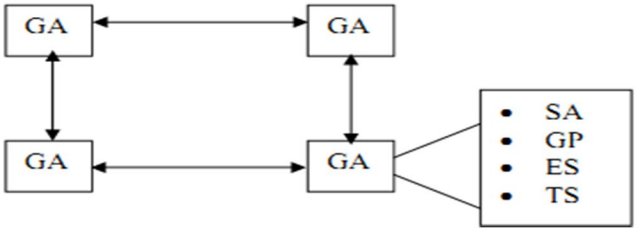


Fig. 3.17 - Exemple d'hybridation de haut niveau co-évolutionnaire

3.6.3.3 – Classification selon la stratégie de contrôle

Le critère utilisé pour différentier entre les différentes approches hybrides est celui de la stratégie de contrôle : collaborative ou intégrative :

*1 - **Stratégie intégrative*** **:** c'est la plus populaire, les composants d'un algorithme dit subordonné sont incorporés dans un autre algorithme. L'exemple le plus proche est celui des algorithmes mémétiques où plusieurs techniques de recherche locale sont introduites dans un algorithme évolutionnaire.

*2 - **Stratégie coopérative*** **:** les algorithmes échangent de l'information, mais aucun ne fait partie de l'autre.

3.6.3.4 - Classification générale

Une approche hybride peut être **homogène ou hétérogène**, **globale ou partielle et spécialisée ou générale.**

- Une hybridation est dite hétérogène lorsque les métaheuristiques combinées sont différentes.
- Une hybridation globale fait en sorte que toutes les métaheuristiques explorent l'ensemble de l'espace des solutions. Par contre une hybridation est dite partielle lorsqu'elle décompose un problème en sous-problèmes ayant leur propre espace de solutions, et que chaque sous-problème est résolu par un algorithme.
- Les hybridations générales sont celles où tous les algorithmes hybridés résolvent le même problème d'optimisation. Les hybridations spécialisées sont celles où chaque algorithme résout un problème d'optimisation différent (Fig. 3.18).

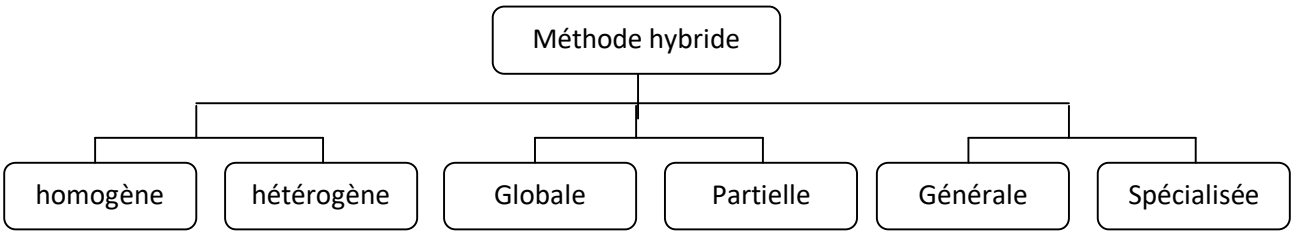


Fig. 3.18 - Classification générale de l'hybridation

3.7 - CONCLUSION

Plusieurs critères sont à considérer pour comparer entre les différentes métaheuristiques, pour la résolution des problèmes d'optimisation dont on peut citer :

- la performance et la robustesse, analysées en termes de qualité des solutions, de durée d'exécution et d'occupation mémoire, (générant des solutions proches de l'optimum en temps acceptable avec un minimum de ressources) ;
- la complexité des algorithmes ;
- la capacité de l'exploitation et l'exploration de l'espace de recherche ;
- les propriétés adaptatives du système de résolution dans un contexte dynamique et stochastique;
- la simplicité (facile à comprendre et à mettre en œuvre) ;
- la flexibilité, (aisément extensibles à des problèmes plus complexes) ;
- la potentialité de généralisation à d'autres domaines d'application.

Une méthode de recherche est rarement aussi efficace pour exploiter que pour explorer l'espace de recherche. La solution est d'associer à une méthode dont la capacité d'exploration est très élevée à une méthode caractérisée par une bonne exploitation de l'espace de recherche, d'où l'émergence actuelle de méthodes hybrides, qui s'efforcent de tirer parti des avantages spécifiques d'approches différentes en les combinant à différents niveaux.

Une hybridation idéale engendre une méthode hybride qui cumule les bonnes propriétés de ses constituants tout en inhibant leurs points faibles.

Les algorithmes évolutionnaires par exemple (comme les algorithmes génétiques et l'évolution différentielle) sont devenus très utilisés pour résoudre les problèmes complexes. Ses avantages sont multiples, à savoir la simplicité de l'approche, sa robustesse face aux changements de l'environnement et enfin sa flexibilité. Les algorithmes évolutionnaires sont faciles à implémenter par rapport à d'autres métaheuristiques. Par contre, pour plusieurs problèmes complexes, les algorithmes évolutionnaires ne réussissent pas à trouver la solution optimale, d'où la nécessité de les hybrider avec d'autres méthodes. Les raisons de cette hybridation sont :

- améliorer les performances d'un algorithme évolutionnaire (vitesse de convergence) ;
- améliorer la qualité de la solution obtenue ;
- élargir le domaine d'application des algorithmes évolutionnaire.

Il existe plusieurs manières pour hybrider les algorithmes évolutionnaires, qu'on peut les résumer comme suit :

- la population initiale peut être créée avec une heuristique spécifique au problème ;

- quelques solutions obtenues par un algorithme évolutionnaire peuvent être améliorés par une recherche locale (recuit simulé par exemple) ;
- des opérateurs de croisement et de mutation peuvent exploiter les connaissances préalables du problème ;

Dans le chapitre suivant, nous allons présenter les algorithmes génétiques, l'évolution différentielle, l'optimisation par essaims particulaires et le recuit simulé pour résoudre le problème de l'optimisation du plan de tension et de la puissance réactive d'un réseau électrique. Nous allons présenter aussi deux algorithmes hybrides pour résoudre le même problème : l'un entre l'essaim particulaire et les algorithmes génétiques, l'autre entre l'évolution différentielle et le recuit simulé.

4 - SIMULATION ET RESULTATS

4.1 – INTRODUCTION

Les méta-heuristiques sont des algorithmes stochastiques capables de résoudre une large gamme de problèmes pour lesquels il n'existe pas de méthodes conventionnelles efficaces connues. Ces techniques sont souvent inspirées de la biologie (algorithmes génétiques, évolution différentielle), de la physique (recuit simulé) et de l'ethnologie (colonies de fourmis, optimisation par essaim de particules).

Les algorithmes méta-heuristiques ont été largement utilisés pour résoudre les problèmes d'optimisation dans un temps d'exécution raisonnable et sans nécessité d'avoir une connaissance approfondie du problème traité. La performance d'une métaheuristique nécessite un compromis entre l'exploitation et l'exploration de l'espace de recherche. Cependant, il est rare d'avoir les deux caractéristiques dans la même méthode de recherche, où l'émergence actuelle des méthodes hybrides. L'intelligence inspirée de la nature devient, de plus en plus, populaire et un nombre important de méthodes entraînées par des concepts issus de la nature ou de la biologie ont été développées.

Dans ce chapitre, on va essayer de résoudre le problème du contrôle tension/puissance réactive par l'application de plusieurs métaheuristiques tels que :

- l'optimisation par essaims particulaires PSO,
- un algorithme génétique amélioré IGA,
- le recuit simulé SA,
- l'évolution différentielle DE.

Afin d'exploiter les avantages des méthodes utilisées et d'améliorer les résultats, ce chapitre propose deux types d'hybridation :

- l'une entre l'essaim particulaire PSO et un algorithme génétique amélioré IGA avec une probabilité de mutation adaptative et un codage réel.
- l'autre entre l'évolution différentielle DE (basée sur une population de solution) et le recuit simulé SA (basée sur une solution unique).

Pour les variables de contrôle, une représentation mixte (continue/discrète) est proposée. Les méthodes utilisées sont testées sur les réseaux Ward-Hale 6 bus et IEEE 30 bus.

4.2 - DESCRIPTION DES RÉSEAUX ÉLECTRIQUES ÉTUDIÉS

Les différentes techniques d'optimisation sont testées sur deux réseaux de simulation : le réseau à 6 JdB appelé « Ward-Hale 6 bus » et le réseau à 30 JdB appelé « IEEE 30 bus ».

4.2.1 - RESEAU À 6 JDB

Soit le réseau électrique appelé « Ward-Hale 6 bus system », représenté sur la figure (4.1).

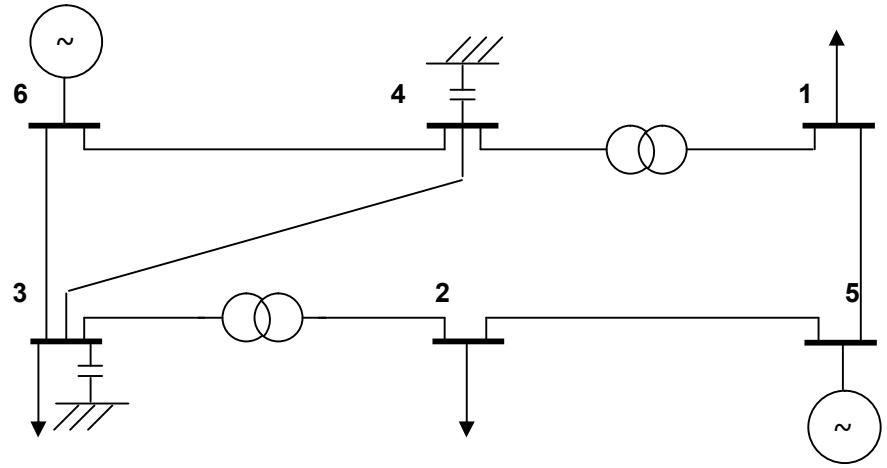


Fig. 4.1 – Réseau de test : Ward-Hale 6 bus system

- Le JdB 6 est le JdB d'équilibre.
- Le JdB 5 est un JdB du type PV (générateur).
- Les JdBs 3 et 4 contiennent des batteries de condensateurs shunts.
- Les branches qui contiennent des transformateurs réglables sont les branches 2-3 et 1- 4.

Les données du système proviennent de [8] et [49]. Les données des branches de ce réseau sont représentées dans le tableau (4.1). Les contraintes qui concernent les variables de contrôle sont décrites dans le tableau (4.2). Les contraintes qui concernent les variables dépendantes sont décrites dans le tableau (4.3). Les données des nœuds du réseau sont présentées dans le tableau (4.4) où la puissance de base utilisée est S_B = 100 MVA.

Tab. 4.1 – Données des branches du réseau étudié (à 6 JdB)

Ligne	r [pu]	x [pu]	r_t [pu]	x_t [pu]	Rapport de transformation
6-3	0.123	0.518	0	0	0
6-4	0.080	0.370	0	0	0
4-3	0.097	0.407	0	0	0
2-5	0.282	0.640	0	0	0
5-1	0.723	1.050	0	0	0
3-2	0	0	0	0.300	1.025
4-1	0	0	0	0.133	1.100

Tab. 4.2 – Contraintes des variables de contrôle pour le réseau étudié (réseau à 6 JdB)

	Rapport de transformation		Tension du JdB générateur [pu]		Installation de la puissance réactive [MVAr]	
	$T_{2\text{-}3}$	$T_{1\text{-}4}$	V_6	V_5	Q_{g_3}	Q_{g_4}
La limite inférieure	0.910		1.0	1.1	0.0	0.0
La limite supérieure	1.110		1.1	1.15	5.5	5.0

Tab. 4.3 – Contraintes des variables dépendantes pour le réseau étudié (réseau à 6 JdB)

	Tension des JdB du type PQ (charge) en [pu]	Puissance réactive des JdB des générateurs [MVAr]
La limite inférieure	0.9	- 20
La limite supérieure	1.1	100

Tab. 4.4 – Données des nœuds du réseau étudié (réseau à 6 JdB)

Nœud	Tension		Charge		Génération	
	V [pu]	θ[degré]	P_d [pu]	Q_d [pu]	P_g [pu]	Q_g [pu]
1	1	0	0.55	0.13	0	0
2	1	0	0.30	0.18	0	0
3	1	0	0.50	0.05	0	0
4	1	0	0	0	0	0
5	1.1	0	0	0	0.501	0
6	1.05	0	0	0	0	0

4.2.2 - RESEAU STANDARD À 30 JDB

Afin de tester et de valider nos approches, le système de test appelé IEEE 30-bus est utilisé (Fig. 4.2). Le réseau se compose de 30 jeux de barres, 41 branches, 6 générateurs, 4 transformateurs réglables et 9 sources de puissance réactive.

- JdB 1 est le JdB d'équilibre.
- JdB 2, 5, 8, 11 et 13 sont des JdB du type PV (générateur).
- JdB 10, 12, 15, 17, 20, 21, 23, 24 and 29 contiennent des batteries de condensateurs shunts.
- Branches (6-9), (6-10), (4-12) et (27-28) contiennent des transformateurs réglables.

Il convient de mentionner que les données du système proviennent de [15], [54], [33] et [4]. Les données des branches, les contraintes de variables de contrôle, les contraintes de variables d'état et les données des JdB sont indiquées respectivement dans les tableaux (4.5), (4.6), (4.7) et (4.8) où la puissance de base utilisée est $S_B = 100$ (MVA).

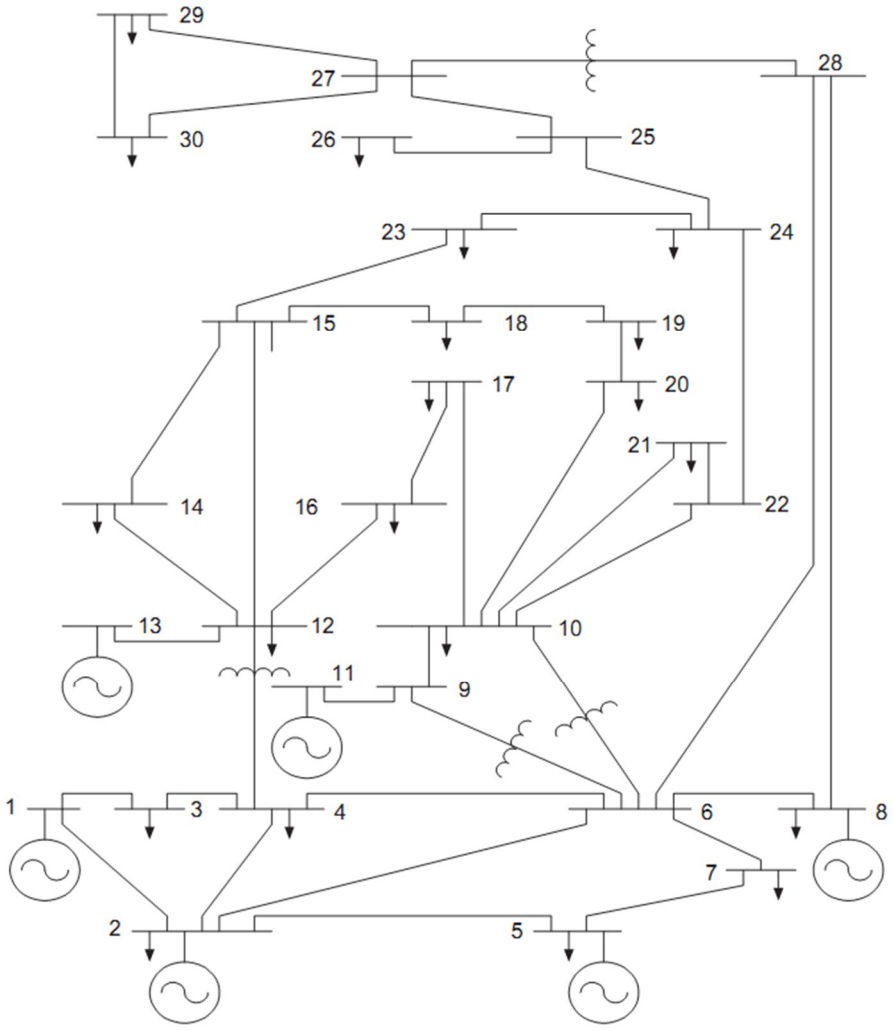


Fig. 4.2 - Schéma unifilaire du réseau à 30 JdB IEEE 30-bus [54]

Tab. 4.5 - Données des branches du réseau étudié (réseau à 30 JdB)

Branche	de	à	R [pu]	X [pu]	R_t [pu]	X_t [pu]	B [pu]	Rapport de transformation
1	1	2	0.0192	0.0575	0	0	0.0264	0
2	1	3	0.0452	0.1852	0	0	0.0204	0
3	2	4	0.0570	0.1737	0	0	0.0184	0
4	3	4	0.0132	0.0379	0	0	0.0042	0
5	2	5	0.0472	0.1983	0	0	0.0209	0
6	2	6	0.0581	0.1763	0	0	0.0187	0
7	4	6	0.0119	0.0414	0	0	0.0045	0
8	5	7	0.0460	0.1160	0	0	0.0102	0
9	6	7	0.0267	0.0820	0	0	0.0085	0
10	6	8	0.0120	0.0420	0	0	0.0045	0
11	6	9	0	0	0	0.2080	0	1.0780
12	6	10	0	0	0	0.5560	0	1.0690
13	9	11	0	0.2080	0	0	0	0
14	9	10	0	0.1100	0	0	0	0
15	4	12	0	0	0	0.2560	0	1.0320
16	12	13	0	0.1400	0	0	0	0
17	12	14	0.1231	0.2559	0	0	0	0
18	12	15	0.0662	0.1304	0	0	0	0
19	12	16	0.0945	0.1987	0	0	0	0
20	14	15	0.2210	0.1997	0	0	0	0
21	16	17	0.0824	0.1932	0	0	0	0
22	15	18	0.1070	0.2185	0	0	0	0
23	18	19	0.0639	0.1292	0	0	0	0
24	19	20	0.0340	0.0680	0	0	0	0
25	10	20	0.0936	0.2090	0	0	0	0
26	10	17	0.0324	0.0845	0	0	0	0
27	10	21	0.0348	0.0749	0	0	0	0
28	10	22	0.0727	0.1499	0	0	0	0
29	21	22	0.0116	0.0236	0	0	0	0
30	15	23	0.1000	0.2020	0	0	0	0
31	22	24	0.1150	0.1790	0	0	0	0
32	23	24	0.1320	0.2700	0	0	0	0
33	24	25	0.1885	0.3292	0	0	0	0
34	25	26	0.2544	0.3800	0	0	0	0
35	25	27	0.1093	0.2087	0	0	0	0
36	28	27	0	0	0	0.3960	0	1.0680
37	27	29	0.2198	0.4153	0	0	0	0
38	27	30	0.3202	0.6027	0	0	0	0
39	29	30	0.2399	0.4533	0	0	0	0
40	8	28	0.0636	0.2000	0	0	0.0214	0
41	6	28	0.0169	0.0599	0	0	0.0065	0

Tab. 4.6 - Contraintes des variables de contrôle (réseau à 30 JdB)

	Rapport de transformation	Tension du JdB générateur [pu]	Installation de puissance réactive [pu]
Limite inférieure	0.9	0.95	0.0
Limite supérieure	1.1	1.1	0.05

Tab. 4.7 - Contraintes des variables dépendantes (réseau à 30 JdB)

	Tension des JdB de type PQ [pu]	Puissance réactive des JdB de type PV (générateurs) [pu]						
		Bus	1	2	5	8	11	13
Limite inférieure	0.95		-0.2	-0.2	-0.15	-0.15	-0.1	-0.15
Limite supérieure	1.05		0.25	1	0.8	0.6	0.5	0.6

Tab. 4.8 - Données des nœuds du réseau étudié (réseau à 30 JdB)

JdB	Tension		Charge		Génération	
	V [pu]	θ [degré]	P_d [pu]	Q_d [pu]	P_g [pu]	Q_g [pu]
1	1.05	0	0	0	0	0
2	1.04	0	0.217	0.127	0.8	0
3	1	0	0.024	0.012	0	0
4	1	0	0.076	0.016	0	0
5	1.01	0	0.942	0.190	0.5	0
6	1	0	0	0	0	0
7	1	0	0.228	0.109	0	0
8	1.01	0	0.300	0.300	0.2	0
9	1	0	0	0	0	0
10	1	0	0.058	0.020	0	0
11	1.05	0	0	0	0.2	0
12	1	0	0.112	0.075	0	0
13	1.05	0	0	0	0.2	0
14	1	0	0.062	0.016	0	0
15	1	0	0.082	0.025	0	0
16	1	0	0.035	0.018	0	0
17	1	0	0.090	0.058	0	0
18	1	0	0.032	0.009	0	0
19	1	0	0.095	0.034	0	0
20	1	0	0.022	0.007	0	0
21	1	0	0.175	0.112	0	0
22	1	0	0	0	0	0
23	1	0	0.032	0.016	0	0
24	1	0	0.087	0.067	0	0
25	1	0	0	0	0	0
26	1	0	0.035	0.023	0	0
27	1	0	0	0	0	0
28	1	0	0	0	0	0
29	1	0	0.024	0.009	0	0
30	1	0	0.106	0.019	0	0

4.3 – APPLICATION DES METHODES PROPOSEES SUR LE RESEAU À 6 JDB

4.3.1 - CALCUL DE L'ECOULEMENT DE PUISSANCE

L'étude de l'écoulement de puissance par l'application de la méthode de Newton-Raphson a donné les résultats présentés dans le tableau (4.9).

On peut remarquer que la tension V_1 est hors limite. Pour cela, on est obligé d'ajuster les moyens de contrôle qui sont : les tensions des générateurs (V_5, V_6), les rapports de transformation des transformateurs réglables ($T_{1\text{-}4}$, $T_{2\text{-}3}$) et les valeurs de puissance réactive des batteries de condensateurs aux JdB 3 et 4 pour avoir un plan optimal de tension tout en minimisant les pertes totales de la puissance active dans le réseau. Ceci revient à résoudre un problème d'optimisation (minimisation) avec contraintes. On va essayer de résoudre ce problème par l'application de plusieurs métaheuristiques ainsi que l'utilisation de l'hybridation.

Tab. 4.9 - Résultats de l'écoulement de puissance (réseau à 6 JdB)

	Tension		Charge		Génération	
JdB	V [pu]	θ [degré]	P_d [pu]	Q_d [pu]	P_g [pu]	Q_g [pu]
1	0.8552	-13.8149	0.55	0.13	0	0
2	0.9009	-13.3980	0.30	0.18	0	0
3	0.9332	-12.6355	0.50	0.05	0	0
4	0.9525	-9.9117	0	0	0	0
5	1.1000	-6.0947	0	0	0.5010	0.3478
6	1.0500	0	0	0	0.9652	0.3812
Pertes actives totales : 11.6197 [MW]						

4.3.2 – APPLICATION DE L'ALGORITHME GENETIQUE AMÉLIORÉ (IGA)

Selon la situation de base de l'écoulement de puissance, les variables dépendantes seront vérifiées. Si un de ces variables est hors de la limite, les AGs vont ajuster les variables de contrôle pour éliminer cette violation des contraintes et minimiser les pertes de puissance. On propose un algorithme génétique amélioré en utilisant une **probabilité de mutation adaptative avec un codage réel.** Les principaux avantages de ce codage sont la simplicité, la facilité et la vitesse de convergence rapide. La modification dans la probabilité de mutation améliore la convergence de l'algorithme génétique. Dans la pratique, quelques types des variables de contrôle ont des valeurs discrètes et non pas continues. Ce travail consiste à représenter les tensions des générateurs comme valeurs continues. Par contre, les batteries de condensateurs shunts et les transformateurs réglables

sont représentés en valeurs discrètes. L'organigramme de l'application des algorithmes génétiques améliorés pour résoudre le problème du contrôle optimal tension - puissance réactive est présenté sur la figure (4.3).

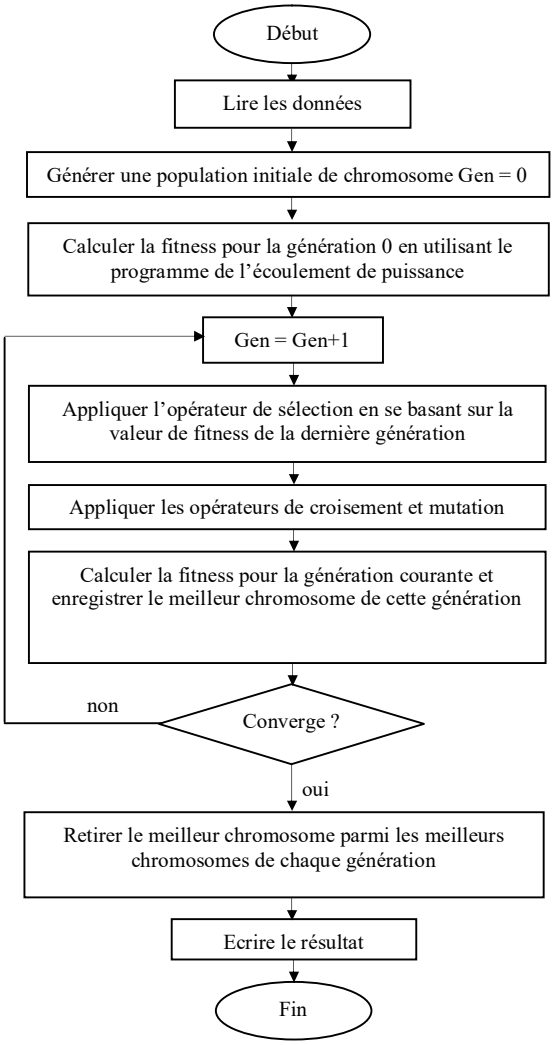


Fig. 4.3 - Organigramme de l'application de l'IGA pour résoudre le problème du contrôle optimal tension - puissance réactive

a) Représentation des variables de contrôle

Une batterie de condensateurs shunts est modélisée par sa puissance réactive injectée. Les trois types des variables de contrôle sont codés en réel dans une seule chaîne X appelé « chromosome » représentée comme un vecteur :

$$X=\left[V_{N-N_{PV}+1} \ \ldots \ V_N \middle| T_1 \ \ldots \ T_{NT} \middle| Q_{gcap_1} \ \ldots \ Q_{gcap\,N_{cap}} \right] \tag{4.1}$$

Chacun des deux transformateurs $T_{2\text{-}3}$ et $T_{1\text{-}4}$ contient 16 gradins. Pour un transformateur *i* à gradins variables, la modification de la valeur du rapport de transformation s'effectue selon l'expression :

$$T_i = T_{i\min} + NT_i \times \Delta T_i \quad \text{avec } NT_i = 0,1,\ldots, NTmax_i \tag{4.2}$$

Où :

NT_i : Une position du gradin parmi les $NTmax_i$ positions possibles dans le transformateur *i*.

ΔT_i : Le pas de la variation du rapport de transformation.

Chacune des deux batteries de condensateurs shunts a 10 variations possibles. Pour une batterie de condensateurs shunts, la modification dans sa puissance réactive générée s'effectue selon l'expression :

$$Q_{g_i} = Q_{gi\min} + NQ_{ci} \times \Delta Qg_i \tag{4.3}$$

Avec :

NQ_{ci} : Le numéro de la variation de la puissance réactive de la batterie de condensateurs shunts au JdB *i*.

ΔQg_i : Le pas de la variation de la puissance réactive générée de la batterie de condensateurs shunts au JdB *i*.

b) Génération initiale

La première génération de chromosomes est créée en utilisant des variables aléatoires uniformes :

$$Y_i = Y_{i\min} + rnd \times \left(Y_{i\max} - Y_{i\min}\right) \tag{4.4}$$

Où rnd est un nombre aléatoire $0 < rnd < 1$

Pour traiter les variables discrètes, on doit ajuster la valeur du variable pour obtenir une formulation comme:

$$Y_i = Y_{i\min} + NY_i \times \Delta Y \tag{4.5}$$

NY_i : un nombre entier qui représente le numéro de la variation du variable Y_i au JdB *i*.

ΔY : Le pas de variation du variable Y_i .

c) Sélection

L'opérateur de sélection proposé est basé sur la méthode de la roulette biaisée.

d) Croisement

Basé sur la probabilité de croisement, les chromosomes sont choisis aléatoirement pour le processus de croisement. Le croisement dans un seul point est utilisé.

e) Mutation

Basé sur la probabilité de mutation, les chromosomes sont choisis aléatoirement pour le processus de mutation.

En général, la probabilité de mutation est fixe pendant toute la procédure de recherche. Cependant, dans des applications pratiques, une petite valeur fixe de probabilité de mutation peut donner des résultats prématurés, alors que la recherche avec une grande valeur fixe de probabilité de mutation ne convergera pas.

Une probabilité de mutation adaptative est donnée pour résoudre le problème comme suit :

$$p_m(k+1)=\begin{cases} p_m(k)-p_{mstep} & \text{si} \quad f_{\max}(k)\ inchangé \\ p_m(k) & \text{si} \quad f_{\max}(k)\ augmente \\ p_{mfinal} & \text{si} \quad p_m(k)-p_{mstep} < p_{mfinal} \end{cases}$$

$$p_m(0)=p_{\min it} \tag{4.6}$$

Où k est le numéro de génération, $p_{\min it}$, p_{mfinal} et p_{mstep} sont des nombres fixés. $p_{\min it}$ est autour 1et *le* p_{mfinal} *serait 0,005*. Le p_{mstep} dépend du nombre maximal de génération. $f_{\max}(k)$ est la fitness maximale de la génération numéro *k*.

f) Fonction Objective

La fonction objective f utilisée est :

$$f=p+\sum_{i\in N_{PQ}}\lambda_{V\text{charge}}(V_i-V_i^{\lim})^2+\sum_{i\in N_{PV}}\lambda_{Qg}(Q_{gi}-Q_{gi}^{\lim})^2 \tag{4.7}$$

p : Pertes actives totales dans le réseau.

$\lambda_{Vcharge}$: Facteur de pénalité des tensions des JdBs de charge.

λ_{Qg} : Facteur de pénalité des puissances réactives des générateurs.

$$V_i^{\lim}=\begin{cases} V_{i\min} & si \quad V_i < V_{i\min} \\ V_{i\max} & si \quad V_i > V_{i\max} \\ V_i & si \quad V_i \le V_{i\min} \le V_{i\max} \end{cases} \tag{4.8}$$

$$Q_{gi}^{\lim}=\begin{cases} Q_{gi\min} & si \quad Q_{gi} < Q_{gi\min} \\ Q_{gi\max} & si \quad Q_{gi} > Q_{gi\max} \\ Q_{gi} & si \quad Q_{gi} \le Q_{gi\min} \le Q_{gi\max} \end{cases} \tag{4.9}$$

Les facteurs de pénalité pour les contraintes des variables dépendantes doivent être d'une valeur plus élevée pour aider l'algorithme génétique à alléger les violations lors de la maximisation de la fitness.

g) Fonction d'Aptitude

Notre objectif est de minimiser les pertes totales de transport. Mais, les GA sont conçus pour trouver des solutions maximales. Pour cela, il est nécessaire de faire une transformation. Dans notre cas, la fonction d'aptitude *g* (fitness) est choisie comme :

$$g = \frac{1}{1+f} \tag{4.10}$$

h) Application de l'IGA sur le réseau Ward-Hale 6 bus

Une représentation mixte (continue/discrète) pour les variables de contrôle est utilisée. Les tensions des générateurs sont représentées en valeurs continues. Par contre, les batteries de condensateurs shunts et les transformateurs réglables sont représentés en valeurs discrètes.

Les paramètres utilisés de l'algorithme génétique amélioré sont :

- Nombre maximale de générations : 20
- Taille de la population : 2000
- Probabilité de croisement : 0.9
- Une probabilité de mutation adaptative est utilisée ici selon l'équation (4.6).

La programmation de l'algorithme génétique a donné après 9 générations, les résultats présentés sur les tableaux (4.10) et (4.11).

Tab. 4.10 - Variables de contrôle obtenues par l'IGA (réseau à 6 JdB)

V_5 [pu]	V_6 [pu]	$T_{2\text{-}3}$	$T_{1\text{-}4}$	Q_{g_3} [pu]	Q_{g_4} [pu]
1.1477	1.0970	0.9350	0.9475	0.0495	0.0500

Tab. 4.11 - Etat du réseau après application de l'IGA (réseau à 6 JdB)

	Tension		Génération	
JdB	V [pu]	θ [degré]	P_g [pu]	Q_g [pu]
1	1.0299	-11.5051	0	0
2	1.0101	-11.0347	0	0
3	0.9753	-10.9179	0	0.0495
4	0.9943	-8.7866	0	0.0500
5	1.1477	-2.7035	0.5010	0.1415
6	1.0970	0	0.9369	0.4225
Pertes actives totales : 8.7933 [MW]				

Les figures (4.4), (4.5) et (4.6) représentent respectivement la meilleure fitness (fonction d'évaluation à maximiser), la meilleure fonction objective (à minimiser) et les pertes dans chaque génération.

Commentaire :

Les résultats de simulation montrent clairement que toutes les variables de contrôle sont ajustées. Les pertes sont réduites de 11.6197 [MW] dans le cas de l'écoulement de puissance à 8.7933 [MW] après exécution de l'algorithme génétique amélioré (une diminution de 24.32%). Le plan de tension est amélioré et toutes les contraintes sont respectées.

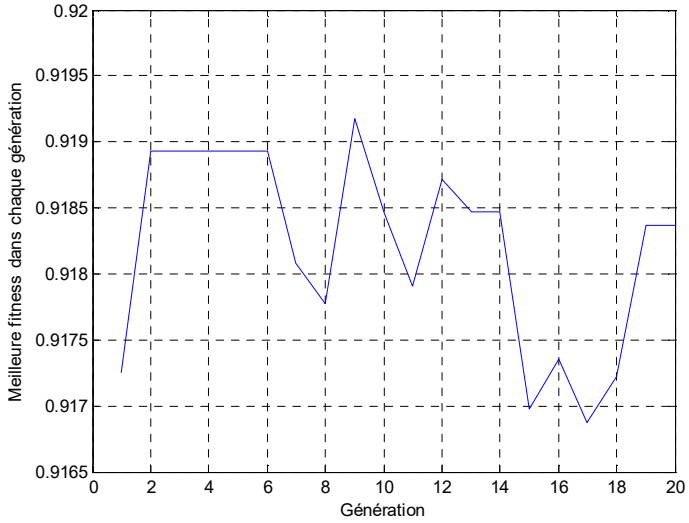


Fig. 4.4 - Meilleure fonction d'évaluation (fitness à maximiser) dans chaque génération de l'IGA (réseau à 6 JdB)

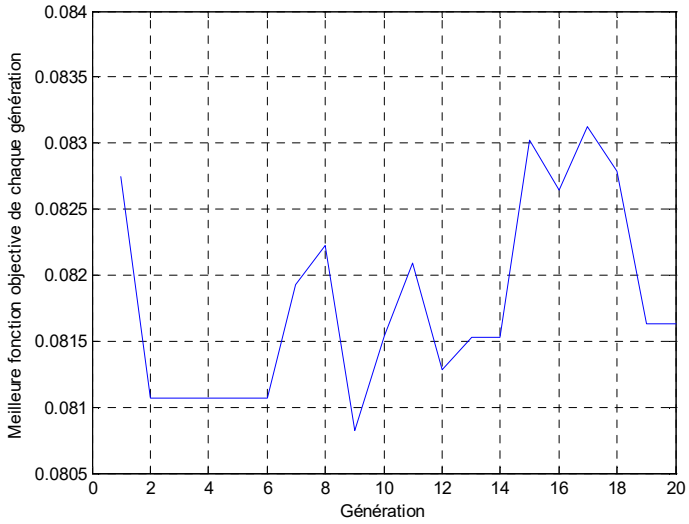


Fig. 4.5 - Meilleure fonction objective (à minimiser) dans chaque génération de l'IGA (réseau à 6 JdB)

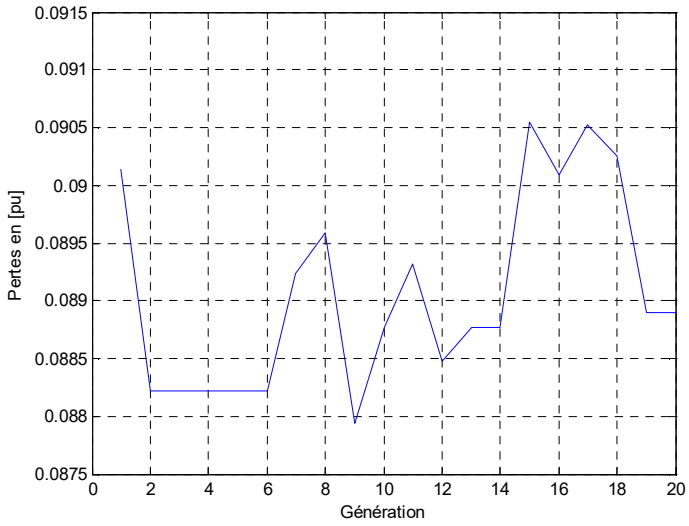


Fig. 4.6 – Pertes dans chaque génération de l'IGA (réseau à 6 JdB)

4.3.3 - APPLICATION DE L'OPTIMISATION PAR ESSAIM PARTICULAIRE PSO

À la différence des algorithmes génétiques GA, l'optimisation par essaims particulaires PSO n'a aucun opérateur d'évolution tel que le croisement et la mutation. Dans le PSO, la population est initialisée aléatoirement et les solutions potentielles, appelées les particules, volent dans l'espace de recherche avec des vitesses qui sont dynamiquement ajustées selon leurs comportements historiques. Dans le PSO, chaque particule est influencée par la meilleure solution qu'elle a découvert jusqu'ici et la meilleure particule découvrée par ses voisins (variante locale de PSO) ou dans la population entière (variante globale de PSO). Ce compromis est formalisé par les équations (4.11)-(4.12).

Le but d'un algorithme de PSO est de réduire au minimum une fonction fitness f qui dépend d'un ensemble de variables inconnues $(x_1, x_2, x_3, \ldots, x_n)$. Une solution candidat pour ce problème est traduis dans le formalisme de PSO comme une position d'une particule i de l'essaim; $S_i = (x_{1i}, x_{2i}, x_{3i} \ldots, x_{ni})$. Soit k une instant de temps (index d'itération dans le contexte de l'optimisation). La nouvelle vitesse et position d'une particule i s'écrit comme suit:

$$v_i^{k+1} = w^k \cdot v_i^k + c_1 \cdot r_1 \cdot \left(pbest_i^k - s_i^k\right) + c_2 \cdot r_2 \cdot \left(gbest^k - s_i^k\right) \quad (4.11)$$

$$s_i^{k+1} = s_i^k + v_i^{k+1} \quad (4.12)$$

Où :

v_i^k : Vitesse courante de l'agent i à l'itération k,

v_i^{k+1} : Vitesse modifiée de l'agent i,

r_1 et r_2 : Nombres aléatoires entre 0 et 1,

s_i^k : Position courante de l'agent *i* à l'itération *k*,

$pbest_i^k$: La meilleure position de la particule *i* à l'itération *k*,

$gbest^k$: La meilleure position globale dans l'essaim à l'itération *k*,

c_1 et c_2 : Constantes cognitives et sociales d'accélération,

w^k : Poids d'inertie, évoluant selon l'équation:

$$w^k = w_{\max} - \left(\frac{w_{\max} - w_{\min}}{k_{\max}} \right) \cdot k \tag{4.13}$$

$w_{\max}$ et $w_{\min}$: Valeurs maximales et minimales du poids,

$k_{\max}$: Le nombre maximal d'itérations,

L'algorithme général du PSO peut être appliqué à n'importe quel problème d'optimisation. Le pseudo code utilisé est décrit sur la figure (4.7).

PSO a été développé ici pour traiter le problème du contrôle de tension/puissance réactive avec variables continues et discrètes. Les tensions des générateurs sont représentées en valeurs continues. Par contre, les batteries de condensateurs shunts et les transformateurs réglables sont représentés en valeurs discrètes.

Chaque transformateur contient 16 gradins. Ainsi, la modification de la valeur du rapport de transformation d'un transformateur *i* sera :

$$T_i = 0.91 + NT_i \times 1.25\% \quad \text{avec} \quad NT_i = 0, 1, \ldots, 16 \tag{4.14}$$

Les limites des rapports de transformation sont $[0.91, 0.91 + 16 \times 1.25\%]$.

Chacune des deux batteries de condensateurs shunts a 10 variations possibles. Alors la modification de la valeur de la puissance réactive générée dans chaque JdB d'une batterie de condensateurs shunts est :

$$Q_{g_3} = 0 + NQ_{c3} \times 0.0055 \quad \text{avec} \quad NQ_{c3} = 0, 1, \ldots, 10 \tag{4.15}$$

$$Q_{g_4} = 0 + NQ_{c3} \times 0.005 \quad \text{avec} \quad NQ_{c4} = 0, 1, \ldots, 10 \tag{4.16}$$

L'optimisation par essaim de particules a trois principaux paramètres, *w, C_1 et C_2*. Les valeurs les plus utilisées pour C_1 et C_2 sont $C_1 = C_2 = 2$ [29]. Le poids d'inertie *w* est égale à 1 dans le PSO original [29], mais dans [47] *w* a des valeurs dans la gamme [0.8 , 1.2].

Début
  Pour chaque particule
    Initialiser la particule avec des nombres aléatoires faisables
  Fin
  Tant que le nombre maximal des itérations (ou tout autre critère de convergent) n'est pas atteint
    Pour chaque particule
      Calculer la valeur de fitness
      Si la valeur courante de fitness est mieux que la meilleure valeur de fitness *(pbest)* obtenue précédemment
        Mettre la valeur courante comme étant la nouvelle valeur de *pbest*
      Fin
    Fin
    Parmi toutes les particules, choisir la particule avec la meilleure valeur de fonction fitness comme étant la nouvelle valeur de *gbest*
    Pour chaque particule
      Calculer la vitesse de particule selon l'équation (4.11) ($-v_{max} \le v \le v_{max}$)
      Changer la valeur de position de particule selon l'équation (4.12) ($s_{min} \le s \le s_{max}$)
    Fin
  Fin
Fin

Fig. 4.7 - Pseudo code utilisé de l'algorithme du PSO

Les paramètres du PSO utilisés ici sont les suivants :

- Nombre maximal de générations : 20
- Nombre de particules : 50
- C_1 : 2
- C_2 : 2
- Poids d'inertie initial w : 0.9
- w_{min} : 0.8
- w_{max} : 1.2
- Vitesse Initial v : 0.05
- Valeur maximal de la vitesse v_{max} : 0.5

La programmation du PSO a donné après 9 itérations (générations), les résultats présentés sur les tableaux (4.12) et (4.13). Les figures (4.8), (4.9) et (4.10) représentent les caractéristiques de convergence de l'algorithme du PSO.

Tab. 4.12 - Variables de contrôle obtenues par le PSO (réseau à 6 JdB)

V_5 [pu]	V_6 [pu]	$T_{2\text{-}3}$	$T_{1\text{-}4}$	Q_{g_3} [pu]	Q_{g_4} [pu]
1.15	1.1	0.935	0.935	0.055	0.050

Tab. 4.13 - Etat du réseau après application du PSO (réseau à 6 JdB)

	Tension		Génération	
JdB	V [pu]	θ [degré]	P_g [pu]	Q_g [pu]
1	1.0445	-11.3859	0	0
2	1.0131	-10.9002	0	0
3	0.9786	-10.8270	0	0.0550
4	0.9956	-8.7299	0	0.0500
5	1.1500	-2.4985	0.5010	0.1293
6	1.1000	0	0.9361	0.4278
Pertes actives totales : 8.7169 [MW]				

Commentaire :

Les résultats de simulation obtenus de l'exécution de l'algorithme du PSO sont meilleurs que ceux obtenus des algorithmes génétiques améliorés (IGA). Les pertes actives totales dans le réseau sont réduites de 8.7933 [MW] (après application d'IGA) à 8.7169 [MW] (après application du PSO), avec une diminution de 24.98% par apport au cas initial de l'écoulement de puissance.

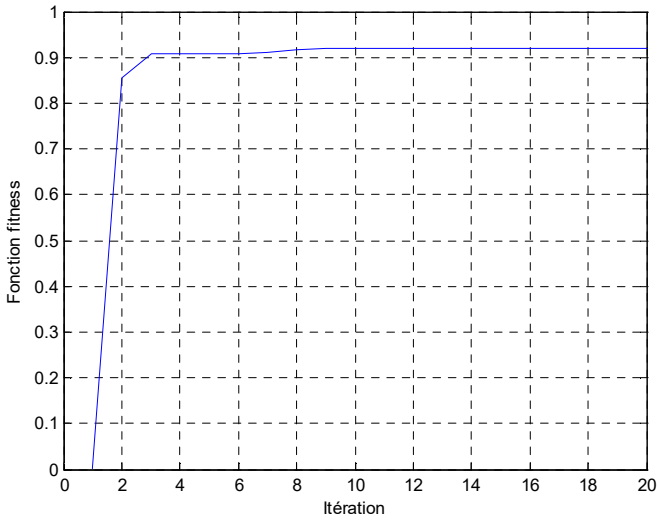


Fig. 4.8 - Meilleure fonction d'évaluation (à maximiser) dans chaque itération du PSO (réseau à 6 JdB)

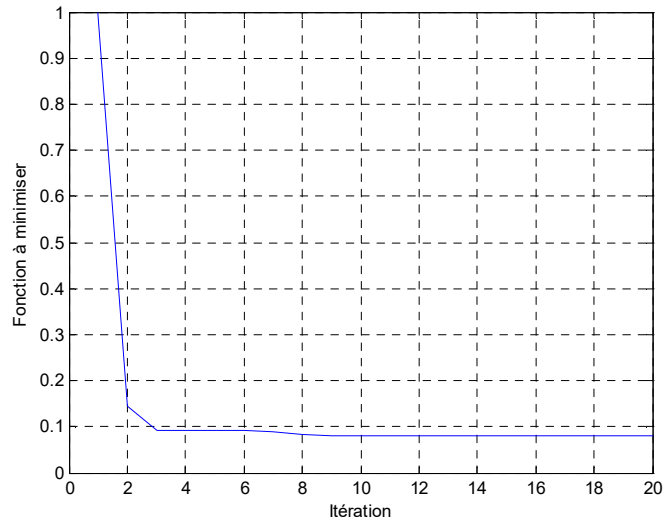


Fig. 4.9 - Meilleure fonction objective (à minimiser) dans chaque itération du PSO (réseau à 6 JdB)

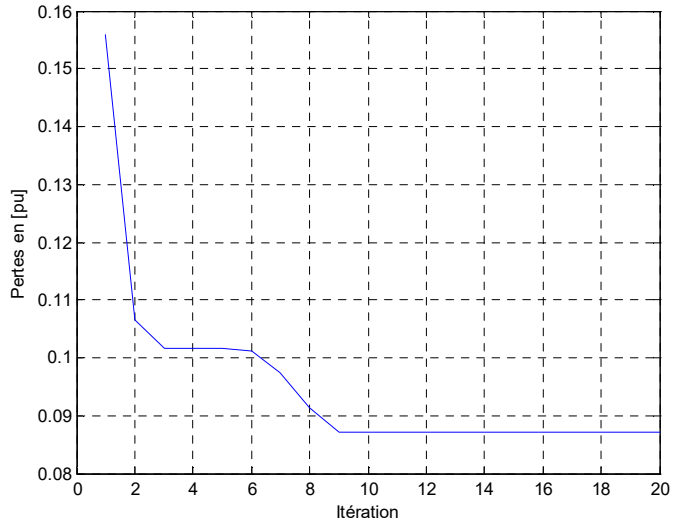


Fig. 4.10 – Pertes dans chaque itération du PSO (réseau à 6 JdB)

4.3.4 - HYBRIDATION ENTRE PSO ET IGA

Pour améliorer plus les résultats, nous avons appliqué une formulation hybride entre le PSO et l'IGA, nommée HPSOIGA. La méthode du PSO a été employée pour générer une population initiale injectée et utilisée dans le programme de l'IGA.

L'hybridation appliquée a donné après 3 générations les résultats présentés dans les tableaux (4.14) et (4.15). Les figures (4.11), (4.12) et (4.13) représentent les caractéristiques de convergence de l'algorithme d'hybridation entre IGA et PSO.

Commentaire :

Nous pouvons voir que toutes les contraintes sont respectées et les pertes actives totales dans le réseau sont réduites jusqu'à 8.7114 [MW] (une diminution de 25.02% par apport au cas initial de l'écoulement de puissance).

Tab. 4.14 - Variables de contrôle obtenues par l'hybridation HPSOIGA (réseau à 6 JdB)

V_5 [pu]	V_6 [pu]	$T_{2\text{-}3}$	$T_{1\text{-}4}$	Q_{g_3} [pu]	Q_{g_4} [pu]
1.15	1.1	0.935	0.9475	0.055	0.050

Tab. 4.15 - Etat du réseau après application de l'hybridation HPSOIGA (réseau à 6 JdB)

	Tension		Génération	
JdB	V [pu]	θ [degré]	P_g [pu]	Q_g [pu]
1	1.0338	-11.4251	0	0
2	1.0143	-10.9550	0	0
3	0.9797	-10.8444	0	0.0550
4	0.9980	-8.7271	0	0.0500
5	1.1500	-2.6331	0.5010	0.1373
6	1.1000	0	0.9361	0.4185
Pertes actives totales : 8.7114 [MW]				

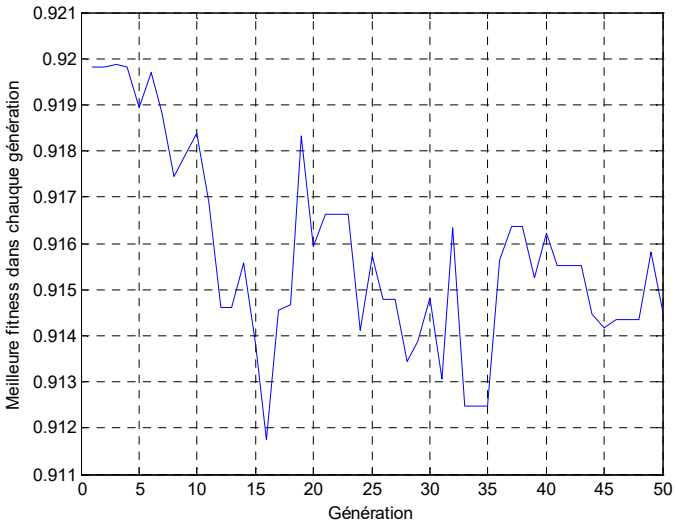


Fig. 4.11 - Meilleure fonction d'évaluation (fitness à maximiser) dans chaque génération de l'algorithme d'hybridation HPSOIGA (réseau à 6 JdB)

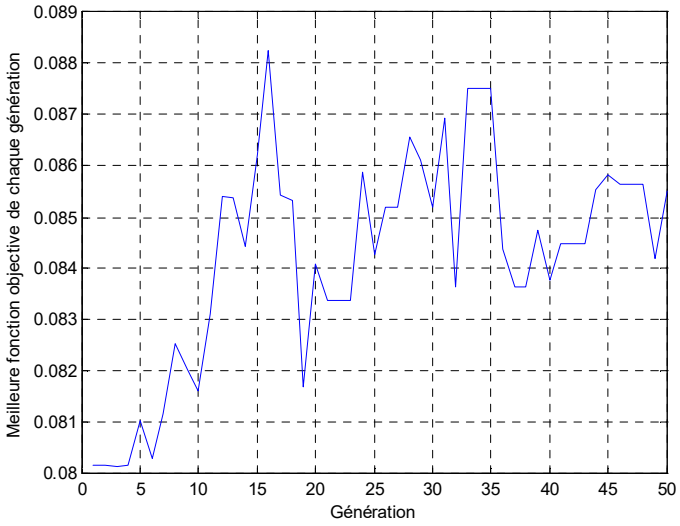


Fig. 4.12 - Meilleure fonction objective (à minimiser) dans chaque génération de l'algorithme d'hybridation HPSOIGA (réseau à 6 JdB)

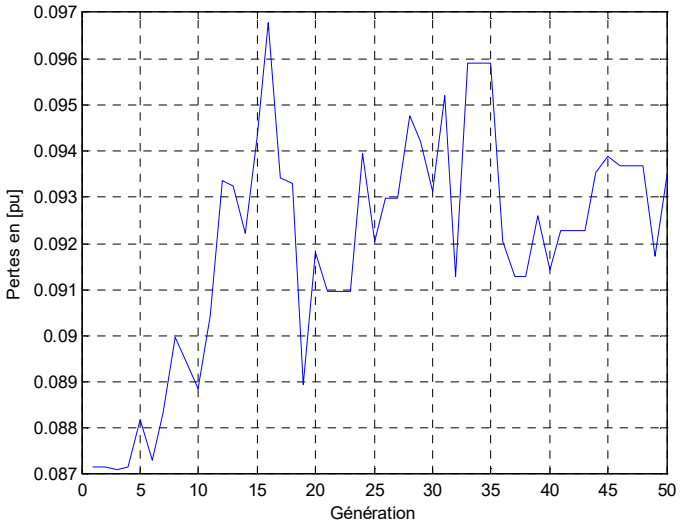


Fig. 4.13 – Pertes dans chaque génération de l'algorithme d'hybridation HPSOIGA (réseau à 6 JdB)

4.3.5 – APPLICATION DU RECUIT SIMULÉ

Selon l'algorithme de la figure (3.10), l'algorithme de base du recuit simulé (SA) comporte 4 paramètres principaux: T_{max}, T_{min}, α et *Trymax*. Afin d'aider l'algorithme de SA à trouver la solution optimale, nous devons utiliser :

- une température initiale à grande valeur,
- une température finale à petite valeur,
- un maximum d'essais d'une grande valeur,
- une diminution lente de la température.

La décroissance de la température à l'itération *(t)* est faite en utilisant : $T(t) = \alpha \, . \, T(t-1)$, Où α est un coefficient proche de 1.

Les paramètres utilisés ici pour le recuit simulé sont:

- Température initiale T_{max} : 1000
- Température finale T_{min} : 0.1
- Constante de la baisse de température α : 0.99
- Maximum d'essaies à chaque température *Trymax* : 50

Les résultats de SA sont présentés dans les tableaux (4.16) et (4.17). La figure (4.14) montre la courbe de décroissance de la température. Les figures (4.15) et (4.16) représentent respectivement la meilleure fonction objective (à minimiser) et les pertes obtenues dans chaque itération de l'algorithme de SA.

Commentaire :

Les pertes de puissances actives totales sont diminuées de 11,6197 [MW] dans le cas de l'écoulement de puissance à 8,9277 [MW] après l'utilisation de SA. Ce qui représente une diminution des pertes de 23.16%.

Tab. 4.16 - Variables de contrôles obtenues par le SA (réseau à 6 JdB)

V_5 [pu]	V_6 [pu]	T_{2-3}	T_{1-4}	Q_{g_3} [pu]	Q_{g_4} [pu]
1.1434	1.0974	0.9475	0.9850	0.0550	0.0350

Tab. 4.17 - Etat du réseau après application du SA (réseau à 6 JdB)

	Tension		Génération	
JdB	V [pu]	θ [degré]	P_g [pu]	Q_g [pu]
1	0.9966	-11.6482	0	0
2	1.0034	-11.2131	0	0
3	0.9804	-10.9900	0	0.0550
4	0.9986	-8.7734	0	0.0350
5	1.1434	-3.0871	0.5010	0.1721
6	1.0974	0	0.9382	0.4012
Pertes actives totales : 8.9277 [MW]				

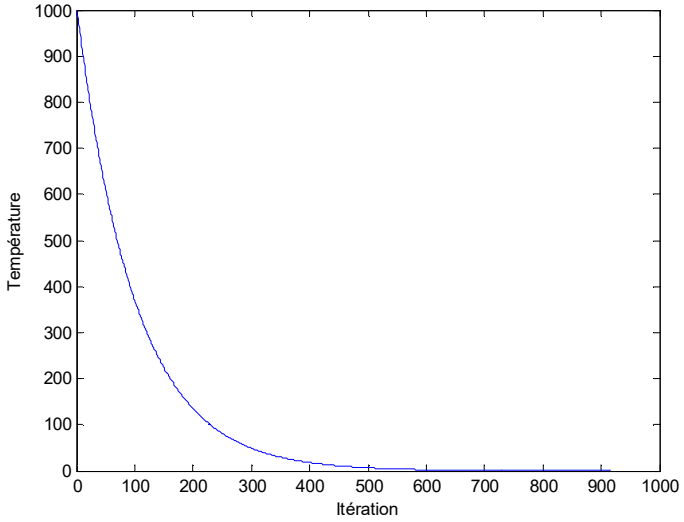


Fig. 4.14 – Courbe de décroissance de la température (réseau à 6 JdB)

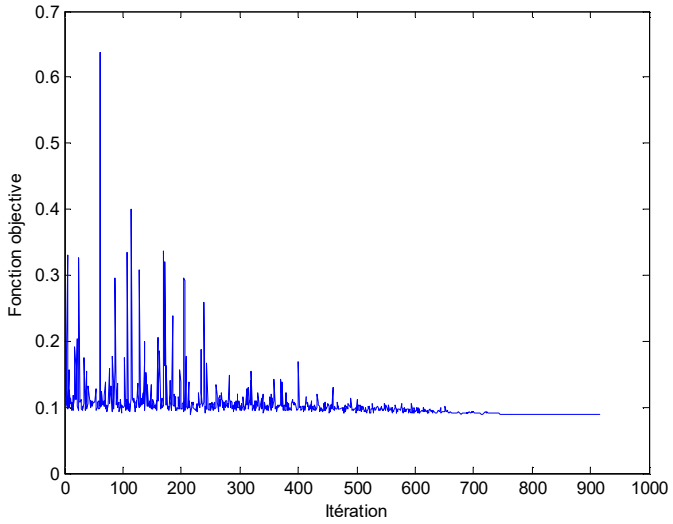


Fig. 4.15 – Meilleure fonction objective (à minimiser) dans chaque itération du SA (réseau à 6 JdB)

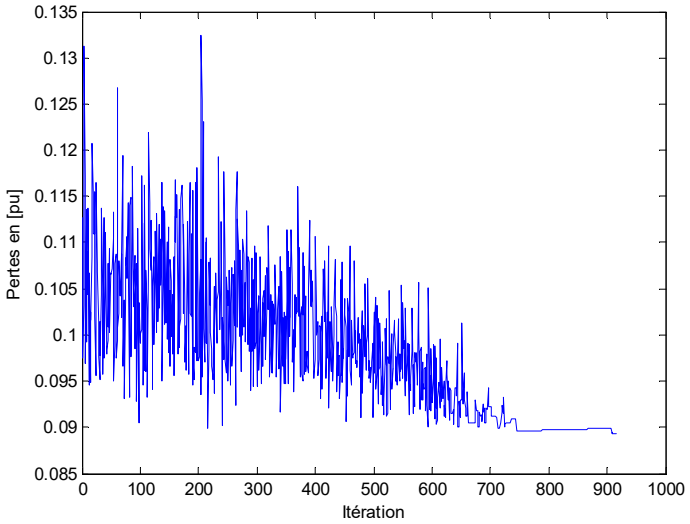


Fig. 4.16 – Pertes dans chaque itération du SA (réseau à 6 JdB)

4.3.6 – APPLICATION DE L'EVOLUTION DIFFERENTIELLE

Il existe plusieurs variantes de l'évolution différentielle (DE). Pour ce travail, on a utilisé la stratégie « DE / rand / 1 ». Cette dernière est la plus couramment utilisée pour résoudre des problèmes pratiques [48].

Les principaux paramètres de l'évolution différentielle DE sont le poids différentiel F (taille du pas) et le coefficient de croisement CR. Le paramètre F contrôle l'échelle de la variation différentielle. Il est généralement sélectionné pour être dans la gamme de $0 \leq F \leq 2$. Si F est trop petit, les vecteurs de différenciation nous conduiront à une recherche locale. Dans le cas où F est trop grand, les nouveaux vecteurs peuvent souvent dépasser les contraintes imposées.

Le coefficient de croisement CR est généralement compris entre $0 < CR < 1$. De plus, il doit être suffisamment proche de 1 pour permettre l'acceptation de nouveaux vecteurs et augmenter le niveau de diversité dans la population.

Dans ce travail, les différents paramètres de DE sont déterminés après plusieurs essais d'exécution de simulation. Les paramètres utilisés pour l'algorithme DE sont:

- Nombre maximal de générations : 1000
- Taille de la population : 20
- Poids differentiel F : 1
- Coefficient de croisement CR : 0.8

Le programme DE a donné après 341 générations les résultats présentés dans les tableaux (4.18) et (4.19). Les figures (4.17), (4.18) et (4.19) représentent respectivement la meilleure fitness (fonction d'évaluation à maximiser), la meilleure fonction objective (à minimiser) et les pertes dans chaque itération.

Commentaire :

Les résultats de la simulation montrent clairement que le profil de tension est amélioré et toutes les contraintes sont respectées. On peut remarquer une diminution importante de 25.04 % des pertes actives totales, allant de 11,6197 [MW] dans le cas de l'écoulement de puissance à 8.7099 [MW] après application de l'évolution différentielle (DE).

Tab. 4.18 - Variables de contrôle obtenues par le DE (réseau à 6 JdB)

V_5 [pu]	V_6 [pu]	$T_{2\text{-}3}$	$T_{1\text{-}4}$	Q_{g_3} [pu]	Q_{g_4} [pu]
1.15	1.1	0.9284	0.9451	0.055	0.05

Tab. 4.19 - Etat du réseau après application du DE (réseau à 6 JdB)

	Tension		Génération	
JdB	V	θ	P_g	Q_g
	[pu]	[degré]	[pu]	[pu]
1	1.0350	-11.4130	0	0
2	1.0180	-10.9458	0	0
3	0.9776	-10.8382	0	0.0550
4	0.9967	-8.7227	0	0.0500
5	1.1500	-2.5751	0.5010	0.1299
6	1.1000	0	0.9360	0.4268
Pertes actives totales : 8.7099 [MW]				

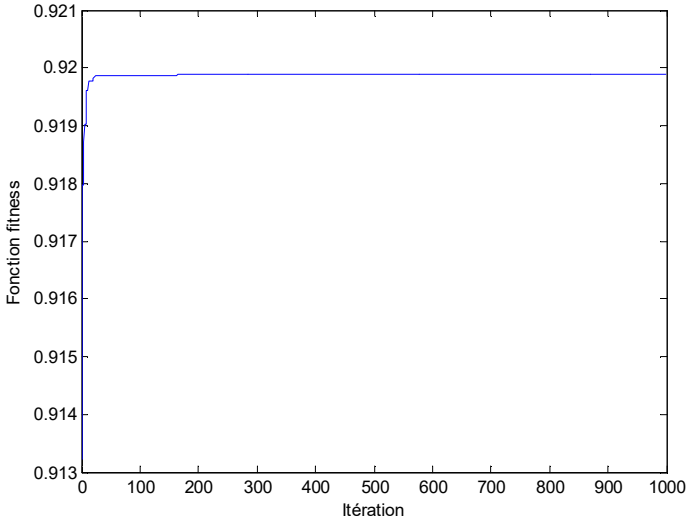


Fig. 4.17 - Meilleure fonction fitness (à maximiser) dans chaque itération du DE (réseau à 6 JdB)

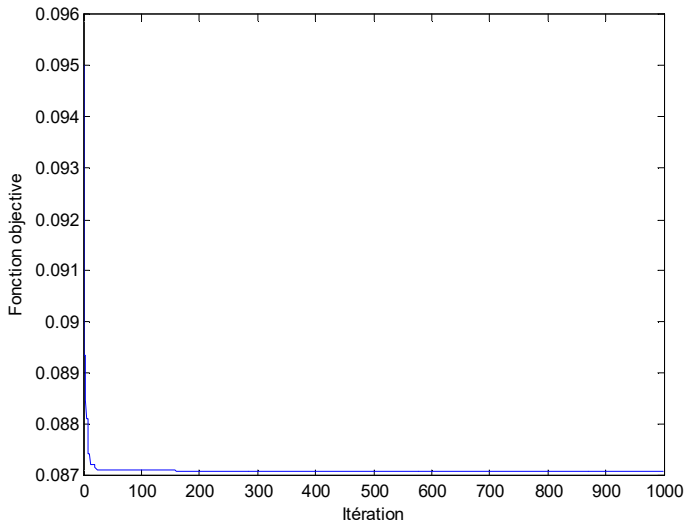


Fig. 4.18 - Meilleure fonction objective (à minimiser) dans chaque itération du DE (réseau à 6 JdB)

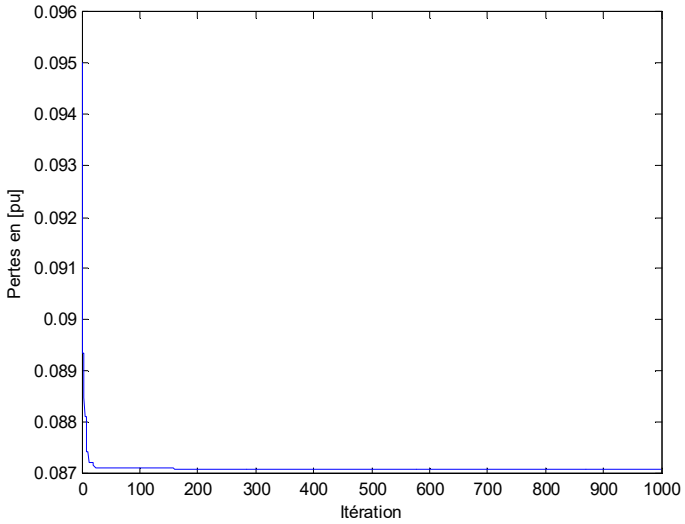


Fig. 4.19 – Pertes dans chaque itération de l'algorithme du DE (réseau à 6 JdB)

4.3.7 – HYBRIDATION PROPSEE ENTRE (DE) ET (SA)

Une formulation hybride entre les algorithmes de l'évolution différentielle et le recuit simulé (HDESA) est proposée pour résoudre le problème de la répartition optimale de la puissance réactive.

DE est une technique d'optimisation globale, puissante, simple et facile à utiliser. Elle est caractérisée par son aptitude à la parallélisation. La différence entre DE et les autres algorithmes évolutionnaires (EA) apparaît dans les phases de mutation et de recombinaison. Dans les techniques EA telles que les GA, la perturbation se produit en fonction d'une quantité aléatoire, tandis que DE utilise des différences pondérées entre les vecteurs de solution pour perturber la population. Lorsque la population perdra complètement sa diversité, elle contiendra des éléments identiques, et elle reste inchangée par la perturbation créée par l'algorithme de l'DE. Pour éviter une convergence prématurée, il est nécessaire de maintenir un niveau raisonnable de diversité dans la population.

De l'autre côté, la méthode SA est caractérisée par sa capacité d'éviter les minimums locaux à cause de sa fonction de probabilité incorporée dans son algorithme pour accepter ou rejeter de nouvelles solutions. Elle n'a pas besoin d'une grande mémoire de stockage.

Dans ce travail, une formulation hybride, nommée HDESA, est proposée. L'hybridation proposée bénéficie de la capacité de recherche globale de DE et de la capacité de recherche locale de SA, en inhibant leurs points faibles. La combinaison de DE et SA fournira un bon équilibre entre exploration et exploitation.

Dans cette étude, nous commençons l'algorithme hybride avec l'algorithme DE décrit dans la (3.3), mais au moment de la sélection de l'individu, nous remplaçons l'étape:

$$x_{i,G+1} = \begin{cases} u_{i,G+1} & si\ f(u_{i,G+1}) < f(x_{i,G}) \\ x_{i,G} & sinon \end{cases} \quad (4.17)$$

par cette expression inspirée de la sélection dans l'algorithme de SA :

$$x_{i,G+1} = \begin{cases} u_{i,G+1} & \text{si } f(u_{i,G+1}) < f(x_{i,G}) \\ \text{sinon} & \begin{cases} u_{i,G+1} & \text{si } r \leq \exp^{-\frac{\Delta E}{T}}, r \in [0,1] \\ x_{i,G} & \text{autrement} \end{cases} \end{cases} \quad (4.18)$$

La baisse de température est calculée ici à chaque génération.

Cette méthode de sélection peut permettre l'exploration de zones n'améliorant pas la valeur de la fonction objective en acceptant des solutions moins bonnes, ce qui peut mener à éviter le point local, résister à une convergence prématurée et augmenter la diversité des solutions de l'évolution différentielle DE. Lorsque cet algorithme modifié de DE est terminé, nous utilisons la solution trouvée comme solution initiale pour l'algorithme SA. Nous utilisons le SA comme un algorithme de recherche locale dans la dernière étape pour améliorer la solution trouvée.

Les résultats de la simulation de l'hybridation proposée sont représentés dans les tableaux (4.20) et (4.21). La figure (4.20) montre la courbe de décroissance de la température. Les figures (4.21) et (4.22) représentent respectivement la meilleure fonction objective (à minimiser) et les pertes obtenues dans chaque itération de l'algorithme de SA.

Commentaire :

L'hybridation a participé à réduire les pertes actives jusqu'à 24.27% par apport au cas initial de l'écoulement de puissance.

Tab. 4.20 - Variables de contrôle obtenues par l'hybridation HDESA (réseau à 6 JdB)

V_5 [pu]	V_6 [pu]	$T_{2\text{-}3}$	$T_{1\text{-}4}$	Q_{g_3} [pu]	Q_{g_4} [pu]
1.1492	1.0999	0.9475	0.9350	0.0550	0.0350

Tab. 4.21 - Etat du réseau après application de l'hybridation HDESA (réseau à 6 JdB)

	Tension		Génération	
JdB	V [pu]	θ [degré]	P_g [pu]	Q_g [pu]
1	1.0425	-11.4010	0	0
2	1.0042	-10.9268	0	0
3	0.9802	-10.8542	0	0.0550
4	0.9936	-8.7287	0	0.0350
5	1.1492	-2.6009	0.5010	0.1444
6	1.0999	0	0.9369	0.4294
Pertes actives totales : 8.7990 [MW]				

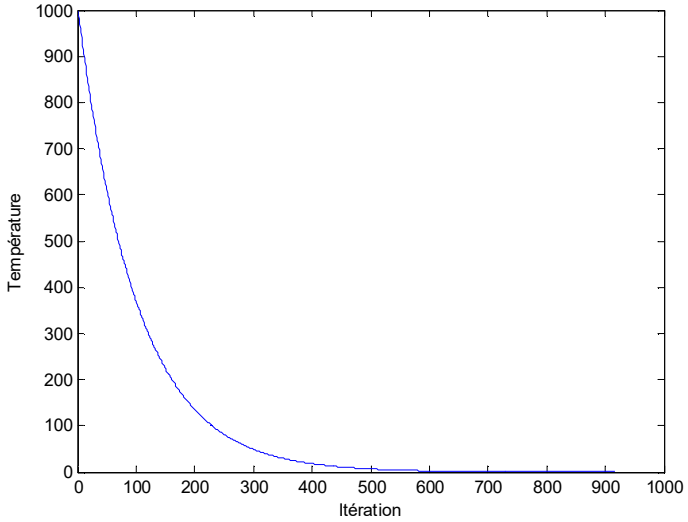


Fig. 4.20 – Courbe de décroissance de la température dans l'algorithme hybride HDESA (réseau à 6 JdB)

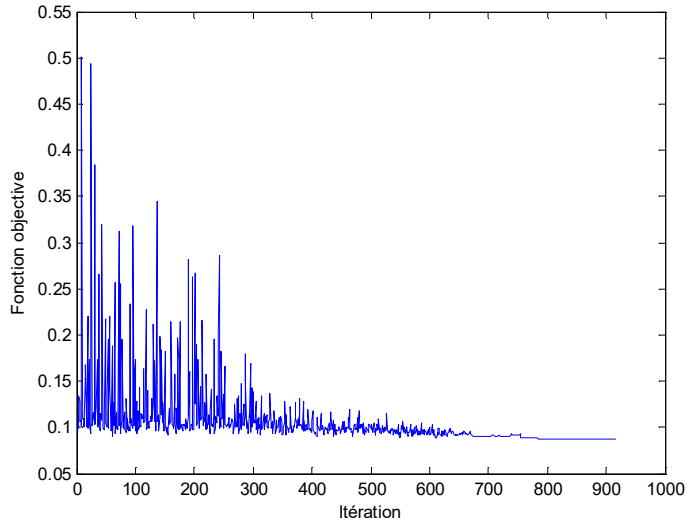


Fig. 4.21 – Meilleure fonction objective (à minimiser) dans chaque itération de l'algorithme hybride HDESA (réseau à 6 JdB)

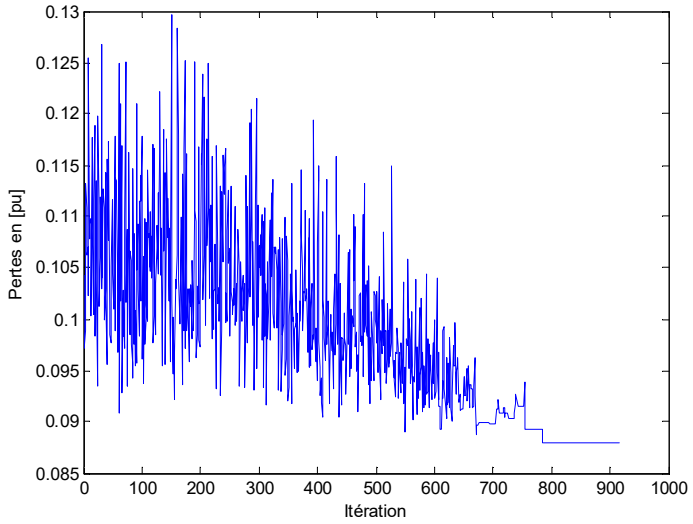


Fig. 4.22 – Pertes dans chaque itération de l'algorithme hybride HDESA (réseau à 6 JdB)

4.3.8 - COMPARAISON DES RESULTATS (RESEAU À 6 JDB)

Les résultats de nos techniques appliquées sur le réseau à 6 JdB « Ward-Hale 6 bus system » sont comparés dans le tableau (4.22) avec :

- les résultats de DE obtenus dans [49].
- les résultats de LP présentés dans [8].

À partir du tableau (4.22) on peut remarquer que :

- Comparés à l'utilisation de la méthode IGA ou PSO seules, les résultats obtenus de l'algorithme d'hybridation HPSOIGA sont les meilleurs.
- On peut remarquer aussi que les résultats obtenus de l'algorithme DE et de l'hybridation HPSOIGA sont proches. Ce qui montre l'efficacité de l'algorithme de l'évolution différentielle appliqué sur des réseaux électriques de petites tailles.

Tab. 4.22 - Variables de contrôle et pertes obtenues par l'exécution de IGA, PSO, HPSOIGA, DE, SA, HDESA et par DE de la référence [49] et LP de [8] (réseau à 6 JdB)

Variables de contrôle	Initial	IGA	PSO	Hybridation (HPSOIGA)	DE	SA	Hybridation (HDESA)	DE [49]	LP [8]
V_5 [pu]	1.1	1.1477	1.1500	1.1500	1.1500	1.1434	1.1492	1.150	1.15
V_6 [pu]	1.05	1.0970	1.1000	1.1000	1.1000	1.0974	1.0999	1.089	1.09
$T_{2\text{-}3}$ [pu]	1.025	0.9350	0.9350	0.9350	0.9284	0.9475	0.9475	0.972	0.96
$T_{1\text{-}4}$ [pu]	1.100	0.9475	0.9350	0.9475	0.9451	0.9850	0.9350	0.991	0.98
Q_{g_3} [pu]	0	0.0495	0.0550	0.0550	0.0550	0.0550	0.0550	0.055	0.05
Q_{g_4} [pu]	0	0.0500	0.050	0.0500	0.0500	0.0350	0.0350	0.055	0.055
Pertes totales [MW]	11.6197	8.7933	8.7169	8.7114	8.7099	8.9277	8.7990	8.890	8.93

4.4 – APPLICATION DES METHODES PROPOSEES SUR LE RESEAU STANDARD IEEE 30 BUS

4.4.1 - CALCUL DE L'ECOULEMENT DE PUISSANCE

L'étude de l'écoulement de puissance par l'application de la méthode de Newton-Raphson a donné les résultats présentés dans le tableau (4.23) où la valeur des pertes de transmission totales est 58223 [MW]. Les tensions hors limite sont V_{19}, V_{20}, V_{21}, V_{22}, V_{23}, V_{24}, V_{25}, V_{26}, V_{27}, V_{29}, et V_{30}. Pour cela, on est obligé d'ajuster les moyens de contrôle pour avoir un plan optimal de tension tout en minimisant les pertes totales de la puissance active dans le réseau.

Tab. 4.23 - Résultats de l'écoulement de puissance (réseau à 30 JdB)

	Tension		Charge		Génération	
JdB	V	θ	P_d	Q_d	P_g	Q_g
	[pu]	[degré]	[pu]	[pu]	[pu]	[pu]
1	1.0500	0	0	0	0.9922	-0.01535
2	1.0400	-1.7623	0.217	0.127	0.8000	0.15644
3	1.0279	-3.9323	0.024	0.012	0	0
4	1.0222	-4.6963	0.076	0.016	0	0
5	1.0100	-6.4824	0.942	0.190	0.5000	0.16406
6	1.0166	-5.4355	0	0	0	0
7	1.0059	-6.3969	0.228	0.109	0	0
8	1.0100	-5.6272	0.300	0.300	0.2000	0.13537
9	0.9755	-7.0162	0	0	0	0
10	0.9547	-9.1959	0.058	0.020	0	0
11	1.0500	-4.6886	0	0	0.2000	0.38002
12	0.9976	-8.7884	0.112	0.075	0	0
13	1.0500	-7.2567	0	0	0.2000	0.39545
14	0.9773	-9.7952	0.062	0.016	0	0
15	0.9680	-9.7932	0.082	0.025	0	0
16	0.9718	-9.2538	0.035	0.018	0	0
17	0.9540	-9.4522	0.090	0.058	0	0
18	0.9501	-10.3964	0.032	0.009	0	0
19	0.9429	-10.5331	0.095	0.034	0	0
20	0.9450	-10.2636	0.022	0.007	0	0
21	0.9408	-9.7516	0.175	0.112	0	0
22	0.9413	-9.7419	0	0	0	0
23	0.9467	-10.1714	0.032	0.016	0	0
24	0.9274	-10.2804	0.087	0.067	0	0
25	0.9204	-10.3073	0	0	0	0
26	0.9008	-10.8220	0.035	0.023	0	0
27	0.9257	-10.0102	0	0	0	0
28	1.0116	-5.8711	0	0	0	0
29	0.9035	-11.5199	0.024	0.009	0	0
30	0.8907	-12.6115	0.106	0.019	0	0
Pertes actives totales : 5.8223 [MW]						

4.4.2 - APPLICATION DE L'ALGORITHME GÉNÉTIQUE AMELIORÉ (IGA)

Une représentation mixte (continue/discrète) pour les variables de contrôle est utilisée. Les tensions des générateurs sont représentées en valeurs continues. Par contre, les batteries de condensateurs shunts et les transformateurs réglables sont représentés en valeurs discrètes.

Les paramètres utilisés de l'algorithme génétique amélioré sont :

- Nombre maximal de générations : 500
- Taille de la population : 100
- Probabilité de croisement : 0.9
- Une probabilité de mutation adaptative est utilisée ici selon l'équation (4.6).

La programmation de l'algorithme génétique a donné, après 25 générations, les résultats présentés sur les tableaux (4.24) et (4.25). Les figures (4.23), (4.24) et (4.25) représentent respectivement la meilleure fitness (à maximiser), la meilleure fonction objective (à minimiser) et les pertes dans chaque génération.

Commentaire :

Les résultats de simulation montrent clairement que toutes les variables de contrôle sont ajustées. On peut clairement percevoir une diminution importante de 11.87% des pertes actives totales, allant de 5.8223 [MW] dans le cas de l'écoulement de puissance à 5.1311 [MW] après application de l'algorithme génétique amélioré. Le plan de tension est amélioré et toutes les contraintes sont respectées.

Tab. 4.24 - Variables de contrôle obtenues par l'application de l'IGA (réseau à 30 JdB)

V_1 [pu]	V_2 [pu]	V_5 [pu]	V_8 [pu]	V_{11} [pu]	V_{13} [pu]	$T_{6\text{-}9}$	$T_{6\text{-}10}$	$T_{4\text{-}12}$	$T_{28\text{-}27}$
1.0706	1.0574	1.0223	1.0311	0.9903	1.0688	1.0625	0.9063	1.0438	0.9938

Q_{c10} [pu]	Q_{c12} [pu]	Q_{c15} [pu]	Q_{c17} [pu]	Q_{c20} [pu]	Q_{c21} [pu]	Q_{c23} [pu]	Q_{c24} [pu]	Q_{c29} [pu]
0.0355	0.016	0.045	0.0415	0.035	0.039	0.0075	0.001	0.029

Tab. 4.25 - Etats du réseau après application de l'IGA (réseau à 30 JdB)

	Tension		Génération	
JdB	V	θ	P_g	Q_g
	[pu]	[degré]	[pu]	[pu]
1	1.0706	0	0.985311	0.057750
2	1.0574	-1.6281	0.800000	0.129805
3	1.0470	-3.7251	0	0
4	1.0410	-4.4456	0	0
5	1.0223	-6.1331	0.500000	0.123362
6	1.0309	-5.1485	0	0
7	1.0196	-6.0723	0	0
8	1.0311	-5.4480	0.200000	0.324515
9	0.9942	-6.5806	0	0
10	1.0101	-8.5635	0	0.035500
11	0.9903	-4.1589	0.200000	-0.014371
12	1.0290	-7.9289	0	0.016000
13	1.0688	-6.4700	0.200000	0.306420
14	1.0155	-8.9072	0	0
15	1.0123	-9.0811	0	0.045000
16	1.0151	-8.5160	0	0
17	1.0086	-8.8226	0	0.041500
18	1.0016	-9.6979	0	0
19	0.9985	-9.8631	0	0
20	1.0024	-9.6461	0	0.035000
21	0.9998	-9.1117	0	0.039000
22	1.0002	-9.0958	0	0
23	0.9998	-9.4785	0	0.007500
24	0.9899	-9.5898	0	0.001000
25	0.9980	-9.6789	0	0
26	0.9800	-10.1152	0	0
27	1.0120	-9.4492	0	0
28	1.0272	-5.6318	0	0
29	1.0006	-10.9603	0	0.029000
30	0.9853	-11.7430	0	0

Pertes actives totales dans le réseau : 5.1311 [MW]

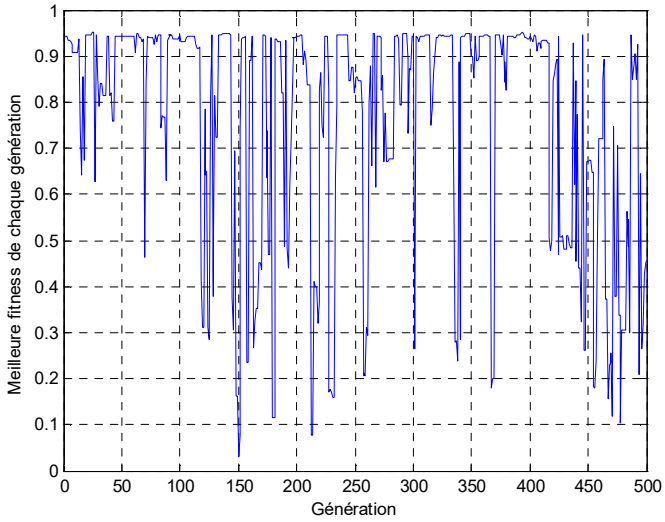


Fig. 4.23 - Meilleure fonction fitness dans chaque génération de l'IGA (réseau à 30 JdB)

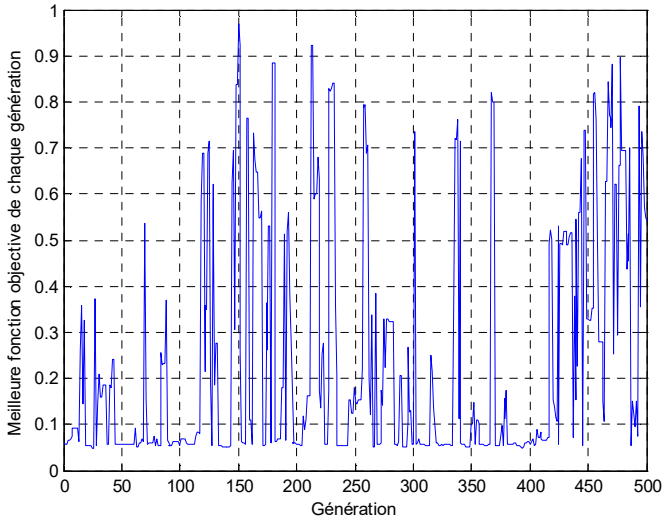


Fig. 4.24 - Meilleure fonction objective dans chaque génération de l'IGA (réseau à 30 JdB)

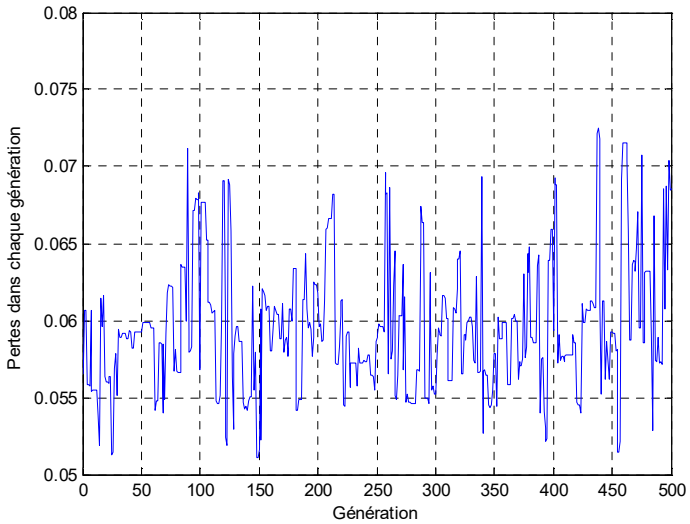


Fig. 4.25 - Pertes dans chaque génération de l'IGA (réseau à 30 JdB)

4.4.3 - APPLICATION DE L'OPTIMISATION PAR ESSAIMS PARTICULAIRES PSO

PSO a été développé ici pour traiter le problème du contrôle de tension/puissance réactive avec variables continues et discrètes. Les tensions des générateurs sont représentées en valeurs continues. Par contre, les batteries de condensateurs shunts et les transformateurs réglables sont représentés en valeures discrètes.

Chaque transformateur contient 32 gradins. Ainsi, la modification de la valeur du rapport de transformation d'un transformateur *i* sera :

$$T_i = 0.9 + NT_i \times 0.625\ \% \quad \text{avec } NT_i = 0, 1, \ldots, 32 \tag{4.19}$$

Les limites des rapports de transformation sont $[0.9, 0.9+32\times0.625\%]$. Chaque batterie de condensateurs shunts a 100 variations possibles.

Les paramètres du PSO utilisés ici sont les suivants :

- Nombre maximal de générations : 500
- Nombre de particules : 100
- C_1 : 2
- C_2 : 2
- Poids d'inertie initial w : 0.9
- w_{min} : 0.8
- w_{max} : 1.2
- Vitesse Initial v : 0.05
- Vitesse maximal v_{max} : 0.5

La programmation du PSO a donné après 500 itérations, les résultats présentés sur les tableaux (4.26) et (4.27). Les figures (4.26), (4.27) et (4.28) représentent les caractéristiques de convergence du PSO.

Tab. 4.26 - Variables de contrôle obtenues par l'application du PSO (réseau à 30 JdB)

V_1 [pu]	V_2 [pu]	V_5 [pu]	V_8 [pu]	V_{11} [pu]	V_{13} [pu]	$T_{6\text{-}9}$	$T_{6\text{-}10}$	$T_{4\text{-}12}$	$T_{28\text{-}27}$
1.0699	1.0615	1.0405	1.0406	1.0654	1.0693	1.0438	0.900	0.9938	0.9813
Q_{c10} [pu]	Q_{c12} [pu]	Q_{c15} [pu]	Q_{c17} [pu]	Q_{c20} [pu]	Q_{c21} [pu]	Q_{c23} [pu]	Q_{c24} [pu]	Q_{c29} [pu]	
0.023	0	0	0.0055	0.05	0.05	0	0.05	0.05	

Tab. 4.27 - Etats du réseau après application du PSO (réseau à 30 JdB)

	Tension		Génération	
JdB	V [pu]	θ [degré]	P_g [pu]	Q_g [pu]
1	1.0699	0	0.983145	-0.052702
2	1.0615	-1.7092	0.800000	0.116200
3	1.0500	-3.7472	0	0
4	1.0448	-4.4735	0	0
5	1.0405	-6.3444	0.500000	0.238461
6	1.0405	-5.2612	0	0
7	1.0328	-6.2074	0	0
8	1.0406	-5.5550	0.200000	0.317662
9	1.0369	-6.6468	0	0
10	1.0442	-8.5108	0	0.023000
11	1.0654	-4.4887	0.200000	0.149593
12	1.0498	-7.6391	0	0
13	1.0693	-6.2098	0.200000	0.151281
14	1.0370	-8.5726	0	0
15	1.0343	-8.7430	0	0
16	1.0403	-8.2921	0	0
17	1.0380	-8.6622	0	0.005500
18	1.0292	-9.4453	0	0
19	1.0293	-9.6702	0	0
20	1.0347	-9.5029	0	0.050000
21	1.0352	-9.0520	0	0.050000
22	1.0356	-9.0400	0	0
23	1.0277	-9.2488	0	0
24	1.0273	-9.5759	0	0.050000
25	1.0340	-9.6092	0	0
26	1.0166	-10.0153	0	0
27	1.0466	-9.3664	0	0
28	1.0380	-5.7553	0	0
29	1.0417	-10.9468	0	0.050000
30	1.0244	-11.5968	0	0
Pertes actives totales dans le réseau : 4.9145 [MW]				

Commentaire :

Comparés avec ceux obtenus par l'application de l'IGA, les résultats de SPO sont les meilleurs. Les pertes actives totales dans le réseau sont réduites de 5.1311 [MW] après application d'IGA à 4.9145 [MW] après application du PSO, avec une diminution de 15.59% par apport au cas initial de l'écoulement de puissance.

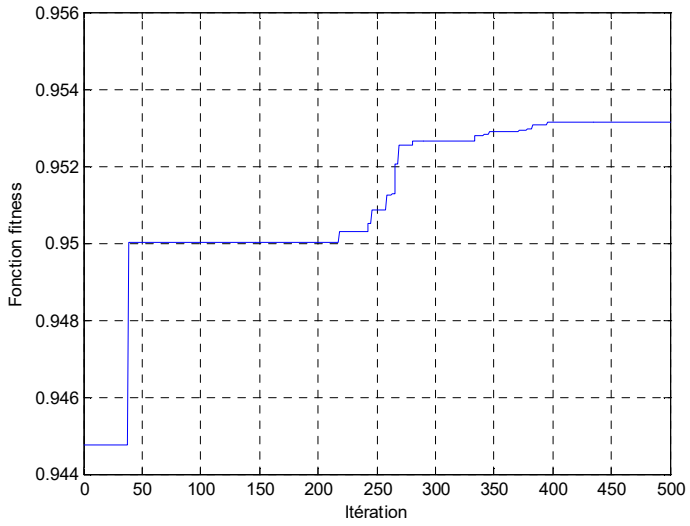


Fig. 4.26 - Meilleure fonction d'évaluation dans chaque itération du PSO (réseau à 30 JdB)

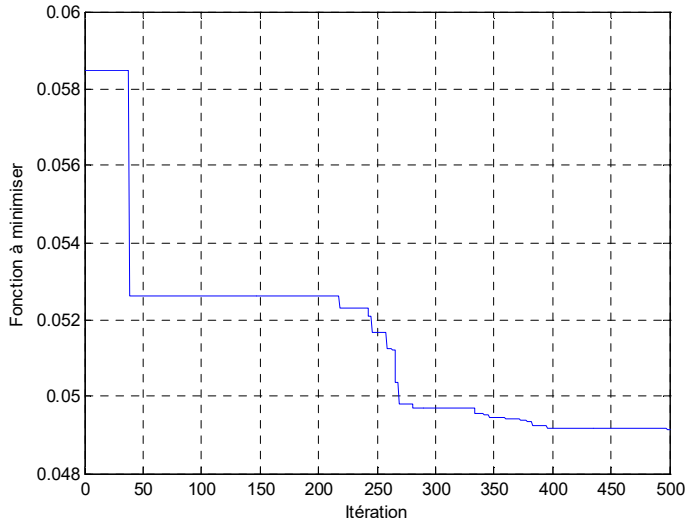


Fig. 4.27 - Meilleure fonction objective dans chaque itération du PSO (réseau à 30 JdB)

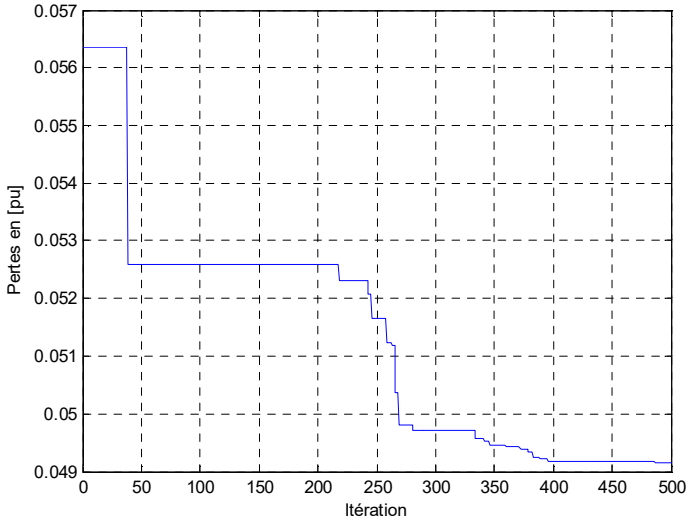


Fig. 4.28 - Pertes dans chaque itération du PSO (réseau à 30 JdB)

4.4.4 - HYBRIDATION ENTRE PSO ET IGA

Pour améliorer plus les résultats, nous avons appliqué une formulation hybride entre PSO et IGA nommée HPSOIGA. La méthode du PSO a été employée pour générer une population initiale injectée et utilisée dans le programme de l'IGA. L'hybridation a donné après la $6^{ième}$ génération de l'algorithme de l'hybridation les résultats présentés dans les tableaux (4.28) et (4.29). Les figures (4.29), (4.30) et (4.31) représentent respectivement la meilleure fitness (fonction d'évaluation à maximiser), la meilleure fonction objective (à minimiser) et les pertes dans chaque génération de l'algorithme d'hybridation entre l'IGA et le PSO.

Tab. 4.28 - Variables de contrôle obtenues après l'hybridation HPSOIGA (réseau à 30 JdB)

V_1 [pu]	V_2 [pu]	V_5 [pu]	V_8 [pu]	V_{11} [pu]	V_{13} [pu]	T_{6-9}	T_{6-10}	T_{4-12}	T_{28-27}
1.0705	1.0609	1.0402	1.0393	1.0663	1.0676	1.0313	0.9000	1.0000	0.9813
Q_{c10} [pu]	Q_{c12} [pu]	Q_{c15} [pu]	Q_{c17} [pu]	Q_{c20} [pu]	Q_{c21} [pu]	Q_{c23} [pu]	Q_{c24} [pu]	Q_{c29} [pu]	
0.0205	0	0	0.0095	0.05	0.05	0.024	0.05	0.0265	

Commentaire :

Nous pouvons voir que toutes les contraintes sont respectées et les pertes actives totales dans le réseau sont réduites à 4.8960 [MW], avec une diminution de 15.90% par apport au cas initial de l'écoulement de puissance. Les résultats sur le réseau IEEE 30-bus montrent les avantages de la méthode hybride par rapport à l'utilisation de la méthode d'IGA ou de PSO seule.

Tab. 4.29 - Etats du réseau après l'hybridation HPSOIGA (réseau à 30 JdB)

	Tension		Génération	
JdB	V [pu]	θ [degré]	P_g [pu]	Q_g [pu]
1	1.0705	0	0.98296	-0.026708
2	1.0609	-1.6880	0.80000	0.095039
3	1.0499	-3.7332	0	0
4	1.0445	-4.4563	0	0
5	1.0402	-6.3349	0.50000	0.245107
6	1.0392	-5.2344	0	0
7	1.0319	-6.1879	0	0
8	1.0393	-5.5266	0.20000	0.318597
9	1.0410	-6.6470	0	0
10	1.0463	-8.5240	0	0.020500
11	1.0663	-4.4993	0.20000	0.133587
12	1.0485	-7.6086	0	0
13	1.0676	-6.1753	0.20000	0.148362
14	1.0367	-8.5554	0	0
15	1.0348	-8.7581	0	0
16	1.0406	-8.2860	0	0
17	1.0398	-8.6746	0	0.009500
18	1.0302	-9.4581	0	0
19	1.0307	-9.6818	0	0
20	1.0363	-9.5147	0	0.050000
21	1.0372	-9.0671	0	0.050000
22	1.0376	-9.0563	0	0
23	1.0311	-9.3417	0	0.024000
24	1.0284	-9.6020	0	0.050000
25	1.0307	-9.5668	0	0
26	1.0132	-9.9754	0	0
27	1.0407	-9.2853	0	0
28	1.0363	-5.7135	0	0
29	1.0289	-10.6926	0	0.026500
30	1.0144	-11.4406	0	0

Pertes actives totales dans le réseau : 4.8960 [MW]

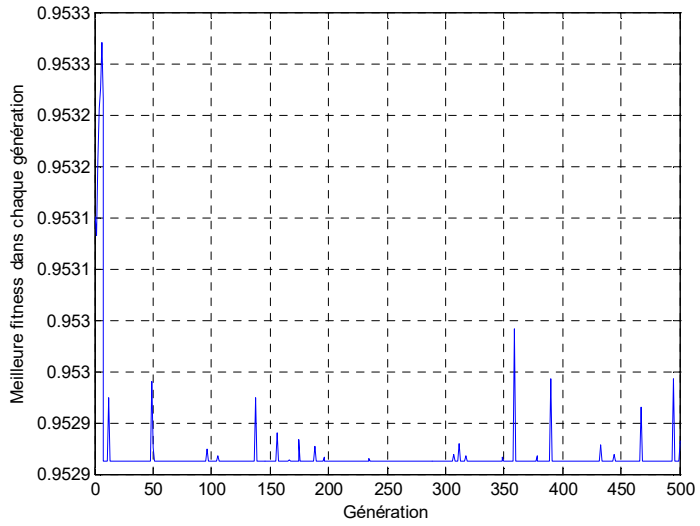


Fig. 4.29 - Meilleure fitness (fonction d'aptitude à maximiser) dans chaque génération de l'hybridation HPSOIGA (réseau à 30 JdB)

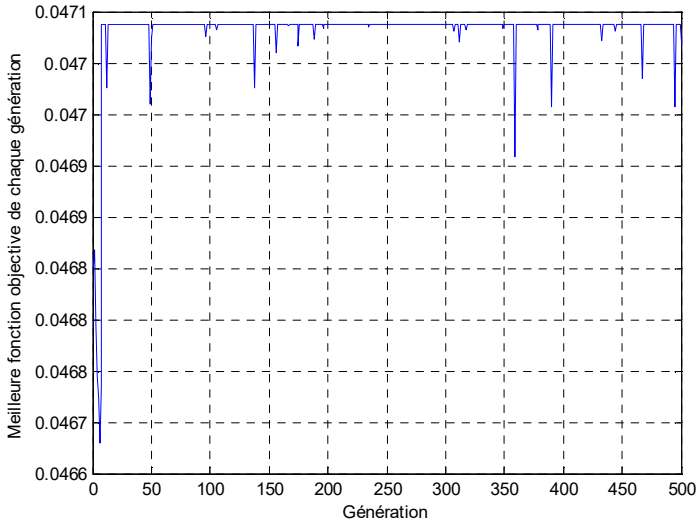


Fig. 4.30 - Meilleure fonction objective (à minimiser) dans chaque génération de l'hybridation HPSOIGA (réseau à 30 JdB)

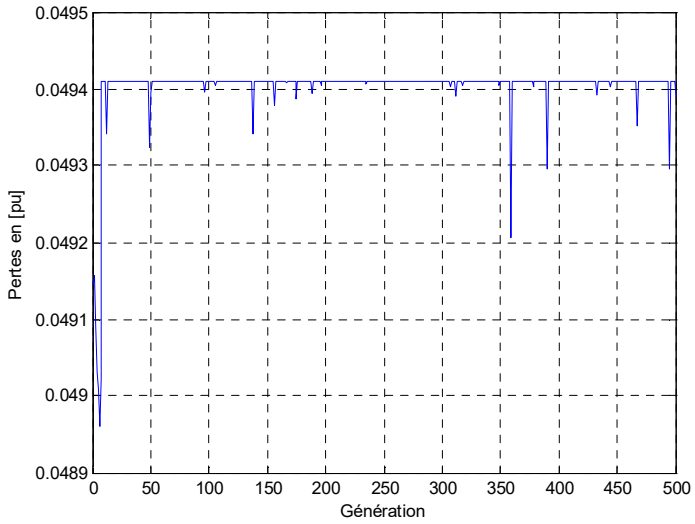


Fig. 4.31 - Pertes en [pu] dans chaque génération de hybridation HPSOIGA (réseau à 30 JdB)

4.4.5 – APPLICATION DU RECUIT SIMULÉ (SA)

Les paramètres utilisés ici pour le recuit simulé SA sont :

- Température initiale T_{max} : 1000
- Température finale T_{min} : 0.1
- Constante de la baisse de température α : 0.99
- Maximum d'essais à chaque température *Trymax* : 50

Les résultats de SA sont présentés dans les tableaux (4.30) et (4.31). Les figures (4.32), (4.33) et (4.34) représentent respectivement la courbe de décroissance de la température, la meilleure fonction objective (à minimiser) et les pertes obtenues dans chaque itération de l'algorithme du recuit simulé.

Commentaire :

Il est clairement démontré que les pertes sont diminuées de 10.59%, allant de 5,8223 [MW] dans le cas de l'écoulement de puissance à 5,2055 [MW] après application de l'algorithme de SA. Le profil de tension est amélioré et toutes les contraintes sont respectées.

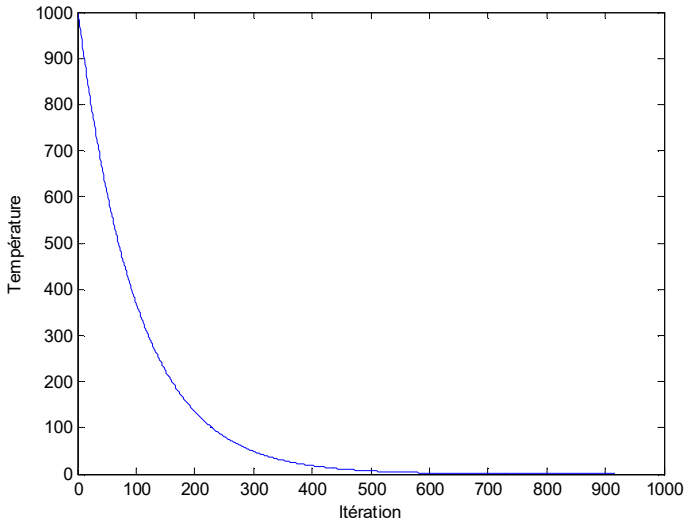


Fig. 4.32 - Courbe de décroissance de la température (réseau à 30 JdB)

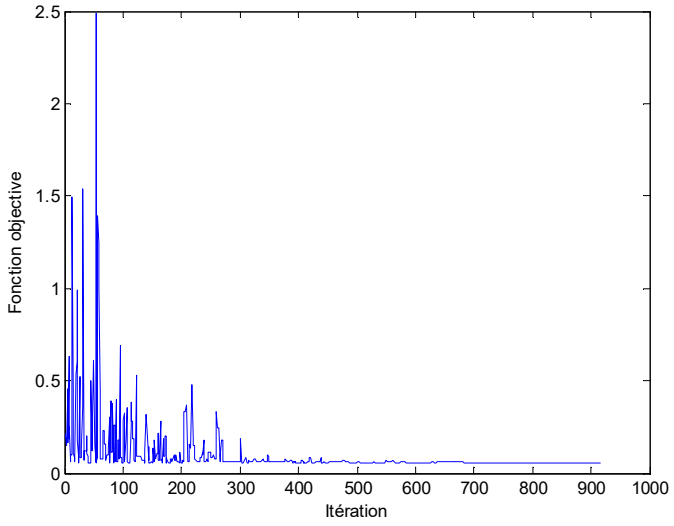


Fig. 4.33 - Meilleure fonction objective (à minimiser) dans chaque itération du SA (réseau 30 JdB)

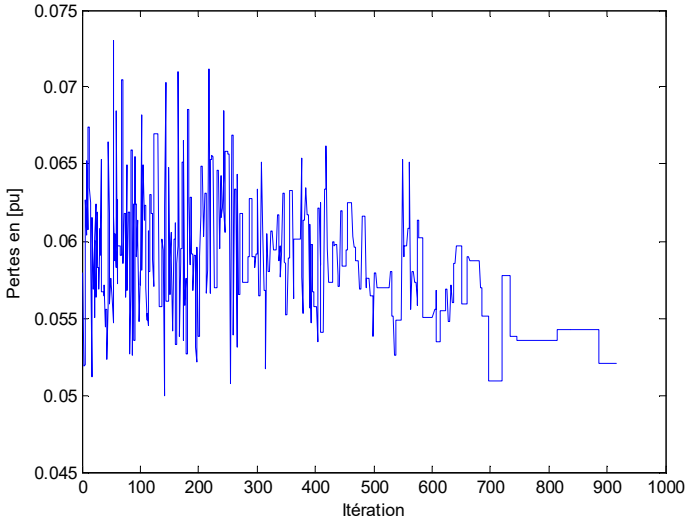


Fig. 4.34 - Pertes dans chaque itération du SA (réseau à 30 JdB)

4.4.6 - APPLICATION DE L'EVOLUTION DIFFERENTIELLE

Les différents paramètres de l'ED sont déterminés après plusieurs essais d'exécution de simulation. Les paramètres utilisés pour l'algorithme DE sont:

- Nombre maximal de générations : 100
- Taille de la population : 500
- Poids differentiel F : 1
- Coefficient de croisement CR : 0.8

Les meilleurs résultats sont atteints après 64 itérations. Les caractéristiques de convergence de l'algorithme DE sont représentées sur les figures (4.35), (4.36) et (4.37). Les résultats du DE sont donnés dans les tableaux (4.30) et (4.31).

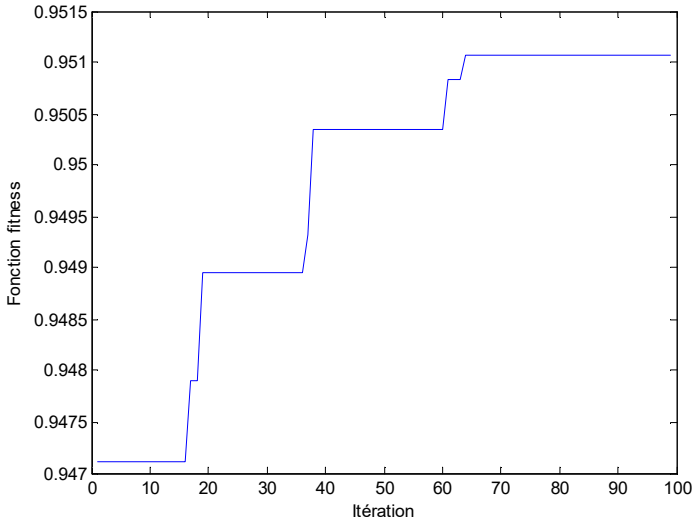


Fig. 4.35 - Meilleure fonction fitness (à maximiser) dans chaque itération du DE (réseau à 30 JdB)

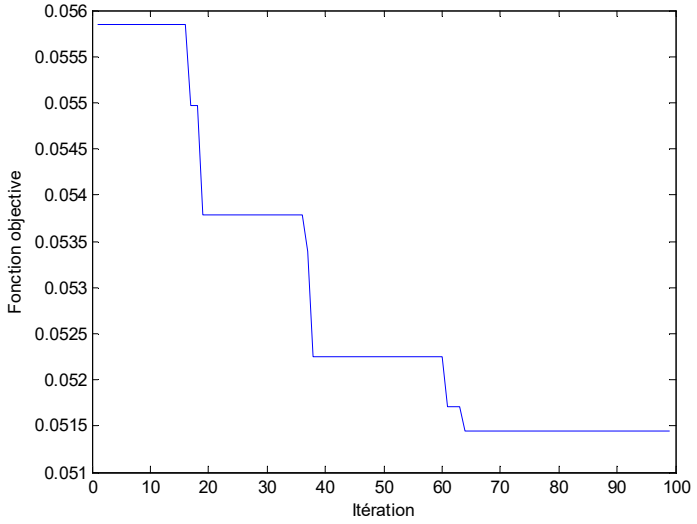


Fig. 4.36 - Meilleure fonction objective (à minimiser) dans chaque itération du DE (réseau à 30 JdB)

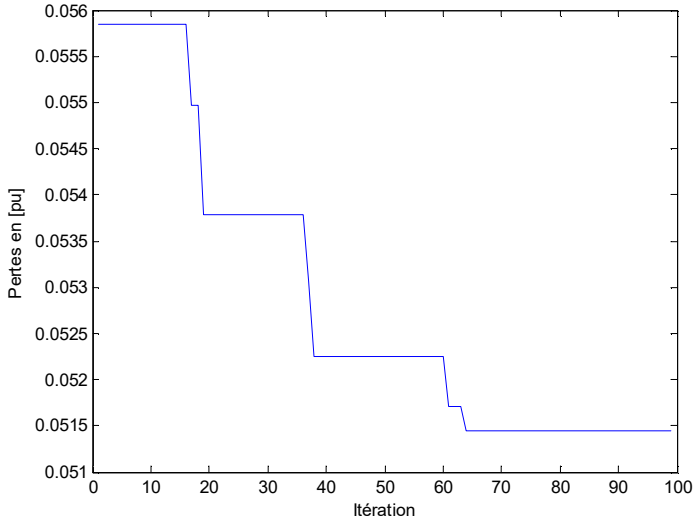


Fig. 4.37 – Pertes dans chaque itération du DE (réseau à 30 JdB)

Commentaire :

Par rapport à l'application de SA, les résultats obtenus par DE sont les meilleurs. Les pertes sont diminuées de 11.65% par rapport au cas initial de l'écoulement de puissance. Elles sont réduites de 5,2055 [MW] dans le cas du SA à 5,1439 [MW] après utilisation du DE.

4.4.7 - HYBRIDATION PROPOSEE ENTRE DE ET SA

Les caractéristiques de convergence de l'algorithme hybride proposé HDESA sont représentées sur les figures (4.38), (4.39) et (4.40). Les résultats de la simulation de l'hybridation proposée sont représentés dans les tableaux (4.30) et (4.31).

Commentaire :

La perte totale est considérablement réduite à 5.1298 [MW] avec une diminution de 11.89% par rapport au cas initial de l'écoulement de puissance. Les contraintes sont respectées. L'utilisation de l'hybridation HDESA a donné de bons résultats par rapport à l'application du DE ou SA seule.

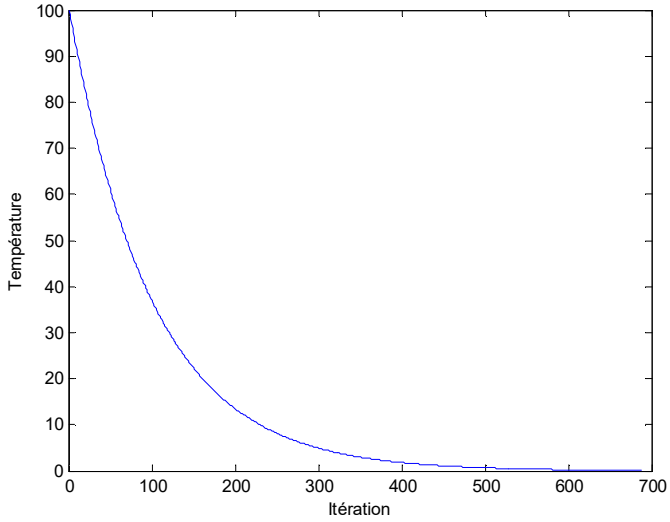


Fig. 4.38 – Courbe de décroissance de la température dans l'algorithme HDESA (réseau à 30 JdB)

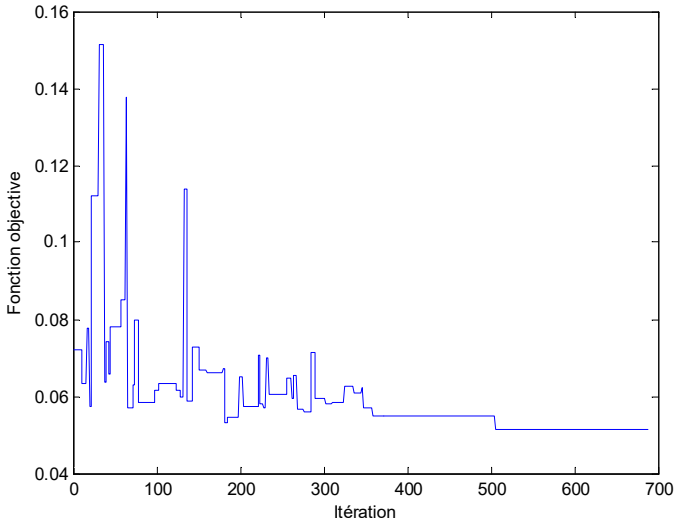


Fig. 4.39 – Meilleure fonction objective dans chaque itération de HDESA (réseau à 30 JdB)

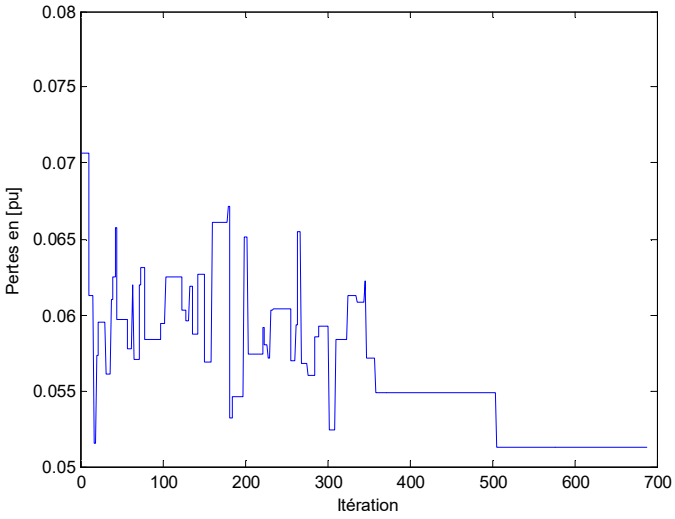


Fig. 4.40 – Pertes dans chaque itération de l'algorithme hybride HDESA (réseau à 30 JdB)

Tab. 4.30 - Variables de contrôle et pertes obtenues du cas initial de l'écoulement de puissance et de l'exécution de SA, DE et HDESA (réseau à 30 JdB)

Variable de Contrôle	Initial	SA	DE	Hybridation (HDESA)
V_1 [pu]	1.05	1.0590	1.0593	1.0744
V_2 [pu]	1.04	1.0545	1.0532	1.0724
V_5 [pu]	1.01	1.0102	1.0279	1.0486
V_8 [pu]	1.01	1.0348	1.0240	1.0498
V_{11} [pu]	1.05	1.0712	1.1000	1.0692
V_{13} [pu]	1.05	1.0717	1.0579	1.0038
$T_{6\text{-}9}$	1.078	0.9688	1.0312	1.0375
$T_{6\text{-}10}$	1.069	0.9750	0.9875	0.9938
$T_{4\text{-}12}$	1.032	1.0313	0.9812	0.9750
$T_{28\text{-}27}$	1.068	0.9750	1.0187	1.0438
Q_{c10} [pu]	0	0.0110	0.0040	0.0330
Q_{c12}[pu]	0	0.0390	0.0500	0.0465
Q_{c15} [pu]	0	0.0140	0	0.0350
Q_{c17} [pu]	0	0.0425	0.0250	0.0335
Q_{c20} [pu]	0	0.0175	0.0020	0.0180
Q_{c21} [pu]	0	0.0090	0.0265	0.0070
Q_{c23} [pu]	0	0.0170	0.0275	0.0170
Q_{c24} [pu]	0	0.0120	0.0395	0.0155
Q_{c29} [pu]	0	0.0335	0.0355	0.0110
Pertes actives totales [MW]	5.8223	5.2055	5.1439	5.1298

Tab. 4.31 - Variables d'état obtenues à partir de l'exécution de SA, DE, HDESA et les paramètres initiaux (réseau à 30 JdB)

Variables d'état	Initial	SA	DE	Hybridation (HDESA)
V_3 [pu]	1.0279	1.0437	1.0386	1.0500
V_4 [pu]	1.0222	1.0394	1.0332	1.0457
V_6 [pu]	1.0166	1.0312	1.0298	1.0471
V_7 [pu]	1.0059	1.0147	1.0212	1.0401
V_9 [pu]	0.9755	1.0500	1.0355	1.0253
V_{10}[pu]	0.9547	1.0345	1.0219	1.0117
V_{12}[pu]	0.9976	1.0399	1.0416	1.0194
V_{14}[pu]	0.9773	1.0271	1.0265	1.0065
V_{15}[pu]	0.9680	1.0244	1.0215	1.0037
V_{16}[pu]	0.9718	1.0317	1.0268	1.0098
V_{17}[pu]	0.9540	1.0307	1.0194	1.0076
V_{18}[pu]	0.9501	1.0169	1.0094	0.9953
V_{19}[pu]	0.9429	1.0157	1.0055	0.9935
V_{20}[pu]	0.9450	1.0205	1.0089	0.9982
V_{21}[pu]	0.9408	1.0226	1.0112	0.9986
V_{22}[pu]	0.9413	1.0232	1.0117	0.9988
V_{23}[pu]	0.9467	1.0175	1.0125	0.9933
V_{24}[pu]	0.9274	1.0123	1.0017	0.9836
V_{25}[pu]	0.9204	1.0209	0.9966	0.9787
V_{26}[pu]	0.9008	1.0033	0.9786	0.9603
V_{27}[pu]	0.9257	1.0349	1.0023	0.9847
V_{28}[pu]	1.0116	1.0284	1.0268	1.0449
V_{29}[pu]	0.9035	1.0251	0.9927	0.9674
V_{30}[pu]	0.8907	1.0097	0.9764	0.9540
Q_{g1} [pu]	-0.0153	-0.1499	-0.0917	-0.1559
Q_{g2} [pu]	0.1564	0.3135	0.2165	0.3396
Q_{g5} [pu]	0.1640	0.0149	0.2078	0.2288
Q_{g8} [pu]	0.1353	0.4204	0.1448	0.3935
Q_{g11}[pu]	0.3800	0.1074	0.3450	0.2295
Q_{g13}[pu]	0.3954	0.2461	0.1256	-0.1090

4.4.8 - COMPARAISON DES RESULTATS (RESEAU À 30 JDB)

Les résultats de nos algorithmes appliqués sur le réseau à 30 JdB sont comparés dans le tableau (4.32) avec :

- les résultats de la répartition de la puissance réactive (RPD) multi-objectifs utilisant l'algorithme PSO classique de [94].
- les résultats d'une nouvelle stratégie multi-objectifs (cas IV) basée sur la technique de l'optimisation de l'accouplement d'abeille (Chaotic Parallel Vector Evaluated Interactive Honey Bee Mating Optimization CPVEIHBMO) [148].

À partir du tableau (4.32), on peut remarquer l'efficacité de nos algorithmes hybrides proposés :

- Les résultats obtenus de l'algorithme d'hybridation HDESA sont meilleurs que ceux obtenus de l'application de la méthode du DE ou SA seule.
- L'algorithme de l'optimisation par essaim particulaire PSO appliqué sur un réseau électrique à 30 JdB a montré son efficacité à résoudre le problème de la répartition optimale de la puissance réactive (une diminution de 15.59% par apport au cas initial de l'écoulement de puissance).
- Comparés à l'utilisation de la méthode d'IGA ou PSO seule, les résultats obtenus de l'algorithme d'hybridation HPSOIGA sont les meilleurs (une diminution des pertes de 15.90% par rapport au cas initial de l'écoulement de puissance).

Tab. 4.32 - Variables de Contrôle et pertes obtenues de l'exécution de IGA, PSO, HPSOIGA, DE, SA, HDESA, PSO de [94] et CPVEIHBMO de [148] (réseau à 30 JdB)

Variable de Contrôle	Initial	IGA	PSO	HPSOIGA	SA	DE	HDESA	PSO [94]	CPVEIHBMO [148]
V_1 [pu]	1.05	1.0706	1.0699	1.0705	1.0590	1.0593	1.0744	1.04189	1.025431
V_2 [pu]	1.04	1.0574	1.0615	1.0609	1.0545	1.0532	1.0724	1.03174	1.035234
V_5 [pu]	1.01	1.0223	1.0405	1.0402	1.0102	1.0279	1.0486	1.00817	1.056352
V_8 [pu]	1.01	1.0311	1.0406	1.0393	1.0348	1.0240	1.0498	1.00711	1.027345
V_{11} [pu]	1.05	0.9903	1.0654	1.0663	1.0712	1.1000	1.0692	1.05427	1.028768
V_{13} [pu]	1.05	1.0688	1.0693	1.0676	1.0717	1.0579	1.0038	1.00921	1.075644
$T_{6\text{-}9}$	1.078	1.0625	1.0438	1.0313	0.9688	1.0312	1.0375	1.08161	0.998322
$T_{6\text{-}10}$	1.069	0.9063	0.9000	0.9000	0.9750	0.9875	0.9938	0.90668	0.974875
$T_{4\text{-}12}$	1.032	1.0438	0.9938	1.0000	1.0313	0.9812	0.9750	0.99907	0.972653
$T_{28\text{-}27}$	1.068	0.9938	0.9813	0.9813	0.9750	1.0187	1.0438	0.97220	1.081762
Q_{c10} [pu]	0	0.0355	0.023	0.0205	0.0110	0.0040	0.0330	0.02752	0.04829834
Q_{c12}[pu]	0	0.0160	0	0	0.0390	0.0500	0.0465	0.04986	0.04836522
Q_{c15} [pu]	0	0.0450	0	0	0.0140	0	0.0350	0.04821	0.04767409
Q_{c17} [pu]	0	0.0415	0.055	0.0095	0.0425	0.0250	0.0335	0.02367	0.04859763
Q_{c20} [pu]	0	0.0350	0.050	0.0500	0.0175	0.0020	0.0180	0.04998	0.04984623
Q_{c21} [pu]	0	0.0390	0.050	0.0500	0.0090	0.0265	0.0070	0.04999	0.04995552
Q_{c23} [pu]	0	0.0075	0	0.0240	0.0170	0.0275	0.0170	0.04954	0.04895767
Q_{c24} [pu]	0	0.0010	0.050	0.0500	0.0120	0.0395	0.0155	0.05000	0.04999823
Q_{c29} [pu]	0	0.0290	0.050	0.0265	0.0335	0.0355	0.0110	0.02591	0.04999931
Pertes totales [MW]	5.8223	5.1311	4.9145	4.8960	5.2055	5.1439	5.1298	5.2278	5.324314

4.5 - CONCLUSION

La répartition optimale de la puissance réactive (ORPD) est un problème d'optimisation multivariable et non linéaire avec contraintes. Pour résoudre ce problème, on a proposé :

- un algorithme génétique amélioré IGA,
- l'optimisation par essaims particulaires PSO,
- l'évolution différentielle DE,
- le recuit simulé SA.

En général, dans un algorithme génétique ordinaire (GA), la probabilité de mutation est fixe pendant toute la procédure de recherche. Cependant, dans des applications pratiques, une petite valeur fixe de probabilité de mutation peut donner des résultats prématurés, alors que la recherche avec une grande valeur fixe de probabilité de mutation ne convergera pas. Dans ce travail, un algorithme génétique amélioré IGA basé sur une probabilité de mutation adaptative est proposé pour résoudre ce problème. Cette probabilité de mutation est variable selon la valeur de la fonction objective.

L'algorithme du PSO a l'avantage de localiser rapidement les bonnes solutions, mais approximativement. Cependant, ils peuvent converger prématurément pour donner une solution relativement mauvaise.

L'évolution différentielle (DE) est une technique d'optimisation globale, puissante, simple et facile à utiliser. Elle est caractérisée par son aptitude à la parallélisation, mais elle présente l'inconvénient de stagnation. La technique DE utilise des différences pondérées entre les vecteurs de solution pour perturber la population. Lorsque la population perdra complètement sa diversité, elle contiendra des éléments identiques, et elle reste inchangée par la perturbation créée par l'algorithme du DE. Pour éviter une convergence prématurée, il est nécessaire de maintenir un niveau raisonnable de diversité dans la population.

Un algorithme du recuit simulé (SA) a l'avantage d'être souple, facile et rapidement implémentable. Il n'a pas besoin d'une grande mémoire de stockage. Il a la propriété d'exploration aléatoire locale, au voisinage d'un point donné. Il est utile pour une recherche locale rapide mais moins adapté pour une recherche globale.

Afin d'exploiter les avantages des méthodes utilisées et d'améliorer les résultats, ce chapitre propose deux types d'hybridation :

- l'un entre l'essaim particulaire PSO et un algorithme génétique amélioré IGA avec une probabilité de mutation adaptative et un codage réel.

- l'autre entre l'évolution différentielle DE (basée sur une population de solution) et le recuit simulé SA (basée sur une solution unique).

L'objectif de la première hybridation entre le PSO et l'IGA est d'exploiter les avantages des deux méthodes et d'améliorer les résultats. La méthode du PSO a été employée pour générer une population initiale injectée et utilisée dans le programme de l'IGA.

Dans le deuxième type d'hybridation, un algorithme modifié de DE est proposé. Ce dernier utilise une expression de sélection inspirée de la sélection de SA. Lorsque cet algorithme modifié de DE est terminé, nous utilisons la solution trouvée comme solution initiale pour l'algorithme de SA. Nous utilisons le SA comme un algorithme de recherche locale dans la dernière étape pour améliorer la solution trouvée.

Le deuxième algorithme d'hybridation proposé combine la capacité de recherche globale du DE et la capacité de recherche locale du SA. Cela fournira un bon équilibre entre l'exploration et l'exploitation. Cette hybridation peut aider à éviter les points locaux, résister la convergence prématurée et augmenter la diversité des solutions de l'évolution différentielle DE.

Pour les variables de contrôle, une représentation mixte (continue/discrète) est proposée. Les résultats sur les deux réseaux électriques (Ward-Hale 6 bus) et (IEEE 30 bus) montrent les avantages des méthodes hybrides par rapport à l'utilisation des méthodes IGA, PSO, DE ou SA seules. Les méthodes hybrides présentent un avantage potentiel. Par conséquent, les techniques d'hybridation sont des approches alternatives prometteuses.

5 – CONCLUSION GENERALE ET PERSPECTIVES

5.1 – CONCLUSIONS

Le transport de la puissance réactive à longue distance présente une série d'inconvénients tels-que les chutes de tension considérables, les pertes de ligne par effet joule et moins de capacité pour transporter la puissance active.

Afin de garantir une bonne qualité d'énergie il est nécessaire de satisfaire l'équilibre offre-demande de l'énergie réactive, de fournir une tension aussi régulière que possible et de respecter un certain nombre de contraintes techniques.

Le contrôle de la tension/puissance réactive a pour objectif de maintenir un profil de tension adéquat dans le réseau électrique. En plus, il doit maintenir des réserves de puissance réactive dans les différentes zones du système pour faire face aux incidents de tension.

L'optimisation du plan de tension d'un réseau électrique, suppose donc, de maîtriser les transits d'énergie réactive et de minimiser les pertes dans le réseau. Cela n'est possible que par un ajustement optimal d'un nombre de moyens de contrôle, tels que les sources de puissance réactive (switching Var sources), les générateurs de tension (en agissant sur leurs tensions) et les transformateurs réglables.

Alors résoudre ce problème revient à résoudre un problème d'optimisation (minimisation) avec contraintes. Les variables pour ce problème sont divisées en deux : variables de contrôle et variables dépendantes. L'ensemble des variables de contrôle est composé des variables qui affectent les injections de la puissance réactive, c'est-à-dire, les tensions des générateurs, les rapports de transformation des transformateurs réglables et les valeurs des batteries de condensateurs shunts. Par contre, les variables dépendantes sont les tensions des nœuds de charge et les injections de puissance réactive aux nœuds des générateurs.

Le but de cette thèse consiste à traiter ce problème d'optimisation par l'application des techniques intelligentes simples et/ou hybrides dans leur forme améliorée (adaptative). Les méthodes utilisées sont testées sur les réseaux Ward-Hale 6 bus et IEEE 30 bus.

Tout d'abord, un calcul d'écoulement de puissance par la méthode de Newton-Raphson est exécuté en condition de fonctionnement initiale. En conséquence, des tensions de quelques jeux de barres de charge ont dépassées ses limites admissibles. Pour cela, on est obligé d'ajuster les moyens

de contrôle pour avoir un plan optimal de tension et, en même temps, de faire minimiser les pertes totales de la puissance active dans le réseau. Ceci revient à résoudre un problème d'optimisation (minimisation) avec contraintes.

Pour traiter ce problème, on a essayé de résoudre le problème du contrôle optimal tension/puissance réactive par l'application des métaheuristiques suivantes :

- un algorithme génétique amélioré IGA avec une probabilité de mutation adaptative et un codage réel.
- l'optimisation par essaims particulaires PSO : caractérisé par sa rapidité de convergence.
- le recuit simulé SA : inspirée du procédé de recuit utilisé en métallurgie pour améliorer la qualité d'un solide. La méthode du SA est caractérisée par sa capacité d'éviter les minimums locaux à cause de sa fonction de probabilité incorporée dans son algorithme pour accepter ou rejeter de nouvelles solutions. Elle n'a pas besoin d'une grande mémoire de stockage.
- l'évolution différentielle DE : une technique d'optimisation globale puissante, simple et facile à utiliser.

Les résultats de simulation de ces techniques sur les réseaux Ward-Hale 6 bus et IEEE 30 bus montrent clairement que toutes les variables de contrôle sont ajustées, les tensions et les puissances réactives sont dans leurs limites et les pertes sont diminuées. Pour les variables de contrôle, une représentation mixte (continue/discrète) est proposée.

Afin d'exploiter les avantages des méthodes utilisées et d'améliorer les résultats, on a proposé aussi deux types d'hybridation :

- *la première hybridation* HPSOIGA: entre l'essaim particulaire PSO et l'algorithme génétique amélioré IGA. Les principaux avantages du codage réel de l'IGA sont la simplicité, la facilité et la vitesse de convergence rapide. La modification dans la probabilité de mutation améliore la convergence de l'algorithme génétique. La méthode du PSO a été employée pour générer une population initiale injectée et utilisée dans le programme de l'IGA.
- *la deuxième hybridation* HDESA: entre l'évolution différentielle DE (basée sur une population de solution) et le recuit simulé SA (basée sur une solution unique).

  Dans cette étude, nous commençons l'algorithme hybride avec l'algorithme de l'évolution différentielle DE mais la sélection des individus est remplacée par celle utilisé dans le recuit simulé SA. Cette méthode de sélection peut permettre l'exploration de zones n'améliorant pas la valeur de la fonction objective en acceptant des solutions moins bonnes, ce qui peut mener à éviter le point local, résister à une convergence prématurée et augmenter la diversité des solutions de l'évolution différentielle. Dès que cet algorithme

modifié de DE est terminé, nous utilisons la solution trouvée comme solution initiale pour l'algorithme SA.

L'hybridation proposée bénéficie de la capacité de recherche globale de DE et de la capacité de recherche locale de SA, en inhibant leurs points faibles. La combinaison de DE et SA fournira un bon équilibre entre exploration et exploitation.

Comparées à l'utilisation de la méthode IGA, PSO, DE ou SA seule, les méthodes hybrides présentent un avantage potentiel. Par conséquent, les techniques d'hybridation sont des approches alternatives prometteuses.

5.2 - PERSPECTIVES

Les perspectives de ce travail s'orientent vers les optiques suivantes :

- Actuellement, le secteur des algorithmes évolutionnaires comme les algorithmes génétiques GA et l'évolution différentielle DE est très actif, on peut avoir beaucoup d'amélioration en agissant sur ses différents opérateurs et paramètres (codage, probabilité de croisement et de mutation, procédure de la sélection, génération de la population initiale, …etc.).
- L'intelligence inspirée de la nature constitue aujourd'hui une source d'inspiration très riche et très active. On propose de résoudre le problème du contrôle optimale de tension / puissance réactive par d'autres méta-heuristiques récentes.
- Nous pouvons hybrider l'algorithme SA avec une autre technique et démontrer l'efficacité de l'algorithme proposé sur différents réseaux de tests.

BIBLIOGRAPHIE

[1] L. K. Kirchmayer, "*Economic operation of power systems"*, John Wiley & sons, Inc., 1958, New York, USA.

[2] J. Carpentier, "*Contribution à l'étude du dispatching économique*", Bulletin de la Société Française des Electriciens", Ser. 8, Vol. 3, August 1962, France.

[3] H. W. Dommel and W. F. Tinney, "*Optimal power flow solutions*", IEEE Transactions on Power Apparatus and Systems, Vol. 87, N° 10, pp. 1866 – 1876, 1968.

[4] O. Alsac, B. Stott, *"Optimal load flow with steady state security"*, IEEE Trans. Power Appar. Syst. PAS – 93, pp. 745 – 751, 1974.

[5] V. H. Quintana, M. Santos-Nieto, *"Reactive-power dispatch by successive quadratic programming"*, IEEE Transactions on Energy Conversion, Vol. 4, N° 3, pp. 425 – 435, September 1989.

[6] M. Bjelogreic, M. S. Calovic, B. S. Babic, *"Application of Newton's optimal power flow in voltage / reactive power control*", IEEE Transactions on Power Systems, Vol.5 , N° 4, pp. 1448 – 1454, November 1990.

[7] J. R. O. Soto, C. R. R. Dornellos, D. M. Falcao, *"Optimal reactive power dispatch using hybrid formulation : Genetic algorithms and Interior point"*, Paper accepted for presentation at PPT 2001, 2001 IEEE Porto Power Tech Conference, September 10 – 13, 2001, Portugal.

[8] N. I. Deeb, S. M. Shahidehpour, *"An efficient technique for reactive power dispatch using a revised linear programming approach",* Electric Power Systems Research, Vol. 15, pp. 121 – 134, 1988.

[9] B. Venkatesh, G. Sadasivam, M. A. Khan, *"Toward on-line optimal reactive power scheduling using ANN memory model based method"*, IEEE, Power Engineering Society 1999 Winter Meeting, Vol. 2, N° 13, pp. 844 – 848, 1999.

[10] K. H. Abbul-Rahman, S. M. Shahidehpour, *"A fuzzy-based optimal reactive power control",* IEEE Transactions on Power Systems, Vol. 8, N° 2, pp. 662 – 670, Mai 1993.

[11] K. H. Abbul-Rahman, S. M. Shahidehpour, *"Optimal reactive power dispatch with fuzzy variables*", IEEE, International Symposium on Circuit and Systems ISCAS'93, Vol. 4, pp. 2188 – 2191, 3 – 6 May 1993.

[12] D. E. Goldberg, "*Genetic algorithms",* Addison – Wesley, 1991, USA.

[13] Q. H. Wu, J. T. Ma, *"Power system optimal reactive power dispatch using evolutionary programming"*, IEEE Transaction on Power Systems, Vol. 10, N° 3, pp. 1243 – 1249, August 1995.

[14] J. T. Ma, L. L. Lai, *"Application of genetic algorithm to optimal reactive power dispatch including voltage-dependent load models"*, IEEE International Conference on Evolutionary Computation ICEC'95, Vol. 1, N° 2, pp. 5 – 10, 29 Nov – 1 Dec. 1995, Perth, WA, Australia.

[15] Y. J. Cao, Q. H. Wu, *"Optimal reactive power dispatch using an adaptive genetic algorithm"*, Second International Conference on Genetic Algorithms in Engineering Systems : Innovations and Applications GALESIA 97, Vol. 2, N° 21, IEEE, pp. 117 – 122, 2 – 4 September, 1997, Glasgow, UK.

[16] W. N. W Abdullah, H. Saibon, A. A. M Zain, R. L. Lo, *"Genetic algorithm for optimal reactive power dispatch"*, IEEE, International Conference on Energy Management and Power Delivery, Proceedings of EMPD'98, Vol. 1, N° 29(S06.4 : numéro 4 de la session 6), pp. 160 – 164, 3 – 5 March 1998, Singapore.

[17] H. Zhang, L. Zhang, *"Reactive power optimization based on genetic algorithm"*, IEEE, International Conference on Power System Technology, Proceeding POWERCON'98, Vol. 2, N° 137, pp. 1448 – 1453, 18 – 21 August 1998, Beijing, China.

[18] M. Xiangping, L. Zhishan, Z. Huaguang, *"Fast synthetic genetic algorithm and its application to optimal control of reactive power flow",* IEEE, International Conference on Power System Technology, Proceeding POWERCON'98, Vol. 2, N° 138, pp. 1454 – 1458, 18 – 21 August 1998, Beijing, China.

[19] H. Aoki, Y. Mizutani, *"Reactive power control by genetic algorithm"*, IEEE, Power Engineering Society Winter Meeting, Vol. 2, pp. 1389 – 1393, 23 – 27 Jan. 2000.

[20] A. Simões, E. Costa: *"On biologically inspired genetic operators: Transformation in the standard genetic algorithm"*, Proceedings of the Genetic and Evolutionary Computation Conference GECCO 2001, pp. 584 – 591, 2001, San Francisco, USA.

[21] A. Terki, N. Mansouri, *"Optimization of nonlinear systems using genetic algorithms with transformation and transposition of genes"*, 3rd International Conference on Electrical Engineering Design and Technologies ICEEDT 09, Oct. 31 – Nov. 02, 2009, Sousse, Tunisia.

[22] P. Subbaraj, P. N. Rajnarayanan, *"Optimal reactive power dispatch using self-adaptive real coded genetic algorithm"*, Electric Power Systems Research, Vol. 79, N° 2, pp. 374 – 381, 2009, Elsevier.

[23] K. Socha, M. Dorigo, *"Ant colony optimization for continuous domains"*, European Journal of Operational Research, Vol. 185, Issue 3, pp. 1155 – 1173, 2008, Elsevier.

[24] F. M. Tuaimah, Y. N. Abd, F. A. Hameed, *"Ant colony optimization based optimal power flow analysis for the iraqi super high voltage grid"*, International Journal of Computer Applications, Vol. 67, N° 11, pp. 13 – 18, April 2013.

[25] K. Y. Lee, J. G. Vlachogiannis, *"Ant colony optimization for active/reactive operational planning"*, Proceedings of the 16th IFAC Word Congress, Vol. 16, Part 1, 2005, Czech Republic.

[26] K. Lenin, M. R. Mohan, *"Ant colony search algorithm for optimal reactive power optimization"*, Serbian Journal of Electrical Engineering, Vol. 3, N° 1, pp. 77 – 88, June 2006, Serbia.

[27] A. R. Abbasy, *"Application of ant colony optimization algorithms to optimal reactive power dispatch"*, 7th WSEAS International Conference on Electric Power Systems, High Voltages, Electric Machines, pp 121 – 126, November 21 – 23, 2007, Venice, Italy.

[28] A. A. Abou El-Ela, A. M. Kinawy, M. T. Mouwafi R. A. El Sehiemy, *"Optimal reactive power dispatch using ant colony optimization algorithm"*, Proceedings of the 14th International Middle East Power Systems Conference (MEPCON'10, Paper ID 315, pp 960 – 965), December 19 – 21, 2010, Cairo University, Egypt.

[29] J. Kennedy, R. Eberhart, *"Particle swarm optimization"*, Proceeding of the IEEE International Conference on Neural Networks (ICNN'95), Vol. 4, pp 1942-1948, November – December 1995, IEEE Service Center, Perth, Western Australia.

[30] A. Stacey, M. Jancic, I. Grundy, *"Particle swarm optimization with mutation"*, The 2003 Congress on Evolutionary Computation CEC '03, Vol. 2, pp. 1425 – 1430, IEEE 2003, Australia.

[31] B. Alatas, E. Akin, A. B. Ozer, *"Chaos embedded particle swarm optimization algorithms"*, Chaos, Solitons & Fractals, Vol. 40, N° 4, pp. 1715 – 1734, 2009 Elsevier, Amsterdam, Netherland.

[32] B. Alatas, E. Akin, *"Chaotically encoded particle swarm optimization algorithm and its applications"*, Chaos, Solitons & Fractals Chaos, Vol. 41, N° 2, pp. 939 – 950, 30 Jul 2009, Elsevier, Amsterdam, Netherland.

[33] J. J. Jiménez-Núñez, J. R. Cedeño -Maldonado, *"A particle swarm optimization approach for reactive power dispatch"*, In Proceeding of the 37th North Amarican Symposium, Ames, IA, pp. 198 – 205, October 23 – 25, 2005 IEEE, USA.

[34] L. L. Lai, T. Y. Nieh, D. Vujatovic, Y. N. Ma, Y. P. Lu, Y. W. Yang, H. Braun, *"Swarm intelligence for optimal reactive power dispatch"*, 2005 IEEE/PES Transmission and Distribution Conference & Exhibition: Asia and Pacific Dalian, pp. 1 – 5, 2005, China.

[35] B. Zhao, C.X. Guo, Y.J. Cao, *"An improved particle swarm optimization algorithm for optimal reactive power dispatch"*, IEEE Power Engineering Society Genaral Meeting, Vol. 1, pp 272 – 279, 12 – 16 June 2005, USA.

[36] A. A. Abou El-Ela1, T. Fetouh, M. A. Bishr, R. A. F. Saleh, *"Power systems operation using particle swarm optimization technique"*, Electric Power Systems Research, Vol. 78, Issue 11, pp. 1906 – 1913, November 2008, Elsevier, Amsterdam, Netherland.

[37] G. Baskar, M. R. Mohan, *"Security constrained economic load dispatch using improved particle swarm optimization suitable for utility system"*, International Journal of Electrical Power and Energy Systems Vol. 30, N° 10, pp. 609 – 613, 2008, Elsevier, Amsterdam, Netherland.

[38] B. K. Panigrahi, V. Ravikumar Pandi, S. Das, *"Adaptive particle swarm optimization approach for static and dynamic economic load dispatch"*, International Journal of Energy Conversion and Management, Vol. 49, N° 6, pp. 1407 – 1415, 2008, Elsevier, Amsterdam, Netherland.

[39] G. Baskar, M.R. Mohan, *"Contingency constrained economic load dispatch using improved particle swarm optimization for security enhancement"*, Electric Power Systems Research Vol. 79, N° 4, pp. 615 – 621, 2009, Elsevier, Amsterdam, Netherland.

[40] R. Labdani, L. Slimani, T. Bouktir, *"Particle swarm optimization applied to the economic dispatch problem"*, Journal of Electrical Systems Vol. 2, N° 2, pp. 95 – 102, 2006, JES 2006 on-line.

[41] D. Ben Attous, Y. Labbi, *"Particle swarm optimization to solve economic dispatch with valve point effects"*, 3rd International Conference on Electrical Engineering Design and Technologies (ICEEDT 09) Oct. 31 – Nov. 02, 2009 Sousse, Tunisia.

[42] M. Saravanan, S. M. R. Slochanal, P. Venkatesh, J. P. S. Abraham, *"Application of particle swarm optimization technique for optimal location of FACTS devices considering cost of installation and system loadability"*, Electric Power Systems Research, Vol. 77, N° 3, pp. 276 – 283, 2007, Elsevier, Amsterdam, Netherland.

[43] A. A. EL-Dib, Hosam K. M. Youssef, M. M. EL-Metwally, Z. Osman, *"Optimum VAR sizing and allocation using particle swarm optimization"*, Electric Power Systems Research Vol. 77, N° 8, pp. 965 – 972, 2007, Elsevier, Amsterdam, Netherland.

[44] L. D. Arya, S. C. Choube, M. Shrivastava, D. P. Kothari, *" Particle swarm optimization for determining shortest distance to voltage collapse"*, International Journal of Electrical Power and Energy Systems Vol. 29, N° 10, pp 796 – 802, 2007, Elsevier, Amsterdam, Netherland.

[45] S. Boubaker, F. M. Sahli, *"Two optimal control strategies based on particle swarm optimization for a hybrid two-tank system"*, 2nd International Conference on Electrical Systems Design & Technologies (ICEEDT 08), Nov. 8 – 10, 2008, Hammamet, Tunisia.

[46] Recioui, H. Bentarzi, A. Azrar, M. Dehmas, M. Challal, *"Linear array antenna optimization for wireless applications using particle swarm optimization technique"*, 3rd International Conference on Electrical Engineering Design and Technologies (ICEEDT 09) Oct. 31 – Nov. 02, 2009, Sousse, Tunisia.

[47] Y. Shi, R. C. Eberhart, *"A modified particle swarm optimizer"*, 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), pp. 69 – 73, 4 – 9 May 1998.

[48] K. Price, R. Storn, J. Lampinen, *"Differential evolution: a practical approach to global optimization",* Springer, 2005.

[49] C. Thammasirirat, B. Marungsri, R. Oonsivilai, A. Oonsivilai, *"Optimal reactive power dispatch using differential evolution"*, Word Academy of Science, Engineering and Technology, Vol. 60, pp. 89 – 94, 2011.

[50] S. Sakthivel, D. Mary, S. Ramya, *"Reactive power optimization and voltage stability limit improvement with TCSC device through DE algorithm under most credible contingency condition"*, International Journal of Scientific & Engineering Research, Vol. 3, Issue 5, pp. 1 – 8, May – 2012.

[51] W. Nakawiro, I. Erlich, *"A new adaptive differential evolution algorithm for voltage stability constrained optimal power flow"*, 17th Power Systems Computation Conference Stockholm, Vol. 17, pp. 1 – 7, August 22 – 26, 2011, Sweden.

[52] R. K. Verma, H. Singh, L. Srivastava, *"Optimal power flow using differential evolution algorithm with conventional weighted sum method"*, International Journal of Computational Engineering, IJCER, Vol. 2, Issue 3, N° 11, pp. 681 – 685, May – June 2012.

[53] C. F. Ionescu, C. Bulac, I. Triştiu, A. Mandiş, *"Application of mixed-integer hybrid differential evolution to optimal reactive power dispatch"*, U.P.B. Sci. Bull., Series C, Vol. 74, Issue 1, pp 155 – 162, 2012.

[54] A. A. Abou El Ela, M. A. Abido, S. R. Spea, *"Differential evolution algorithm for optimal reactive power dispatch"*, Journal of Electric Power Systems Research, Vol. 81, N° 2, pp. 458 – 464, 2011.

[55] M. Varadarajan, K. S. Swarup, *"Differential evolution approach for optimal reactive power dispatch"*, Applied Soft Computing, Vol. 8, N° 4, pp. 1549 – 1561, 2008.

[56] M. Varadarajan, K .S. Swarup, *"Differential evolutionary algorithm for optimal reactive power dispatch"*, Electrical Power and Energy Systems, Vol. 30, N° 30, pp. 435 – 441, 2008.

[57] S. A. Taher, S. A. R. Afsari, *"Optimal location and sizing of UPQC in distribution networks using differential evolution algorithm"*, Hindawi Publishing Corporation, Mathematical Problems in Engineering, Volume 2012 (2012), Article ID 838629, 20 pages, 2012, USA.

[58] A. Najafi, H. Falaghi, M. Ramezani, *"Combined heat and power economic dispatch using improved differential evolution algorithm"*, International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), Vol. 2, Issue 8, pp. 69 – 77, August 2012.

[59] B. Bhattacharyya and S. K. Goswami, *"A decomposition approach for combined heuristic and differential evolution method for the reactive power problem "*, Patern Recognition and Machine Intelligence PReMI 2007, Lecture Notes in Computer Science LNCS Vol. 4815, pp. 261– 268, 2007, Springer-Verlag Berlin Heidelberg 2007, Germany.

[60] D. Zaharie, *" Critical values for the control parameters of differential evolution algorithms"*, Proceedings of the 8th International Conference on Soft Computing, Mendel 2002, Vol. 8, pp. 62 – 67, 2002, Brno, Czech Republic.

[61] A. Ozturk, S. Cobanli, P. Erdogmus, S. Tosun, *"Reactive power optimization with artificial bee colony algorithm"*, Scientific Research and Essays Vol. 5, N° 19, pp. 2848 – 2857, 4 October, 2010.

[62] B. S. Prajapati, L. Srivastava, *"Multi-objective reactive power optimization using artificial bee colony algorithm"*, International Journal of Engineering and Innovative Technology (IJEIT), Vol. 2, Issue 1, pp 126 – 131, July 2012.

[63] S. Jaganathan , S. Palanisamy, *"Foraging algorithm for optimal reactive power dispatch with voltage stability and reliability analysis"*, Journal of Theoretical and Applied Information Technology (JATIT 2013), Vol. 47, N° 1, pp. 143 – 157, 10th January 2013.

[64] J. G. Vlachogiannis, K.Y. Lee, *"Quantum-inspired evolutionary algorithm for real and reactive power dispatch"*, IEEE Transactions on Power Systems, Vol. 23, N° 4, pp. 1627 – 1636, November 2008.

[65] J. G. Vlachogiannis, J. Østergaard, *"Reactive power and voltage control based on general quantum genetic algorithms"*, Expert Systems with Applications, Vol. 36, N° 3, pp. 6118 – 6126, 2009, Elsevier.

[66] K. Liu, W. Sheng, Y. Li, *"Research on reactive power optimization based on immunity genetic algorithm"*, International Conference on Intelligent Computing (ICIC 2006), Kunming, China, August 16-19, 2006. Proceedings, Part I, LNCS 4113, pp. 600 – 611, 2006, Springer – Verlag Berlin Heidelberg 2006, Germany.

[67] X. Hugang, C Haozhong, L. Haiyu, *"Optimal reactive power flow incorporating static voltage stability based on multi-objective adaptive immune algorithm"*, Energy Conversion and Management, Vol. 49, N° 5, pp. 1175 – 1181, 2008, Elsevier.

[68] R. Suresh, C. Kumar and S. Sakthivel, *"Real power loss and voltage deviation minimization by reactive power control through gravitational search algorithm"*, European Journal of Scientific Research, Vol. 90, N° 1, November 2012, pp. 36 – 48.

[69] J. Wang, W. Wang, X. Wang, H. Chen, X. Wang, *"Cooperative co-evolutionary approach applied in reactive power optimization of power system"*, Advances in Natural Computation: Second International Conference (ICNC 2006), Proceeding Part I, pp. 620 – 628, 2006, Springer – Verlag Berlin Heidelberg 2006, Germany.

[70] M. Barkaoui, *"Approche évolutionnaire pour la planification d'itinéraires dans un environnement dynamique "*, Thèse présentée pour l'obtention du grade de Philosophias Doctor (Ph.D.), Encadré par B. Moulin, Département d'informatique et de génie logiciel, Faculté des sciences et de génie, Université Laval, Québec 2010, Canada.

[71] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, "*Optimization by Simulated Annealing"*, Science, New Series, Vol. 220, N° 4598, May 13, 1983, pp. 671 – 680.

[72] K. K. Vishwakarma, H. M. Dubey, M. Pandit, B. K. Panigrahi, *"Simulated annealing approach for solving economic load dispatch problems with valve point loading effects"*, International Journal of Engineering, Science and Technology, Vol. 4, N° 4, pp. 60 – 72, 2012.

[73] N. Sinsuphan, U. Leeton, T. Kulworawanichpong, "*Optimal power flow solution using improved harmony search method"*, Applied Soft Computing, Vol. 13, pp. 2364 – 2374, 2013.

[74] X. S. Yang, S. Deb, "*Cuckoo search via Lévy flights"*, in: NaBIC 2009: Proceedings of the World Congress on Nature & Biologically Inspired Computing, pp. 210 – 214, Coimbatore, India, 2009.

[75] K. Lenin, B. R. Reddy, M. S. Kalavathi, *"Improved Cuckoo Search Algorithm for Solving Optimal Reactive Power Dispatch Problem"*, International Journal of Research in Electronics and Communication Technology (IJRECT 2014), Vol. 1, Issue 1, Jan – March 2014.

[76] M. K. Marichelvam, T. Prabaharan, X. S. Yang, "*Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan"*, Applied Soft Computing, Vol. 19, pp. 93 – 101, 2014.

[77] H. R. E. H. Bouchekara, "*Optimal power flow using black-hole-based optimization approach"*, Applied Soft Computing, Vol. 24, pp. 879 – 888, 2014.

[78] M. H. Sulaiman, Z. Mustaffa, M. R. Mohamed, O. Aliman, "*Using the gray wolf optimizer for solving optimal reactive power dispatch problem"*, Applied Soft Computing, Vol. 32, pp. 286 – 292, 2015.

[79] S. Mirjalili, S. M. Mirjalili, A. Lewis, *"Grey Wolf Optimizer"*, Advances in Engineering Software, Vol. 69, pp. 46 – 61, 2014.

[80] T. Niknam, A. M. Ranjbar, A. R Shirani, A. Ostadi, *"A new approach based on ant algorithm for Volt/Var control in distribution network considering distributed generation"*, Iranian Journal of Science & Technology, Transaction B, Engineering, Vol. 29, N° B4, pp. 385 – 398, 2005, Iran.

[81] D. G. Immanuel, G. Selvekumar, C. Christober, A. Rajan, *"A multi objective hybrid differential evolution algorithm assisted genetic algorithm approach for optimal reactive power and voltage control"*, International Journal of Engineering and Technology (IJET), Vol. 6, N° 1, pp. 199-203, Feb – Mar 2014.

[82] N. A. Rahmat, I. Musirin, *"Hybrid differential evolution-ant colony optimization for economic load dispatch problem"*, Journal of Theoretical and Applied Information Technology, Vol. 48, N° 2, pp. 680 – 690, 20th February 2013.

[83] C. M. Huang, Y. C. Huang, K. Y. Huang, *"A hybrid evolutionary computation algorithm for optimal reactive power dispatch considering voltage magnitude deviation"*, International Conference on Artificial Intelligence and Soft Computing ICAISC 2012, Lecture Notes in Information Technology, Vol. 12, pp. 178 – 184, 2012.

[84] R. Su, L. Kong, S. Song, P. Zhang, K. Zhou, J. Cheng, *"A new ridgelet neural network training algorithm based on improved particle swarm optimization"*, Third International Conference on Naturel Computation (ICNC 2007), Vol. 3, pp. 411 – 415, IEEE 2007.

[85] W. Zhang, Y. Liu, *"Multi-objective reactive power and voltage control based on fuzzy optimization strategy and fuzzy adaptive particle swarm"*, International Journal of Electrical Power and Energy Systems, Vol. 30, N° 9, pp. 525 – 532, 2008, Elsevier.

[86] S. C. Neoh, N. Morad, C. P. Lim, Z. Abdul Aziz, *"A layered matrix cascade genetic algorithm and particle swarm optimization approach to thermal power generation*

scheduling", Soft Computing in Industrial Applications, ASC 39, pp. 241 – 250, 2007, Springer-Verlag Berlin Heidelberg, Germany.

[87] A. Shunmugalatha, S. M. R. Slochanal, *"Optimum cost of generation for maximum loadability limit of power system using hybrid particle swarm optimization"*, International Journal of Electrical Power and Energy Systems, Vol. 30, N° 8, pp. 486 – 490, 2008, Elsevier.

[88] J. Hazra, A. Sinha, *"Application of soft computing methods for economic dispatch in power systems"*, Word Academy of Science engineering and Technology 28, pp. 643 – 648, 2009.

[89] V. Karthikeyan, S. Senthilkumar, V. J. Vijayalakshmi, *"A new approach to the solution of economic dispatch using particle swarm optimization with simulated annealing"*, International Journal on Computational Sciences & Applications (IJCSA) Vol. 3, N° 3, pp. 37 – 49, June 2013.

[90] K. Lenin, B. R. Reddy, M. S. Kalavathi, *"Restarted simulated annealing particle swarm optimization algorithm for solving optimal reactive power dispatch problem"*, Pinnacle Advanced Physics, Vol. 2, N° 4, pp. 142 – 149, 2015.

[91] K. Lenin, B. R. Reddy, *"Abatement of active power loss and enhancement of voltage profile by gravitational search algorithm based on simulated annealing method "*, International Journal of Research in Electrical and Electronics Technology (IJREET), Vol. 1, Issue 3, pp. 38 – 43, August 2014.

[92] K. Lenin, B. R. Reddy, M. S. Kalavathi, *"Reduction of active power loss and improvement of voltage profile index by using simulating annealing based krill herd algorithm"*, International Journal of Novel Research in Electronics and Communication, Vol. 1, Issue 1, pp. 10 – 21, September – October 2014.

[93] K. Lenin, B. R. Reddy, *"Minimization of active power loss by using hybridization of simulated annealing and nelder-mead algorithm"*, Majlesi Journal of Energy Management, Vol. 3, N° 4, pp. 23 – 28, December 2014.

[94] L. Srivastava, H. Singh, *"Hybrid Multi-Swarm Particle Swarm Optimisation Based Multi-Objective Reactive Power Dispatch",* IET Generation, Transmission & Distribution, Vol. 9, N° 8, pp. 727 – 739, 2015.

[95] P. Biswas, B. B. Pal, "*A Trilevel Programming Approach to Solve Reactive Power Dispatch Problem Using Genetic Algorithm Based Fuzzy Goal Programming",* Proceedings of the 48th annual convention of CSI, Vol. 1, pp. 205 – 217, 2014.

[96] B. Bersene, *"Réglage de la tension dans les réseaux de distribution du futur"*, Thèse pour obtenir le grade de Docteur de l'Université de Grenoble, Encadré par N. Hadjsaïd et Y. Bésanger, délivrée par l'Institut Polytechnique de Grenoble, Université de Grenoble, Décembre 2010, France.

[97] G. Rami, *"Contrôle de tension auto adaptatif pour des productions décentralisées d'énergies connectées au réseau électrique de distribution"*, Thèse pour obtenir le grade de Docteur de l'INP Grenoble, Encadré par N. Hadjsaid et T. T. Quoc, Spécialité : Génie Electrique,

préparée au Laboratoire d'Electrotechnique de Grenoble dans le cadre de l'Ecole Doctorale « Electronique, Electrotechnique, Automatique, Télécommunication et Signal », 2006, France.

[98] T. Wildi et G. Sybille, *" Électrotechnique 4ième édition"*, Edited by De Boeck, 2005, Bruxelles.

[99] A. E. Smith, D. W. Coit, *"Constraint-Handling Techniques - Penalty Functions"*, Chapter C5.2 in Handbook of Evolutionary Computation, Institute of Physics Publishing and Oxford University Press, Bristol, U.K, 1997.

[100] L. A. Amine, *"Calcul évolutionnaire, application sur l'optimisation de la planification de puissance réactive"*, Mémoire de Magister en Electrotechnique, Encadré par A. Hellal, Option : Réseaux Electriques et Hautes Tensions, Département de Génie Electrique, Ecole Nationale Polytechnique d'Alger, 2005, Algérie.

[101] O. Hajji, *"Contribution au développement de méthodes d'optimisations stochastiques : Application à la conception des dispositifs électrotechniques"*, Thèse de Doctorat, Encadré par J. M. Kauffman et A. Gerbaud, Ecole centrale de Lille et l'Université des sciences et techniques de Lille, 2003, France.

[102] M. Samir, *"Optimisation multiobjectif par un nouveau schéma de coopération méta/exacte", Mémoire de Magister,* Encadré par M. C. Batouche et D. E. Saïdouni, Faculté d'Ingénieur, Département de l'Informatique, Université Mentouri de Constantine, 2007, Algérie.

[103] P. Rebreyend, *"Algorithmes génétiques hybrides en optimisation combinatoire"*, Thèse de Doctorat, Encadré par A. Ferreira et Y. Robert, Spécialité : Informatique, Ecole Normale Supérieure de Lyon, 1999, France.

[104] L. Jourdan, *"Métaheuristiques pour l'extraction de connaissances : Application à la génomique"*, Thèse pour obtenir le grade de Docteur de L'U.S.T.L, Encadré par E. G. Talbi et C. Dhaenens, Discipline : Informatique, Université des Sciences et Technologies de Lille, U.F.R. D'I.E.E.A, 2003, France.

[105] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, *"Optimization by simulated annealing",* Journal of Science, New Series, Vol. 220, N° 4598, pp. 671 – 680, May 13, 1983.

[106] N. Metropolis, A. W. Rosenbluth, M. N. Rsenbluth, A. H. Teller, *" Equations of state calculations by fast computing machines",* Journal of chemical pysics, Vol. 21, N° 6, pp. 1087 – 1092, June, 1953.

[107] S. Bouallagui, *"Techniques d'optimisation déterministe et stochastique pour la résolution de problèmes difficiles en cryptologie"*, Thèse en vue de l'obtention du titre de Docteur de l'Institut National des Sciences Appliquées de Rouen, Encadré par H.A. Le Thi et T. P. Dinh, 2010, France.

[108] H. Hachimi, *" Hybridations d'algorithmes métaheuristiques en optimisation globale et leurs applications"*, Thèse de Doctorat en cotutelle pour obtenir les grades de Docteur de l'Institut National des Sciences Appliquées de Rouen, Encadré par R. Ellaia et A. E. Hami, Spécialité : Optimisation des structures, Option : Fiabilité, Qualité, Sécurité, et de Docteur en sciences de l'ingénieur Ecole Mohammadia d'Ingénieurs, Université Mohammed 5 Agdal, Spécialité

Mathématiques Appliquées et Informatique, Option : Optimisation, Analyse Numérique, Statistique, 2013, Maroc.

[109] O. El Gerari, *"Contributions à l'amélioration des techniques de la programmation génétique et de la programmation évolutionnaire"*, Thèse de Doctorat de l'Université du Littoral Côte d'Opale, Encadré par C. Fonlupt, Discipline : Informatique, 2011, France.

[110] L. J. Fogel, A. J. Owens, and M. J. Walash. *"Artificial intelligence through simulated evolution"*, John Wiley, 1966, New York, USA.

[111] J. Koza, *"Genetic programming, on the programming of composants by means of natural selection",* MIT press, Cambridge, MA, 1992, USA.

[112] R. Storn, K. Price, *"Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces"*, Journal of Global Optimization, Vol. 11, N° 4, pp. 431 – 359, 1997, USA.

[113] J. Lepagnot, *"Conception de métaheuristiques pour l'optimisation dynamique. Application à l'analyse de séquences d'images IRM. "*, Thèse de Doctorat en informatique, Encadré par P. Siarry, Université Paris-Est, École Doctorale Mathématiques Et Stic, 2011, France.

[114] W. Tfaili, *"Conception d'un algorithme de colonie de fourmis pour l'optimisation continue dynamique"*, Thèse de Doctorat, Encadré par P. SIARRY, Spécialité : Sciences de l'Ingénieur, Option : Optimisation, Université de Paris 12 Val de Marne, 2007, France.

[115] R. Charrier, *"L'intelligence en essaim sous l'angle des systèmes complexes : Etude d'un système multi-agent réactif à base d'itérations logistiques couplées"*, Thèse de Doctorat de l'Université Nancy 2, Encadré par F. Charpillet et C. Bourjot, Département de Formation Doctorale en informatique, 2009, France.

[116] Y. Cooren, *"Perfectionnement d'un algorithme adaptatif d'optimisation par essaim particulaire : Applications en génie médical et en électronique"*, Thèse de Doctorat, Encadré par P. Siarry, Spécialité : Sciences de l'Ingénieur, Option : Optimisation, Université de Paris 12 Val de Marne, 2008, France.

[117] L. Kartobi, *"Optimisation de la synthèse des FACTS par les algorithmes génétiques et les essaims particulaires pour le contrôle des réseaux électriques"*, Mémoire de Magister, Encadré par A. Mekhaldi et M. Boudour, Option : Réseaux Electriques et Haute Tension, Ecole Nationale Polytechnique, 2006, Algérie.

[118] R. Benabid, *"Optimisation multiobjectif de la synthèse des FACTS par les particules en essaim pour le contrôle de la stabilité de tension des réseaux électriques"*, Mémoire de Magister en Electrotechnique, Encadré par M. Boudour, Option : Analyse, Conduite et Commande des Réseaux Electriques, Faculté des Sciences et Sciences de l'Ingénieur, Département Génie Electrique, Université Amar Telidji, Laghouat, 2007, Algérie.

[119] S. Mouassa, *" Optimisation de l'écoulement de puissance par une méthode métaheuristique (technique des abeilles) en présence d'une source renouvelable (éolienne) et des dispositifs FACTS",* Mémoire de Magister en Electrotechnique, Encadré par T. Bouktir, Option : Réseaux électriques, Faculté de Technologie, Département d'Electrotechnique, Université Ferhat Abbas, Sétif, 2012, Algérie.

[120] K. Radja, *"Contribution au développement de méthodologies de fusion/classification contextuelles d'images satellitaires multi-sources"*, Thèse de Doctorat en Electronique, Encadré par A. B. Aissa, Université des Sciences et de la Technologie Houari Boumediene, Faculté d'électronique et Informatique, Algérie, 2008, Algérie.

[121] S. Dermoul, *" Etude de la contribution des systèmes immunitaires artificiels au pilotage de systèmes de production en environnement perturbé"*, en vue d'Obtenir le Grade de Docteur d'Université, Encadré par H. Pierreval et S. H. Gabouj, Spécialité Productique (Université Blaise Pascale, France) et Informatique industrielle (Université 7 Novembre Carthage, Tunisie), 2010, France.

[122] I. Labed, *" Proposition d'un système immunitaire artificiel pour la détection d'intrusions",* Mémoire de Magister en Informatique, Encadré par M. K. Kholladi, Option : Information & Computation, Université Mentouri de Constantine, Faculté des Sciences de l'ingénieur, Département d'informatique, Laboratoire Lire, 2006, Algérie.

[123] A. Houacine, *"Système immunitaire artificiel parallèle appliqué aux flow shop hybride (FSH) ",* Mémoire de Magister, Encadré par K. Belkadi, Spécialité : Informatique, Option: Systèmes, Réseaux et Bases de Données (SRBDD), Université des Sciences et de la Technologie d'Oran -Mohamed Boudiaf, Faculté des Sciences, Département d'Informatique, 2011, Algérie.

[124] A. Blais, *" Calcul quantique universel sur qubits supraconducteurs",* Mémoire présenté au Département de Physique en vue de l'Obtention du Grade de Master ès sciences (M.Sc.), Encadré par S. Lacelle et A. M. Tremblay, Faculté des Sciences, Université de Sherbrooke, Québec, septembre 1999, Canada.

[125] M. L. Bellac, *"Introduction à l'information quantique",* Cours donnée à l'Ecole Supérieure de Sciences Informatiques (ESSI), Institut Non Lineaire de Nice, 2003, Edited by HAL, 2006, France.

[126] F. Omary, *"Applications des algorithmes évolutionnistes à la cryptographie",* Thèse de Doctorat d'état, Encadré par AboubakerLbekkouri, Spécialité : Informatique, Université Mohammed V – Agdal, Faculté des sciences Rabat, 2006, Rabat – Maroc.

[127] J. H. Holland, *"Adaptation in natural and artificial systems"*, University of Michigan press, 1975, USA.

[128] L. Slimani, *"Contribution à l'application de l'optimisation par des méthodes métaheuristiques à l'écoulement de puissance optimal dans un environnement de l'électricité dérégulé"*, Thèse de Doctorat en sciences, Encadré par Tarek Bouktir, Université de Batna Faculté des Sciences de l'ingénieur, Département d'Electrotechnique, 2009, Algérie.

[129] A. El Dor, *"Perfectionnement des algorithmes d'Optimisation par Essaim Particulaire. Applications en segmentation d'images et en électronique"*, Thèse de Doctorat en Informatique, Encadré par Patrick Siarry et Maurice Clerc, Université Paris – Est, École Doctorale mathematiques Et Stic, 2012, France.

[130] Y. Shi, R. C. Eberhart, *"Empirical study of particle swarm optimization"*, Proceedings of the 1999 Congress on Evolutionary Computation, (CEC 99), Vol. 3, pp. 1945 – 1950, 1999, Washington, USA.

[131] M. Clerc, J. Kennedy, *"The particle swarm : explosion, stability, and convergence in multi-dimensional complex space"*, IEEE Transactions on Evolutionary Computation, Vol. 6, N° 1, pp. 58 – 73, 2002.

[132] R. Eberhart, Y. Shi, *"Comparing inertia weights and constriction factors in particle swarm optimization"*, In : Proceedings of the 2000 Congress on Evolutionary Computation, pp. 84 – 88, IEEE Press, 2000.

[133] S. Labed, *"Méthodes bio-inpirées hybrides pour la résolution de problèmes complexes"*, Thèse de Doctorat en Sciences en Informatique, Encadré par Pr. Chikhi Salim, Université Constantine 2, Faculté des Nouvelles Technologies de l'Information et de la Communication, Département Informatique Fondamentale et Applications, 2013, Algérie.

[134] D. Duvivier, *"Etude de l'hybridation des méta-heuristiques, application à un problème d'ordonnancement de type jobshop",* Thèse pour obtenir le grade de Docteur, Encadré par Ph. Preux, Discipline : Informatique, Université du Littoral, Côte d'Opale, France.

[135] A. Spalanzani, *"Algorithmes évolutionnaires pour l'étude de la robustesse des systèmes de reconnaissance de la parole"*, Thèse présentée pour obtenir le titre de Docteur de l'Université Joseph Fourier - Grenoble I, Encadré par J. Caelen et H. Kabré, Spécialité Informatique Systèmes et Communication, 1999, France.

[136] E. D. Talbi, *"Sélection et règlage des paramètres pour l'optimisation de logiciels d'ordonnancement industriel"*, Thèse de Doctorat de l'Institut National Polytechnique de Toulouse, Encadré par B. Grabot et L. Geneste, Ecole Doctorale Systèmes, Spécialité : Informatique Industrielle, 2004, France.

[137] D. Meignan, *"Une approche organisationnelle et multi-agent pour la modélisation et l'implantation de métaheuristiques : Application aux problèmes d'optimisation de réseaux de transports"*, Thèse de Doctorat en Informatique, Encadré par A. Koukam et J. C. Créput, de l'Université de Franche-Comté et de l'Université de Technologie de Belfort-Montbéliard, 2008, France.

[138] A. Lemouari, *"Systèmes adaptatifs et apprentissage incrémental: de l'inspiration biologique à la résolution collective des problèmes"*, Doctorat en Science de l'Université de Constantine, Encadré par M. Benmohammed, Spécialité : Informatique, Université Mentouri de Constantine, Faculté des Sciences de L'ingénieur, Département de l'informatique, 2009, Algérie.

[139] T. Chaari, *"Un algorithme génétique pour l'ordonnancement robuste : Application au problème du flow shop hybride"*, Thèse en Co-Tutelle pour Obtenir le Grade de Docteur de l'Université de Valenciennes et du Hainaut-Cambrésis, Encadré par T. Loukil, D. Trentesaux et S. Chaabane, Discipline : Automatique, Spécialité : Automatique, Génie Informatique, et le grade de Docteur de la Faculté des Sciences Économiques et de Gestion de Sfax, Discipline : Méthodes Quantitatives, 2010, France / Tunisie.

[140] L. Jourdan, *"Métaheuristiques pour l'extraction de connaissances : Application à la Génomique"*, Thèse de Doctorat en Informatique, Encadré par E. G. Talbi et C. Dhaenens, Université des Sciences et Technologies de Lille, 2003, France.

[141] E. Ziyati, *"Optimisation de requêtes OLAP en entrepôts de données approche basée sur la fragmentation génétique"*, Thèse de Doctorat en Informatique et Télécommunications, Encadré par D. Aboutajdine, Université de Mohammed V – Agdal, Faculté des sciences, Rabat, 2010, Maroc.

[142] S. Sumathi, T. Hamsapriya, P. Surekha, *"Evolutionary intelligence an introduction to theory and applications with Matlab"*, Springer-Verlag Berlin and Heidelberg, 2008, Berlin, Germany.

[143] A. Zilouchian, M. Jamshidi, *"Intelligent control systems using soft computing methodologies"*, Edited by Ali Zilouchian and Mo Jamshidi, CRC Press LLC, 2001, USA.

[144] S. Sumathi, P. Surekha, *"Computational intelligence paradigms, theory and applications using MATLAB"*, by Taylor and Francis Group, LLC, 2010, USA.

[145] S. Araghinejad, *"Data-driven modeling : Using Matlab in water resources and environmental engineering"*, Springer Science+Business Media Dordrecht, 2014, Pays – Bas.

[146] W. Y. Yang, W. Cao, T.S. Chung, J. Morris, *"Applied numerical methods using Matlab"*, by John Wiley & Sons, Inc, 2005, New Jersey, USA.

[147] S. M. Mikki, A. A. Kishk, *"Particle swarm optimization : A physics-based approach"*, by Morgan & Claypool, 2008, USA.

[148] A. Ghasemi, K. Valipour, A. Tohidi, *"Multi objective optimal reactive power dispatch using a new multi objective strategy"*, Electrical Power and Energy Systems, Vol. 57, pp. 318 – 334, 2014.

TRAVAUX SCIENTIFIQUES

a) PUBLICATIONS INTERNATIONALES

1) F. Laouafi, A. Boukadoum, S. Leulmi, *"A Hybrid Formulation between Differential Evolution and Simulated Annealing Algorithms for Optimal Reactive Power Dispatch"*, TELKOMNIKA, Vol.16, N° 2, pp : 513 - 524, April 2018.

2) F. Laouafi, S. Leulmi, A. Boukadoum, *"Reactive Power Dispatch with Hybrid Formulation : Particle Swarm Optimization and Improved Genetic Algorithms with Real Coding"*, International Review of Electrical Engineering (I.R.E.E.), Part B, Vol. 5, N°2, pp : 601 - 607, March-April 2010.

b) COMMUNICATIONS INTERNATIONALES

1) Leulmi Salah, Laouafi Farida, Boukadoum Ahcene *, « Algorithmes génétiques pour le contrôle de tension - puissance réactive »,* 4th International Conference on Electrical Engineering, CEE-06 , pp. 455 – 460, 07 - 08 November 2006, Batna.

2) F. Laouafi, S. Leulmi, S. Boukadoum, A. Maache, F. Bouhennoufa, S. Kissar *"Application of genetic algorithms in voltage /reactive power dispatch optimization"*, 2nd International Conference on Electrical Engineering Design and Technologies (ICEEDT'08), November 8-10, 2008, Hammamet, Tunisia.

3) F. Laouafi, S. Leulmi, S. Kissar, *"Improved Genetic Algorithms with real coding applied in Reactive Power Dispatch Optimization"*, 3rd International Conference on Electrical Engineering Design and Technologies (ICEEDT'0 9), (ICEEDT'09), Oct. 31 – Nov. 02, 2009, Sousse, Tunisia.

c) PROJETS DE RECHERCHE

1) Projet de recherche CNEPRU

Thème : *« Optimisation du plan de tension et de la répartition de la puissance réactive par les techniques hybrides classiques & intelligentes améliorées : Application au réseau « SONELGAZ » Est-Algérien »*
Code du projet : J020 16 2007 0100
Début du projet **2008**
Fin du projet **2011**