الجمهوريــة الجزائريــة الديمقراطيــة الشعبيــة
République Algérienne Démocratique et Populaire
وزارة التعلـيـم العـالي و البـحـث العـلمـي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université MUSTAPHA Stambouli
Mascara

جامعة مصطفى أسطمبولي
معسكر

Faculté des Sciences Exactes
Département d'Informatique

THESE de DOCTORAT LMD

Spécialité : Informatique

Option : Technologie de l'information et de la communication

Intitulée

Optimisation des algorithmes évolutionnaires par hybridation pour la résolution des problèmes multi-objectifs

Présentée par : **BOUKHARI Noureddine**

Le lundi 04 janvier 2021

Devant le jury :

Présidente	**YAHYAOUI Khadidja**	**MCA**	**Université M. S. de Mascara**
Examinateur	**SALLEM Mohamed**	**MCA**	**Université M. S. de Mascara**
Examinateur	**BACHIR BOUIADJRA Rochdi**	**MCA**	**Université M. S. de Mascara**
Examinateur	**BOUKLI HACÈNE Sofiane**	**Professeur**	**Université D. L. de Sidi Bel Abbés**
Encadreur	**DEBBAT Fatima**	**Professeur**	**Université M. S. de Mascara**
Co-encadreur	**MONMARCHÉ Nicolas**	**Professeur**	**Université de Tours, France**

Année Universitaire : 2020-2021

Je dédie cette thèse

À mes chers parents,

À ma famille,

À mes amis,

À toute personne qui m'aime et que j'aime,

À tous ceux qui m’ont soutenu.

Remerciements

Tout d'abord, je tiens à remercier Allah, le clément et le miséricordieux, de m'avoir donné le courage et la patience pour la réalisation de ce travail.

J'ai le privilège d'avoir le professeur DEBBAT Fatima comme encadreur de thèse. Je tiens à la remercier grandement pour ses conseils avisés, son encadrement exemplaire et ses idées, son accessibilité, sa grande disponibilité et sa gentillesse. Je lui suis profondément reconnaissant. Je remercie également mes co-encadreurs MONMARCHÉ Nicolas et SLIMANE Mohamed pour leurs encadrements, leurs commentaires précieux, leurs relectures minutieuses et leurs soutiens constants tout au long de ma thèse.

J'adresse mes vifs remerciements à Madame YAHYAOUI Khadidja, Maître de Conférences de l'Université Mustapha Stambouli de Mascara de m'avoir honoré en acceptant de présider le jury, à Monsieur SALEM Mohamed, Maître de Conférences de l'Université Mustapha Stambouli de Mascara, à Monsieur BACHIR BOUYADJRA Rochdi, Maître de Conférences de l'Université Mustapha Stambouli de Mascara, et à Monsieur BOUKLI HACÈNE Sofiane, Professeur de l'Université Djillali Liabès de Sidi Bel Abbès, d'avoir accepté d'examiner et d'évaluer ce travail et de participer au jury.

Je remercie tous les enseignants et membres de notre équipe pour leur gentillesse et leur amitié pendant ces années de thèse. Je tiens à témoigner tout particulièrement ma sympathie et ma reconnaissance à Monsieur MEFTAH Boudjelal, Maître de Conférences de l'Université Mustapha Stambouli de Mascara, pour sa générosité intellectuelle, ses excellents conseils et sa disponibilité.

J'offre mes salutations et ma bénédiction à tous ceux qui m'ont soutenu à tous égards.

Enfin, je remercie mes parents et ma famille de m'avoir toujours soutenu mentalement, émotionnellement et physiquement durant la réalisation de cette recherche.

Résumé

Optimisation des algorithmes évolutionnaires par hybridation pour la résolution des problèmes multi-objectifs

Actuellement, les problèmes d'optimisation sont omniprésents et deviennent de plus en plus complexes. L'objectif de l'optimisation est de minimiser les temps calculs, le coût et le risque ou maximiser le profit ou le gain, la qualité et l'efficacité etc.... La résolution satisfaisante d'un problème d'optimisation difficile, qui comporte un grand nombre de solutions sous-optimales, justifie le recours à des méthodes métaheuristiques puissante. Les algorithmes d'optimisation bio-inspirés sont des approches émergentes, robustes et concurrentes qui reposent sur les principes inspirées de l'évolution biologique de la nature. La majorité des algorithmes utilisés pour résoudre ces problèmes d'optimisation sont les approches évolutionnaires ou à population. Ces derniers sont des algorithmes stochastiques faisant évoluer une population de solutions candidates (individus) dans l'espace de recherche de manière itérative, dans l'espoir de converger vers des solutions satisfaisantes. Plusieurs travaux ont été proposés sur des versions améliorées de ces méta-heuristiques en ajoutant quelques techniques afin de leurs permettre de résoudre les problèmes d'optimisation d'une manière plus efficace. Le but de cette thèse est de proposer et de concevoir de nouvelles approches évolutionnaires hybrides et de les appliquer pour la résolution à des problèmes d'optimisation réels multi objectifs.

Cette thèse vise à fournir des informations nouvelles et pertinentes sur la conception des paradigmes d'optimisation hybride inspirés de la nature. Ils combinent des méthodes évolutives avec des méthodes déterministes et d'apprentissages. Ces méthodes évolutionnaires hybrides donnent de meilleures performances lorsqu'elles sont appliquées à des tâches d'optimisation globale complexes et des recherches récentes ont montré l'importance de ces politiques d'hybridation.

Ce travail de thèse présente deux nouvelles versions améliorées des algorithmes évolutionnaires (Hybrid evolutionary strategy Algorithm with simplex method : ES-NM, hybrid multiobjective differential evolution Algorithm with opposition based learning method : MODE-OBL) visant à accélérer son fonctionnement et la localisation de l'optimum global pour différentes catégories de problèmes d'optimisation, à savoir les problèmes d'optimisation mono et multi-objectifs.

Les nouveaux algorithmes hybrides développées ont été soumis à différents Benchmark de fonctions complexes de différents types ainsi que le problème de portfolio. Les résultats des expériences sont ensuite comparés à la variante standard et à différentes méthodes de la littérature en termes de solutions trouvées et de vitesse de convergence. L'analyse des résultats obtenus a généralement confirmé que les modifications proposées sur les AEs amélioraient nettement leurs performances.

Mots clefs : Intelligence artificielle, Méta-heuristiques, algorithmes évolutionnaires, hybridation, Problème d'optimisation multi-objectifs.

Abstract

Optimizing evolutionary algorithms by hybridization for solving multi-objective problems

Currently, optimization problems are pervasive and becoming more and more complex. The objective of optimization is to minimize computational time, cost and risk or maximize profit or gain, quality, and efficiency etc. The satisfactory resolution of a difficult optimization problem, which involves large number of suboptimal solutions, justifies the use of powerful metaheuristic methods. Bio-inspired optimization algorithms are emerging, robust and competing approaches that are based on the principles inspired of nature's biological evolution. Most algorithms used to solve these optimization problems are population based evolutionary approaches. These are stochastic algorithms that evolve a population of candidate solutions (individuals) in the research space in an iterative way, in the hope of converging towards satisfactory solutions. Several works have been proposed on improved versions of these meta-heuristics by adding some techniques to enable them to solve optimization problems in a more efficient way. The aim of this thesis is to propose and design new hybrid evolutionary approaches and apply them for solving to real multi-objective optimization problems.

This thesis aims to provide new and relevant information on the design of hybrid optimization paradigms inspired by nature. It combines evolutionary methods with deterministic and learning methods. These hybrid evolutionary methods perform better when applied to complex global optimization tasks, and recent research has shown the importance of these hybridization policies.

This thesis work presents two new and improved versions of evolutionary algorithms (Hybrid evolutionary strategy Algorithm with simplex method: ES-NM, multi-objective hybrid differential evolution Algorithm with opposition based learning method: MODE-OBL) aimed at accelerating its operation and locating the global optimum for different categories of optimization problems, namely mono and multi-objective optimization problems.

Newly developed hybrid algorithms were subjected to different Benchmark of complex functions of different types as well as the portfolio problem. The results of the experiments are then compared to the standard variant and different methods of the literature in terms of found solution and convergence speed. Analysis of the results generally confirmed that the proposed changes to the AEs significantly improved their performance.

Keywords: Artificial intelligence, Metaheuristics, Evolutionary algorithms, Hybridization, Multi-objective optimization problems.

ملخص

تحسين الخوارزميات التطورية عن طريق التهجين لحل المشاكل المتعددة الأهداف

حاليا، مشاكل التحسين منتشرة واصبحت أكثر فأكثر تعقيدا .والهدف من التحسين هو تقليص وقت الحسابات، التكلفة، المخاطر أو تحقيق أقصى قدر من الفائدة، الجودة، والكفاءة... الخ .إن الحل المرضي لمشكلة معقدة من مشاكل التحسين، التي تتضمن على عدد كبير من الحلول المثلى، هو الذي يبرر استخدام أساليب قوية من أساليب البحث الحديثة. خوارزميات التحسين المستوحاة من البيولوجيا هي طرق ناشئة، قوية ونهج تنافسية مبنية على اساس التطور البيولوجي للطبيعة .معظم الخوارزميات المستخدمة في حل مشاكل التحسين هي خوارزميات المجتمعات أو النهج التطورية .هاته الأخيرة هي خوارزميات تعتمد على مبدأ العشوائية بحيث تقوم بتطوير مجموعة من الحلول المرشحة (الأفراد) في مجال البحث بطريقة متكررة، على أمل مصادفة حلول مرضية. وقد اقترحت العديد من الأعمال على إصدارات محسنة من هذه المناهج بإضافة بعض التقنيات لتمكينها من حل مشاكل التحسين بطريقة أكثر كفاءة .والهدف من هذه الأطروحة هو اقتراح وتصميم نهج تطورية هجينة جديدة وتطبيقها لحل مشاكل التحسين المتعددة الأهداف.

تهدف هذه الأطروحة إلى تقديم معلومات جديدة وذات صلة حول تصميم نماذج التحسين الهجينة المستوحاة من الطبيعة. فهي تجمع بين الأساليب التطورية وأساليب التعلم القطعية والذكاء الصناعي. هذه الطرق التطورية الهجينة أداءها أفضل عند تطبيقها على مشاكل التحسين المعقدة، وقد برهنت الأبحاث الحديثة على أهمية سياسات التهجين.

هذه الأطروحة تقدم نسختين جديدتين ومحسنتين من الخوارزميات التطورية الهجينة تهدفان إلى تسريع العمليات وتحديد الحلول المثلى لفئات مختلفة من مشاكل التحسين، وهي مشاكل التحسين أحادية ومتعددة الأهداف. حيث خضعت الخوارزميات الهجينة الجديدة التي تم تطويرها إلى عدة مقاييس الأداء المعقدة من مختلف الأنواع وكذلك مشكلة المحفظة. ثم تم مقارنة نتائج التجارب مع عدة خوارزميات تطورية وغيرها من طرق الذكاء الحديثة من حيث الحلول الذي تم العثور عليها وسرعة التقارب. وقد أكد تحليل النتائج عموماً أن التغييرات المقترحة على الخوارزميات التطورية قد حسنت أدائها بشكل كبير.

الكلمات المفتاحية: الخوارزميات التطورية، التهجين، مشاكل التحسين متعددة الأهداف، الذكاء الاصطناعي.

Table des matières

LISTE DES FIGURES

.....

Liste des Tableaux

INTRODUCTION GÉNÉRALE

Contexte

Comme le montre la littérature, l'utilisation des algorithmes évolutionnaires (AE) s'est avérée efficace pour résoudre les problèmes d'optimisations. Les AE sont des approches basées sur la population qui s'inspirent de l'évolution biologique, et elles sont généralement stochastiques par nature et comprennent plusieurs caractéristiques générales. Tout d'abord, les AE sont capables d'échantillonner plusieurs solutions candidates en une seule exécution de simulation. Une autre caractéristique importante des évaluations environnementales est l'adoption de l'idée de la survie du plus apte qui permet de conserver les meilleures solutions candidates. Enfin, les AE utilisent plusieurs mécanismes stochastiques inspirés de l'évolution biologique qui incluent la reproduction, la mutation, la recombinaison et la sélection pour explorer l'espace de recherche. Grâce à ces caractéristiques, les chercheurs ont réussi d'appliquer et d'adapter les AE dans un large éventail de problèmes d'application. Parmi les applications des AE, nous pouvons citer : la planification, les applications réseau, les télécommunications, optimisation du portefeuille en finance et autres.

Au fil des décennies, plusieurs AE ont été développées pour résoudre les problèmes d'optimisations, incluant les algorithmes génétiques (GA), la programmation évolutionnaire (EP), les stratégies évolutives (ES) et l'évolution différentielle (DE), et autres versions. Cependant, il existe certaines limites associées aux AE. Premièrement, rien ne garantit que des solutions optimales pourront être trouvées par les AE dans un laps de temps limité. L'utilisation des AE implique également certains paramètres de contrôle qui doivent être réglés pour des performances optimales, et cela peut impliquer un processus fastidieux d'essais et d'erreurs pour le réglage des paramètres avant de pouvoir être utilisés pour la résolution des problèmes de manière efficace. En plus, les AE peuvent également souffrir d'autres faiblesses telles que la perte de diversité, la convergence lente et la stagnation de la population, et ces faiblesses deviendront encore plus importantes dans le cas où les AE présentent des problèmes multimodaux et multi-objectifs.

Dans les problèmes complexes qui peuvent être observés dans certaines applications pratiques, les AE peuvent également prendre un temps considérable pour localiser les optima globaux ou peuvent même être piégés dans des optima locaux. Par conséquent, cela créera des difficultés pour les chercheurs en termes d'équilibrage de la précision de la solution et du taux de convergence pour ce type de problèmes dans l'optimisation multi-objectifs.

Motivation et Problématique

Dans notre vie quotidienne, nous sommes constamment confrontés à une prise de décision à critères multiples dans de nombreux domaines différents. En fait, de nombreux problèmes du monde réel dans un large éventail d'applications, comme l'ingénierie, la finance, la logistique, la bio-informatique et bien d'autres, impliquent la tâche difficile d'optimiser simultanément plusieurs objectifs contradictoires. Par exemple, un gestionnaire de portefeuille d'investissement devra tenir compte à la fois de la maximisation des rendements et de la minimisation des risques lorsqu'il prend une décision d'investissement. C'est le cas du compromis risque-rendement, dans lequel un investissement ne peut générer des rendements plus élevés que s'il est soumis à un risque plus élevé. Les deux objectifs sont donc en conflit.

Un tel type de problème est communément appelé problème d'optimisation multi-objectif (MOOP). MOOP n'est pas un problème d'optimisation trivial car aucune solution unique n'est optimale pour tous les objectifs du problème. À ce titre, les méthodes de recherche prises en charge pour MOOP doivent être en mesure de trouver un ensemble de solutions alternatives représentatives du compromis entre les différents objectifs en conflit.

De plus, la présence d'un espace de recherche complexe ou même de grande dimension, avec des fonctions non linéaires, non différentiels pourrait entraîner des difficultés supplémentaires lors de la résolution des MOOP. En raison de ces défis, on constate que la plupart des méthodes déterministes rencontrent des difficultés lors du traitement des MOOP et ne sont donc pas en mesure d'obtenir des solutions raisonnables avec des ressources de calcul limitées.

Pour résoudre ces problèmes, l'utilisation de techniques de recherche stochastique est considérée comme une meilleure technique alternative aux méthodes déterministes pour traiter les MOOP. Les algorithmes d'optimisation évolutionnaires multi-objectifs (EMO), couramment utilisés pour trouver un ensemble de solutions représentant le front optimal de Pareto, sont souvent critiqués pour leur convergence lente, l'absence de preuve de convergence théorique et pour l'absence de critère de terminaison efficace.

Depuis une décennie, plusieurs études ont cherché à améliorer les performances de ces algorithmes basés sur les opérateurs évolutionnaires en améliorant certaines de ses parties par hybridation avec d'autres techniques d'optimisations et des méthodes de l'intelligence artificielle pour résoudre certains problèmes multi-objectifs et surmonter les limites de ces algorithmes.

Objectifs et Contributions

Afin de surmonter ces limites, plusieurs travaux d'amélioration ont été effectués sur les AE afin d'optimiser la solution et le taux de convergence tout en maintenant la diversité de la population pour différents types de problèmes mono et multi-objectifs.

Afin d'atteindre simultanément une vitesse de convergence rapide et une capacité de recherche globale efficace, nous avons exploré l'hybridation de différentes variantes des AE avec d'autres techniques de l'intelligence artificielle et de recherche locale dans l'objet d'assister et de guider les AE vers des performances meilleures.

Une méthode de recherche globale est rarement efficace pour trouver et garder l'équilibre entre l'exploitation et l'exploration de l'espace de recherche. La solution est de l'associer à une méthode dont la capacité d'exploration est très élevée avec 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.

Les méthodes hybrides ont rapidement gagné du terrain en réussissant à produire des meilleurs résultats pour de nombreux problèmes. Cependant, il y a encore plus de potentielles pour l'amélioration des AEs, et cette thèse a pour objectif principal le développement de nouveaux algorithmes évolutionnaires hybrides pour la résolution de problèmes d'optimisation mono et multi-objectifs.

Pour atteindre cet objectif, nous avons adopté la démarche suivante :

- Étudier les différentes méthodes d'optimisations évolutionnaires et directes proposées dans la littérature en analysant leurs principaux avantages et inconvénients en termes de convergence et cout de calcule.
- Étudier l'état de l'art concernant les méthodes évolutionnaires hybrides, y compris leurs fondements, les principaux mécanismes, les mesures de performance, les opérateurs, et leurs avantages en optimisation mono et multi-objectifs.

- Proposer et développer une stratégie hybride qui combine les stratégies évolutionnaire auto-adaptative (SA-ES) avec une méthode classique sans gradient (Simplex Nelder-Mead).
- Proposer et développer un schéma hybride qui optimise les performances de l'algorithme différentielle multi-objectifs (MODE) assisté par l'apprentissage par opposition (OBL).

Plan de la thèse

La thèse est organisée en quatre chapitre. Le plan est détaillé ci-dessous.

Le premier chapitre, à caractère introductif et bibliographique, est consacré à un état de l'art sur l'optimisation et les différentes approches couramment employées pour résoudre un problème d'optimisation en accordant une importance particulière aux méthodes utilisées dans nos travaux de recherche; notamment l'algorithme de Nelder-Mead, stratégie évolutionnaire, l'algorithme de l'évolution différentielle, ainsi qu'une description du métaphore évolutionnaire, les concepts de base et les différentes classes de l'optimisation multi-objectif.

Dans le chapitre II, nous élaborons un état de l'art sur les méthodes hybrides qui combinent les méthodes évolutionnaires, les algorithmes bio-inspirés et les heuristiques spécifiques, tout en décrivant quelques classifications de stratégies d'hybridation en optimisation mono et multi-objectifs. Nous terminons finalement ce chapitre par une conclusion dans laquelle nous proposerons des perspectives, sur la base des travaux effectués.

Dans le chapitre III, présentant notre première contribution, L'approche proposée tentent de trouver un compromis entre l'exploration et l'exploitation de l'espace de recherche en combinant l'algorithme de stratégie évolutionnaire auto-adaptative (SA-ES) avec une méthode de recherche locale (Nelder-Mead). Les résultats du nouveau schéma hybride montrent des performances supérieures en optimisation continue mono et multi-objectif en termes de vitesse de convergence et la capacité de surmonter les points locaux.

Le dernier chapitre présente une deuxième contribution, est un schéma hybride pour les problèmes d'optimisation multi-objectifs via l'utilité de deux algorithmes établis. Le schéma hybride proposé se compose de deux parties qui comprennent l'algorithme évolutionnaire multi-objectif basé sur l'évolution différentielle MODE assisté par la technique d'apprentissage par opposition OBL en deux phase. Le schéma hybride proposé est appliqué de certaines fonctions de référence et au problème du portefeuille et les résultats numériques montrent que le schéma hybride proposé fournit des résultats compétitifs supérieurs à ceux des algorithmes existants.

Pour conclure, le manuscrit expose finalement une vision globale de l'ensemble des contributions de cette recherche, ainsi que et les travaux futurs à envisager comme perspectives.

CHAPITRE I :

ÉTAT DE L'ART SUR L'OPTIMISATION ÉVOLUTIONNAIRE

I.1 INTRODUCTION

Les algorithmes évolutionnaires (Evolutionary Algorithms ou Evolutionary Computation) discipline de l'intelligence artificielle, sont des algorithmes d'optimisation stochastique simulant l'évolution naturelle. Ces algorithmes sont inspirés du paradigme de l'évolution darwinienne des espèces, telle qu'elle a été définie par Charles Darwin (Darwin, 1860). Les espèces naturelles sont en compétition pour survivre et seules les plus aptes survivent à la sélection naturelle. Au cours de leur évolution, ces mêmes individus auront la possibilité de transmettre leur patrimoine génétique à la génération suivante par reproduction. L'itération de ce principe permet pendant des générations de faire apparaître dans la population des individus plus adaptés à leur environnement.

Les algorithmes évolutionnaires sont particulièrement utiles pour la résolution des problèmes où les algorithmes classiques d'optimisation, d'apprentissage ou de conception automatique sont incapables de donner des résultats satisfaisants. L'introduction du principe évolutionnaire date pas d'hier (Fogel, 1966) (Holland, 1976), mais ce n'est que vers les années quatre-vingt-dix qu'on a commencé à profiter de la puissance des machines pour traiter des problèmes réels de taille importante.

Ce chapitre présente quelques concepts de base liés à l'optimisation évolutionnaire. L'objectif le plus important de ce chapitre est que le lecteur se familiarise avec les concepts de base, les définitions et les notations utilisées dans le reste de cette thèse et de fournir la base conceptuelle et théorique de l'optimisation globale. Une classification des différentes méthodes de programmation mathématique pour résoudre les problèmes d'optimisation non linéaire est présentée ainsi qu'une brève description sur les techniques d'optimisations et la métaphore darwinienne et les paradigmes les plus importants pour résoudre les problèmes d'optimisation mono-objectif. Le chapitre introduire aussi les concepts et méthodes disponibles au sein de l'optimisation évolutionnaire multi-objectif et décrit les avantages et les inconvénients de l'utilisation de ces approches et leurs domaines d'applications.

I.2 NOTIONS D'OPTIMALITÉ

En mathématiques, l'optimisation se réfère au processus de détermination du point minimum ou maximum d'une fonction en choisissant systématiquement les valeurs de ses variables de décision correspondantes dans un certain espace de recherche. Pour être plus précis, un problème d'optimisation générique peut être formellement énoncé comme suit.

I.2.1 Définition (Problème d'optimisation)

Trouver le vecteur x qui minimise la fonction $f(x)$ tels que $x \in \Omega$, où $\Omega \subseteq R^n$ est la région faisable qui satisfait les p contraintes d'inégalité (Kuhn & Tucker, 2014) :

$$\begin{cases} g_i(x) \leq 0;\ i = 1\ldots,p \\ \text{Et les } q \text{ contraintes d}'\text{égalité :} \\ h_j(x) = 0; \qquad j = 1\ldots,q \end{cases} \tag{I.1}$$

Où Ω est le sous-espace des solutions réalisables et f est communément appelé fonction objective. La solution faisable $x* \in \Omega$ qui correspond à la valeur minimale de la fonction objective dans tout l'espace de recherche est appelé optimum global.

La Figure I.1 montre un espace de conception bidimensionnel où la région réalisable est indiquée par des lignes hachurées. L'espace de conception bidimensionnel est délimité par des lignes droites montré sur la figure. C'est le cas lorsque les contraintes sont linéaires. Cependant, les contraintes peuvent également être non linéaires et l'espace de conception sera délimité par des courbes dans ce cas.

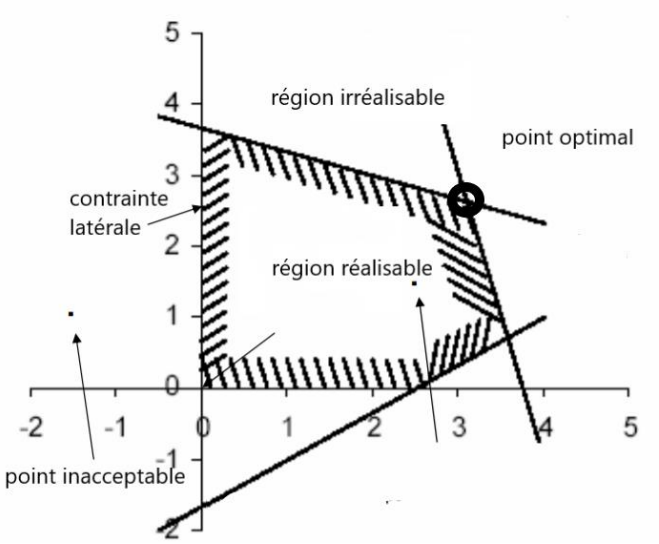

Figure II-1 Région réalisable dans un espace de conception à deux dimensions (Janga Reddy & Nagesh Kumar, 2012)

La difficulté d'un problème d'optimisation est déterminée par les différents types de relations mathématiques entre la fonction objective, les contraintes et l'étendue des variables de décision. Pour comprendre la complexité de la résolution d'un problème d'optimisation, les définitions suivantes sont introduites :

I.2.2 Définition (Minimum Global)

Étant donné une fonction f : $\Omega \subseteq R^n \rightarrow R$, $\Omega \neq \emptyset$, pour $x^* \in \Omega$ La valeur $f^* = f(x^*) > -\infty$ est appelé minimum global, si et seulement si (Kuhn & Tucker, 2014) :

$$\forall x \in \Omega : f(x*) <= f(x) \qquad \text{(I.2)}$$

Où vecteur x^* est un point minimum global.

I.2.3 Définition (Minimum Local)

Étant donné une fonction $f : \Omega \subseteq R^n \rightarrow R$, une solution $x^0 \in \Omega$ est appelée point minimum local, si et seulement si (Kuhn & Tucker, 2014) :

$$\forall x \in \Omega : f(x^0) <= f(x), \text{ tels que : } ||x - x^0|| < \varepsilon \qquad \text{(I.3)}$$

Où le $\varepsilon > 0$ et la valeur $f(x^0)$ est appelée minimum local.

I.2.4 Définition (Fonction Convexe)

Une fonction f : $R^n \rightarrow R$ est appelé convexe, si pour deux vecteurs $x_1, x_2 \in R^n$ (Kuhn & Tucker, 2014):

$$f(\lambda x_1 + (1-\lambda)x_2) <= \lambda f(x_1) + (1-\lambda) f(x_2) \quad \text{(I.4)}$$

Où $\lambda \in [0,1]$.

De cette façon, si la fonction objective et toutes les contraintes sont convexes, il est possible de trouver de manière exacte la solution optimale à l'échelle global, et de résoudre le problème jusqu'à un très grand nombre de variables de décision. D'autre part, si la fonction est non-convexe, le problème est beaucoup plus difficile à résoudre et il devient beaucoup plus difficile de localiser la région faisable et, par conséquent, de trouver l'optimum global.

En règle générale, les problèmes d'optimisation impliquent de multiples exigences de conception, souvent contradictoires. En conséquence, la recherche d'une solution optimale est difficile. La Figure I.2 illustre des vues typiques des espaces de solution dans les problèmes d'optimisation locaux et globaux. Pour les problèmes d'optimisation locale (Figure I.2a), une seule solution optimale existe, mais ces problèmes sont pour la plupart idéalistes. Les problèmes mondiaux (figure I.2b), en revanche, impliquent une combinaison de nombreuses solutions optimales locales et globales. En fait, la majorité des tâches d'optimisation sont de type global. Quoi qu'il en soit, les méthodes d'optimisation devraient permettre de trouver la solution optimale globale avec un coût de calcul minimal possible.

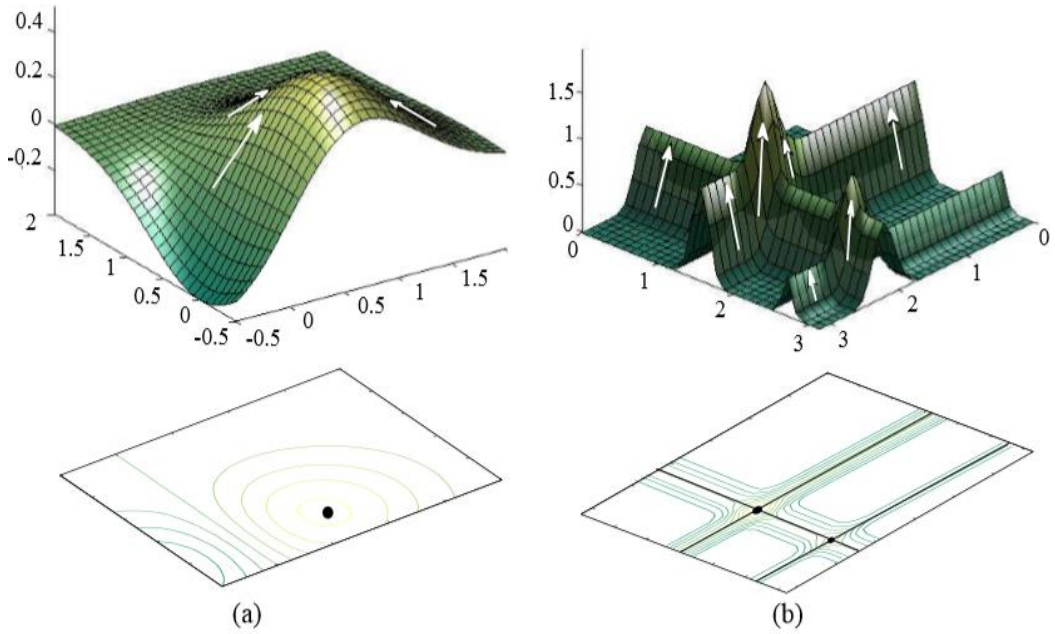

Figure II-2 Une illustration des paysages d'optimisation locaux et globaux (Bashir, 2014).

I.3 CLASSIFICATION DES PROBLÈMES D'OPTIMISATION

Les problèmes d'optimisation peuvent être classés en fonction du type de contraintes, de la nature des variables de conception, de la nature des équations impliquées, de la nature déterministe des variables, de la valeur admissible des variables de conception, et du nombre des fonctions objectifs. Ces classifications sont brièvement discutées ci-dessous : (Chambers & Fletcher, 2001).

I.3.1 Classification basée sur l'existence de contraintes

Qui concerne l'existence ou non de limitations sur les variables (contraintes) qui peuvent être simplement des bornes géométriques ou des égalités ou inégalités non linéaires. Dans cette catégorie, les problèmes d'optimisation peuvent être classés en deux groupes comme suit (Bellman, 1954) :

a) Problèmes d'optimisation à contraintes : soumis à une ou plusieurs contraintes.

b) Problèmes d'optimisation sans contraintes : dans lesquels aucune contrainte n'existe.

I.3.2 Classification basée sur les variables de décision

Sous cette classification, les fonctions objectives peuvent être classées comme des problèmes de programmation entiers et à valeur réelle (Astolfi, 2008).

a) Problème de programmation entière (discret)

Si certaines ou toutes les variables de conception d'un problème d'optimisation sont limitées pour prendre uniquement des valeurs entières (ou discrètes), le problème est appelé un problème de programmation entier. Par exemple, l'optimisation consiste à trouver le nombre d'articles nécessaires pour une opération avec le moins d'effort. Ainsi, la minimisation de l'effort requis pour l'opération étant l'objectif, les variables de décision, c'est-à-dire le nombre d'articles utilisés ne peuvent prendre que des valeurs entières. D'autres restrictions sur le nombre minimum et maximum de ressources utilisables peuvent être imposées.

b) Problème de programmation à valeur réelle (continue)

Un problème à valeur réelle est celui dans lequel on cherche à minimiser ou à maximiser une fonction réelle en choisissant systématiquement les valeurs des variables réelles dans un ensemble autorisé. Lorsque l'ensemble autorisé ne contient que des valeurs réelles, cela s'appelle un problème de programmation à valeur continue.

I.3.3 Classification basée sur la nature des équations impliquées

En fonction de la nature des équations de la fonction objectif et des contraintes, les problèmes d'optimisation peuvent être classés comme des problèmes de programmation linéaires, non linéaires, géométriques et quadratiques. La classification est très utile d'un point de vue informatique car de nombreuses méthodes spéciales prédéfinies sont disponibles pour une solution efficace d'un type particulier de problème (Dan, 2013).

a) Problème de programmation linéaire

Si la fonction objectif et toutes les contraintes sont des fonctions «linéaires» des variables de conception, le problème d'optimisation est appelé problème de programmation linéaire (LPP). Un problème de programmation linéaire est souvent énoncé sous la forme standard:

$$\text{Trouver } \mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{Bmatrix}, \text{ Qui maximise } f(\mathbf{X}) = \sum_{i=1}^{n} c_i x_i \qquad \text{(I.5)}$$

Subject to the constraints $\sum_{i=1}^{n} a_{ij} x_i = b_j$, $j = 1, 2, \ldots, m$

$x_i \geq 0$, $j = 1, 2, \ldots, m$

Ou c_i, a_{ij}, et b_j sont des constantes.

b) Problème de programmation non linéaire

Si l'une des fonctions parmi les fonctions d'objectif et de contrainte est non linéaire, le problème est appelé problème de programmation non linéaire (NLP). Il s'agit de la forme la plus générale d'un problème de programmation et tous les autres problèmes peuvent être considérés comme des cas particuliers du problème NLP.

c) Problème de programmation géométrique

Un problème de programmation géométrique (GMP) est un problème dans lequel la fonction objectif et les contraintes sont exprimées sous forme de polynômes dans X. Une fonction $h(X)$ est appelée un polynôme (avec des termes) si h peut être exprimé comme

$$h(X) = c_1 x_1^{a11} x_2^{a21} \cdots x_n^{an1} + c_2 x_1^{a12} x_2^{a22} \cdots x_n^{an2} + \cdots + c_m x_1^{a1m} x_2^{a2m} \cdots x_n^{anm} \qquad \text{(I.6)}$$

Où c_j ($j = 1, \cdots, m$) et a_{ij} ($i = 1, \cdots, n$ and $j = 1, \cdots, m$) sont des constantes avec $c_j \geq 0$ and $x_i \geq 0$.

Ainsi, les problèmes GMP peuvent être posés comme suit :

Trouvez X qui minimise $f(x) = \sum_{j=1}^{N_0} c_j \left(\prod_{i=1}^{n} x_i^{a_{ij}} \right)$, $c_j > 0$, $x_i > 0$ (I.7)

Sujet à

$$g_k(x) = \sum_{j=1}^{N_k} a_{jk} \left(\prod_{i=1}^{n} x_i^{q_{ijk}} \right) > 0, \qquad a_{jk} > 0, \quad x_i > 0, \; k = 1,2,\ldots..,m$$

Où N_0 et N_K désignent le nombre de termes dans la fonction objectif et dans la $k^{\text{ème}}$ fonction de contrainte, respectivement.

d) Problème de programmation quadratique

Un problème de programmation quadratique est le problème de programmation non linéaire qui se comporte le mieux avec une fonction objectif quadratique et des contraintes linéaires et il est concave (pour les problèmes de maximisation). Il peut être résolu en modifiant convenablement les techniques de programmation linéaire. Il est généralement formulé comme suit (Madani BEZOUI, 2011):

$$f(\boldsymbol{X}) = c + \sum_{i=1}^{n} q_i x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} Q_{ij} x_i x_j \qquad \text{(I.8)}$$

Sujet à $\sum_{i=1}^{n} a_{ij} x_i = b_j$, $j = 1,2,\ldots.,m$

$x_i \geq 0$, $i = 1,2,\ldots.,n$

Où c, q_i, Q_{ij}, a_{ij}, et b_j sont des constantes.

I.3.4 Classification basée sur le nombre de fonctions objectives

Dans cette classification, les fonctions objectifs peuvent être classées comme des problèmes de programmation à objectif unique et à objectifs multiples (Dan, 2013).

a) Problème de programmation à objectif unique :

Dans lequel il n'y a qu'une seule fonction d'objectif à optimiser (minimiser ou maximiser).

b) Problème de programmation multi-objectif :

Un problème de programmation multi-objectif peut être déclaré comme suit :

Trouvez X qui minimise $f_1(X), f_2(X),... f_k(X)$ (I.9)

Sujet à $\boldsymbol{g_j(X) \leq 0,\ j = 1,2,\ldots,m}$

Où $f_1, f_2, \ldots f_k$ désignent les fonctions d'objectif à minimiser simultanément.

Par exemple, dans certains problèmes de conception, il pourrait être nécessaire de minimiser le coût et le poids de l'élément structurel pour l'économie et, en même temps, de maximiser la capacité de charge sous des contraintes données.

I.4 TECHNIQUES D'OPTIMISATION

Au fil des ans, un grand nombre de techniques de programmation mathématique pour résoudre les problèmes d'optimisation non linéaire ont été proposées. Des études exhaustives de ces méthodes de programmation mathématique peuvent être trouvées dans (Conn et al., 2009),(Bard, 1997). Le développement de ces méthodes d'optimisation a été motivé par différents problèmes dans le monde réel. Au fil des ans, différentes taxonomies pour classer les méthodes d'optimisation ont été proposées (voir, par exemple, celles présentées dans (Deb et al. , 2002), (Bard, 1997). Ces méthodes sont classées en deux catégories différentes : les techniques de programmation mathématique et les techniques stochastiques, voir Figure I.3.

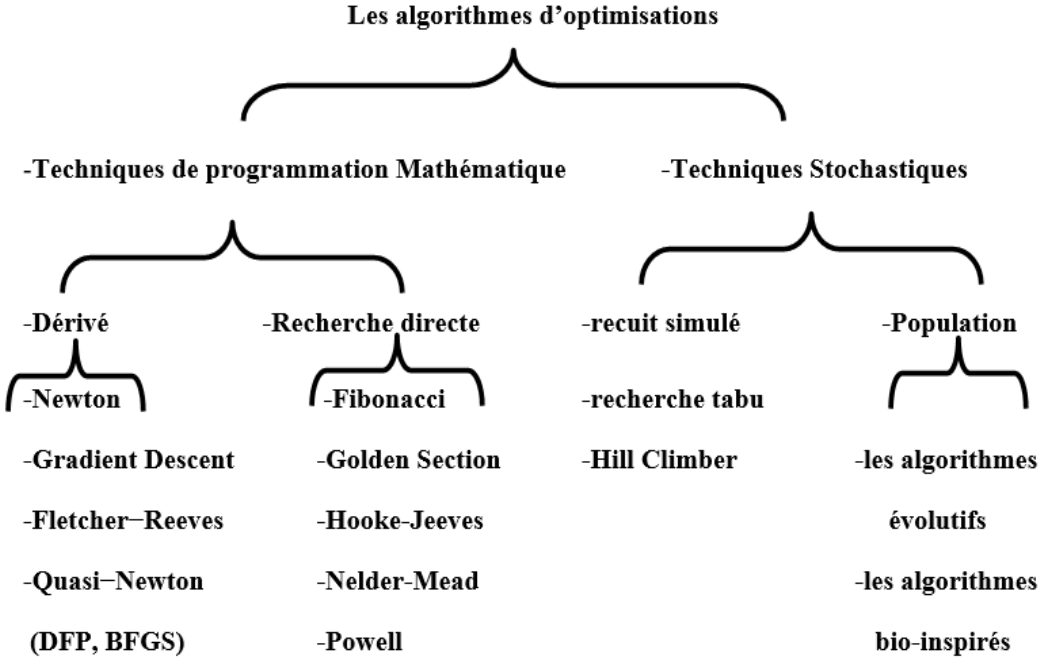

Figure II-3 taxonomie des techniques d'optimisation (Janga Reddy & Nagesh Kumar, 2012)

I.4.1 Méthodes Déterministe

Les méthodes de programmation classique ou mathématique sont des algorithmes déterministes caractérisés par des règles spécifiques pour passer d'une solution à l'autre. Ces méthodes ont été utilisées pendant un certain temps et ont été appliquées avec succès dans de nombreux problèmes d'ingénierie. Il existe différentes variantes de ces méthodes (Conn et al., 2009), (Nocedal & Wright, 2000), (Ravindran et al., 2007). Sur la base de leurs fondements conceptuels, ces méthodes peuvent être divisées en deux grands groupes : les méthodes de programmation mathématique de gradient et de non-gradient.

Généralement, les méthodes déterministes résolvent des problèmes formulés en fonction de leurs propriétés analytiques et de certaines hypothèses théoriques. En outre, lorsqu'il existe des informations préalables disponibles sur le problème étudié, les algorithmes de branchement et de liaison (Branch-and-Bound) dans l'optimisation globale déterministe sont les méthodes les plus capables de rechercher les minima globaux (Antoniou & Lu, 2007). Les méthodes déterministes peuvent trouver un minimum global avec une tolérance prédéfinie en cas de calcul exact sur une longue période. Dans ce type d'optimisation, la plage de toutes les variables définit l'espace de recherche. De plus, ces méthodes se caractérisent par une assurance théorique de trouver la solution globale. Cela signifie que la valeur de la fonction objective lors de l'application d'un optimum local ne diffère que de petite valeur par rapport à l'optimum globale. Les anciens algorithmes déterministes souffrent de certaines limitations telles que des limites de bornes, et ils ne peuvent pas traiter des problèmes pratiques avec plus de dix variables. Cependant, certaines nouvelles méthodes déterministes ont été améliorées pour obtenir de bons résultats, comme la méthode Hit-and-Run (Törn et al., 1999).

Malheureusement, aucune de ces méthodes mathématiques ne garantit la convergence à l'optimum global lorsqu'il s'agit du problème général d'optimisation non linéaire. Ces limites des méthodes déterministes ont motivé la transition vers les méthodes stochastiques.

I.4.2 Méthodes Stochastiques

Les méthodes non classiques ou stochastiques, sont des algorithmes qui utilisent généralement des règles de transition probabilistes. Les algorithmes de recherche stochastique sont conçus pour les problèmes de bruit aléatoire inhérent ou les problèmes déterministes résolus par l'aléatoire injecté (Glover et al., 2000). La recherche privilégie les designs avec de meilleures performances. Une caractéristique importante des algorithmes de recherche stochastiques est qu'ils peuvent effectuer une recherche étendue de l'espace de conception et ainsi éviter les optima locaux. De plus, les algorithmes de recherche stochastiques ne nécessitent pas de gradients pour guider la recherche, ce qui les rend bien adaptés aux problèmes discrets (Fu, 2015). Cependant, il n'y a pas de condition nécessaire pour une solution optimale et l'algorithme doit s'exécuter plusieurs fois pour s'assurer que les solutions obtenues sont robustes (Eberhart & Kennedy, 1995).

Ces méthodes comprennent le calcul évolutionnaire, qui est devenu très populaire, en particulier lorsqu'il s'agit de problèmes d'optimisation difficiles (Antoniou & Lu, 2007). Ce type d'approches, en comparaison, sont nouveaux et très utiles, parce qu'ils ont certaines propriétés que les algorithmes déterministes n'ont pas, et elles ne nécessitent pas d'informations préalables du problème. En effet, ils n'ont pas besoin ni d'un point de recherche initial ni d'informations de gradient, (Gheraibia & Moussaoui, 2013).

I.5 LES ALGORITHME ÉVOLUTIONNAIRES

I.5.1 Présentation

Les algorithmes évolutionnaires (AE) sont des méthodes inspirées de la sélection naturelle (en particulier, le principe de « survie du plus fort »). Les AE sont des techniques basées sur l'utilisation d'une population, c'est-à-dire qu'elles fonctionnent sur un ensemble de solutions au lieu d'opérer sur une solution à la fois, comme les méthodes d'optimisation traditionnelles. À chaque itération d'une évaluation générale, un mécanisme de sélection concurrentiel qui tend à préserver les solutions les plus aptes est appliqué. Les solutions avec les valeurs de la fonction objective les plus élevées ont la plus grande probabilité d'être recombiné avec d'autres solutions pour mélanger l'information et former de nouvelles solutions, qui devraient être meilleures que leurs prédécesseurs. Ce processus est répété jusqu'à ce qu'une condition de terminaison soit atteinte.

I.5.2 La Métaphore Darwinienne

L'évolution est un processus issu du paradigme néodarwinien d'inspiration biologique, c'est-à-dire le principe de survie du plus fort (Fogel & Ghozeil, 1997). Les algorithmes évolutionnaires (EA) sont conçus pour imiter les mécanismes intrinsèques de l'évolution naturelle et pour fournir progressivement des solutions améliorées à un large éventail de problèmes d'optimisation. Il est usuel d'introduire les AEs au moyen de la métaphore "darwinienne" (Le Riche et al., 2007) Un point X de S est assimilé à un individu d'une espèce. La théorie de Darwin est que les espèces évoluent en étant soumises à deux mécanismes, les variations aveugles lors de la reproduction, et la sélection naturelle qui favorise les individus les plus adaptés, permettant ainsi à terme l'émergence d'espèces adaptées à leur environnement. Lorsque les individus sont des points de S et que l'adaptation à l'environnement est quantifiée par la valeur de la fonction objective f, une telle évolution peut alors être vue comme la résolution d'un problème d'optimisation.

Les principaux termes de cette métaphore sont récapitulés en Table. I.1. Plus généralement, on trouvera dans la littérature des AEs des allusions fréquentes à des concepts de la biologie, de la génétique et de l'évolution. Cette métaphore est souvent utilisée car elle facilite l'explication et peut même stimuler l'intuition. Ces concepts sont illustrés dans la Figure I.4.

Une structure ou un individu est une solution codée à un problème. Typiquement, un individu est représenté comme une chaine de code similaire à un génotype biologique. Ce génotype définit un organisme individuel lorsqu'il est exprimé (décodé) en un phénotype. Un génotype est composé d'un ou plusieurs chromosomes, ou` chacun est composé de gènes distincts. Ainsi, chaque individu décodé est un ensemble de paramètres utilisés comme entrée de la fonction objectif. Finalement, un ensemble donné de chromosomes est appelé une population. Tout comme dans la nature, les opérateurs évolutionnaires manipulent une population d'individus et tentent de générer des solutions. Les trois opérateurs évolutionnaires de base pour les AEs sont : la mutation, le croisement (recombinaison) et la sélection. Chaque parent est coupé et recombiné avec une pièce de l'autre (Goldberg, 1991). Les meilleurs

individus dans la population sont sélectionnés pour devenir membres de la prochaine génération.

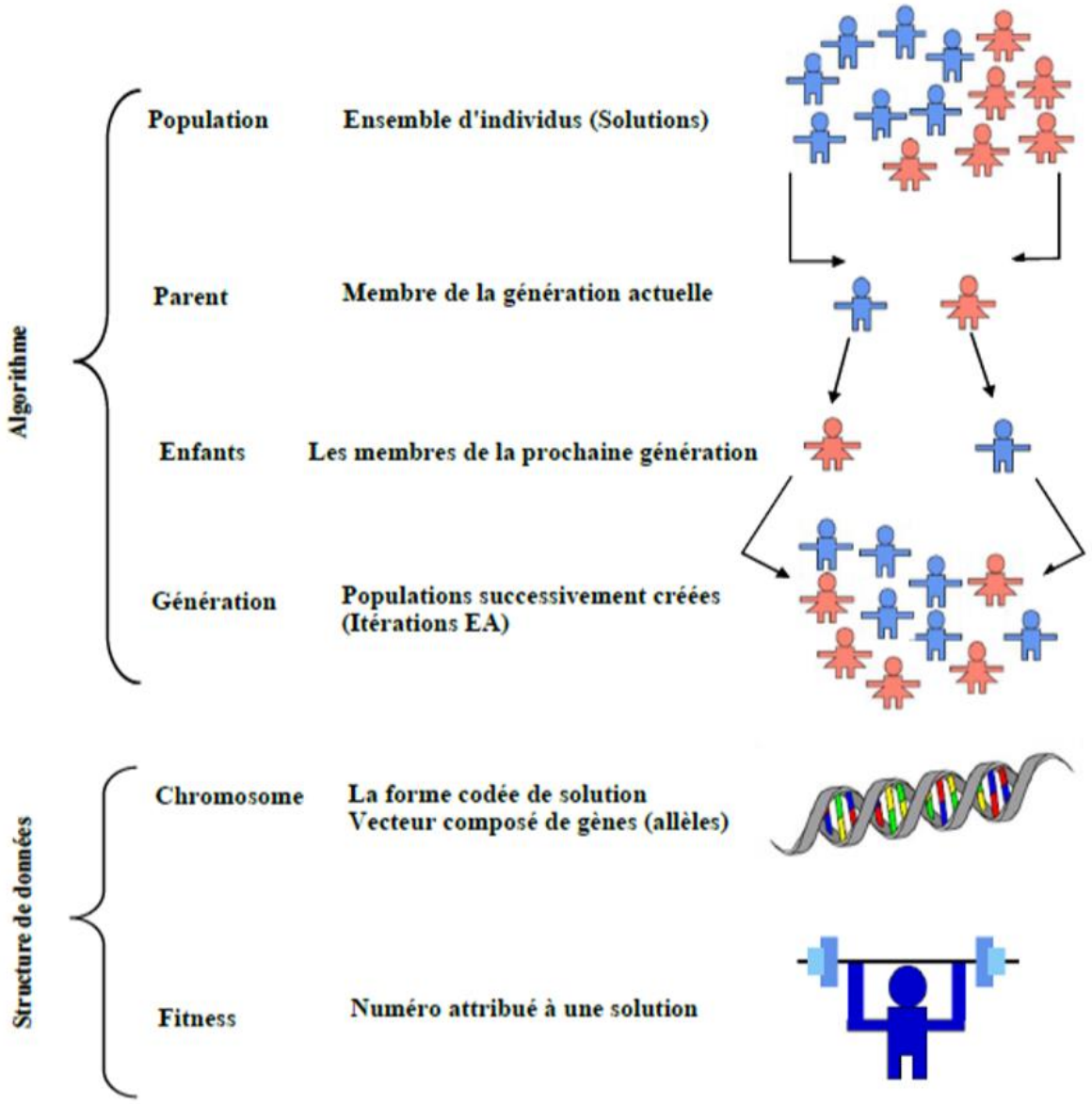

Figure II-4 Composants clés de l'AE (C. A. *Coello* Coello, 2006)

Tableau II-1 La métaphore darwinienne et nomenclature évolutive (Le Riche et al., 2007)

Terminologie	Symbole	Description
Individu	X	Une solution échantillon arbitraire, également appelée solution candidate
Gène	-	En biologie, c'est une unité héréditaire de création, en évolution simulée c'est une unité de représentation qui constitue une solution individuelle (Une composante du codage de X)

Chromosome	-	Une structure de données (par exemple une chaîne binaire) de longueur fixe de gènes codant une solution
Allèle	-	Valeur possible qu'un gène peut prendre à n'importe quel locus d'un chromosome, par ex. une valeur de 0 ou 1 pour un bit binaire
Population	P	$P = \{x1,...,x^{\mu}\} \in S^{\mu}$ Un ensemble de configuration de solution candidate subir un cycle évolutif
Parent	P_a, P_{ai}	Un ensemble ou tout autre exemple de solution impliqué dans un processus évolutif susceptible de produire de nouvelles solutions
Descendant	O, O_i	Une nouvelle solution ou un ensemble de solutions formé en faisant évoluer une seule solution ou un groupe de solutions parentes
Espèce	-	Une catégorie / variété de solutions susceptibles de subir certains processus évolutifs
Niche	-	Un lot de solutions localisées partageant des fonctionnalités communes et / ou occupant une région spécifique d'un espace de recherche évolutif
Phénotype	$\mathfrak{p}$	Une solution dans sa forme originale qui peut être évaluée
Espace phénotypique	$\boldsymbol{\mathfrak{p}}$	L'espace du problème d'origine dans lequel les solutions sont évaluées
Génotype	G	Une représentation de la solution dans un codage donné utile pour le croisement et la mutation
Espace génotype	$\mathfrak{g}$	L'espace de codage ou de représentation de la solution, par ex. un espace binaire, à valeur réelle, etc.
Generation	**G**	Un cycle/itération/époque complet d'un modèle évolutif
Fitness	$\mathcal{F}$	Une qualité de solution candidate qui se traduit par des chances de survie lors de la sélection

Dans le cadre de la métaphore évolutionnaire, l'espace d'application des opérateurs de variation, et l'espace de recherche sur lequel est calculée la fonction objectif (et donc dans lequel a lieu la sélection) sont qualifiés de "génotypiques" et de "phénotypiques", respectivement Figure I.3.

Opérations phénotypiques (sélection et remplacement) : La sélection effectue un tirage biaisé par la performance (la fonction fitness *f*) dans une certaine population. L'idée de la sélection en probabilité est qu'il peut être bénéfique pour l'optimisation de garder une probabilité non nulle de sélectionner des individus de performance moyenne voire mauvaise. En effet, dans les problèmes difficiles à optimiser, il y a généralement une mauvaise corrélation entre la valeur de la performance et la distance aux optima. Il peut donc y avoir un avantage à ne pas

sélectionner de manière déterministe à partir de f. L'exemple le plus classique de sélection est le tournoi de taille τ. Son pseudocode est le suivant (Le Riche et al., 2007) :

```
SelectionTournoi(P)        % En général, P ⊂S^μ,
                           % population de parents
      Choisir τ individus par tirage uniforme sans remise dans P
      Retourner l'individu de performance optimale
Fin
```

La procédure est répétée dans les AEs ou` plusieurs individus sont sélectionnés.

Le remplacement, comme son nom l'indique, remplace certains des parents par certains des enfants en fonction de leurs fitness f respectives.

Opérations de variations génotypiques : Par opérateurs de variation, nous entendons ici un ensemble croisement, produisant x', puis mutation produisant un "enfant" x^e. L'opérateur de croisement s'applique à un ensemble de ρ points préalablement sélectionnés, soit directement par l'opérateur de sélection, soit préalablement par le remplacement (les ρ points sont alors tirés au hasard dans la population courante P^t). Certains AEs n'utilisent que la mutation et pas le croisement. Dans ces cas, x' est la copie d'un individu sélectionné. L'objectif général du croisement est de recombiner des caractéristiques de points préalablement sélectionnés pour créer de nouveaux points. En particulier, on demande au croisement de transmettre aux enfants les caractéristiques communes aux ρ individus sélectionnés. L'objectif de la mutation est d'effectuer des perturbations stochastiques de la population courante. Sans la mutation, l'action combinée de la sélection (ou/et du remplacement) et du croisement aboutit à une uniformisation rapide de la population autour d'un point dépendant de la population initiale, ce qui ne peut pas constituer un processus d'optimisation globale. Une telle uniformisation de P hors des optima globaux est appelée convergence prématurée.

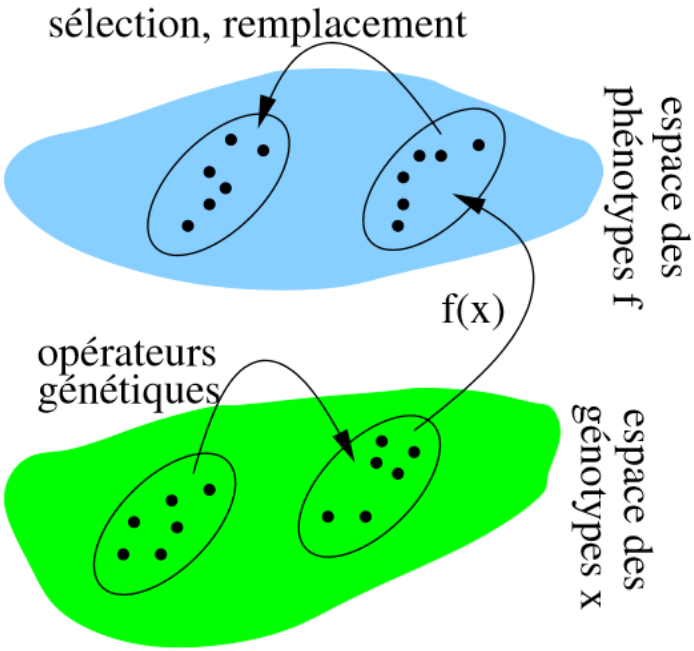

Figure II-5 espaces d'application des opérateurs évolutionnaire (Le Riche et al., 2007)

I.5.3 Squelette D'un Algorithme Évolutionnaire

Le pseudo code d'un algorithme évolutionnaire est montré dans Algorithm1. Les principaux composants, procédures et opérateurs qui doivent être spécifiés afin de définir une AE particulière sont décrits ci-dessous (Martínez & Coello Coello, 2012) :

```
Algorithme 1 : Schéma général d'un algorithme évolutionnaire AE
1 debut
2    t = 0;
3    INITIALISATION : P(t)∈ Iμ
4    Evaluation: Φ(P(t));
5    tant que (ι(P(t)) ≠ JUSTE) faire
6        Recombination: P'(t) = r(P(t))
7        Mutation: P''(t) = m(P'(t));
8        Evaluation: Φ(P''(t));
9        Selection: P(t+1) = s(P(t)∪ P''(t));
10       t = t+1;
11    Fin
12 Fin
```

Représentation (codage des individus) : Pour lier un problème du monde réel à un AE, nous devons utiliser une représentation particulière des variables de décision. Il y a deux niveaux de représentation utilisés dans un AE : génotypique et phénotypique. Le génotype est l'encodage adopté dans le chromosome et ses gènes correspondants. Le phénotype est le résultat du décodage des valeurs du chromosome dans l'espace variable de décision du problème.

La fonction de fitness (fonction objective) : La fonction objective d'un individu est liée à la valeur objective de la fonction et représente la tâche à résoudre par l'algorithme évolutif. La fonction d'évaluation attribue une mesure de qualité à chaque individu qui permet de la comparer aux autres solutions de la population.

Population : La population est l'ensemble des solutions adoptées par l'AE pour effectuer la recherche. La population est responsable du maintien de la diversité, de sorte que l'AE ne se coince pas dans l'optimisation locale. Ainsi, il est important que la population initiale contienne des solutions qui sont réparties sur tout l'espace de recherche.

Mécanisme de sélection des parents : Ce mécanisme permet aux meilleurs individus de la population de devenir parents de la prochaine génération et guide la recherche de solutions avec une meilleure fonction objective. Une personne est choisie comme parent si elle survit au processus de sélection. Différents types de schémas de sélection sont possibles. Par exemple : proportionnel, stochastique, déterministe ou basé sur des tournois.

Operateurs de variation (recombinaison et mutation) : Leur fonction est de modifier la façon dont les parents sont combinés pour former la progéniture. Le multi-croisement (ou opérateur de recombinaison) utilise deux parents ou plus pour générer une ou deux descendantes. Le principe principal derrière la recombinaison est de produire une progéniture qui combine les parents sélectionnés pour former de meilleures personnes dans l'espace de recherche. L'opérateur de mutation est appliqué à une seule solution et modifie légèrement les informations génétiques de la progéniture. L'idée générale de l'opérateur de mutation est de permettre des sauts (ou se déplacer brusquement) d'une région à l'autre dans l'espace de recherche.

Mécanisme de sélection des survivants (remplacement) : Ce mécanisme aide l'AE à distinguer entre les individus en fonction de leur fonction objective ou de qualité, favorisant ceux qui ont la plus haute qualité.

Un AE est une stratégie d'optimisation basée sur la population structurée avec une procédure itérative (Figure I.6). Un algorithme révolutionnaire commence par une étape d'initialisation, où les génomes d'une taille de population spécifique sont généralement initialisés au hasard. Après l'étape d'initialisation, l'algorithme boucle en quelques étapes jusqu'à ce qu'un point de terminaison soit atteint. Les phénotypes de la population de génomes sont évalués pour acquérir des valeurs de fitness pour chaque génome. À partir de là, l'algorithme évolutionnaire passe par les étapes suivantes :

1- Sur la base de ces valeurs d'ajustement, un opérateur de sélection est utilisé pour sélectionner les parents qui peuvent créer la source à partir de la prochaine génération.

2- À leur tour, la progéniture est produite et leur génome est modifié en fonction de la mutation / des opérateurs croisés.

3- L'adéquation de la progéniture est ensuite déterminée sur la base de leur phénotype.

4- Le placement fonctionne-t-il pour déterminer la façon dont la progéniture est intégrée à la population existante.

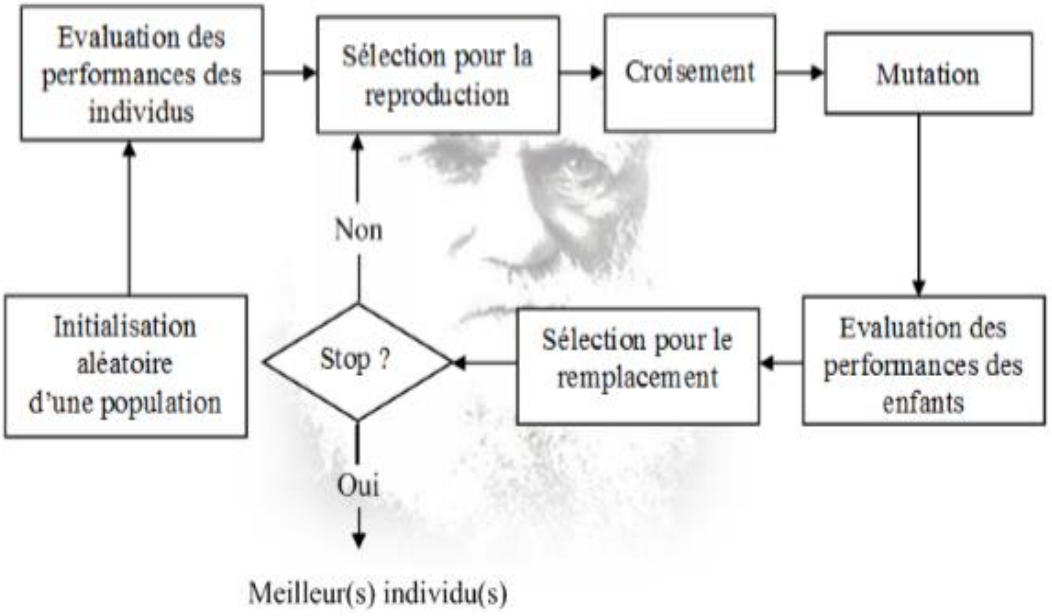

Figure II-6 Procédure de base d'un algorithme évolutionnaire (Coello.Coello et al., 2006)

Ces quatre étapes sont répétées jusqu'à ce que l'algorithme ait atteint un certain type d'exigence de terminal, qui est généralement défini par un nombre maximal de générations ou un nombre maximal d'évaluations.

Il existe deux variantes principales d'AE : un AE générationnel et un AE stationnaire. Un algorithme évolutionnaire générationnel remplace entièrement les enfants d'une population par la population existante (Petrowski & Ben Hamida, 2016). Un AE stationnaire ne remplace les individus de la population existante que lorsqu'ils sont dépassés par les enfants. Ainsi, les mises

en œuvre ne varient que dans leur opérateur de remplacement. Dans une mise en œuvre stationnaire, au lieu de rejeter entièrement la population existante d'individus et de la remplacer par les enfants, l'opérateur de remplacement compare l'adéquation de l'enfant spécifique avec les individus de la population existante et ne remplace un individu existant par un enfant que si la valeur de l'adéquation est plus élevée. Un algorithme stationnaire peut ainsi être comparé à une population d'individus immortels qui ne peuvent être remplacés que lorsqu'ils sont hors compétition. Il existe cependant diverses manières dont les algorithmes d'état stationnaire peuvent encore se débarrasser de certains individus « indésirables », comme dans la suppression d'individus de la population (Jiang & Yang, 2017).

De plus, l'opérateur de sélection des algorithmes évolutionnaires peut être ajusté pour donner aux meilleurs individus une plus grande chance de créer des enfants par rapport aux individus moins performants. Les opérateurs de sélection peuvent être complètement aléatoires, mais généralement la sélection du tournoi ou la sélection du choix proportionnel (également connue sous le nom roulette wheel) est utilisée (De Jong, 2007). En outre, ces techniques peuvent également être mises en œuvre dans l'opérateur de remplacement, où les enfants et la population existante peuvent être soumises à une sélection proportionnelle au tournoi ou au fitness proportionnel afin de réduire la taille de la population à la limite de la population.

L'élitisme est une autre caractéristique couramment mise en œuvre dans les algorithmes évolutionnaires (Du, et al., 2018). L'élitisme retient généralement un certain nombre d'individus parmi les meilleurs (élite) à la prochaine génération sans modifier leur code génétique. Cela augmente généralement la robustesse de l'AE car le meilleur génome n'est pas perdu. Les algorithmes stationnaires ne nécessitent pas d'opérateur d'élitisme, car l'élite ne peut être remplacée que lorsqu'un meilleur individu la remplace, favorisant ainsi intrinsèquement les élites dans la population.

Étant donné que les AE sont des méthodes stochastiques, rien ne garantit que les meilleures solutions finales aient atteint l'optimum global au moment où la condition d'arrêt a été atteinte. Par conséquent, la condition de résiliation est une question importante dans les AE. Si le problème d'optimisation a une solution optimale connue, l'AE doit s'arrêter lorsque la fonction objective atteint le niveau de précision souhaité (Neri, et al., 2012). Si ce n'est pas le cas, l'utilisateur doit déterminer le nombre de générations autorisées, le temps maximal autorisé de l'unité centrale de traitement (CPU), le nombre maximal d'évaluations de la fonction objective ou un autre critère similaire.

I.6 PARADIGMES DE CALCUL ÉVOLUTIONNAIRE

Dans le calcul évolutionnaire (CE), il existe 04 paradigmes principaux :

1) Stratégie évolutionnaire (Evolution Strategy : ES).

2) Programmation génétique (Genetic Programming: GP).

3) Algorithme génétique (Genetic Algorithm : GA).

4) Evolution différentielle (Differentiel Evolution : DE).

Dans ce qui suit, une brève description des paradigmes les plus importants est présentée (voir Figure I.7).

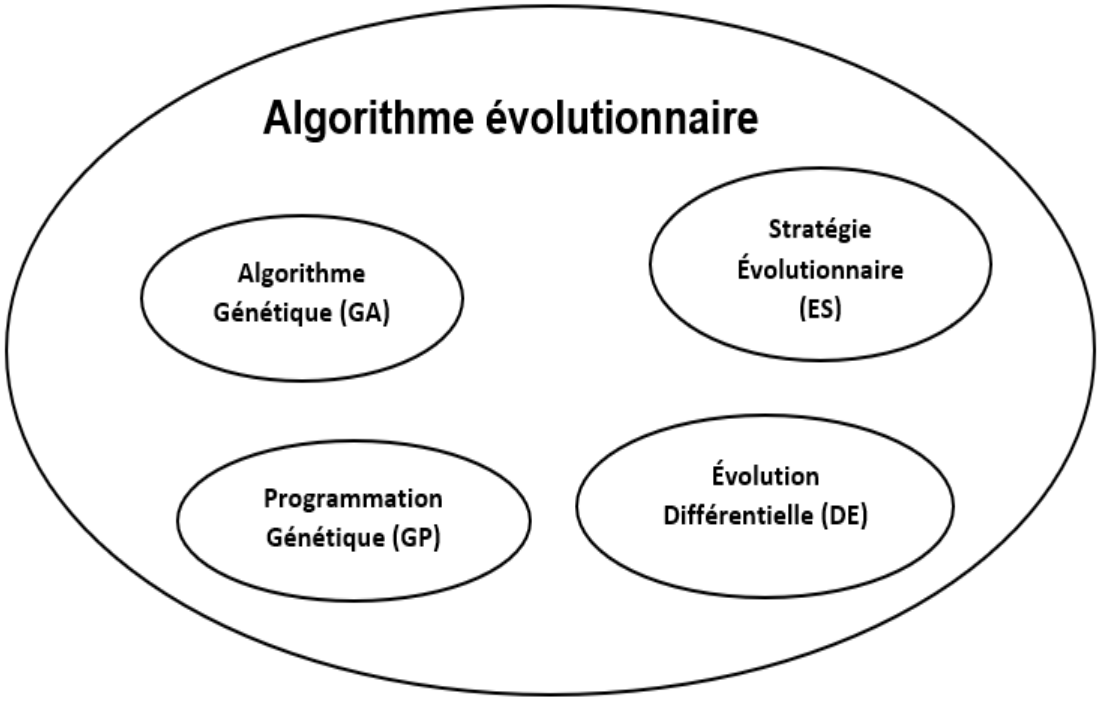

Figure II-7 Classification des algorithme évolutionnaires (Rai & Tyagi, 2013)

I.6.1 Stratégies D'évolution (ES)

Les ES ont été proposées en 1965 par (Rochenberg, 1965) et (Schwefel & Schwefel, 1977) en Allemagne. Ces techniques ont été développées à l'origine pour résoudre les problèmes d'optimisation hydrodynamique ayant un degré élevé de complexité. Les ES évoluent non seulement les variables de décision du problème, mais aussi les paramètres des techniques (un tel mécanisme est appelé autoadaptation). Cette technique simule le processus évolutif à un niveau individuel, et, par conséquent, la recombinaison est possible bien que, il est normalement un opérateur secondaire (c.-à-d., moins important que la mutation). La proposition initiale d'ES ne reposait pas sur une population, mais uniquement sur l'utilisation d'une seule personne. Cependant, les ES à base de population ont été introduits par (Tripathy & Schwefel, 1982) quelques années plus tard. Les ES adoptent normalement l'un des deux schémas de sélection possibles :

i. Sélection plus ($\mu+\lambda$) : dans ce cas, la population suivante est générée par l'union des parents et des enfants.

ii. Sélection de virgules (μ, λ): dans ce cas, la population suivante n'est générée que par les enfants.

L'algorithme 2 montre un contour d'un ES simple à l'aide d'une population initiale.

```
Algorithme 2 : Stratégies évolutionnaire ES
Entrée : Un nombre μ d'individus
Sortie : Une population évoluée P(t)
1 Debut
2    t = 0;
3    INITIALISATION : P(t)∈ I^μ
4    Evaluation: Φ(P(t));
5    tant que (ι(P(t)) ≠ JUSTE) faire
6       Recombination: P'(t) = r(P(t))
7       Mutation: P''(t) = m(P'(t));
8       Evaluation: Φ(P''(t));
9       Selection:
                    { s(P(t)∪ P''(t))(μ+λ)-selection
        P(t+1) =    {                                          ;
                    { P''(t)  (μ,λ)-selection
10      t = t+1 ;
11    Fin
12 Fin
```

I.6.2 Programmation Génétique (GP)

Introduite par Koza au début des années 90 (Koza, 1992), la programmation génétique (genetic programming, ou GP) est un type d'algorithme évolutionnaire qui se caractérise par sa capacité à faire évoluer des expressions symboliques. Le terme "expressions symboliques" regroupe ici toute construction pouvant être exprimée comme une ou plusieurs séquences d'actions ou de choix, lesdites actions étant en nombre fini. Les algorithmes, les programmes informatiques, de même que la plupart des expressions mathématiques ou booléennes peuvent donc être représentés en programmation génétique, sous la forme d'individus. Ces individus, regroupés dans ce que l'on appelle une population, sont évolués itérativement, chaque itération étant, en lien avec la biologie, nommée génération. Une solution peut être encodée de différentes manières dans un individu. Dans la littérature, des listes (Linear GP), des graphes (Cartesian GP), voire du code machine ont été utilisés comme représentations afin de mieux s'adapter à différentes applications. Toutefois, la variante de GP la plus répandue est sans conteste la variante originellement proposée par Koza. Celle-ci se base sur un encodage sous forme d'arbres. Voir Algorithme 3.

```
Algorithme 3 : programmation génétique
Entrée : Un nombre μ d'individus
Sortie : Une population évoluée P(t)
1: Créer une population initiale (au hasard)
2: g ← 0
3: répéter
4:       Évaluer les individus de la population
5:       Sélectionner un certain nombre d'individus à conserver
6:       Créer une nouvelle population en utilisant les opérateurs de variation
7:       g ← g + 1
8: jusqu'à le Critère d'arrêt atteint
9: Retourner le meilleur individu
```

I.6.3 Algorithme Génétique (AG)

Les AG ont été appelés à l'origine des « plans de reproduction » génétiques et ont été développés au début des années 1960 par Holland (Sampson, 1976), dans le but de résoudre les problèmes d'apprentissage automatique. Ce type d'algorithme évolutionnaire se caractérise principalement par le codage des individus (traditionnellement à l'aide d'une chaîne binaire), et pour avoir un mécanisme de sélection probabiliste. Le croisement joue un rôle majeur dans les AG, mais un opérateur de mutation est également adopté pour maintenir de bonnes capacités exploratoires, voir Algorithme 4. Les AG travaillent au niveau génotypique et n'adoptent normalement pas un mécanisme d'autoadaptation en tant qu'ES, bien que certaines propositions à cet égard aient été étudiées dans la littérature spécialisée (J. D. (James D. Schaffer & Spears, 1989), (J. D. Schaffer & Morishima, 1987). En outre, il existe un autre opérateur appelé élitisme qui joue un rôle crucial dans les AG. Cet opérateur conserve le meilleur individu produit à chaque génération, et le transmet intact (c'est-à-dire sans être recombiné ou muté) à la génération suivante. Rudolph (Rudolph, 1994) a montré qu'une AG exige de l'élitisme pour converger vers l'optimum. Pour cette raison, l'élitisme est un mécanisme qui est devenu une norme dans les AE.

```
Algorithme 4 : Schéma général d'un algorithme génétique
Entrée : Un nombre μ d'individus
Sortie : Une population évoluée P(t)
1 début
2    t = 0 ;
3    INITIALISATION : P(t)∈ I^μ
4    tant que (ι(P(t)) ≠ JUSTE) faire
5        Evaluation: Φ(P(t));
6        Selection: P_p(t) = s(P(t));
7        Recombination: P_r(t) = r(P_p(t))
8        Mutation: P_m(t) = m(P_r(t));
9        P(t+1)) = P_m(t);
10       t = t+1;
11    Fin
12 Fin
```

Actuellement, il existe de nombreuses variantes de AG, avec différents codages de solutions, ainsi qu'une variété d'opérateurs de sélection, de croisement et de mutation (M. Kumar et al., 2020). Néanmoins, les caractéristiques d'un AG sont l'utilisation de la représentation binaire, la sélection proportionnelle de fonction objective, la mutation bit-flip (un opérateur qui est

appliqué avec une faible probabilité), et le croisement de un point (qui est appliqué avec une haute probabilité) et d'autres opérateurs récemment développés (Hassanat et al., 2019).

I.6.4 Évolution Différentielle (DE)

L'algorithme à évolution différentielle (DE : Differential Evolution) a été proposé par R. Storn et K. Price dans les années 1990 (Storn & Price, 1997) afin de résoudre le problème d'ajustement par polynômes de Tchebychev (Chebyshev polynomial fitting problem).

Déférentes variantes de DE ont été suggérées par Price (K. V. Price et al, 2005) et sont classiquement appelées $DE/x/y/z$, où DE désigne l'évolution différentielle, x fait référence au mode de sélection de l'individu de référence ou l'individu cible (rand ou best) pour la mutation, y est le nombre de différenciations utilisées pour la perturbation du vecteur cible x et z désigne le schéma de croisement, qui peut être binomial ou exponentiel. À chaque itération du processus d'évolution, chaque individu est d'abord muté, puis croisé avec son mutant. La phase de mutation implique que, pour chaque individu de la population Xi (target individual), un nouvel individu Vi (mutant individual) est généré, en ajoutant à ses composantes la différence pondérée d'autres individus pris aléatoirement dans la population. Le vecteur d'essai Ui (trial individual) est ensuite généré après croisement entre l'individu initial Xi et son mutant Vi. La phase de sélection intervient juste après, par compétition entre l'individu père Xi et son descendant Ui, le meilleur étant conservé pour la génération suivante. Ce processus est répété pour chaque individu de la population initiale, et mène donc à la création d'une nouvelle population de taille identique.

Algorithme 5 : L'évolution différentielle (DE)

Entrée : Un nombre µ d'individus

Sortie : Une population évoluée P(t)

1 début

2 t = 0 ;

3 INITIALISATION : générer la population initiale d'individus

4 faire

5 Pour chaque individu j dans la population Choisissez trois nombres X1, X2 et X3, c'est-à-dire $1 \leq X1, X2, X3 \leq N$ avec $X1 \neq X2 \neq X3 \neq j$

6 Générer un entier aléatoire $i_{rand} \in (1, N)$

7 Pour chaque paramètre i

8 $V_{i,} = x^{r1,} + F\ (x_{r2,} - x_{r3,})$

9 $U_{ijg} = \begin{cases} V_{ijg} \text{ if rand } () \leq CR \text{ or } j = j_{rand} \\ X_{ijg} \textit{ autrement} \end{cases}$

10 fin Pour

11 Remplacer $X_{i,}$ avec l'enfant U_{ijg} si U_{ijg} est mieux

12 fin pour

13 Jusqu'à ce que la condition d'arrêt soit atteinte

I.7 PROBLÈME D'OPTIMISATION MULTI-OBJECTIFS

Les problèmes à objectifs multiples se posent et surgissent de façon naturelle dans la plupart des disciplines. Leur résolution a été un défi pour les chercheurs depuis longtemps (Jain & Deb, 2014). Malgré la variété considérable des techniques développées en recherche opérationnelle et d'autres disciplines pour résoudre ces problèmes, la complexité de leurs solutions exige des approches alternatives.

Dans l'optimisation à objectif unique, il est possible de déterminer entre une paire de solutions donnée si l'une est meilleure que l'autre. En conséquence, nous obtenons généralement une seule solution optimale (l'optimum global). Cependant, dans l'optimisation multi-objective, il n'existe pas de méthode simple pour déterminer si une solution est meilleure qu'une autre. La méthode la plus couramment adoptée en optimisation multi-objectif pour comparer des solutions est appelée relation de dominance de Pareto (A. Zhou et al., 2011) qui, au lieu d'une seule solution optimale, conduit à un ensemble d'alternatives avec différents compromis entre les objectifs. Ces solutions sont appelées solutions optimales de Pareto ou solutions non dominées.

I.7.1 Formulation du problème d'optimisation multi-objectif :

Un problème d'optimisation multi-objectif (MOP) est défini comme (Jaimes, 2011) :

Minimiser $f_1(X), f_2(X),\ldots f_k(X)$ (I.10)

Sujet à

$$g_j(\boldsymbol{X}) \leq 0, \qquad j = 1,2,\ldots,m$$

Le vecteur $x \in \Re^n$ est formé de n variables de décision représentant les quantités pour lesquelles des valeurs doivent être choisies dans le problème d'optimisation. L'ensemble réalisable $X \subseteq \Re^n$ est implicitement déterminé par un ensemble de contraintes d'égalité et d'inégalité de la forme $gi\ (x) < 0\ et\ hi\ (x) = 0$, respectivement.

La fonction vectorielle $f: X \to \Re^k$ est composée de $k > 2$ fonctions d'objectifs scalaires $fi: Rn \to R\ (i = 1,\ldots,k)$. Dans l'optimisation multiobjective, les ensembles $\Re^n$ et $\Re^k$ sont appelés respectivement espace variable de décision et espace de fonction objectif. L'image de X sous la fonction f est un sous-ensemble de l'espace de fonction objectif désigné par $Z = f\ (X)$ et appelé l'ensemble réalisable dans l'espace de fonction objectif. La Figure I.8 présente un exemple avec 2 variables de décision et 2 fonctions objectives. Dans cette figure, nous pouvons apprécier la façon dont la fonction vectorielle f met en correspondance les solutions de l'ensemble faisable, X, avec l'ensemble faisable, Z, dans l'espace de fonction objectif. L'espace de fonction objectif attire beaucoup d'attention dans l'optimisation multi-objective puisque les performances de chaque solution sont évaluées dans cet espace.

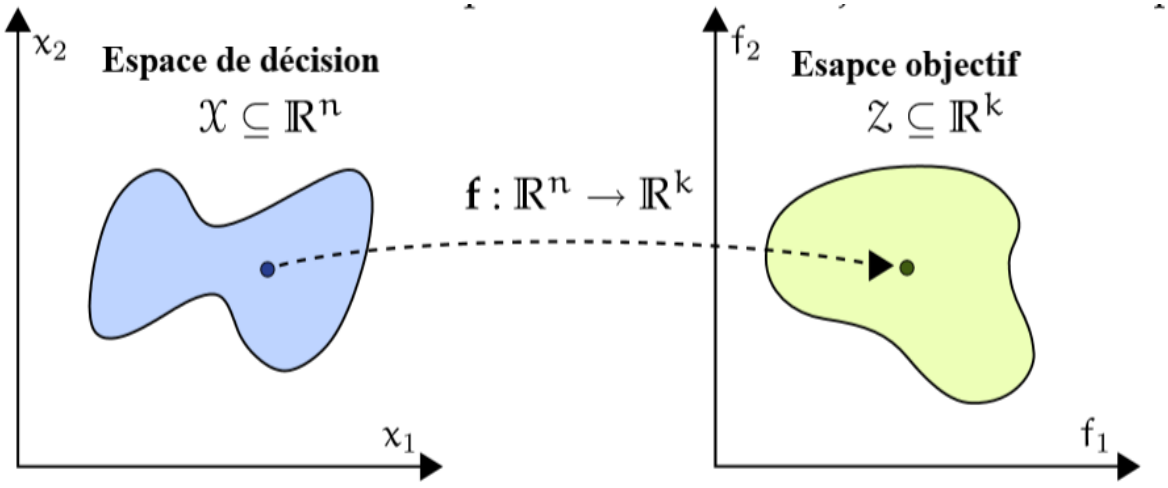

Figure II-8 Recherche des éspaces dans des problèmes d'optimisation multi-objectifs (Jaimes, 2011)

I.7.2 Notions d'optimalité dans l'optimisation multi-objective

Afin de définir plus précisément le problème d'optimisation multi-objectif, nous devons établir le sens de minimiser dans $\Re^k$. C'est-à-dire qu'il faut définir comment les vecteurs $f\ (x)\ \in\ \Re^k$ doivent être comparés pour différentes solutions$x\ \in\ Rn$. La relation « inférieure ou égale » ($\leq$) est utilisée dans l'optimisation à objectif unique pour comparer les valeurs d'objectifs scalaires. En utilisant cette relation, il peut y avoir de nombreuses solutions optimales différentes $x\ \in\ X$, mais une seule valeur optimale $fmin\ =\ min\{f\ (x)\ |\ x\ \in\ X\}$ puisque la relation 6 induit un ordre total dans R (c'est-à-dire que chaque paire de solutions est comparable afin que nous puissions trier les solutions de la meilleure à la pire). En revanche, dans les problèmes d'optimisation multi-objectifs, il n'y a pas d'ordre canonique sur $\Re^k$, et donc, nous avons besoin de définitions d'ordre plus faibles pour comparer les vecteurs dans $\Re^k$.

La notion d'optimalité dans le formalisme multi-objectif est basée sur ce principe de dominance proposée à l'origine par Edgeworth en 1881, et plus tard généralisée par l'économiste français Vilfredo Pareto en 1896. Pour cela on introduit les définitions suivantes (Emmerich & Deutz, 2018):

Relation de pareto dominance : On dit qu'un vecteur z^1 domine le vecteur z^2, noté $z^1\ \prec_{pareto}\ z^2$, si et seulement si:

$$\forall i\ \in\ \{1,\ldots,k\}:\ z_i^1\ \leq\ z_i^2\ and\ \exists i\ \in\ \{1,\ldots,k\}:\ z_i^1\ <\ z_i^2. \qquad \text{(I.11)}$$

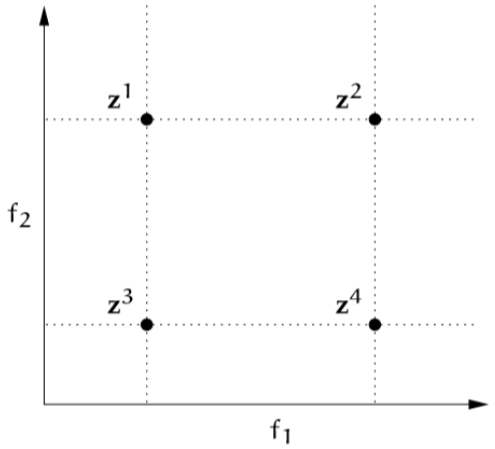

Figure II-9 illustration du concept de relation de dominance de Pareto (Jaimes, 2011)

La Figure I.9 illustre la relation de dominance de Pareto avec un exemple avec quatre vecteurs bidimensionnels. Le vecteur z^3 est strictement inférieur à z^2 dans les deux objectifs, donc $z^3\ \prec_{pareto}\ z^2$. Le vecteur z^3 domine également z^1 puisque par rapport à f_1 ces vecteurs sont égaux, mais dans f_2, z^3 est strictement inférieur à z^1. Puisque $\prec_{pareto}$ n'est pas un ordre total, certains éléments peuvent être incomparables comme c'est le cas avec z^1 et z^4, c'est-à-dire $z^1\ \nprec_{pareto}\ z^4$ et $z^4\ \nprec_{pareto}\ z^1$. Les autres comparaisons sont les suivantes : $z^3\ \prec_{pareto}\ z^2$, et $z^4\ \prec_{pareto}\ z^2$.

Ainsi, pour trouver les solutions optimales d'un MOP, nous devons trouver les solutions $x\ \in\ X$ dont les images, $z\ =\ f\ (x)$, ne sont dominées par aucun autre vecteur dans l'espace réalisable. Dans l'exemple de la Figure I.9, aucun vecteur ne domine z^3 et, par conséquent, nous disons que z^3 n'est pas dominé. Si pour deux solutions, z^1 et z^2, il arrive que $z^1\ \nprec_{pareto}\ z^2$ et $z^2\ \nprec_{pareto}\ z^1$, alors il est dit que ces solutions sont mutuellement non dominées.

Pareto Optimalité : Une solution $x^* \in X$ est dite Pareto optimale s'il n'existe pas une autre solution $x \in X$ telle que $f(x) \prec_{pareto} f(x^*)$. L'ensemble des solutions optimales de Pareto et son image dans l'espace objectif est défini comme suit (Van Veldhuizen, 1999).

Ensemble optimal de Pareto : L'ensemble optimal de Pareto, P_{opt}, est défini comme :

$$P_{\text{opt}} = \{x \in X \mid \nexists\, y \in X : f(y) \prec_{pareto} f(x)\} \qquad \text{(I.12)}$$

Front de Pareto : Pour un ensemble optimal de Pareto, P_{opt}, le front de Pareto, PF_{opt}, est défini comme: $PF_{opt} = \{f(x) = (f_1(x), \ldots, f_k(x)) \mid x \in P_{\text{opt}}\}$ (I.13)

La Figure I.10 illustre le concept d'ensemble optimal de Pareto (Jaimes, 2011) et son image dans l'espace objectif, le front de Pareto. Les points plus foncés désignent les vecteurs optimaux de Pareto. Dans l'espace variable, ces vecteurs sont appelés vecteurs de décision optimaux de Pareto, tandis que dans l'espace objectif, ils sont appelés vecteurs objectifs optimaux de Pareto. Comme nous pouvons le voir dans la Figure I.10, le front de Pareto n'est composé que de vecteurs non dominés. La résolution d'un problème d'optimisation multi-objectif conduit à l'obtention d'une multitude de solutions et seul un nombre réduit de ces solutions va être retenue. Ce choix délicat des solutions est basé sur la relation de dominance. Cette relation existe entre la solution considérée et les autres solutions.

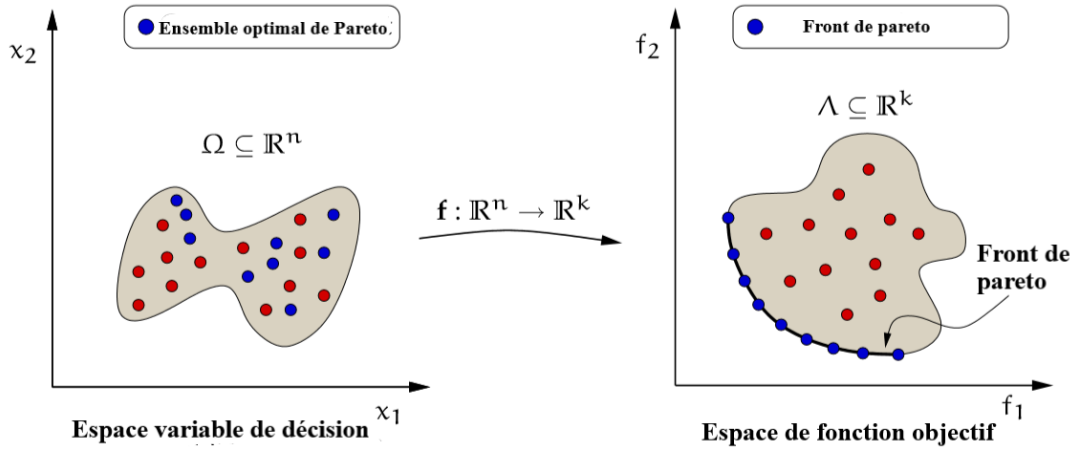

Figure II-10 illustration de l'ensemble optimal de Pareto et de son image, le front de Pareto. (Jaimes, 2011)

I.7.3 Structure générale d'un algorithme évolutionnaire multi-objective MOEA

Les AE mono-objectifs et les MOEA partagent une structure similaire. La principale différence est le mécanisme d'affectation de fitness car un MOEA traite des vecteurs de forme de dimension k ($k \geq 2$). Comme l'ont souligné différents auteurs (Carlos A. Coello Coello et al., 2007), trouver une approximation du front de Pareto est un problème bi-objectif dont les objectifs sont:

- Minimiser la distance des vecteurs générés au front optimal de Pareto, et
- Maximisez la diversité de l'approximation du front de Pareto obtenue.

Par conséquent, l'affectation de fitness doit tenir compte de ces deux objectifs.

L'algorithme 6 décrit la structure de base d'un algorithme évolutif multi-objectif.

```
Algorithme 6 : Pseudo code d'un algorithme évolutionnaire multi-objective
1: t ← 0
2: Générer une population initiale P(t)
3: tant que le critère d'arrêt n'est pas atteint faire
4:            Évaluer le vecteur objectif f pour chaque individu dans P(t)
5:            Attribuer une fitness pour chaque individu dans P(t)
6:            Sélectionner parmi P(t) un groupe de parents P'(t) préférant les
              meilleurs
7:            Recombiner les individus de P'(t) pour obtenir une population de
              descendants P''(t)
8:            Muter les individus en P''(t)
9:            Combinez P(t) et P''(t) et sélectionnez les meilleurs individus pour
              obtenir P(t + 1)
10:           t ← t + 1
```

Habituellement, la population initiale est générée de manière aléatoire. Cependant, si nous avons des connaissances sur les caractéristiques d'une bonne solution, il est sage d'utiliser ces informations pour créer la population initiale. L'affectation de fitness nécessite de classer les individus selon une relation de préférence, puis d'attribuer une valeur de forme scalaire à chaque individu en utilisant un tel classement. La sélection pour la reproduction (ligne 6) est effectuée comme dans le cas à objectif unique, par exemple, en utilisant la sélection de tournoi. En revanche, la sélection pour la survie (ligne 9), destinée à maintenir les meilleures solutions jusqu'à présent (l'élitisme), utilise une relation de préférence pour supprimer certaines solutions et maintenir la taille de la population constante. Pour assurer la diversité de l'ensemble d'approximation, le mécanisme de sélection est également basé sur un estimateur de densité de l'espace de fonction objectif.

I.7.4 Éléments clés d'un MOEA

Dans cette section, nous décrirons en détail les éléments les plus importants d'un MOEA qui doivent être pris en compte pour sa conception.

1- ***Affectation de fitness :*** une relation de préférence est utilisée pour induire un ordre partiel de la population dans l'espace fonctionnel objectif. Ensuite, un score scalaire (rang) est attribué à chaque solution en fonction de la façon dont la solution se compare par rapport aux autres solutions. Par exemple, les systèmes de classement par dominance comptent le nombre d'individus par lesquels un individu donné est dominé. Dans les schémas de comptage de dominance, la fitness d'un individu correspond au nombre d'individus qu'il domine. Comme nous l'avons remarqué dans une section précédente, la dominance de Pareto est la relation de préférence la plus couramment adoptée dans les MOEA.

2- ***Élitisme :*** L'élitisme est le mécanisme destiné à éviter la perte des meilleures solutions trouvées lors de la recherche en raison d'effets stochastiques. Ce concept joue un rôle majeur dans les MOEA modernes car, avec la mutation, il garantit la convergence globale. Dans l'optimisation multi-objective, la mise en œuvre de l'élitisme est plus complexe que dans l'optimisation à objectif unique. Puisque nous comptons avec des ressources de mémoire limitées, si plus de solutions non dominées apparaissent que celles qui peuvent être stockées, alors de bonnes solutions doivent être rejetées. Par conséquent, la stratégie élitiste adoptée détermine si le MOEA est globalement convergent ou non. Actuellement, nous pouvons principalement distinguer deux approches pour mettre en œuvre l'élitisme. L'une d'elles consiste à combiner les anciennes et les nouvelles populations, puis à utiliser une sélection déterministe pour conserver les meilleures solutions de la prochaine

génération (Deb, 2011). L'autre approche consiste à maintenir un ensemble externe d'individus appelé « archive » qui stocke les solutions non dominantes trouvées au cours du processus de recherche. La Figure I.11 illustre ces deux approches.

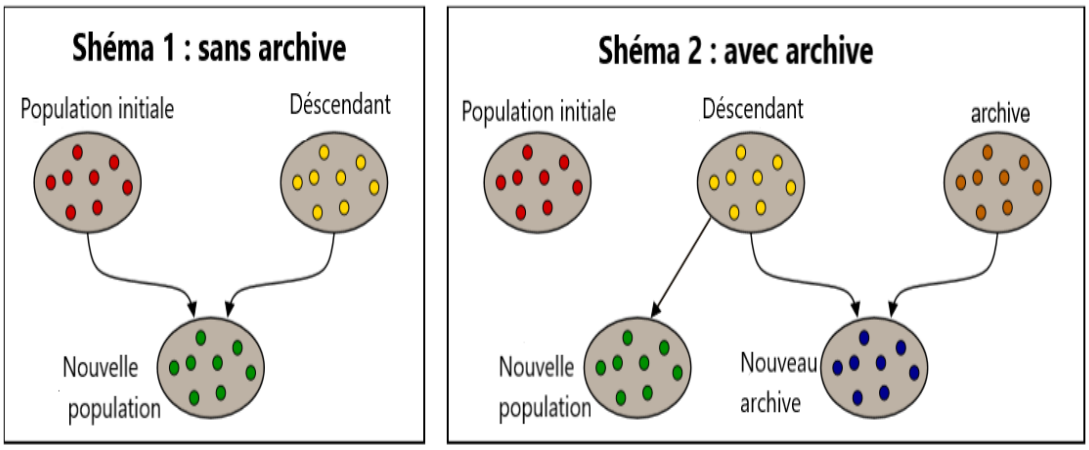

Figure II-11 Schémas pour mettre en œuvre l'élitisme (Jaimes, 2011).

3- ***Diversité :*** L'un des objectifs du MOEA est d'obtenir des solutions non dominantes bien réparties sur le front de Pareto. Dans ce qui suit, nous décrivons quelques techniques pour maintenir la diversité de la population :

- ***partage de fitness*** : L'objectif du partage de fitness (Goldberg, 1989) est de former et de maintenir des sous-populations (niches) réparties sur l'espace des fonctions objectives. L'idée est de considérer la fonction objective comme une ressource qui doit être partagée entre les individus d'une même niche. Ainsi, plus le nombre d'individus dans la niche est grand, plus la fonction objective assignée à chaque individu est petite.

$$f_{s_i} = \frac{f_i}{\sum_{j=1}^{N} \varphi(d_{ij})} \qquad \text{(I.14)}$$

Où f_i est la fitness de l'individu i, et d_{ij} est la fonction de partage, définie par:

$$\varphi(dij) = \begin{cases} 1 - (\frac{d_{ij}}{\sigma_{sh}}, d_{ij} < \sigma_{share} \\ 0 \qquad\qquad , otherwise \end{cases} \qquad \text{(I.15)}$$

Où σ_{share} est le rayon de niche et d_{ij} est la distance entre les individus i et j.

- ***Hyper-grilles :*** Une hyper-grille divise l'espace de fonction objectif dans les hyper-cubes à région scallée. Chaque solution non dominée occupe un hyper-cube comme le montre la Figure I.12. L'idée est seulement d'accepter des solutions non dominées appartenant à des hyper-cubes sous-peuplés. Bien que le nombre de divisions dans l'hyper-grille dans chaque dimension soit constant, la position et l'extension de la grille peuvent être adaptées au cours du processus de recherche.

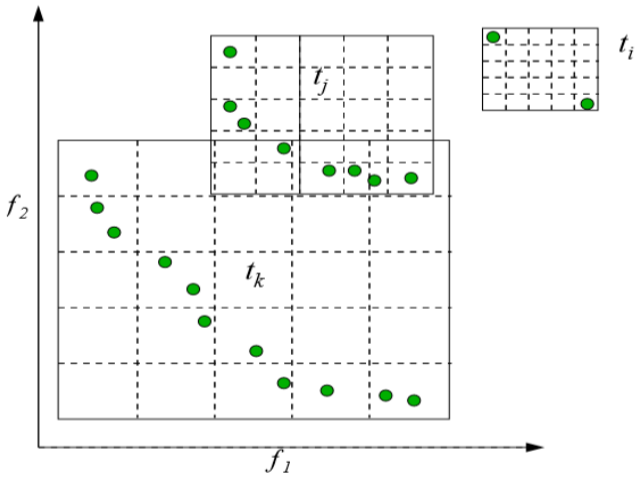

Figure II-12 Hypergrille pour maintenir la diversité dans l'archive (Jaimes, 2011)

- ***Regroupement*** : *(clustering)* L'objectif d'un algorithme de regroupement (E.-G. Talbi, 2009) est de partitionner un ensemble de points de telle manière que: (a) chaque groupe contient des points très similaires les uns aux autres; et (b) les points d'un groupe sont très différents de ceux d'autres groupes. Dans un MOEA, nous utilisons le regroupement pour préserver la diversité dans l'archive et réduire sa taille. Ce processus comprend trois étapes (Figure I.13) :

    1. Partitionnez l'archive à l'aide d'un algorithme de clustering.
    2. Sélectionnez un individu représentatif de chaque groupe, c'est-à-dire le centroïde.
    3. Supprimez toutes les autres individus du cluster.

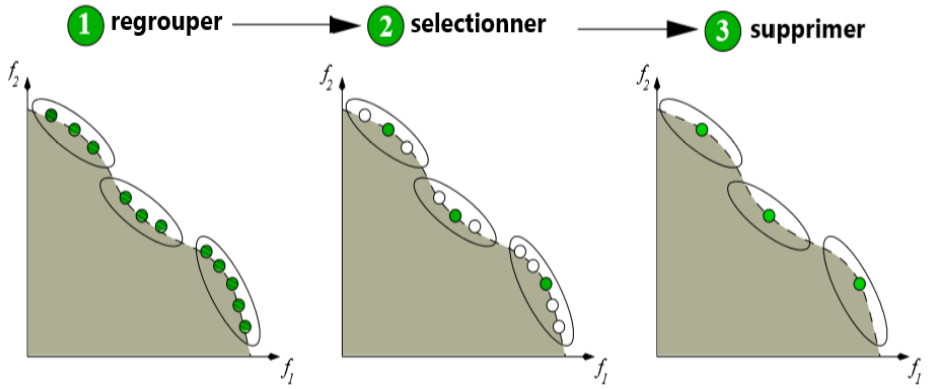

Figure II-13 Technique de clustering pour maintenir la diversité dans l'archive (Jaimes, 2011).

1.7.5 Chronologie et Approche des algorithmes évolutionnaires multi-objectives

Au cours des dernières décennies, les algorithmes évolutionnaires multi-objectifs se sont développés rapidement et sont divisés en quatre phases dans le tableau I.2. (W. Huang et al, 2019)

La première phase du tableau I.2 est antérieure aux années 1990. En 1967, Rosenberg a proposé de résoudre la MOP avec la recherche génétique, mais il ne l'a pas mise en œuvre. (James D. Schaffer & Spears, 1989) a proposé VEGA, qui a été le premier à utiliser l'AE pour traiter les MOP. En 1989, Goldberg a proposé de combiner la théorie de Pareto avec les AE.

La deuxième phase du tableau I.2 est au début des années 1990, le concept de dominance de Pareto a été incorporé dans ces algorithmes. (Fonseca & Fleming, 1993) ont proposé MOGA, (Srinivas & Deb, 1994) ont proposé NSGA, (Horn et al., 1994) ont proposé NPGA. Ce qui doit être amélioré pendant cette période est de savoir comment réduire la complexité de calcul et trouver une meilleure stratégie de préservation de la diversité.

La troisième phase du tableau I.2 est de 1999 à 2002, les algorithmes de cette période ont adopté le concept de préservation d'élite et de meilleures stratégies de préservation de la diversité. Certaines des études les plus importantes sont SPEA proposées par (Eckart Zitzler et al., 2001), PAES (Knowles & Corne, 1999), PESA (2000) et PESA-II (2001) proposées par (Corne et al., 2001), (Erickson et al., 2001) proposait NPGA, NPGA2, Micro-GA proposait par (Carlos A. Coello Coello & Pulido, 2001). SPEA2 proposait par (Eckart Zitzler et al., 2001) et NSGA-II proposait par (Deb et al., 2002) sont les algorithmes les plus représentatifs. De nombreux algorithmes au cours de cette période ont dominé le domaine du MOEA pendant de nombreuses années, mais ne sont toujours pas adaptés aux problèmes d'optimisation à plusieurs objectifs.

La quatrième phase du tableau I.2 a commencé en 2003, elle a présenté un développement plus diversifié : (1) le MOEA basé sur les indicateurs et la décomposition a été largement utilisé dans les problèmes d'optimisation à plusieurs objectifs. NSGA-III proposé par (Deb & Jain, 2014) a attiré une grande attention. (2) Introduire de nouvelles stratégies évolutives, telles que MOPSO (Sierra & Coello, 2005). (3) Recherche sur les opérateurs de sélection environnementale, tels que le principe de domination ε (Laumanns et al., 2005) et le principe de domination μ (Koduru et al., 2007). (4) De nombreux chercheurs ont successivement conçu des problèmes de référence plus efficaces, tels que ZDT, DTLZ proposés par (E. Zitzler et al., 2000). (Huband et al., 2006) proposait WFG. (5) Les recherches sur les MOEA combinées à la technologie d'apprentissage automatique sont également menées avec énergie. Par exemple, (Q. Zhang et al., 2008) a proposé un algorithme d'estimation multi-objectif basé sur un modèle de régularité (RMEDA), MOEAs basé sur l'apprentissage basé sur l'opposition (X. Ma et al., 2014).

Tableau II-2 les phases de développement des MOAE (W. Huang et al, 2019)

Phase	Algorithme
La première phase : (avant 1990)	VEGA, etc.
La deuxième phase : (1990 à 1998)	MOGA, NSGA, NPGA, etc.
La troisième phase : (1999 à 2002)	SPEA, PAES, PESA, PESA-II , NPGA, NPGA2, Micro-GA , SPEA2 , NSGA-II, etc.
La quatrième phase : (après 2002)	MOEA/D, NSGA-III , MOPSO, MODE, RMEDA, IBEA, SMS-EMOA, etc.

Récemment, des recherches ont été menées dans deux directions: (1) établir une terminaison plus sûre d'un MOEA en garantissant la convergence de solutions proches de leur optimalité Pareto (Abouhawwash et al., 2017). (2) MOEA interactif, qui intègre les procédures de prise de décision dans MOEA et produit directement les solutions optimales préférées.

Selon les caractéristiques des MOEA actuels, ils peuvent être grossièrement divisés en quatre catégories :

(1) MOEA basés sur la domination : Les individus reçoivent une fonction objective basée sur le principe de Pareto-dominance pour atteindre une bonne convergence, et un schéma explicite de préservation de la diversité est fourni pour maintenir la diversité des solutions.

NSGA-II (Jain & Deb, 2014) utilise un tri rapide non dominé pour classer les individus et réduit la complexité temporelle. L'opérateur de comparaison bondée sans spécifier de paramètre de partage remplace le mécanisme de partage de fitness. La stratégie de préservation d'élite de $(\mu + \lambda)$ (Andrew, 1998) est introduite, les parents et les bons gènes étant en compétition pour produire la prochaine génération. Mais la plupart des comparaisons entre les solutions dans le tri non dominé sont redondantes ou inutiles. Les chercheurs ont proposé une variété de méthodes plus rapides basées sur elle, comme le tri déductif (McClymont & Keedwell, 2012) et le tri efficace non dominé (Zhang et al., 2015), etc. Les informations sur la densité fonctionneront pour maintenir la diversité lorsque différents individus ont le même rang de non-domination. Les techniques d'estimation de la densité incluent la technologie de niche (Horn & Nafpliotis, 1994), la distance de surpopulation (Deb et al. ,2002), le plus proche k-voisin (Zitzler et al., 2001) et l'hyperbox (Knowles & Corne, 2001). SPEA2 (Zitzler et al., 2001) incorpore une valeur de densité d'un individu qui est l'estimation de la densité déterminée par la distance de l'individu à son ke plus proche voisin dans l'affectation de fitness. SPEA2 simplifie la méthode de mise à jour de la population externe basée sur un cluster dans SPEA. Bien que la complexité temporelle soit toujours $O(M^3)$, l'uniformité de distribution de la solution obtenue est meilleure.

Il y a encore quelques inconvénients dans les MOEA basés sur la domination. La vitesse de convergence lente vers PF, les mauvaises performances sur les MOP avec PF complexe et le véritable comportement de convergence de chacune des solutions non dominées de l'ensemble optimal de Pareto sont inconnus. Récemment, (Abouhawwash et al., 2017) ont développé une mesure de proximité KKT (KKTPM) pour estimer la proximité d'une solution de l'ensemble optimal de Pareto.

Les MOEA dominés par Pareto ne conviennent pas pour une optimisation à plusieurs objectifs. Parce qu'au fur et à mesure que le nombre d'objectifs augmente, la proportion de solutions non dominées augmente de façon exponentielle, réduisant ainsi la pression de sélection et ralentissant le processus évolutif, la diversité et la convergence sont toutes altérées. Ces dernières années, une variété de nouveaux principes de domination ont été proposés, l'objectif principal étant d'assouplir les critères de comparaison, afin que les solutions réalisables puissent être comparées entre elles, telles que: principe de domination ε (Laumanns et al., 2005), principe de domination μ (Koduru et al., 2007), etc. L'inconvénient est qu'ils dépendent trop des paramètres, et bien que la diversité et la convergence soient améliorées, les solutions finales non dominées ne sont pas garanties de couvrir uniformément l'ensemble du PF.

NSGA-III (Deb et al., 2014) est un algorithme efficace pour traiter les problèmes d'optimisation à plusieurs objectifs. Son cadre de base reste similaire à l'algorithme NSGA-II d'origine. Le plus grand changement consiste à utiliser des points de référence bien répartis pour maintenir la diversité. En plus de souligner la relation de dominance, le nombre d'individus associés à chaque point de référence est également souligné. Lorsque les points de référence sont uniformément répartis sur tout l'hyperplan, les solutions résultantes peuvent être uniformément réparties et proches du PF. Cependant, en raison de la faible pression de sélection de la domination de Pareto, sa convergence doit encore être améliorée.

(2) MOEA basés sur des indicateurs : les indicateurs de performance de la mesure de la qualité de la solution sont intégrés dans le MOEA comme critères de sélection de la sélection environnementale, guident la recherche et optimisent en permanence les attributs attendus de l'ensemble de la population.

Le premier MOEA basé sur des indicateurs bien connus est l'IBEA, proposé par Zitzler et Kunzl en 2004, qui définit la préférence du décideur comme un indicateur binaire pour comparer la qualité relative de deux ensembles de solutions approximatives. Cet indicateur est soumis à la règle de Pareto, de sorte que le calcul de la fonction objective peut être effectué en tant que stratégie d'affectation de fonction objective basée sur Pareto. Il fournit un cadre général pour les MOEA basés sur des indicateurs, permettant aux futures études d'aller dans cette direction.

SMS-EMOA (Beume et al., 2007) utilise l'indicateur HV (hyper-volume) qui mesure la quantité d'espace objectif dominée par la solution approximative définie comme critère de sélection, en écartant les solutions qui ont le plus petit HV sur le pire front, en évitant le coût de calcul élevé provoqué par les comparaisons basées sur la domination de Pareto. (Bader & Zitzler, 2011) ont proposé un algorithme évolutif basé sur HV rapide nommé HypE pour réduire les coûts de calcul excessifs. L'indicateur HV est très attractif dans les problèmes pratiques car il n'a pas besoin de connaître PF et contient des informations d'approximation, de diversité et de corrélation, et enfin évalue la qualité de l'ensemble de solutions approchées (Helbig & Engelbrecht, 2013). Bien sûr, il reste encore beaucoup de travaux de recherche à faire pour résoudre le coût de calcul du HV.

Certaines études ont montré que le remplacement d'autres indicateurs par des indicateurs HV a un coût de calcul moindre et des propriétés théoriques tout aussi bonnes. Par exemple, (Trautmann, Wagner, & Brockhoff, 2013) a proposé l'indicateur R2 pour résoudre les MOP. (Tian et al., 2018) a proposé un indicateur de distance de génération amélioré nommé IGD-NS pour distinguer les solutions qui ne contribuent pas à l'indicateur IGD.

L'algorithme basé sur des indicateurs présente de nombreux avantages, mais sa pression de sélection plus élevée que la domination de Pareto fait que les solutions préfèrent certaines régions spécifiques de PF, provoquant la disparition de certaines solutions optimales au cours du processus évolutif, puis les solutions finales peuvent ne pas être réparties uniformément le long du PF.

(3) MOEA basés sur la décomposition : Bien que la méthode de décomposition soit largement utilisée dans les méthodes de programmation mathématiques traditionnelles pour résoudre les MOP. La plupart des MOEA ne considèrent que la MOP dans son ensemble et s'appuient sur la domination pour mesurer la qualité de la solution, et ces solutions peuvent alors ne pas être uniformément réparties sur PF. L'évaluation de la fonction objective basée sur la fonction scalaire a une évolutivité par rapport au nombre d'objectifs et une forte capacité de recherche, sa complexité de calcul n'augmente pas de manière exponentielle avec l'augmentation des objectifs. Par conséquent, de nombreux MOEA basés sur la décomposition ont été proposés.

Le MOEA basé sur la décomposition le plus représentatif est le MOEA/D proposé par (Q. Zhang & Li, 2007). Diverses méthodes d'optimisation à objectif unique traditionnelles et méthodes de recherche locale peuvent être appliquées au cadre MOEA/D. L'idée de base est d'utiliser la stratégie de décomposition pour décomposer un MOP en un certain nombre de sous-problèmes d'optimisation scalaire. Les solutions de chaque sous-problème sont optimisées en effectuant des opérations évolutives parmi ses plusieurs sous-problèmes voisins, et ces sous-problèmes sont organisés de manière organique et résolus simultanément. Les structures

voisines entre les sous-problèmes sont définies en fonction de la distance entre leurs vecteurs de poids. La complexité de calcul de MOEA/D est $O(MNT)$, T est le nombre de vecteurs de poids, ce qui est inférieur à la complexité de calcul $O(MN^2)$ de NSGA-II. Les méthodes de décomposition couramment utilisées sont l'approche à somme pondérée (Miettinen, 2008), l'approche de Tchebycheff (Miettinen, 2008) et l'approche par intersection de limites basée sur les pénalités (I. Das & Dennis, 1998).

MOEA/D a encore de nombreuses limites. Par exemple, les vecteurs de poids sont générés en utilisant la méthode de conception de réseau simplex (I. Das & Dennis, 1998). Les inconvénients sont que la taille de la population augmente de façon non linéaire avec l'augmentation du nombre d'objectifs et ne peut pas être fixée arbitrairement, et la distribution des vecteurs de poids n'est pas très uniforme pour trois objectifs ou plus. De plus, lorsque le PF est inconnu ou très irrégulier, même un vecteur de poids uniformément réparti ne peut garantir que les solutions sont uniformément réparties sur le PF. MOEA/D est difficile de choisir une méthode de décomposition appropriée pour différents problèmes.

Ces dernières années, il existe de nombreuses recherches et travaux d'application dans le cadre de MOEA/D: comme de nouvelles méthodes de génération de vecteurs poids ont été proposées, le MOEA/D-AWA proposé par (Qi et al., 2014) utilise une stratégie d'ajustement adaptatif du vecteur de poids (AWA) pour les MOP avec des PF complexes. En développant une nouvelle stratégie de décomposition, (Lshibuchi et al., 2010) ont proposé une approche basée sur la combinaison de différentes fonctions de scalarisation dans le cadre MOEA/D; Une allocation de ressources de calcul plus raisonnable, (Q. Zhang et al., 2009) proposition de MOEA/D-DRA.

(4) MOEA hybrides : il est naturel de combiner les caractéristiques de différents algorithmes et d'utiliser leurs avantages pour faire face à des MOP complexes, proposant ainsi une série de MOEA hybrides. Les deux problèmes les plus importants des algorithmes hybrides sont: quelles techniques utiliser et comment les hybrider (A. Zhou et al., 2011).

Hybridation de différentes méthodes de recherche. Par exemple, combiner la recherche globale et les méthodes de recherche locales, comme l'introduction d'une méthode de recherche locale dans les EA pour améliorer localement les individus générés par les opérateurs EA communs, ce qui est bon pour obtenir de meilleures solutions non dominées.

Diviser le processus de recherche en différentes phases et utiliser différentes stratégies de recherche. Par exemple, (Yang et al., 2009) utilisent différentes stratégies de recherche en différentes phases pour mettre l'accent sur les solutions dominées, équilibrer les solutions dominées et non dominées, et se concentrer sur les solutions non dominées, respectivement. Combinant l'idée de conception des MOEA hybrides avec le nouveau modèle évolutif, (Xie et al., 2017) ont proposé un algorithme amélioré d'optimisation de feux d'artifice multi-objectifs.

La plupart des recherches actuelles se concentrent sur la façon de combiner différentes techniques pour générer une population de descendants.

I.8 AVANTAGES ET INCONVÉNIENTS DES AE

I.8.1 Avantages Des Algorithmes Évolutionnaires

- Un large domaine d'application les algorithmes évolutionnaires ont été utilisés avec succès dans une large gamme de problèmes. Ceci est principalement dû au fait de la simplicité et de l'intuitivité du processus de l'évolution naturelle (Janga Reddy & Nagesh Kumar, 2012).

- Adaptés aux espaces de recherche complexes (non linéaires, multimodaux, bruyant…) contrairement aux algorithmes évolutionnaires, la définition des autres méthodes heuristique à des problèmes complexes est une tâche très difficile.
- Faciles à paralléliser le concept de population et la faiblesse, ou voir la non existence, de dépendances entre les individus rend la parallélisation très facile et permet ainsi de réduire considérablement le temps d'exécution.
- Les AE offrent un cadre tel qu'il est relativement facile d'intégrer des connaissances spécifiques sur le domaine, et ces capacités d'adaptation peuvent être avantageuses lorsqu'il s'agit d'environnements dynamiques et distribués.
- Les AE peuvent être appliquées à pratiquement n'importe quel problème qui peut être formulé comme une tâche d'optimisation.
- L'espace de recherche peut être exploité en utilisant des méthodes mathématiques ou stochastiques (y compris des algorithmes mémétiques).

I.8.2 Inconvénients Des Algorithmes Évolutionnaires

- La simplicité de l'idée de base des algorithmes évolutionnaires sera payée par une grande consommation en termes de ressources.
- Difficulté d'ajustement de paramètres. Certains algorithmes sont conçus pour résoudre des problèmes spécifiques et leur adaptation sera très difficile et nécessitent un grand nombre de paramètre à ajuster.
- La nature heuristique des AE implique, qu'ils constituent des méthodes d'approximation qui ne donnent aucune garantie d'exactitude ou d'optimalité.
- Il peut être difficile d'affiner leurs paramètres pour bien fonctionner sur n'importe quel problème spécifique.
- Il n'existe aucune façon de prédire leurs temps d'exécution.
- Les algorithmes évolutionnaires sont très sensibles à la fonction d'évaluation et un seul changement dans le problème peut mener à un comportement tout à fait différent.
- Il reste inconnu comment faire les meilleurs choix de paramètres, ou comment construire la meilleure fonction objective pour un problème donné (Janga Reddy & Nagesh Kumar, 2012).

I.9 DOMAINES D'APPLICATION DES AE

Les applications des algorithmes évolutionnistes sont très larges (Figure I.14). À partir du moment où le problème en question peut être formulé sous forme d'un problème d'optimisation et que l'on dispose d'un moyen de représenter et évaluer les solutions, il est possible d'utiliser un AE. Outre les domaines traditionnels d'optimisation que l'on peut retrouver en mathématiques ou ingénierie traditionnellement :

- **Parcours de graphe** : S'inspirant de la sélection naturelle, les AE sont une approche fascinante pour résoudre les problèmes de recherche classique comme problème de voyageur de commerce TSP (Liao et al, 2012) , coloration de graphe GCP (Hindi et al, 2012), partitionnement des graph GPP (Datta et al, 2008)
- **Optimisation de process, d'emplois du temps (scheduling)** : Des problèmes de planification existent dans de nombreux systèmes de fabrication et de production, dans

le transport et la distribution de personnes et de biens, et dans d'autres types d'industries. Les trois éléments à tracer sont le temps, les tâches et les ressources (Xia et al, 2011)

- **Optimisation de formes d'objets :** dans les problèmes d'ingénierie du monde réel. Par exemple, l'industrie automobile, de l'industrie aéronautique et aérospatiale et de l'ingénierie civile, structurelle et mécanique, les AE apporter une opportunité et également un défi aux chercheurs pour améliorer et progresser dans la conception et l'optimisation des produits (Greiner et al, 2018)
- **Conception de circuits** : La conception des circuits est similaire aux instructions d'assemblage de bas niveau en ce qu'elle a été explorée avec succès à l'aide d'EA (Sekanina, 2010). Les règles de conception des circuits sont relativement simples et les EA peuvent donc explorer l'espace problématique relativement facilement.
- **Robot marchant** : Les robots qui apprennent à marcher ont toujours été un problème difficile. Les EA participent à l'apprentissage de la marche depuis des décennies. Les EA ont deux attributs qui les rendent particulièrement utiles pour gérer ces types de problèmes (Saputra, Takeda, & Kubota, 2015). Premièrement, les AE peuvent apprendre à s'améliorer à chaque cycle évolutif (améliorations incrémentielles) et deuxièmement, ils peuvent s'adapter aux changements de l'environnement. Par exemple, si un composant physique change ou fonctionne mal, les EA peuvent s'adapter à ce changement et continuer à marcher.
- **Optimisation des variables et des paramètres :** L'optimisation des paramètres et des variables est un processus permettant de trouver le meilleur ensemble de valeurs de paramètres pour un problème(S. Das, Mullick, & Suganthan, 2016). Les problèmes en eux-mêmes sont compliqués ou le nombre de paramètres est extrêmement important.

Figure II-14 Applications des AE

On observe de nouveaux cas d'utilisations pour les algorithmes évolutionnistes avec des applications notables sur des problèmes de données :

- **Optimisation multi-objectifs** : avec des applications dans le clustering et classification (Ramos & Vellasco, 2018)
- **Réseaux de neurones** : des approches comme NEAT (Neuro-Evolution of Augmenting Topologies) (Jha & Josheski, 2016) qui offrent des résultats compétitifs aux algorithmes par renforcement et optimisation des paramètres en problème d'apprentissage et génération des règles de façon évolutionnaire.
- **Les réseaux sans fil** : des AE ont été appliqués avec succès à de nombreux problèmes d'optimisation multi-sauts mobiles (Reina, et al., 2016) tels que la gestion de la topologie, les algorithmes de diffusion, les protocoles de routage et les modèles de mobilité, et d'autres problèmes connexes.
- **Configuration d'ordinateurs neuromorphiques** : Les EA sont utilisés pour concevoir des ANN simples pour configurer la plate-forme. Cette technique permet d'utiliser pleinement un SNN complexe pour créer de petits réseaux afin de résoudre des problèmes spécifiques. Les EA peuvent explorer tout l'espace des paramètres du matériel spécifique (Buckley, et al., 2018).
- **Optimisation et prévision des marchés financiers** : Les AG sont utilisés par les marchands institutionnels quantitatifs et d'autres domaines du monde financier (Mahfoud & Mani, 2017). Concevoir un portefeuille d'investissement afin de maximiser le rendement attendu dans des niveaux de risque acceptables.
- **Conception d'antenne** : Les antennes sont compliquées et sont principalement conçues à la main. Cela prend du temps et nécessite de nombreuses ressources. Les AE ont été « utilisés pour rechercher dans l'espace de conception et trouver automatiquement de nouvelles conceptions d'antennes ». Dans l'article (Basak & Lohn, 2013), un groupe de scientifiques de la NASA a réussi à concevoir une antenne en utilisant l'évolution numérique. L'antenne s'est avérée efficace pour une variété d'applications. Il était unique car le design final n'aurait pas été créé par un humain.

I.10 CONCLUSION

Nous avons vu dans ce premier chapitre les différents éléments constitutifs des algorithmes évolutionnistes ainsi que les architectures et structures classiques d'AE. Le cycle : sélection – évolution – évaluation de la population est l'idée centrale derrière les algorithmes évolutionnaires. Le plus souvent lors de la conception d'un algorithme évolutionniste la difficulté se trouve dans la conception de la représentation des individus et des méthodes d'évaluation qui sont directement spécifiques au problème. Un processus d'optimisation multi-objectif au sens de Pareto doit résoudre les deux tâches suivantes : guider le processus de recherche vers la frontière de Pareto, et de maintenir une diversité des solutions pour assurer une bonne répartition sur la frontière de Pareto. L'accomplissement de ces tâches est très délicat car les difficultés rencontrées dans un problème multi-objectif sont identiques à celles d'un problème simple objectif mais elles sont amplifiées par la présence d'objectifs dépendants les uns des autres.

Dans le chapitre suivant nous allons présenter un état de l'art sur les différentes techniques d'hybridation des algorithmes évolutionnaires avec les différentes méthodes d'optimisation qui existent dans la littérature et leurs impacts sur le perfectionnement des AE dans l'objectif et de proposer des approches efficaces afin de réduire la complexité des AE.

CHAPITRE II :

HYBRIDATION DES ALGORITHMES ÉVOLUTIONNAIRES

II.1 INTRODUCTION

Le calcule évolutionnaire est devenu une méthodologie de résolution de problèmes importante pour de nombreux chercheurs. Le processus d'apprentissage collectif basé sur la population, l'adaptation et la robustesse sont quelques-unes des caractéristiques clés des algorithmes évolutionnaires par rapport aux autres techniques d'optimisation globales (Blum et al., 2011). Même si le calcul évolutionnaire a été largement accepté pour résoudre plusieurs applications pratiques importantes dans les domaines de l'ingénierie, science, finance, etc., leurs performances ne sont parfois que marginales. La sélection inappropriée de divers paramètres, représentations, etc. est souvent mise en cause. Il y a peu de raisons de penser que l'on peut trouver un algorithme uniformément optimal pour résoudre tous les problèmes d'optimisation. Ceci est en accord avec le principe No Free Lunch, qui précise que, quel que soit l'algorithme, une performance élevée sur une classe de problèmes est payée exactement pour une performance sur une autre classe.

Le comportement de l'AE est déterminé par la relation d'exploitation et d'exploration conservée tout au long de l'analyse. Tous ces éléments illustrent clairement la nécessité d'approches évolutionnaires hybrides dont la tâche principale est d'optimiser les performances de l'approche évolutionnaires directe. Récemment, l'hybridation d'algorithmes évolutionnaires est de plus en plus populaire en raison de leurs capacités à traiter plusieurs problèmes du monde réel impliquant complexité, imprécision, incertitude et multi-objective.

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. Les méthodes hybrides ont rapidement gagné du terrain en réussissant à produire les meilleurs résultats pour de nombreux problèmes.

Dans ce chapitre, nous soulignons d'abord les motivations et la nécessité des algorithmes évolutionnaires hybrides, puis nous illustrons les diverses possibilités d'hybridation d'un algorithme évolutionnaires et nous présentons également certaines des architectures évolutionnaires génériques hybrides qui ont évolué au cours des deux dernières décennies. Nous passons en revue les différents algorithmes évolutionnaires hybrides pour l'optimisation continu et multi-objectif et nous illustrons également un état de l'art sur les approches hybride efficace combinant les AE avec des méthodes de recherche globale et locale.

II.2 LES MÉTHODES HYBRIDES

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.

Les origines des algorithmes hybrides remontent probablement aux travaux décrits dans (Mühlenbein et al., 1988) (Grefenstette, 1987). Les auteurs ont clairement démontré l'effet bénéfique de l'intégration d'une méthode de descente à l'intérieur d'une méthode évolutive. D'autres travaux ont montré que lors de la résolution du problème de coloration des graphes, les meilleurs résultats sont obtenus en hybridant une méthode évolutionnaire avec une méthode de recherche locale (Hao et al., 1999). Plusieurs publications ont été réalisées dans ce domaine, qui devient de plus en plus intéressant (Tvrdík & Křivý, 2015) (Vakil-Baghmisheh & Ahandani, 2014) (Gonzalez et al., 2017) (Chen et al., 2018).

Actuellement, les approches hybrides gagnent en popularité car ce type d'algorithme produit généralement les meilleurs résultats pour plusieurs problèmes d'optimisation combinatoire (E. G. Talbi et al., 2007). En effet, selon (E.-G. Talbi, 2009), les approches hybrides ont permis d'obtenir de bons résultats dans une grande variété de problèmes théoriques d'optimisation combinatoire tels le problème du voyageur de commerce, le problème de coloration de graphe, le problème d'affectation quadratique, le problème de tournée de véhicules, le séquençage d'ADN ou encore le calcul des trajectoires des satellites. Les premières approches hybrides proposées ont combiné des métaheuristiques à base de populations (algorithmes génétiques) avec des métaheuristiques à solution unique (recuit simulé ou recherche tabou).

II.3 MOTIVATION DE L'HYBRIDATION

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 permet 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.

Les méthodes de recherche modernes pour l'optimisation considèrent les algorithmes évolutionnaires hybrides (AEH) ceux qui utilisent l'algorithme évolutionnaires (AE) et les optimiseurs locaux travaillant ensemble. L'hybridation vient de l'équilibre des procédures de recherche globales et locales. L'inspiration dans la nature a été poursuivie pour concevoir des modèles de calcul flexibles, cohérents et efficaces. Les principaux axes de ces modèles sont les problèmes du monde réel, compte tenu de la faible efficacité connue des algorithmes évolutionnaires dans leur traitement.

Pour plusieurs problèmes, un algorithme évolutionnaire simple peut suffire à trouver la solution désirée. Comme rapporte la littérature, il existe plusieurs types de problèmes pour lesquels un algorithme évolutionnaire direct pourrait échouer à obtenir une solution commode ou optimale (Tseng & Liang, 2006). Cela ouvre clairement la voie à la nécessité d'hybridation d'AE avec d'autres algorithmes d'optimisation, de techniques d'apprentissage automatique, d'heuristiques, etc. Certaines des raisons possibles de l'hybridation sont les suivantes (Sinha et al., 2006):

1- Améliorer les performances de l'AE (exemple: vitesse de convergence)

2- Améliorer la qualité des solutions obtenues par l'AE

3- Incorporer l'algorithme d'évolution dans le cadre d'un système plus vaste

(Wolpert & Macready, 1997) ont montré que tous les algorithmes qui recherchent un extremum d'une fonction de coût fonctionnent exactement de la même façon, lorsqu'ils sont en

moyenne sur toutes les fonctions de coûts possibles. Selon les auteurs, si l'algorithme A surpasse l'algorithme B sur certaines fonctions de coûts, alors vaguement parlant il doit exister exactement autant d'autres fonctions où B surpasse A. Par conséquent, d'un point de vue de résolution de problèmes, il est difficile de formuler un universel algorithme d'optimisation qui pourrait résoudre tous les problèmes. L'hybridation peut être la clé pour résoudre des problèmes pratiques. Pour illustrer la popularité des approches hybrides, nous avons cherché le nombre de publications apparaissant dans certaines des bases de données scientifiques populaires à savoir ScienceDirect (“sciencedirect”), IEEE-Xplore (“IEEE-Xplore”), et SpringerLink (“SpringerLink”) en utilisant les mots clés "hybrid evolutionary" et " génétique hybride » et les résultats de la requête sont compilés ci-dessous. Étant donné qu'un filtrage a été utilisé dans la requête, ou la période explorer est les dix dernières années entre 2010 et 2018 comme illustre la Figure II-1 et la Figure II-2

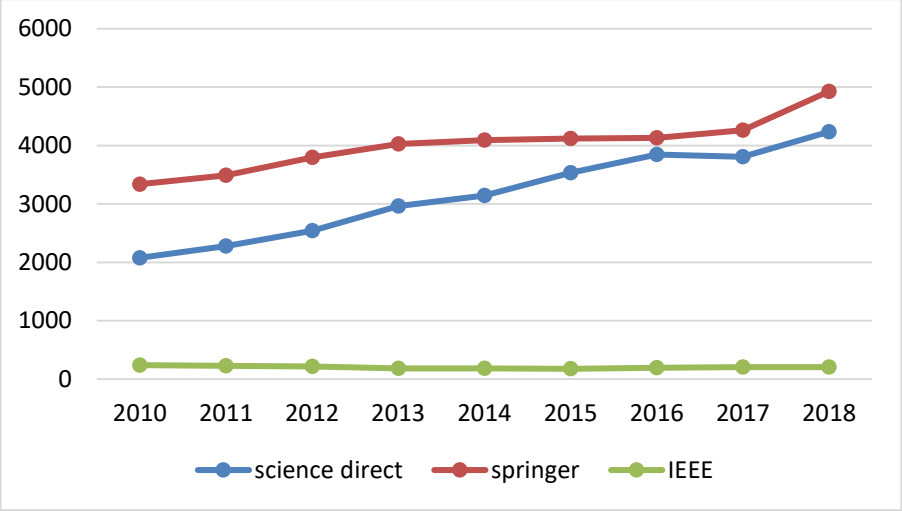

Figure III-1 L’évolution du terme "hybrid evolutionary"

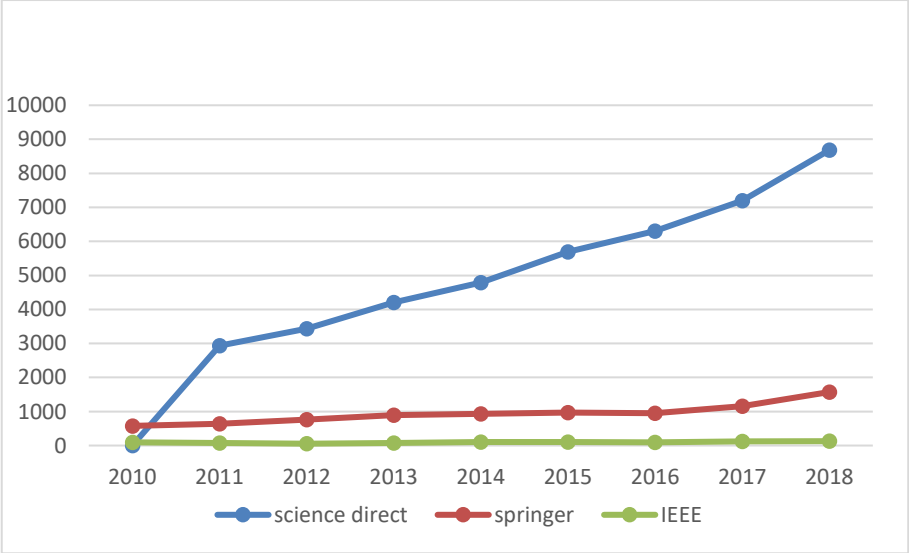

Figure III-2 L’évolution du terme "hybrid multi-objective"

La Figure II.3 illustre certaines des architectures génériques (Grosan & Abraham, 2007) pour les différents types d'hybridation. Un problème, un problème d'approximation de fonctions optimisées et un paradigme intelligent se rapportent à toute technique d'intelligence computationnelle, recherche locale, algorithmes d'optimisation, etc.

La Figure II.3 A, B représente une architecture concurrente dans laquelle tous les composants sont nécessaires au bon fonctionnement du modèle. Comme illustré à la Figure II.4 A, l'algorithme évolutionnaire agit comme un préprocesseur et le paradigme intelligent est utilisé pour ajuster les solutions formulées par l'AE. Sur la Figure II.3 B, le paradigme intelligent agit en tant que préprocesseur et l'algorithme évolutionnaire est utilisé pour ajuster les solutions formulées par le paradigme intelligent. La Figure II.3 C représente un système hybride transformationnel dans lequel l'AE est utilisé pour ajuster les performances du paradigme intelligent, tandis que le paradigme intelligent est utilisé pour optimiser les performances de l'AE. Les informations requises sont échangées entre les deux techniques au cours du processus de recherche (résolution de problème). Dans un modèle coopératif, le paradigme intelligent n'est utilisé que pour l'initialisation ou pour la détermination de certains paramètres de l'algorithme évolutionnaire. Comme illustré à la Figure II.3 D, par la suite, les paramètres intelligents ne sont pas nécessaires au bon fonctionnement du système. En outre, il existe plusieurs façons d'hybrider deux techniques ou plus.

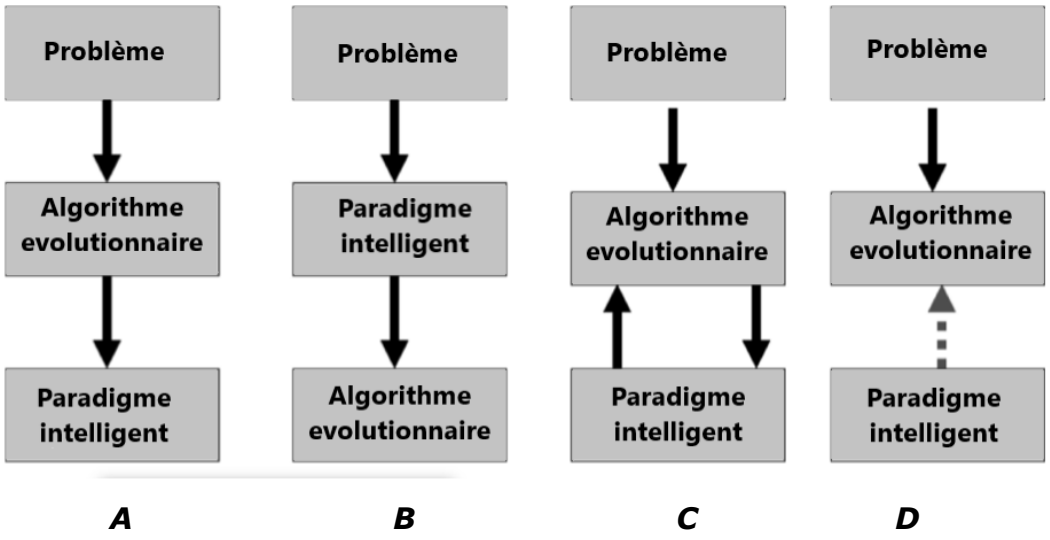

Figure III-3 Architectures génériques d'AEs hybrides (Grosan & Abraham, 2007)

II.4 CLASSIFICATION DES STRATÉGIES 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-dessus :

II.4.1 Classification 1

Dans (Duvidier, 2000), l'auteur propose une classification des techniques d'hybridation en s'appuyant sur les travaux du groupe PERFORM (Parallel Genetic algorithms for optimization) du laboratoire LIFL de Lille. Les approches hybrides sont classifiées en trois catégories, selon leurs architectures :

- Hybridation séquentielle
- Hybridation parallèle synchrone

- Hybridation parallèle asynchrone

Hybridation séquentielle *:* c'est le type d'hybridation le plus populaire. Elle consiste à appliquer plusieurs méthodes de telle manière 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 II.4).

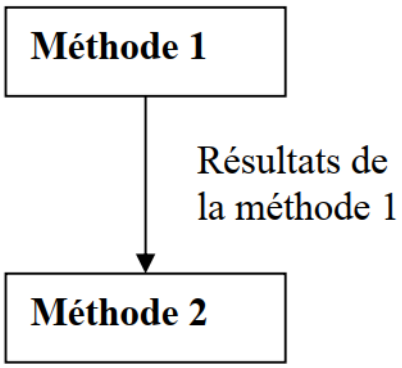

Figure III-4 Hybridation séquentielle (Duvidier, 2000)

Hybridation parallèle synchrone *:* cette hybridation est réalisée par incorporation d'une méthode de recherche particulière dans un opérateur. Elle est plus complexe à mettre en œuvre que la précédente. L'objectif est de combiner une recherche locale avec une recherche globale dans le but d'améliorer la convergence. Par exemple une recherche tabou est utilisée pour générer des solutions initiales pour un algorithme de strategie évolutionnaire (Figure II.5).

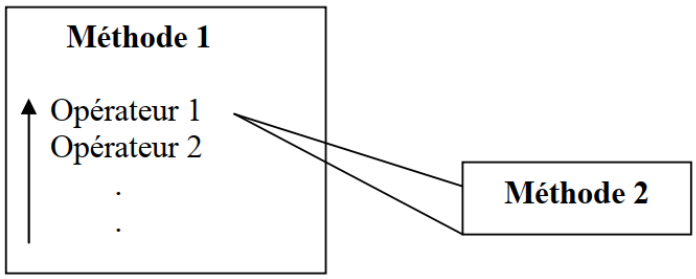

Figure III-5 Hybridation parallèle synchrone (Duvidier, 2000)

Hybridation parallèle asynchrone (coopérative) *:* 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 (Figure II.6).

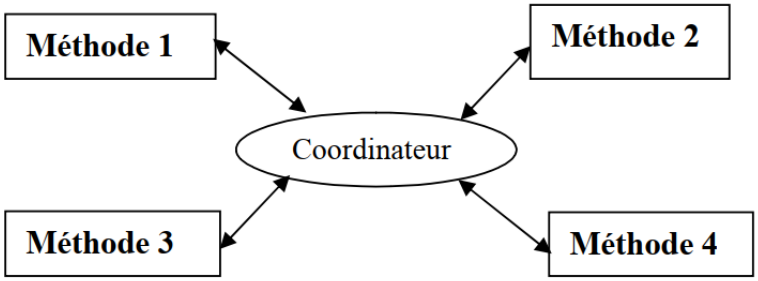

Figure III-6 Hybridation parallèle asynchrone (coopérative) (Duvidier, 2000)

II.4.2 Classification 2

Une taxonomie a été proposée par (E.-G. Talbi, 2009), permettant de classifier les différentes hybridations et de les comparer de façon qualitative. Cette taxonomie comporte deux aspects principaux. Une classification hiérarchique permettant d'identifier la structure générale de l'hybridation et une classification générale spécifiant les détails des algorithmes impliqués dans l'hybridation.

La classification hiérarchique : se subdivise en deux classes : une hybridation de bas niveau et une autre de haut niveau. On parle d'une hybridation de bas niveau lorsqu'une fonction d'une métaheuristique est remplacée par une autre métaheuristique. Cependant, 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 séquentielle, 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 :

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 (Figure II.7).
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 l'évolution différentielle (DE) comme solution initiale d'une recherche tabou (Figure II.8).
4. ***Hybridation de haut niveau co-évolutionnaire :*** implique un ensemble d'algorithme évolutionnaires complètes qui travaillent en parallèle et coopèrent pour trouver la solution optimale du problème. Par exemple coopérer entre une recherche locale et une recherche taboue, de sorte que, à intervalles réguliers, les deux méthodes échangent des informations (Figure II.9).

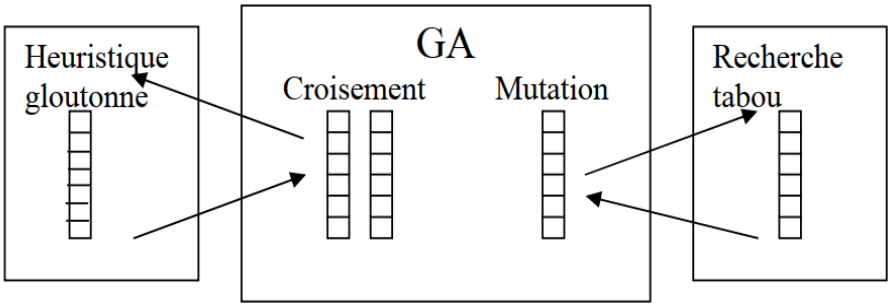

Figure III-7 Exemple d'hybridation de bas niveau co-évolutionnaire(E.-G. Talbi, 2009)

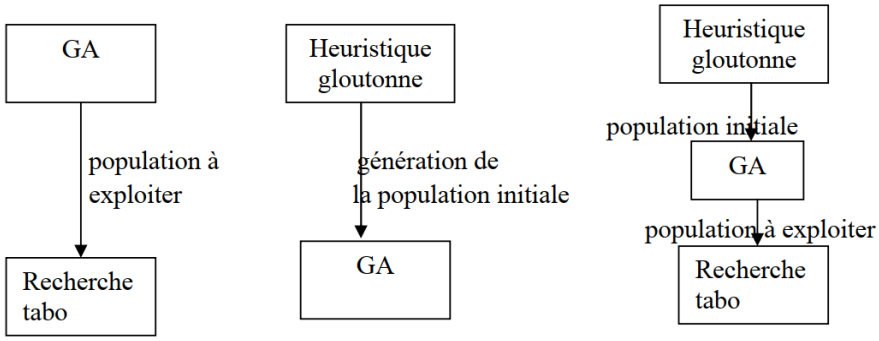

Figure III-8 Exemple d'hybridation de haut niveau à relais(E.-G. Talbi, 2009)

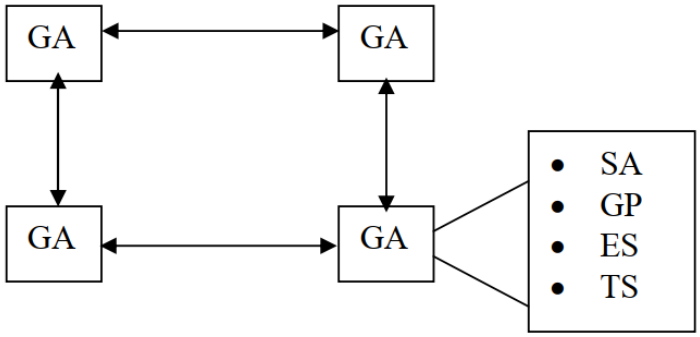

Figure III-9 Exemple d'hybridation de haut niveau co-évolutionnaire(E.-G. Talbi, 2009)

La 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 (voir Figure II.10)

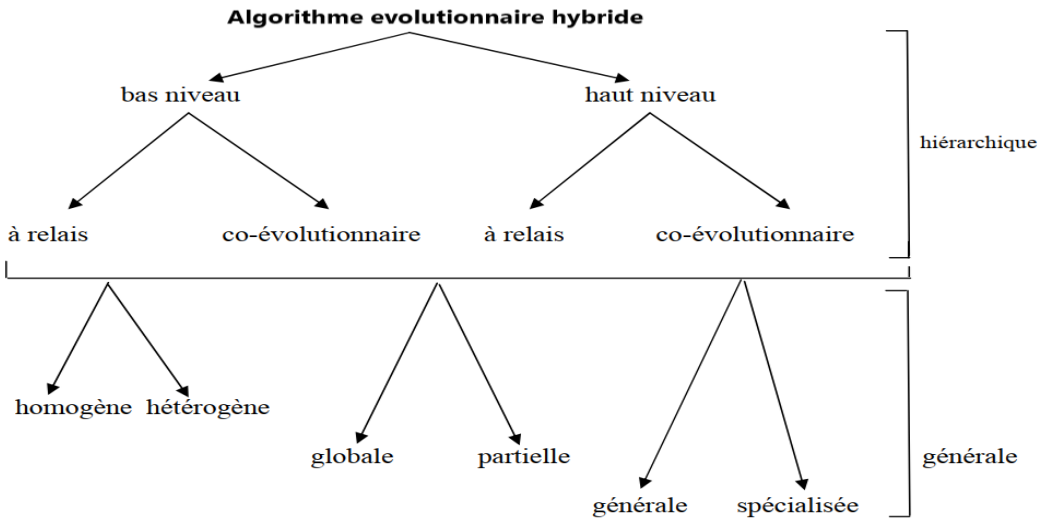

Figure III-10 Classification des AEs hybrides(E.-G. Talbi, 2009)

II.4.3 Classification 3

Une autre classification axée sur l'hybridation de méthodes exactes et approximatives a été proposée dans (Raidl, 2005). Les méthodes hybrides sont divisées en deux catégories : les hybridations collaboratives et intégratives. Les algorithmes qui échangent des informations de façon séquentielle, parallèle ou entrelacée font partie de la catégorie des hybridations collaboratives. Les algorithmes à hybridation intégrative font en sorte qu'une technique est une composante incorporée à une autre technique.

L'hybridation collaborative séquentielle consiste à exécuter de façon que la méthode exacte soit un prétraitement de l'AE, ou vice-versa. Dans (Applegate et al., 1998) les auteurs proposent une méthode collaborative séquentielle pour résoudre le problème de voyageur de commerce. Une recherche locale est exécutée pour créer un ensemble de solutions qui seront utilisées à leur tour par un algorithme exact. (Hachimi et al., 2013) appliquent la solution de la relaxation d'un programme linéaire à une fonction d'arrondissement aléatoire pour créer les solutions de départ d'un AG, afin de résoudre le problème d'assignation généralisé. Si une solution est infaisable, une fonction de réparation aléatoire est utilisée pour la corriger. Quant à l'hybridation collaborative parallèle, elle a lieu quand la communication entre les AEs et les méthodes exactes s'effectue parallèlement. Par exemple, utiliser la méthode Branch and Bound pour accélérer la recherche et l'algorithme génétique pour de réduire l'espace de recherche. Les combinaisons d'une méthode exacte et d'un AE de façon intégrative se font de manière qu'un des deux algorithmes soit une composante intégrée à l'autre algorithme. On peut donc intégrer une méthode exacte à un algorithme évolutionnaire, et inversement. Par exemple, utiliser Branch and Bound pour avoir le meilleur croisement possible entre individus dans un algorithme évolutionnaire (Stanojević et al., 2015). D'autre part, on peut utiliser une métaheuristique pour calculer les bornes des solutions sortantes pour les approches Branch and Bound (Festa, 2014).

II.5 L'HYBRIDATION EN OPTIMISATION CONTINUE

Lorsqu'un algorithme hybride est conçu, deux des caractéristiques les plus pertinentes à prendre en compte sont : 1) le coût de la recherche locale ; (2) le paysage de recherche sous-jacent. Afin

de trouver des solveurs mémétiques efficaces, en optimisation continue, ces caractéristiques doivent être abordées différemment par rapport au cas discret.

Dans le domaine continu (R. G. G. R. K. Kumar, 2013), la situation est un peu plus simple, dans la mesure où il existe un paysage sous-jacent naturel en R^n, à savoir celui induit par des mesures de distance telles que la distance euclidienne. En d'autres termes, en optimisation continue, l'ensemble des points pouvant être atteints par l'application d'opérateurs unaires à un point de départ peut être représenté par des sphères fermées de rayon. Au contraire, l'ensemble des points accessibles par les opérateurs de recombinaison peut être visualisé au moyen d'un hyper-cubes dans l'espace de décision. Les images intuitives des optima locaux et des bassins d'attraction s'intègrent naturellement ici et permettent au concepteur d'exercer un certain contrôle sur la dynamique de recherche en ajustant avec soin les propriétés d'intensification / diversification des opérateurs utilisés.

Ces deux problèmes mentionnés ci-dessus ont été traités dans la littérature sur les algorithmes mémétique pour une optimisation continue de différentes manières. En commençant par le premier (le coût de la recherche locale), il souligne la nécessité de choisir avec soin quand et comment la recherche locale est appliquée (évidemment, il s'agit d'un problème général qui s'applique également aux problèmes combinatoires, mais qui revêt une importance capitale pour les problèmes continus). Cette prise de décision est très difficile en général (Krasnogor & Smith, 2008), mais certaines stratégies ont été avancées dans des travaux antérieurs. Une solution assez simple consiste à recourir au lamarckianisme partiel (Houck et al., 1997) en appliquant de manière aléatoire une recherche locale avec une probabilité $Pls < 1$. Évidemment, la fréquence d'application n'est pas le seul paramètre pouvant être ajusté pour ajuster le coût de calcul de la recherche locale : La recherche locale (c.-à-d. pendant combien de temps une tentative d'amélioration locale est-elle tentée sur une solution particulière) est un autre paramètre à modifier. Cet ajustement peut être effectué à l'aveuglette (c'est-à-dire en préfixant une valeur constante ou une planification de variation sur l'ensemble de l'exécution) ou de manière adaptative.

Par exemple, (R. G. G. R. K. Kumar, 2013) définissent trois classes de solutions différentes (sur la base de l'aptitude) et associent un ensemble différent de paramètres de recherche locale pour chacune d'entre elles. À ce sujet, (Nguyen et al., 2007) considèrent une approche stratifiée, dans laquelle la population est triée et divisée en n niveaux (n étant le nombre d'applications de recherche locale), et un individu par niveau étant sélectionné de manière aléatoire. Ceci est montré pour fournir de meilleurs résultats que la sélection aléatoire. Nous nous référons à (Bambha et al., 2004) pour une analyse empirique approfondie des compromis temps / qualité lors de l'application de la recherche locale paramétrée au sein d'algorithmes mimétique. Ce paramétrage adaptatif a également été exploité dans des chaînes dites de recherche locale (Molina et al., 2010), en sauvegardant l'état de la recherche locale à la fin sur une certaine solution pour une utilisation ultérieure si la même solution est sélectionnée à nouveau pour une amélioration locale. Notons enfin, en ce qui concerne cette question de paramétrage, que les stratégies d'adaptation peuvent être poussées plus loin et entrer dans le domaine de l'autoadaptation (Boukhari et al., 2018).

En ce qui concerne le compromis exploitation / exploration, la composante population est généralement utilisée pour naviguer dans l'espace de recherche, offrant ainsi des points de départ intéressants pour intensifier la recherche via l'opérateur d'amélioration locale. L'aspect diversification de la recherche basée sur la population peut être renforcé de plusieurs manières, par exemple en utilisant plusieurs sous-populations, ou des stratégies de remplacement orientées vers la diversité. Ces derniers sont courants dans la recherche scatter (Glover et al.,

2000) (SS), un paradigme d'optimisation étroitement lié aux algorithmes mémétiques dans lequel la population (ou la référence définie dans le jargon SS) est divisée en niveaux : leur entrée est gagnée par la solution de fitness dans un cas ou de diversité dans l'autre cas. De plus, les SS incorporaient souvent des mécanismes de reprise pour introduire de nouvelles informations dans la population lors de la convergence de celle-ci. La diversification peut également être introduite via un couplage sélectif.

Une stratégie connexe a été proposée par (Manuel et al., 2004) par le biais de l'utilisation d'un appariement assortatif négatif: après avoir choisi une solution pour la recombinaison, un ensemble de partenaires potentiels est sélectionné et la plus diversifiée est utilisée. D'autres stratégies ont adopté l'utilisation du clustering (Sindhya et al., 2013) (pour détecter des solutions susceptibles de se trouver dans le même bassin d'attraction sur lequel il peut ne pas être fructueux d'appliquer la recherche locale), à l'utilisation de techniques standard de préservation de la diversité dans des contextes multimodaux tels que le partage ou encombrement. Il convient également de mentionner que la composante d'intensification de l'algorithme mémétique est parfois fortement imbriquée dans le moteur basé sur la population, sans recourir à une composante de recherche locale distincte.(Floreano & Mattiussi, 2008) ont utilisé une stratégie d'intensification différente en considérant une procédure exacte pour trouver la meilleure combinaison de valeurs variables à partir des parents. Cela nécessite évidemment que la fonction objective soit susceptible de donner lieu à l'application d'une procédure efficace d'exploration du potentiel souches (ensemble d'enfants possibles) des solutions en cours de recombinaison. Nous renvoyons à (M. Lozano & García-Martínez, 2010) pour une analyse détaillée des stratégies de diversification / intensification dans les méta-heuristiques hybrides (en particulier dans les algorithmes mémétiques).

Dans certains cas, il peut être nécessaire de détecter plusieurs optima locaux plutôt que seulement l'optimum global. Ce problème est généralement indiqué en tant que problème d'optimisation multimodale. De toute évidence, cette situation ne se produit que lorsqu'il existe un paysage continu, car dans l'optimisation discrète, il n'existe pas de concept absolu d'optimum local. Les approches MC ont été utilisées dans divers contextes pour résoudre ce problème. Bien que ce ne soit pas le sujet de cette enquête, il convient de mentionner quelques approches mémétiques proposées dans la littérature. Par exemple, dans (Delvecchio, et al., 2006) une approche hybride composée d'opérations séquentielles à seuil, la recherche globale et locale permet la détection de plusieurs optima sous des contraintes de forme. Dans (Qu et al., 2012), une cartographie heuristique est proposée afin de promouvoir la convergence multiple dans un cycle évolutionnaire unique.

II.6 L'HYBRIDATION EN OPTIMISATION MULTI-OBJECTIVE

Les algorithmes d'optimisation multi-objectifs évolutionnaires (MOEA), bien qu'ils soient largement utilisés, sont souvent critiqués pour leur lente convergence vers le front optimal de Pareto (Goel & Deb, 2002). Par conséquent, il existe un besoin d'algorithmes MOEA avec une vitesse de convergence améliorée pour gérer les problèmes d'optimisation multi-objectifs coûteux en calcul. Les algorithmes MOEA hybrides sont un type d'améliorations efficaces pour les algorithmes évolutionnaires multi-objectifs. Un algorithme MOEA hybride est un algorithme dans lequel une procédure de recherche locale composée d'un opérateur de recherche local est combinée avec un algorithme MOEA. L'opérateur de recherche local peut être considéré comme un opérateur supplémentaire dans un algorithme hybride MOEA. Un algorithme MOEA joue le rôle d'un solveur global pour trouver des régions prometteuses d'importance et une procédure de recherche locale améliore localement les individus d'une

population aux optima locaux les plus proches. Nous pouvons obtenir les avantages suivants en utilisant un algorithme hybride MOEA :

a) une vitesse de convergence accrue vers le front optimal de Pareto,

b) une convergence garantie vers le front optimal de Pareto

c) un critère de terminaison efficace.

Ensuite, nous présentons une classification pour l'hybridation d'algorithmes MOEA avec une procédure de recherche locale. Au moins deux approches différentes peuvent être utilisées pour l'hybridation d'un algorithme MOEA avec une procédure de recherche locale (Vasant, 2015):

II.6.1 Approche hybride simultanée

Un cadre général d'une approche hybride simultanée est illustré à la Figure II.11. Dans une approche hybride simultanée, une procédure de recherche locale est utilisée dans un algorithme MOEA pour améliorer certains individus d'une population dans une génération. De plus, une procédure de recherche locale peut être appliquée à tous les individus de la population finale pour garantir la convergence vers le front optimal de Pareto (au moins localement). Nous appelons un algorithme MOEA hybride basé sur une approche hybride simultanée comme algorithme MOEA hybride simultané. En améliorant localement quelques individus d'une population, la vitesse de convergence d'un algorithme MOEA hybride simultané peut être considérablement améliorée. Il existe un problème de comportement abordé concernant la mise en œuvre d'une approche hybride simultanée telle que :

a) fréquence de la procédure de recherche locale (une utilisation excessive d'une procédure de recherche locale peut consommer davantage d'évaluations de fonctions et une utilisation clairsemée de la procédure de recherche locale peut ne pas améliorer la vitesse de convergence),

b) choix des individus pour la procédure de recherche locale,

c) une procédure de recherche locale peut perturber l'équilibre dans l'étendue de l'exploration et de l'exploitation par un algorithme MOEA hybride simultané, et doit donc être rééquilibrée et d) un critère de terminaison efficace pour un algorithme MOEA hybride.

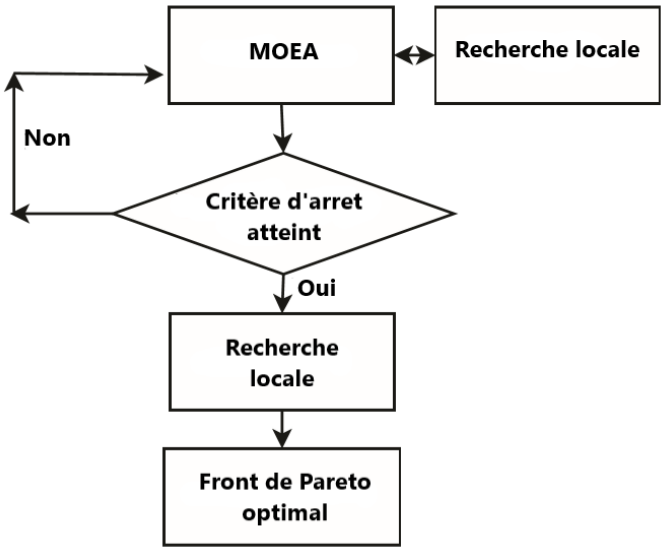

Figure III-11 Approche hybride simultanée (Vasant, 2015)

Plusieurs algorithmes MOEA hybrides simultanés peuvent être trouvés dans la littérature. (Ishibuchi & Murata, 1998) ont présenté probablement le premier algorithme multi-objectif hybride appelé MOGLS en hybridant l'algorithme génétique multi-objectif avec une procédure de recherche locale. La procédure de recherche locale a été utilisée sur chaque progéniture générée par des opérations génétiques. L'opérateur de recherche locale a consisté à utiliser une somme pondérée d'objectifs comme fonction de scalarisation avec des poids aléatoires et une recherche de voisinage, c'est-à-dire qu'un petit nombre de solutions voisines autour de chaque progéniture a été examinée pour trouver une progéniture améliorée. L'algorithme a été testé sur un problème d'ordonnancement de l'atelier de flux. (Jaszkiewicz, 2002) a proposé un nouvel algorithme de recherche génétique locale (GLS) pour l'optimisation combinatoire à objectifs multiples. Ici, une fonction d'utilité linéaire pondérée ou une fonction d'utilité de Chebyshev pondérée, avec des poids aléatoires, a été utilisée comme fonction de scalarisation dans l'opérateur de recherche local. (Goel & Deb, 2002) ont proposé des algorithmes hybrides basés sur l'approche hybride et série et ont testé sur un certain nombre de problèmes d'ingénierie d'optimisation de forme. Une somme pondérée d'objectifs est utilisée comme fonction de paramétrage pour l'opérateur de recherche local et une recherche de quartier pour trouver une progéniture améliorée. La procédure de recherche locale a été utilisée sur tous les descendants dans l'algorithme MOEA hybride simultané.

(Lara et al., 2010) ont proposé une nouvelle stratégie de recherche locale appelée le grimpeur avec pas de côté (HCS). Ici, la géométrie des cônes directionnels des problèmes d'optimisation est utilisée et peut fonctionner avec et sans informations de gradient. Selon la distance de la solution actuelle de l'ensemble optimal (localement) de Pareto, des solutions peuvent être générées à la fois vers et le long de l'ensemble optimal (localement) de Pareto. La force de cette stratégie est démontrée dans les problèmes de test de la suite de tests DTLZ (Deb, Thiele, et al., 2002) en hybridant HCS avec NSGA-II et SPEA2. En dehors des algorithmes mentionnés ici, plusieurs algorithmes MOEA hybrides simultanés proposés dans la littérature ont été résumés dans l'article (Sindhya et al., 2011).

II.6.2 Approche hybride série

Dans une approche hybride série, une procédure de recherche locale n'est utilisée qu'après la fin d'un algorithme MOEA. Nous appelons un algorithme MOEA hybride basé sur une approche hybride série un algorithme MOEA hybride série. L'utilisation d'une procédure de recherche locale après la fin d'un algorithme EMO garantit la pareto optimalité (locale) de la population finale d'un algorithme MOEA hybride série. Cependant, les algorithmes MOEA hybrides série peuvent présenter les inconvénients suivants :

(a) La vitesse de convergence d'un algorithme MOEA hybride série n'est pas améliorée pendant l'exécution d'un algorithme MOEA.

(b) Un critère de terminaison utilisant une procédure de recherche locale ne peut pas être conçu pour terminer un algorithme MOEA hybride série. Lorsqu'une approche hybride série est utilisée, la proximité de la population finale d'un algorithme MOEA hybride série ne peut pas être déterminée de manière efficace, c'est-à-dire que la population finale peut être soit loin du front optimal de Pareto soit à proximité du front optimal de Pareto. Ainsi, des évaluations excessives des fonctions peuvent être utilisées dans la procédure de recherche hélicoïdale appliquée à tous les individus d'une population finale qui est loin du front optimal de Pareto. De plus, tous les résultats individuels de la procédure de recherche locale peuvent se trouver sur un front optimal local de Pareto lorsque la population finale est loin du front optimal de Pareto.

Certains algorithmes basés sur des approches hybrides en série peuvent être trouvés dans la littérature. (Deb et al., 2016) ont proposé un algorithme MOEA hybride en série en plus de l'algorithme MOEA hybride simultané. Un opérateur de recherche locale utilisant la recherche de quartier a été utilisé dans la procédure de recherche locale et appliqué à toutes les solutions non dominées obtenues par NSGA-II. Une somme pondérée d'objectifs a été utilisée comme fonction de scalarisation dans l'opérateur de recherche local. (E. G. Talbi et al., 2001) ont utilisé une procédure de recherche locale comme moyen d'atteindre l'accélération et le raffinement de la recherche génétique. Ici, une fois qu'un ensemble de solutions non dominées a été obtenu, une procédure de recherche locale a été appliquée à tous les individus. Une somme pondérée d'objectifs a été utilisée comme fonction de scalarisation et une recherche de voisinage est effectuée pour trouver des solutions améliorées. L'algorithme a été testé sur des problèmes de flowshop. (Harada et al., 2006) a proposé un nouvel algorithme hybride utilisant la méthode de descente de Pareto (PDM) comme opérateur de recherche local. Ici, les directions de descente de Pareto ont été trouvées et des solutions améliorées ont été trouvées dans ces directions. Ainsi, une amélioration simultanée de tous les objectifs peut être obtenue. Un algorithme MOEA hybride série s'est avéré meilleur que l'algorithme MOEA hybride simultané, lorsqu'il a été testé sur des problèmes de référence avec des espaces de recherche continue disponibles dans la littérature.

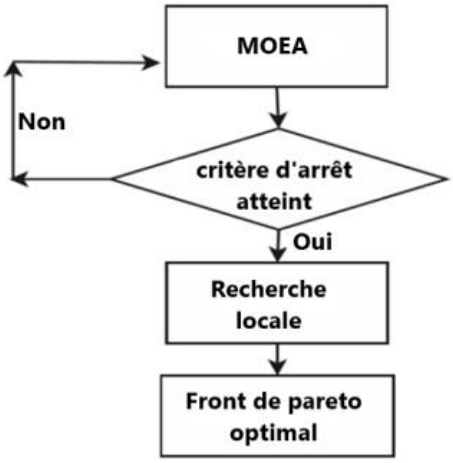

Figure III-12 Approche hybride série (Deb et al., 2016)

On peut voir ci-dessus que, dans la plupart des algorithmes, une somme naïve pondérée d'objectifs avec une procédure de recherche de voisinage est utilisée dans un opérateur de recherche local. Comme mentionné au chapitre I, une somme pondérée d'objectifs ne peut pas trouver toutes les solutions optimales de Pareto lorsque le front optimal de Pareto est non convexe. De plus, on peut également observer que la vitesse de convergence d'un algorithme MOEA ne peut être améliorée qu'en utilisant une approche hybride simultanée.

II.7 ÉTAT DE L'ART : HYBRIDATION BASÉ SUR LES AES

Comme indiqué dans la littérature, plusieurs techniques et méthodes heuristiques, méta-heuristiques et exactes ont été utilisées pour améliorer l'efficacité générale des AE. Nous examinerons brièvement certaines des architectures décrites ci-dessous.

La Figure II.3 illustre certaines possibilités d'hybridation. De l'initialisation de la population à la génération de descendants (Grosan & Abraham, 2007), il existe de nombreuses possibilités d'incorporer d'autres techniques/algorithmes, etc. La population peut être paraphée en incorporant des solutions connues ou en utilisant des heuristiques, des recherches locales, etc.

Les méthodes de recherche locales peuvent être incorporées dans les membres de la population initiale ou parmi les descendants. Les algorithmes évolutionnaires peuvent être hybridés en utilisant des opérateurs d'autres algorithmes (ou algorithmes eux-mêmes) ou en incorporant des connaissances spécifiques au domaine. Le comportement de l'AE est déterminé par la relation d'exploitation et d'exploration maintenue tout au long de l'exécution. Des algorithmes évolutionnaires adaptatifs ont été construits pour induire des relations d'exploitation et d'exploration qui évitent le problème de convergence prématurée et optimisent les résultats finaux. Les performances de l'algorithme évolutif peuvent être améliorées en combinant des connaissances spécifiques à des problèmes pour des problèmes particuliers.

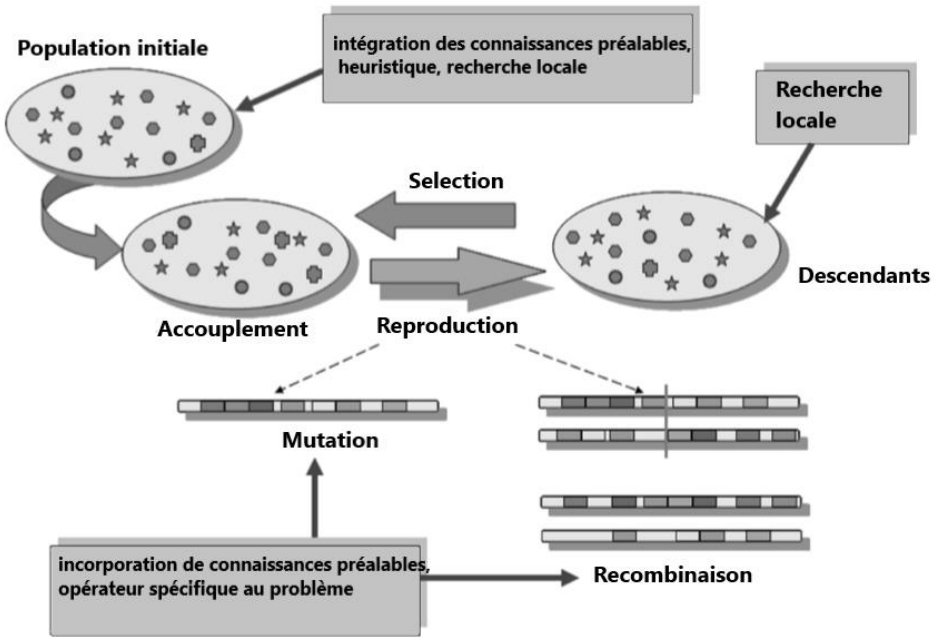

Figure III-13 Perspectives d'hybridation dans un AE (Grosan & Abraham, 2007)

L'intégration de différentes techniques d'apprentissage et d'adaptation, permettant de surmonter les limitations individuelles et d'obtenir des effets synergiques par hybridation ou fusion de ces techniques, a contribué au cours des dernières années à la mise au point d'un grand nombre de nouveaux systèmes évolutifs hybrides (Kramer, 2016). La plupart de ces approches suivent toutefois une méthodologie de conception ad hoc, justifiée par le succès rencontré dans certains domaines d'application. En l'absence d'un cadre commun, il reste souvent difficile de comparer conceptuellement les différents systèmes hybrides et d'évaluer leurs performances de manière comparative. Il existe plusieurs façons d'hybrider un algorithme évolutif classique pour résoudre les problèmes d'optimisation. Certains d'entre eux sont résumés ci-dessous (Cotta & Moscato, 2018):

- Les solutions de la population initiale d'EA peuvent être créées par une heuristique spécifique à un problème.
- Certaines ou toutes les solutions obtenues par l'AE peuvent être améliorées par la recherche locale. Ce type d'algorithme est connu sous le nom d'algorithme memetic.

- Les solutions peuvent être représentées de manière indirecte et un algorithme de décodage mappe n'importe quel génotype à une solution phénotypique correspondante. Dans ce mappage, le décodeur peut exploiter des caractéristiques spécifiques à un problème et appliquer des méthodes heuristiques, etc.
- Les solutions peuvent être représentées de manière indirecte et un algorithme de décodage mappe n'importe quel génotype à une solution phénotypique correspondante. Dans ce mappage, le décodeur peut exploiter des caractéristiques spécifiques à un problème et appliquer des méthodes heuristiques, etc.
- Les opérateurs de variation peuvent exploiter la connaissance du problème. Par exemple, lors de la recombinaison, des propriétés plus prometteuses d'une solution mère peuvent être héritées avec des probabilités plus élevées que les propriétés correspondantes de l'autre ou des autres parents. La mutation peut également être biaisée pour inclure dans les solutions des propriétés prometteuses avec des probabilités plus élevées que les autres.

II.7.1 Algorithmes évolutionnaires assistés par un autre AE

(Han, Mo, & Gao, 2018) proposent un algorithme d'évolution différentielle hybride adaptatif ADE pour l'identification dynamique des paramètres des machines tournantes. L'algorithme génétique (GA) est d'abord utilisé pour réduire la plage des paramètres d'identification afin de gagner du temps de calcul. Les mutations adaptatives de Cauchy et gaussiennes sont ensuite appliquées dans un algorithme d'évolution différentielle (DEA) et mises à jour dynamiquement au cours du processus évolutif en fonction du processus d'identification, pour obtenir plus facilement et plus rapidement la solution optimale globale.

(Molina et al., 2010) un optimiseur puissant est introduit en combinant deux AEs populaires, à savoir GA et CMA-ES. L'algorithme résultant fonctionne parfaitement sur les problèmes de faible dimension. La tâche principale de l'AG est évidemment l'exploration, tandis que CMA-ES est utilisé ici pour LS. Il faut noter que le CMA-ES fourni les meilleurs résultats par rapport aux fonctions uni-modales et en définissant la taille de pas initiale σ sur une petite valeur, il peut en effet être considéré comme une routine LS appropriée.

(Kämpf & Robinson, 2009) proposent une hybridation de CMA-ES et des algorithmes d'évolution différentielle hybride (HDE) couplés à une technique efficace de traçage des rayons vers l'arrière. Le nouvel algorithme surpasse les algorithmes autonomes CMA-ES et HDE dans les tests de référence et dans l'optimisation multi-objectif du problème d'optimisation solaire.

En optimisation multi-objective une approche hybride efficace proposée (Kaikai, 2015) pour améliorer la précision de recherche et la diversité de l'algorithme génétique de tri non dominé, NSGA-II est hybridé avec une évolution différentielle DE. L'algorithme utilise l'opérateur de guidage de mutation et l'opérateur de croisement de DE pour remplacer l'opérateur de croisement dans NSGA-II afin d'améliorer la capacité de recherche locale et d'améliorer la précision de la recherche. Tout en conservant l'opérateur de mutation de NSGA-II pour améliorer la diversité.

II.7.2 Algorithmes évolutionnaires assistés par réseaux de neurones (NN)

NN présente plusieurs inconvénients (Inthachot et al., 2016) tels qu'une longue durée d'apprentissage, une convergence indésirable vers une solution optimale locale plutôt que

globale et un grand nombre de paramètres ; par conséquent, il y a eu des tentatives pour remédier à certains de ces inconvénients en combinant ANN avec des AEs.

La plupart des travaux dans ce domaine ont commencé à voir le jour en 2017. Une étude expérimentale montre que l'intégration de l'algorithme génétique GA dans une architecture de réseau de neurones à convolution CNN améliore la précision du problème de reconnaissance numérique manuscrite (Trivedi et al., 2018). Dans les travaux proposés par (Suganuma et al., 2017), la programmation génétique GP est plutôt utilisée pour faire évoluer l'architecture du CNN. Pendant ce temps, (Bochinski, et al., 2018) ont proposé IEA-CNN, une approche utilisant une stratégie évolutionnaire ES, innovant en triant les couches évoluées en décroissant la complexité, réduisant efficacement l'espace de recherche de manière factorielle sur le nombre de couches.

Enfin, (Kramer, 2018) est présenté une approche basée sur un algorithme évolutionnaire qui ne repose que sur l'opérateur de mutation et a introduit un mécanisme pour prendre en charge l'héritage des paramètres, afin que les descendants au cours du processus évolutionnaire n'ont pas besoin d'apprendre les poids à partir de début.

II.7.3 Algorithmes évolutionnaire assistés par logique floue

Les contrôleurs de logique floue FLC ont été utilisés pour concevoir des AEs adaptatifs. L'idée principale est d'utiliser un FLC dont les entrées sont une combinaison quelconque de performances de EA et de valeurs de paramètre de contrôle actuelles et dont les sorties sont des valeurs de paramètre de contrôle EA. (Lotfy et al., 2018) propose AG-FLC qui utilise l'AG pour régler les paramètres de la fonction d'appartenance de FLC pour obtenir des performances optimales et en même temps pour éliminer le principal inconvénient de FLC présenté dans le fait que la décision des paramètres d'appartenance d'entrée et de sortie de FLC est une tâche longue et complexe qui souvent, ils ne conduisent pas à des résultats satisfaisants.

(Salehpour et al., 2017) une nouvelle version de l'évolution différentielle (DE) avec facteur de mutation adaptatif. L'algorithme proposé utilise un système d'inférence de logique floue pour régler dynamiquement le facteur de mutation de DE et améliorer son exploration et son exploitation. Deux facteurs, nommés, le nombre de générations et la diversité de la population sont considérés comme des entrées et, un facteur, nommé, le facteur de mutation comme une sortie du système d'inférence à logique floue.

(Snášel et al., 2016) ont rapporté des méthodes d'adaptation étroitement couplées, découplées et faiblement couplées. Trois niveaux d'adaptation étroitement couplés peuvent être mis en œuvre au niveau des individus, du niveau des sous-populations et du niveau de la population. Dans une adaptation découplée, un mécanisme adaptatif totalement séparé ajuste les performances de l'EA. Il est à noter qu'une approche non couplée ne repose pas sur l'évaluation environnementale pour le mécanisme d'adaptation. Dans la méthode à couplage lâche, l'EA est partiellement utilisée pour le mécanisme adaptatif, c'est-à-dire que la population ou les opérateurs génétiques sont utilisés d'une certaine manière.

II.7.4 Algorithmes évolutionnaires assistés par l'optimisation de l'essaim de particules (PSO)

PSO intègrent des comportements réconfortants observés chez des oiseaux, des bancs de poissons ou des essaims d'abeilles, voire un comportement social humain, à partir duquel l'idée est née (Eberhart & Kennedy, 1995). (G. Tian et al., 2016) utilisent un algorithme hybride à objectifs multiples DE + PSO afin de créer un ensemble de solutions Pareto pour le problème

de la planification à double objectif des véhicules de sauvetage afin de distinguer les incendies de forêt. Les résultats montrent que l'approche proposée est capable de produire rapidement des solutions Pareto satisfaisantes par rapport aux algorithmes GA et PSO.

(Cherki et al., 2019) proposent une hybridation séquentielle pour résoudre le problème de flux de puissance réactif optimal. L'algorithme de cette hybridation commence par l'exécution de la méthode GA puis par l'exécution de la méthode PSO. Cela signifie que la solution donnée par l'algorithme génétique est choisie comme solution initiale de l'optimisation de l'essaim de particules.

(Millie & Radha, 2011) proposent une approche hybride pour résoudre les problèmes d'optimisation globale nommée DE-PSO qui consiste à exécuter alternativement les deux phases DE et PSO. Le paramètre déterminant de l'algorithme PSO est le coefficient d'hybridation (HC), qui exprime le pourcentage de la population qui, à chaque itération, évolue avec GA. Donc, HC = 0 signifie que la procédure est un PSO pur (la population entière est mise à jour en fonction des opérateurs de PSO), HC = 1 signifie GA pure, alors que $0 < HC < 1$ signifie que le pourcentage correspondant de la population est développé par GA et que reste par PSO.

(Mao et al., 2018) Combinent les avantages du DE et du PSO, une nouvelle stratégie hybride DEMPSO est proposée qui vise à atteindre à la fois une vitesse de convergence rapide et une optimisation globale efficace. Étant donné que la population de PSO tombe facilement dans un optimum local, la méthode DEMPSO proposée utilise d'abord l'algorithme DE pour réduire l'espace de recherche, puis les populations obtenues sont reprises par le MPSO en tant que population initiale pour obtenir un taux de convergence rapide vers l'optimum global final. L'algorithme hybride peut obtenir la valeur minimale globale basée sur la fonction de fitness, qui est conçue pour la solution numérique pour la cinématique avant d'un manipulateur parallèle.

II.7.5 Algorithmes évolutionnaires assistés par l'optimisation de la colonie de fourmis (ACO)

ACO traite des systèmes artificiels inspirés du comportement alimentaire des fourmis réelles, utilisés pour résoudre des problèmes d'optimisation discrets (Lenoir & Monmarché, 2009).

L'algorithme d'optimisation par colonie de fourmis est caractérisé par sa robustesse, ainsi il est facile à combiner avec d'autres méthodes d'optimisation, mais il présente l'inconvénient de stagnation, ce qui limite son application dans les différents domaines. Pour surmonter cet inconvénient, une approche hybride est proposée dans (Rahmat et al., 2014). Cette approche est basée sur l'hybridation d'un algorithme d'optimisation par colonies de fourmis avec un algorithme d'évolution différentielle. L'approche proposée a été testée sur le problème du voyageur de commerce (TSP). Les résultats expérimentaux montrent son efficacité. Une autre approche hybride pour le même problème a été proposée par (Emel Soylu & Uysal, 2017), ou AG est utilisé pour optimiser les meilleures valeurs des paramètres de l'algorithme de la colonie de fourmis. De cette façon, le taux de réussite de l'algorithme de la colonie de fourmis est maximisé.

Une approche hybride adaptative combinant l'algorithme génétique et l'optimisation des colonies de fourmis pour la planification et l'ordonnancement intégrés, l'approche hybride GA-ACO proposée par (Uslu et al., 2019) exécute simultanément GA et ACO, et la plus performante est sélectionnée comme algorithme principal

Dans (Chang et al., 2012), un algorithme évolutionnaire hybride basé sur la stratégie d'évolution différentielle DE et l'optimisation par colonies de fourmis ACO a été proposé pour résoudre le problème de la conception d'antennes à large bande. Cette hybridation a prouvé son efficacité par son application pour optimiser plusieurs fonctions mathématiques.

Une nouvelle approche multi-objectif appelée acNSGA-III (Mnasri et al., 2017) combine NSGA-III avec ACO est mis en œuvre avec succès pour résoudre le problème du déploiement de nœuds 3D dans un réseau de capteurs sans fil (WSN).

II.7.6 Algorithmes évolutionnaires assistés par la recherche de bactéries (BFO)

Le comportement de recherche de nourriture sociale d'Escherichia coli a récemment fait l'objet d'une grande attention et a été utilisé pour résoudre des problèmes complexes d'optimisation de la recherche (S. Das, Biswas, Dasgupta, & Abraham, 2009). Le comportement alimentaire de la bactérie Escherichia coli est imité. Ils subissent différentes étapes telles que la chimiotaxie, l'essaimage, la reproduction, l'élimination et la dispersion. Au stade de la chimiotaxie, il peut avoir une chute suivie d'une chute ou une chute suivie d'une course. En revanche, en essaimage, chaque bactérie E. coli en signalera une autre via des attractifs pour essaimer ensemble. Lors de l'élimination et de la dispersion, toute bactérie est éliminée de l'ensemble total simplement en la dispersant dans un emplacement aléatoire sur le domaine d'optimisation.

Un algorithme hybride sequentiel (Manikandan & Ramyachitra, 2017) nommé Bacterial Foraging Optimization-Genetic Algorithm (BFO-GA) vise à améliorer les multi-objectifs et à réaliser des mesures d'alignement de séquences multiples.

Un algorithme un autre BFO combiné avec DE pour faire face à la convergence prématurée de l'opérateur de reproduction. Dans ICDEOA (Yildiz et al., 2015), l'opérateur de reproduction de BFOA est remplacé par l'opérateur de repositionnement probabiliste pour améliorer l'intensification et la diversification de l'espace de recherche. Dans une autre approche hybride (Vaisakh et al., 2013), la chimiotaxie computationnelle dans le BFOA sert de recherche locale basée sur la descente de gradient stochastique. L'algorithme hybride résultant est appelé ici CDE (Chemotactic Differential Evolution) incorporant une étape chimiotactique adaptative empruntée au domaine du BFOA dans le DE.

II.7.7 Algorithmes évolutionnaires incorporant des connaissances préalables

Il existe plusieurs approches existantes qui n'utilisent pas une population initiale générée aléatoirement pour les algorithmes évolutifs. Si des connaissances préalables existent ou peuvent être générées à un faible coût de calcul, de bonnes estimations initiales peuvent générer de meilleures solutions avec une convergence plus rapide (Wang et al., 2011).

(Fan et al., 2017) ont proposé une connaissance préalable du DE guidé qui adaptent les paramètres de contrôle pour répondre aux exigences d'exploration et d'exploitation à différentes étapes au cours de l'évolution. (Lwin et al., 2014) proposent une stratégie de génération de solutions guidée par l'apprentissage est intégrée au processus d'optimisation multi-objectifs pour favoriser la convergence efficace en guidant la recherche évolutive vers les régions prometteuses de l'espace de recherche.

Une technique d'apprentissage nouvelle et efficace appelée apprentissage basé sur l'opposition OBL est hybridée avec des algorithmes évolutionnaires comme stratégie d'initialisation (Mahdavi et al., 2018) pour générer une population et améliorer la convergence. OBL est

également combiner avec des algorithmes évolutionnaires multi-objectif comme nouvel mécanisme de sélection (Wang et al., 2019).

II.7.8 Approches hybrides intégrant recherche locale et autres

Le terme Algorithme Mimétique (AM) a été introduit pour la première fois en 1989 par Moscato (Moscato, 1989). Le terme « mémétique » trouve ses racines dans le mot « meme » introduit par (Dawkins, 1976) pour désigner l'unité de l'imitation dans la transmission culturelle. L'idée essentielle derrière les MA est la combinaison de techniques locales de raffinement de recherche avec une stratégie basée sur la population, comme les algorithmes évolutionnaires.

La principale différence entre les algorithmes génétiques et mémétiques est l'approche et la vue des techniques de transmission de l'information. Dans les AG, l'information génétique transmise par les gènes est généralement transmise intacte à la progéniture, alors que dans les MA, les unités de base sont les soi-disant « mèmes » et elles sont généralement adaptées par l'information de transmission individuelle. Alors que les AG sont bons à explorer l'espace de solution à partir d'un ensemble de solutions candidates, les AM explorent à partir d'un seul point, permettant d'exploiter des solutions qui sont proches des solutions optimales.

Exploration de l'espace de recherche (diversification) et exploitation des meilleures solutions trouvées (intensification) sont deux critères contradictoires dont il faut tenir compte lors de la conception ou de l'utilisation d'une métaheuristique (Humeau et al, 2013). Selon ces critères, les algorithmes évolutionnaires peuvent être classés en deux familles ; Algorithmes basés sur la population (AE) qui sont orientés vers l'exploration et basés sur une solution unique (recherche locale) qui sont axés sur l'exploitation. Un bon équilibre entre ces deux objectifs améliorera les performances de l'algorithme de recherche. Un choix pour atteindre cet équilibre consiste à utiliser un modèle hybride où au moins deux techniques sont combinées pour améliorer les performances de chaque technique. L'hybridation entre algorithmes évolutionnaires et recherche locale s'appelle algorithmes mémétique. Ce dernier, a été utilisée par de nombreux chercheurs dans le domaine de l'optimisation et ont montré des performances supérieures pour résoudre de nombreux problèmes pratiques ou académiques par rapport à d'autres algorithmes de recherche locaux ou globaux. Les algorithmes Mémétique se sont avérés être des ordres de grandeur plus rapides et plus précis que les algorithmes évolutionnaires pour différentes classes de problèmes. Comme indiqué dans la littérature, des méthodes hybrides combinant des méthodes probabilistes et des méthodes déterministes ont permis de résoudre des problèmes d'optimisation complexes (Cotta et al., 2018).

Les MA (Memetic Algorithm) sont progressivement devenus l'un des domaines de recherche en croissance récents dans le calcul évolutif. Ils combinent divers algorithmes évolutionnaires avec différentes méthodes LS (Local search) pour équilibrer l'exploration et l'exploitation. Les exemples existants d'algorithmes mémétiques sont NM-BRO (Ahandani et al, 2014), MA-LSCh-CMA (Molina et al., 2010), LBBO (Simon et al, 2014), IMMA (Sun, Garibaldi, Krasnogor, & Zhang, 2013). Dans le cadre des AG, les opérateurs LS sont utilisés pour exécuter une exploitation supplémentaire pour les individus générés par des opérations d'AE communes, ce qui est utile pour améliorer la capacité de l'AE à résoudre des problèmes complexes.

Voici quelques décisions importantes qui devraient être prises lors de la conception des MA :

a. À quel moment du processus d'évolution la recherche locale doit-elle être effectuée ?

b. À quelle fréquence la recherche locale doit-elle être appliquée tout au long du processus évolutionnaire ?

c. À partir de quelles solutions la recherche locale doit-elle être lancée ?

La combinaison d'opérateurs d'amélioration locale s'inscrit dans les étapes évolutives d'une évaluation de l'algorithme évolutionnaire afin d'améliorer les solutions qui sont proches de devenir optimales. Cela a été démontré dans plusieurs domaines d'application pour apporter des améliorations aux résultats standard obtenus par les AG autonomes en termes de qualité des résultats et de vitesse de convergence. En général, il n'existe pas de méthode spécifique pour concevoir une AM.

(Asafuddoula et al, 2014) ont proposé un algorithme DE hybride adaptatif (AH-DEa) qui a trois caractéristiques. Le premier est son utilisation de taux de croisement adaptatifs à partir d'un ensemble donné de valeurs discrètes. La seconde est une stratégie de croisement adaptatif à différentes étapes de l'évolution. Le dernier est l'inclusion d'une stratégie de recherche locale pour améliorer encore la meilleure solution.

(Kelner et al, 2008) ont proposé un algorithme hybride comme une combinaison d'un GA et une stratégie de recherche locale basée sur la méthode du point intérieur, pour la résolution de modèles mathématiques multi-objectifs contraints.

(de Freitas et al, 2014) ont proposé une méthode hybride basé sur les stratégies évolutionnaires (ES), la recherche tabou l'autoadaptation des paramètres, où la recherche locale est appliquée avec modération (intensive et faible) dans plusieurs phases de l'exécution.

Un autre algorithme memétique basé sur les stratégies évolutionnaires est proposé par (Coelho et al., 2016), combiner par la puissance de la recherche de voisinage variable réduite (RVNS) et l'algorithme de recherche de trajectoire à travers une adaptation technique.

(Dumas et al., 2009) ont proposé de combiner une évaluation environnementale, un processus de regroupement et une procédure de recherche locale pour la conception évolutive des réseaux de neurones. L'approche proposée sélectionne un sous-ensemble des meilleurs individus, effectuez une analyse de groupe pour les regrouper et optimisez uniquement le meilleur individu de chaque groupe. L'utilisation d'un algorithme de classification permet la sélection d'individus représentant différentes régions de l'espace de recherche. De cette manière, les individus optimisés sont plus susceptibles de converger vers différents optima locaux.

Plusieurs recherches locales, peuvent être intégrées dans un algorithme hybride. Par exemple, dans (Iacca et al., 2014), un optimiseur mémétique basé sur DE est proposé en étendant l'algorithme EPSDE, avec un pool PLS supplémentaire de diverses routines de recherche locale : méthode simplex Nelder-Mead, la méthode de direction conjuguée de Powell et algorithme de Rosenbrock. Nelder Mead contient, dans une certaine mesure, des fonctionnalités de recherche globale et a donc le potentiel de sauter en dehors d'un bassin d'attraction tandis que les algorithmes de Powell et Rosenbrock exploitent de manière déterministe la solution de départ vers l'optimum le plus proche.

II.8 CONCLUSION

Ce chapitre fournit un état de l'art sur l'hybridation des algorithmes évolutionnaires avec d'autres méthodes d'optimisations stochastiques et exactes de la littérature. Nous avons commencé une introduction qui définit les critères permettant de comparer les différents algorithmes, leurs limites, telles que le temps de calcul qui est souvent élevé, le problème d'ajustement des paramètres et la convergence prématurée. L'introduction des approches hybrides est nécessaire pour surmonter ces problèmes. Par exemple, les algorithmes évolutionnaires sont très performants lorsqu'il s'agit d'explorer l'espace de recherche, mais ils s'avèrent ensuite incapables d'exploiter efficacement la zone vers laquelle la population

des solutions converge. Il est alors plus intéressant d'utiliser à ce stade une autre stratégie hybride qui assure l'équilibre entre la recherche globale (exploitation) et la recherche locale (exploitation) à travers une combinaison efficace entre les différentes méthodes d'optimisations. Nous avons fait le point sur les différentes stratégies de classification qui existent dans la littérature, ainsi que sur quelques approches hybrides proposées. Nous avons constaté que la taxonomie proposée par (E.-G. Talbi, 2009) est très simple et plus générale car elle englobe tous les types d'hybridations qui sont adaptables à un très grand nombre de problèmes d'optimisation qui permet de mieux classifier les différentes hybridations en les comparant de façon qualitative.

Dans le contexte de cette thèse, nous allons présenter et développer de nouvelles approches hybrides basée sur la combinaison des AE avec d'autres techniques d'optimisations et de l'intelligence artificielle afin de réduire le cout de calcul et améliorer les performances des AE.

La première approche (chapitre III) propose un algorithme adaptatif hybride à deux phases basées sur les stratégie évolutionnaire (ES) et assisté par une méthode de regroupement (clustering) et autre de recherche locale (Nelder-Mead) pour accélérer la convergence de ES et surmonter les optima locaux à travers un critère de contraction.

La deuxième approche (chapitre IV) vise essentiellement à réduire la complexité des algorithmes évolutionnaires multi-objectif à travers une approche hybride coopératif basé sur l'algorithme différentielle multi objectif (MODE) et une technique d'apprentissage par opposition (OBL) afin de générer une diversité de solution efficace pour le décideur (DM).

CHAPITRE III :

CONTRIBUTION 1 : APPROCHE HYBRIDE PROPOSÉE POUR RESOUDRE DES PROBLEMES D'OPTIMISATION CONTINUE

III.1 INTRODUCTION

Des efforts ont été faits dans de nouvelles méthodes, dont le processus évolutionnaire n'est qu'une partie de l'ensemble du processus de recherche. En raison de leurs caractéristiques intrinsèques en tant que solveur global, les AE sont utilisés comme un générateur de zones de recherche, qui sont plus intensivement inspectés par un composant heuristique. Ce scénario vient renforcer le concept collaboratif entre les composants de l'optimiseur.

Les méthodes de recherche locale ont été combinées à des AE de différentes manières pour résoudre des problèmes particuliers plus efficacement. Le gradient ainsi que les méthodes de recherche directe ont été utilisés comme outil d'exploitation dans l'optimisation continue. Dans les cas où il n'y a pas de dérivés disponibles pour la fonction objective, les méthodes de recherche de directes sont très utiles pour fournir une convergence plus robuste (Martinez & Coello Coello, 2013).

Il a été constaté (de Freitas et al., 2014) que la majorité des modèles hybrides qui ont une architecture d'intégration bénéficient de caractéristiques de convergence robustes. En effet, les architectures intégratives appliquent un raffinement local à presque tous les exemples de solutions à chaque itération ; cela permet une exploration intense de l'espace de recherche, mais il entrave l'efficacité globale de la recherche. D'un autre côté, les architectures collaboratives appliquent le raffinement local à seulement une ou plusieurs solutions sélectionnées et seulement à certains stades définis de l'évolution. De telles architectures favorisent l'efficacité de la recherche mais risquent de faibles garanties de convergence. Par conséquent, il est extrêmement difficile d'équilibrer l'efficacité de la recherche et de solides garanties de convergence lorsque l'une ou l'autre de ces méthodologies est utilisée. Par conséquent, une question d'actualité est ici - pourquoi utiliser les systèmes hybrides ? Le fait est que si les méthodes stochastiques permettent pour la plupart une exploration globale efficace, ce sont les méthodologies déterministes qui permettent une bonne exploitation locale ; ainsi, l'hybridation est probablement le meilleur moyen d'atteindre les deux.

Le principal défi de ces méthodes hybrides consiste à définir des stratégies efficaces couvrant tous les espaces de recherche, en appliquant la recherche locale que dans les zones de recherche réellement prometteuses. L'inspiration dans la nature a été poursuivie pour concevoir des modèles de calcul flexibles, cohérents et efficaces. L'élitisme joue un rôle important dans la réalisation de cet objectif, car les meilleures solutions représentent une région prometteuse. Une recherche de regroupement est proposée comme un moyen générique qui combine l'AE de recherche avec le clustering pour détecter les zones de recherche prometteuses avant d'appliquer des procédures de recherche locales. Le processus de regroupement vise à rassembler des informations similaires sur l'espace de recherche en groupes, en maintenant une solution représentative associée à ces informations.

Ce chapitre propose une approche hybride générique basé sur la collaboration des composants y compris AE, la recherche par regroupement, recherche locale, critère d'attraction, réinitialisation. Cette approche générique est réalisée à la fois par la possibilité d'utiliser n'importe quel EA et également en s'appliquant aux problèmes d'optimisation continue.

III.2 INSPIRATION DE L'APPROCHE PROPOSÉ

Pour plusieurs problèmes, un simple algorithme évolutionnaire peut suffire à trouver la solution désirée. Comme indiqué dans la littérature, il existe plusieurs types de problèmes pour lesquels un algorithme d'évolution directe pourrait ne pas obtenir une solution commode (optimale) (Dumas et al., 2009). Cela ouvre clairement la voie à la nécessité d'hybridation d'algorithmes évolutifs avec d'autres algorithmes d'optimisation, de techniques d'apprentissage

automatique, d'heuristiques, etc. Certaines des raisons possibles de l'hybridation sont les suivantes (de Freitas et al., 2014):

1. Améliorer les performances de l'algorithme évolutionnaire (vitesse de convergence)
2. Améliorer la qualité des solutions obtenues par l'algorithme évolutionnaire
3. Incorporer l'algorithme d'évolution dans le cadre d'un système plus vaste

Plusieurs processus naturels peuvent être simulés par des algorithmes évolutifs, enrichissant leurs capacités, étendant leur applicabilité et améliorant leurs performances. La migration des individus entre les zones de recherche peut être prise avec une interprétation du comportement social. Une société correspond à un groupe de points dans l'espace de recherche et l'ensemble de toutes les sociétés compose une civilisation. Chaque société a ses dirigeants qui aident les autres individus à s'améliorer grâce à un échange d'informations intra société.

L'échange d'informations intra-société est analogue à une recherche locale intensifiée autour d'un point plus performant, ce qui entraîne le déplacement des points dans le cluster. Les dirigeants d'une société ne peuvent s'améliorer que grâce à un échange d'informations inter sociétés qui se traduit par la migration d'un dirigeant d'une société à une autre. L'échange d'informations inter-sociétés est analogue à une recherche autour de régions globalement prometteuses dans l'espace de recherche.

Une autre interprétation des clusters et de la migration des individus provient de l'analogie avec les écosystèmes dans la nature (H. Ma et al., 2017). Chaque zone de recherche peut être considérée comme une région géographique où des espèces ou des sociétés distinctes évoluent en même temps. Chaque groupe d'individus vit et se reproduit dans sa région respective. Ils recherchent des ressources naturelles par leur stratégie particulière d'exploration. Parfois, la région est inappropriée pour être explorée par de longues périodes et les ressources se raréfient, obligeant les individus à migrer vers une autre région ou même à y mourir. La migration de groupes d'individus permet de découvrir d'autres zones, en gardant habité les plus riches en ressources.

Les méthodes de recherche locales ont été combinées aux AEs de différentes manières pour résoudre plus efficacement des problèmes particuliers. Les méthodes de recherche par gradient et directe ont été utilisées comme outil d'exploitation dans l'optimisation continue. Dans les cas où aucun dérivé n'est disponible pour la fonction fitness, les méthodes directes sont utiles pour fournir une convergence plus robuste (Oliveira & Lorena, 2007).

Nous espérons que des groupes de solutions proches les unes des autres pourront correspondre à des domaines d'attraction pertinents dans la plupart des AE de recherche. Les zones de recherche pertinentes peuvent être traitées avec un intérêt particulier par l'algorithme dès qu'elles sont découvertes. Les clusters fonctionnent comme des fenêtres coulissantes, encadrant les zones de recherche et donnant un point de référence aux procédures de recherche locale spécifiques à un problème. De plus, le centre de cluster lui-même est toujours mis à jour par une interaction permanente avec des solutions internes.

Cette idée de base a été utilisée pour proposer la recherche de classification évolutive appliquée à l'optimisation continue sans contrainte (Dumas et al., 2009). Postérieurement, la recherche guidée par le regroupement a été étendue à l'optimisation multi-objective (Saha & Bandyopadhyay, 2013) afin d'améliorer le taux de convergence de l'algorithme, ainsi que son exploration.

Bien que les algorithmes évolutionnaires puissent atteindre le minimum global d'une fonction, la quantité d'effort de calcul et le temps requis par eux peuvent être importants. Cette

approche démontre une réduction de cet effort et de ce temps en trouvant initialement des solutions approximatives en utilisant l'algorithme ES auto-adaptative, puis en appliquant la méthode d'optimisation Nelder-Mead (NM) pour les affiner davantage. NM est une méthode d'optimisation qui est plus efficace pour les fonctions sans contraintes, et ne peut donc pas être appliquée directement au problème d'optimisation dans de nombreux cas car elle peut conduire à une solution irréalisable.

La section suivante décrit les composants conceptuels de la nouvelle approche et l'effet de chaque composant sur le processus générique d'optimisation.

III.3 LES COMPOSANTS DE L'ALGORITHME HYBRIDE PROPOSÉ : ES-NM

Dans cette section, nous décrivons en détail cinq opérations principales de l'algorithme hybride proposé, y compris, l'algorithme de stratégie évolutionnaire auto-adaptative (SA-ES), un critère de contraction, une stratégie de clustering, la Méthode exacte du simplex (Nelder-Mead) adaptative et schéma de redémarrage. Un schéma général (voir Figure II.1) de la nouvelle méthode algorithme évolutionnaire\méthode exacte appelé ES-NM est donné à la Figure III.2 et défini par pseudo code dans l'algorithme 10. Le but de la combinaison des deux algorithmes est de tirer profit des forces des deux méthodes. ES s'est avéré avoir de bonnes performances d'exploration comme recherche globale mais a des problèmes avec l'exploitation dans une zone prometteuse de l'espace de recherche, cependant NM est une procédure de recherche locale très efficace en raison de son bon taux de convergence et de la simplicité de l'algorithme. La méthode NM ne nécessite généralement qu'une ou deux évaluations de fonction par itération (sauf en rétrécissement, ce qui est rare dans la pratique), Par conséquent, il peut être utilisé pour améliorer les capacités d'intensification ES.

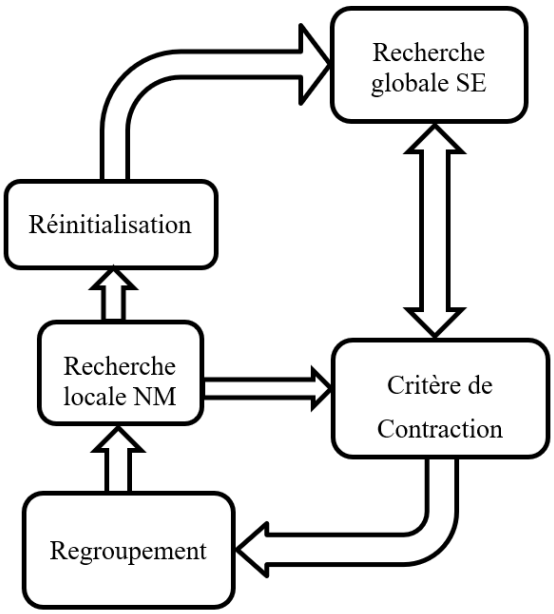

Figure IV-1 schéma général de l'approche proposé

Un défi dans les algorithmes évolutifs hybrides est d'employer des stratégies pour couvrir tout l'espace de recherche, en appliquant la recherche locale uniquement dans les zones de recherche réellement prometteuses. L'élitisme joue un rôle important dans la réalisation de cet objectif, car les meilleures solutions représentent un voisinage prometteur. L'inspiration dans la nature a été recherchée pour concevoir des modèles informatiques flexibles, cohérents et efficaces. Dans ce chapitre, la recherche de clusters est proposée comme un moyen générique

de combiner des métaheuristiques de recherche avec des clusters pour détecter des zones de recherche prometteuses avant d'appliquer des procédures de recherche locales. Le processus de clustering vise à rassembler des informations similaires sur l'espace de recherche en groupes, en maintenant une solution représentative associée à ces informations.

L'organigramme du nouvel algorithme est présenté à la Figure III-2, qui illustre le schéma hybride général avec différentes phases. Partant de l'exploration assurée par la stratégie d'évolution (ES), puis passant à l'exploitation fournie par la méthode de Nelder-Mead (NM) qui s'appliquait à un certain nombre d'individus. La stratégie de réinitialisation a lieu lorsqu'il n'y a pas d'amélioration de nouvelles solutions.

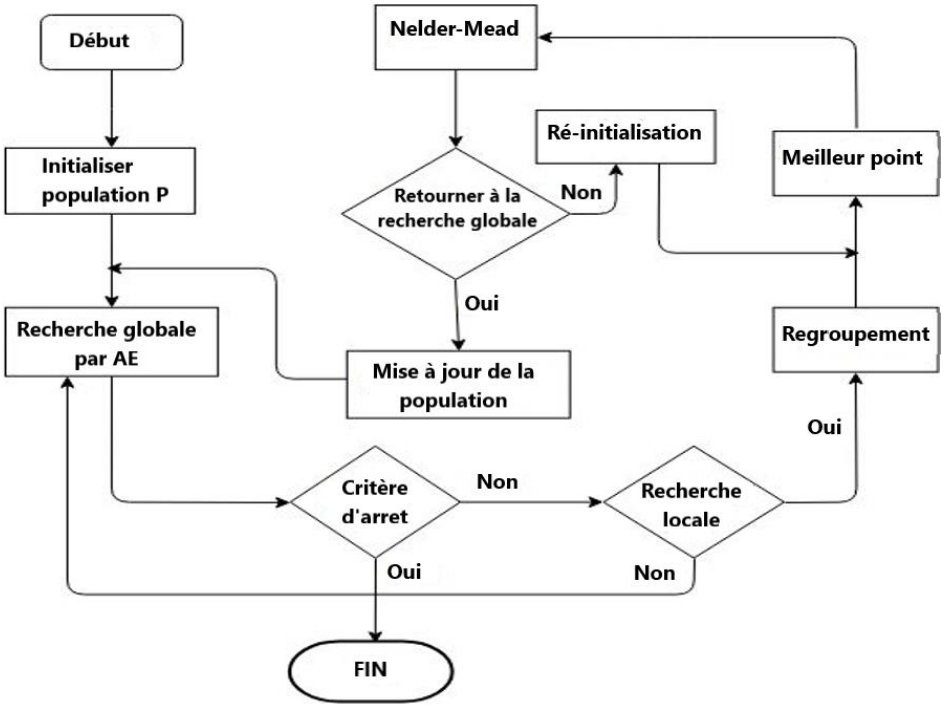

Figure IV-2 Organigramme de l'algorithme hybride ES-NM (Boukhari et al., 2019)

III.3.1 Algorithme évolutionnaire

À ce jour, l'algorithme de stratégie évolutionnaire (SE) a réussi à optimiser diverses fonctions non linéaires continues dans la pratique. Comme c'est le cas pour les algorithmes génétiques, il consiste à suivre l'évolution d'une population de solutions potentielles à travers les trois mêmes principes stochastiques, sélection, recombinaison et mutation. Cependant, contrairement aux algorithmes génétiques, le processus majeur est le processus de mutation et la sélection est rendue déterministe (Dumas et al., 2009). Il est connu de la littérature que bien que l'algorithme ES finisse par localiser la solution désirée, l'utilisation pratique d'une technique de calcul évolutif pour résoudre les problèmes d'optimisation multimodale est fortement limitée par le coût de calcul élevé du taux de convergence lent (Gonzalez-Fernandez & Chen, 2014). Le taux de convergence des ES est également généralement plus lent que ceux des techniques de recherche locales, car ils ne dépendent pas simplement des informations locales pour déterminer une direction de recherche des plus prometteuses.

La stratégie de mutation appliquée dépend des paramètres du vecteur y_i. Dans le cas le plus simple, le vecteur de paramètres contient une seule valeur, $y_i = (\sigma_i)$. Dans ce cas, la stratégie de mutation consiste à ajouter une variable normale univariée $r_i \sim N(0, \sigma_i)$, et les paramètres endogènes contrôlent la force de la mutation (variance de la distribution aléatoire). Le vecteur

de paramètre de la nouvelle progéniture y_i est également muté, en utilisant une règle différente. Chacun des parents μi porte son propre σ_i. Après avoir généré chacun des λ_i, le paramètre σ_i' hérité du parent est muté en appliquant la règle log-normale mentionnée dans l'équation (1) :

$$\sigma_i' \rightarrow \sigma_i * \exp(\tau * N(0,1) \qquad \text{(III.1)}$$

Où le paramètre d'apprentissage τ contrôle la force de la mutation et $N\ (0, 1)$ est une variable normale univariée standard. Le taux d'apprentissage généralement recommandé est $\tau = 1 / \sqrt{N}$ où N est la dimension de l'espace de recherche. Pour les fonctions hautement multimodales, des valeurs telles que $\tau = 1 / \sqrt{2N}$, ou même plus petites, sont recommandées. Un aperçu général est présenté dans l'algorithme 1.

Algorithme 7 : pseudo-code pour l'algorithme de stratégie d'évolution (μ, λ)-ES

```
P ← { }
pour λ fois faire
    xk ← point de départ aléatoire
    yk ← f(xk)
    σk ← variance de départ aléatoire
    P ← P U {(xk, yk, σk)}
fin pour
pour toutes générations faire
    Q ← meilleurs μ solutions dans P
    P ← { }
     pour tout parent Qk faire
           pour λ fois faire
                 (xk, yk, σk) ← Qi
                  xk ← xk + N(1, σi)n
                  yk ← f(xk)
                  σk ← σk · et·N(0,1)
                  P ← P U {(xk, yk, σk)}
           fin pour
     fin pour
fin pour
retourner xk ∈ P avec le minimum yk
```

Les variantes modernes de ES incorporent un plus grand nombre de paramètres endogènes et des stratégies d'autoadaptations plus avancées (Wierstra et al., 2014) (Krause et al., 2016) (Beyer & Sendhoff, 2017). Nous prenons une voie alternative, en revenant a une variante classique auto-adaptative (μ, λ) -ES sans le mécanisme de la matrice de covariance qui a montré sa robustesse dans l'optimisation continue mais il est resté enfermer dans des complications mathématiques et limiter face à des problèmes et fonctions multimodales et à large échelle (Piad-Morffis et al., 2015). La plupart des nouvelles recherches ont tendance à hybrider les stratégies d'évolution avec d'autres techniques stochastiques comme outil d'exploitation en optimisation continue (Freitas et al., 2013) (Piad-Morffis et al., 2015) (Coelho et al., 2016) (X. Qu et al., 2019) (Lou et al., 2019), mais seulement quelques algorithmes le combinant avec gradient et direct des techniques de recherche ont été proposées (Giacobini et al., 2003) (Chenet al., 2009) (Kramer, 2010) (Kramer, 2016). Cependant, la principale motivation de cette recherche n'est pas simplement d'améliorer les performances des ES dans les paysages multimodaux, mais de fournir des preuves à l'appui de notre affirmation selon laquelle la règle d'apprentissage log-normale fausse la recherche.

III.3.2 Critère de contraction

Le passage d'une recherche globale à une recherche locale est utile lorsque la capacité d'exploration de la recherche globale n'est plus efficace. Dans ce but, un critère de contraction qui représente une mesure de la diversité de la population est développé (Boukhari et al., 2019). Chaque algorithme de recherche doit traiter de l'exploration et de l'exploitation d'un espace de recherche. L'exploration consiste à visiter de toutes nouvelles régions d'un espace de recherche, tandis que l'exploitation consiste à visiter ces régions d'un espace de recherche situé à proximité de points précédemment visités. Pour réussir, un algorithme de recherche doit établir un bon rapport entre exploration et exploitation. À cet égard, plusieurs critères ont été utilisé dans la littérature pour basculer entre l'intensification et la diversification (Neri et al., 2012). Le passage d'une recherche globale à une recherche locale est utile lorsque la capacité d'exploration de la recherche globale n'est plus efficace. Pour intégrer Nelder Mead à EA avec succès, il faut un critère de contraction pour décider quand la recherche locale doit commencer. Dans EA-NM, nous proposons une nouvelle condition de déclenchement inspirée de la littérature combinant deux critères : le premier coefficient de variation (CV) est une mesure de diversité de la population dans l'espace objectif, elle est égale au ratio de l'évaluation moyenne de la population actuelle X sur sa population. Écart type correspondant :

$$CV = m / \sigma = mean\{J(x), x \epsilon X\} / \sqrt{Var\{J(x), x \epsilon X} \quad \text{(III.2)}$$

Le deuxième critère est la distance euclidienne ED qui est une mesure de la diversité de la population dans un espace de décision défini comme suit :

$$ED = max(\|xi - xj\|), \forall xi, xj \in X \quad \text{(III.3)}$$

Une recherche locale est appelée lorsque le critère développé augmente entre deux générations consécutives de la recherche globale.

III.3.3 Regroupement (clustering)

Afin d'étendre autant que possible la recherche locale dans l'ensemble du domaine, nous avons décidé de ne l'appliquer qu'occasionnellement et sélectivement. Pour ce faire, la population est divisée en un certain nombre de sous-populations, appelées clusters. Une stratégie simple et classique est adoptée pour créer un certain nombre de clusters où chaque cluster est construit de telle sorte que tous les éléments associés soient plus proches de son centre de masse que de tout autre, comme mentionné par (Dumas et al., 2009). Après cette étape préliminaire appelée clustering, la recherche locale est appliquée un certain nombre de fois à la meilleure personne de chaque cluster. Dans cette méthode, le nombre de clusters diminue progressivement au cours du processus d'optimisation. Cela correspond à l'idée naturelle que l'ensemble du processus se concentrera progressivement sur un nombre réduit de minima locaux. Notez que l'effet du nombre initial de clusters a également été examiné sur les fonctions de test. Il semble que le nombre initial de clusters entre 5% et 20% de la population donnera les meilleurs résultats. Au-dessous de 5%, une convergence prématurée peut être observée alors qu'au-dessus de 20%, la vitesse de convergence peut être réduite. La valeur de 10% sera donc adoptée pour toutes les simulations présentées dans l'expérimentation.

III.3.4 Recherche locale (Simplex Nelder-Mead)

La méthode Simplex est un algorithme de recherche local robuste, facile à programmer et rapide, non dérivé. De nombreuses tentatives ont été faites pour hybrider les AE avec la méthode simplex (Chelouah & Siarry, 2003)(Yen et al., 1998). Simplex peut être utilisé comme

méthode de recherche locale après mutation. La recherche Simplex, introduit par (Nelder & Mead, 1965), est une méthode de recherche basée sur la direction déterministe capable d'explorer l'espace objectif très rapidement.

Un simplex est une collection de n + 1 points dans un espace à n dimensions. Dans un problème d'optimisation impliquant n variables, la méthode simplex recherche une solution d'optimisation en évaluant un ensemble de n + 1 points. La méthode forme en permanence de nouveaux simplices en remplaçant le point ayant les plus mauvaises performances dans un simplexe par un nouveau point. Le nouveau point est généré par les opérations de réflexion, d'agrandissement et de contraction.

Dans un espace multidimensionnel, la soustraction de deux vecteurs signifie un nouveau vecteur commençant par un vecteur et se terminant par l'autre, comme $x_2 - x_1$. Nous nous référons souvent à la soustraction de deux vecteurs comme une direction. L'ajout de deux vecteurs peut être mis en œuvre de manière triangulaire, en déplaçant le début d'un vecteur à la fin de l'autre pour former un autre vecteur. L'expression $x_3 + (x_2 - x_1)$ peut être considérée comme la destination d'un point en mouvement qui commence à x_3 et a une longueur et une direction de $x_2 - x_1$.

Pour chaque nouveau simplex, plusieurs points sont attribués en fonction de leurs valeurs objectives. La recherche simplex répète ensuite la réflexion, l'expansion, la contraction et la réduction, de manière très efficace et déterministe. Les sommets du simplex se déplaceront vers le point optimal et le simplex deviendra de plus en plus petit. Les critères d'arrêt peuvent être sélectionnés comme un nombre prédéterminé d'itérations maximales, la longueur du bord ou le taux d'amélioration de *B*.

La recherche simplexe pour la minimisation est montrée dans l'algorithme 8. Les coefficients pour les opérations de réflexion, de dilatation, de contraction et de rétraction sont typiquement choisis comme suit : $\alpha = 1$, $\beta = 2$, $\gamma = -1/2$ et $\delta = 1/2$. Le simplexe initial est important. La recherche peut facilement rester bloquée pour un simplexe trop petit. Ce simplex doit être sélectionné en fonction de la nature du problème.

Algorithme 8 : Méthode du Simplex Nelder-Mead Method (NM)

1 Initialiser les paramètres
2 Randomiser l'ensemble des individus xi
3 Répéter :
4 **a. Trouvez les pires et les meilleurs individus comme x_h et x_r**
5 **Calculez le centroïde de tous les x_i, i ≠ h, comme$\bar{x}$**
6 **b. Entrer le mode de réflexion :**
7 $x_r = \bar{x} + \alpha(\bar{x} - x_h)$
8 **c. Si** $f(x_l) < f(x_r) < f(x_h)$, $x_h \leftarrow x_r$;
9 **Sinon Si** $f(x_r) < f(x_l)$, **entrer le mode expansion :**
10 $x_e = \bar{x} + \beta(\bar{x} - x_h)$;
11 **Si** $f(x_e) < f(x_l)$, $x_h \leftarrow x_e$;
12 **Sinon** $x_h \leftarrow x_r$;
13 **Fin**
14 **Sinon Si** $f(x_r) > f(x_i), \forall i \neq h$, **entrer le mode contraction :**
15 $x_c = \bar{x} + \gamma(\bar{x} - x_h)$;
16 **Si** $f(x_c) < f(x_h), x_h \leftarrow x_c$;
17 **Sinon entrer le mode rétrécissement :**
18 $x_i = x_l + \delta(x_i - x_l), \forall i \neq l$;

```
19     Fin
20     Fin
21 Jusqu'à ce que la condition de terminaison soit satisfaite
```

L'algorithme commence par un simplexe initial et évolue en utilisant quatre transformations géométriques élémentaires : réflexion, expansion, contraction et retrait. Premièrement, la méthode commence par un simplex initial conçu avec les points V_1, V_2 et V_3, comme indiqué dans la Figure III.3

Un point de réflexion est créé par le centre de gravité des autres points jusqu'au point S.

$$S = M + (M - V_1) \quad \text{(III.4)}$$

Supposant $f(V_3) < f(V_2) < f(V_1)$. À ce stade, trois situations peuvent se présenter :

- Si $f(S) < f(V_3)$, une extension est faite au point S_e selon (III.5), où χ est appelé le coefficient d'expansion.

$$S_e = M + \chi(M - V_1) \quad \text{(III.5)}$$

- Si $f(S) > f(V_3)$, une contraction est faite au point S_c or $S_{c'}$ selon (III.6) et (III.7), selon que $f(V_1)$ ou $f(S)$ est inférieur.

$$S_c = M + \rho(M - V_1) \quad \text{(III.6)}$$

$$S_{c'} = M + \rho(M - V_1) \quad \text{(III.7)}$$

Si $f(S_c)$ ou $f(S_{c'})$ est supérieur à $f(V_3)$, alors la contraction a échoué et nous effectuons une opération de retrait. L'opération de retrait réduit la taille du simplex en déplaçant presque tout le meilleur point V_3 à mi-chemin vers V_3.

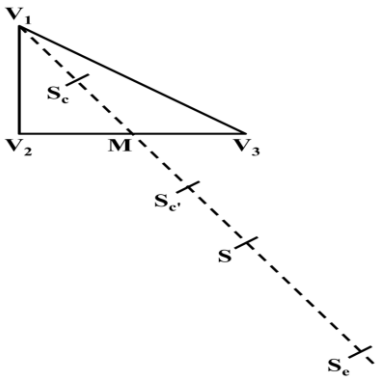

Figure IV-3 Illustration de la méthode Simplex Nelder-Mead (Lepagnot et al, 2013)

III.3.5 Stratégie de réinitialisation

Si la meilleure solution n'a pas été améliorée après la recherche locale, une réinitialisation de l'ensemble de la population est utilisée pour donner aux algorithmes plus d'occasions de trouver l'optimum global. Dans EA-NM, nous appliquons la même stratégie de redémarrage proposée

dans (Peng et al., 2017), où un compteur C garde une trace du nombre de redémarrages, et la population réinitialisée comme décrit dans l'algorithme 9.

Algorithme 9 : Stratégie de réinitialisation

1: std = (U_i− L_i) / D
2: Si Compteur< C_{max}
3: Générer Xi aléatoirement
4: Sinon
5: Pour i = 1 à 2/3 * K faire
6: X_i = randreal (L_i , U_i)
7: Fin pour
8: Pour i = 2/3 * K à K faire :
9: X_i = Gaussien(X_{best}, std)
10: Fin pour
11: Fin Si

L'algorithme 10 montre un aperçu général de la méthode hybride ES-NM. Le schéma d'hybridation a deux phase (série) est proposé ainsi qu'une technique d'adaptation selon les critères de contractions développées (III.2) et (III.3), ce qui représente une indication permettant de passer d'une recherche globale ES à une recherche locale NM. Une stratégie efficace, démontrant l'utilisation de manière rationnelle la recherche locale au sein du solveur évolutionnaire (SE). Le processus évolutionnaire (recherche globale) se déroule normalement jusqu'à ce qu'une zone prometteuse soit détectée (ligne1 à 21). La zone prometteuse est détectée lorsqu'un critère de contraction basé sur une mesure de diversité entre les individus dans l'espace objective et l'espace de décision est concrétisée (ligne22) pour assurer une bonne balance entre l'exploitation et l'exploration. Ensuite, le domaine de recherche est réduit par une méthode de regroupement qui regroupe les individus selon leurs centres de masse pour modérer et appliquer la recherche locale seulement aux individus qui ont le potentiel d'être améliorer l'espace d'objective et alléger le cout d'exécution (ligne23 à 24). Une recherche locale basée sur Nelder et Mead Simplex est lancée. Les mouvements d'exploitation sont lancés à l'aide de la méthode du simplexe autour du meilleur individu trouvé dans le cycle d'exploration (ligne 25). Le critère utilisé pour déclencher une exécution de NM est satisfait si l'algorithme ES montre des signes d'une éventuelle convergence prématurée. L'exploitation est démarrée une fois, après perte de diversité, et le processus évolutionnaire ne peut pas être poursuivi après, à moins qu'une nouvelle population ne se produise, sinon une stratégie de réinitialisation est adoptée (ligne27 à 23) pour surmonter les points locaux et renforcer la diversité.

l'utilisation de NM peut guider les ES vers de nouvelles zones prometteuses de l'espace de recherche, ainsi que vers la stratégie de redémarrage qui offre à l'algorithme une certaine diversification si la méthode de recherche locale ne permet pas d'obtenir une nouvelle région attirante.

Algorithme 10 : Pseudo code d'un adaptative hybride EA-NM

1: P ← { }
2 : conteur = 0
3: pour λ fois faire
4: x_k ← point de départ aléatoire
5: y_k ← f(x_k)
6: σ_k ← variance de départ aléatoire
7: P ← P U {(x_k, y_k, σ_k)}
8: fin pour
9: pour toutes générations faire

```
Q ← meilleurs μ solutions dans P
P ← { }
 pour tout parent Qk faire
        pour λ fois faire
              (xk, yk, σk) ← Qi
               xk ← xk + N(1, σi)n
               yk ← f(xk)
               σk ← σk · eτ·N(0,1)
               P ← P U {(xk, yk, σk)}
        fin pour
 fin pour
fin pour
: Si le critère d'arrêt basé sur CV et ED rencontré Alors
:  M ← regroupement P
:  Xstart ← meilleurs m solutions in M
:  Xbest ← RECHERCHELOCALE Nelder-Mead(Xstart)
Fin Si
Si f(Xbest)< f(Xstart) alors
   conteur = conteur + 1
   tant que (conteur > Cmax)
         stratégie de réinitialisation
         reinistialiser de P comme il est décrit en algorithme 9
         conteur = 0
   Fin tant que
Fin si
retourner xk ∈ P avec le minimum yk
```

III.4 CONFIGURATION ET RÉSULTATS EXPÉRIMENTAUX

Dans cette section, la performance du EA-NM proposé a été testée avec un ensemble de fonctions de référence (Benchmarks) sans contrainte, comme le montre le tableau III.1. Toutes les fonctions de référence utilisées dans ce chapitre sont des problèmes de minimisation. Ces benchmarks sont choisis en fonction de leur popularité et leurs caractéristiques. Celles-ci impliquent certaines caractéristiques telles qu'une large gamme de bornes de solutions, des régions de solutions réalisables larges ou étroites, des variables de haute dimensionnalité et un extérieur multimodal. Ainsi, les fonctions de test de référence de différents espaces de recherche topologiques variable peuvent représenter certains des aspects clés des problèmes du monde réel utiles à l'évaluation des techniques d'optimisation. La performance est ensuite validée par comparaison avec d'autres méthodes utilisant les mêmes problèmes de test. De cette façon, les forces et les faiblesses d'un algorithme peuvent être identifiées afin que d'autres améliorations puissent être apportées.

Afin d'évaluer les performances et l'efficacité de l'algorithme hybride proposé EA-NM, un ensemble de 15 fonctions continues présentant une gamme variée de propriétés est adopté et implémenté. Cette collection de benchmarks est suffisamment vaste pour inclure divers types de fonctions telles que les fonctions unimodales, multimodales, séparables, non séparables, régulières, irrégulières et multidimensionnelles. Il est à noter que la combinaison de ces caractéristiques détermine la complexité de ces benchmarks. Ces fonctions de test affichent des espaces de recherche topologique différentes. Ainsi, elles peuvent représenter certains des aspects clés des problèmes du monde réel qui sont utiles dans l'évaluation des techniques d'optimisation. La plage (l'espace de recherche ou l'intervalle) initiale, l'équation complète, les

dimensions, la théorie, les solutions globales optimales théoriques et les propriétés de ces fonctions sont présentées dans le tableau III.1, tel que rapporté dans (Boussaïd et al., 2011).

Une fonction est considérée comme unimodale si elle n'a qu'un seul optimum. Il peut facilement localiser le point ; ils sont donc utiles pour évaluer la capacité d'exploitation d'un algorithme d'optimisation. Si une fonction benchmark a deux ou plusieurs optima locaux, cette fonction serait appelée multimodale. Les fonctions multimodales conviennent à l'analyse comparative de la capacité d'exploration globale d'un algorithme d'optimisation. Si l'opération d'exploration d'une méthode est médiocre au point qu'elle ne peut effectuer une recherche efficace dans tout l'espace, cette méthode est piégée dans des minima locaux. Les fonctions multimodales sont difficiles pour les algorithmes ainsi que les benchmarks qui ont des surfaces planes, parce que la planéité du benchmark ne fournit à l'algorithme aucun type d'information permettant de diriger le processus de recherche vers les minima. Une fonction est considérée comme séparable si elle peut être exprimée comme une somme de fonction provenant d'une seule variable. Les benchmarks non séparables ne peuvent pas être exprimés sous cette forme. Par conséquent, les benchmarks non séparables sont beaucoup plus difficiles par rapport aux benchmarks séparables. Une fonction régulière est différentiable (analytique) à chaque point de son domaine (Simon, 2008). Par ailleurs, la dimensionnalité de la zone de recherche reflète le nombre de paramètres à optimiser. C'est une caractéristique essentielle qui détermine la difficulté du problème (Karaboga & Akay, 2009b).

Tableau IV-1 Fonctions benchmarks de test utilisé pour l'analyse des performances (Karaboga & Akay, 2009b)

Benchmarks	Function	Plage	Min	Propriété
F1 (Sphere)	$\sum_{i=1}^{n} x_i^2$	[-5.12,+5.12]	0.00	Unimodal/ Séparable
F2 (Rastrigin)	$100 + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$	[-5.12,+5.12]	0.00	Multimodal/ Séparable
F3 (Rosenbrock)	$\sum_{i=1}^{n} \left[100\left(x_i^2 - x_{i+1}\right)^2 + \left(1 - x_i\right)^2\right]$	[-30,30]	0.00	Unimodal/ Non séparable
F4 (Ackley)	$-20e^{-0.2\sqrt{\frac{\sum_{i=1}^{n} X_i^2}{10}}} - e^{\frac{\sum_{i=1}^{n}\cos(2\pi X_i)}{10}} + 20 + e$	[-32,+32]	0.00	Multimodal/ Non séparable
F5 (Griewank)	$\frac{1}{4000}\sum_{i=0}^{n}(x_i - 100)^2 - \prod_{i=0}^{n}\cos\left(\frac{x_i - 100}{\sqrt{i+1}}\right)$	[-600,+600]	0.00	Multimodal/ Non séparable
F6 (Salomon)	$1 - \cos\left(2\pi\sqrt{\sum x_i^2}\right) + 0.1\sqrt{\sum x_i^2}$	[-100,+100]	0.00	Multimodal/ Non séparable
F7 (Powel)	$\sum_{i=1}^{n}\begin{bmatrix}\left(x_{ni-3} + 10x_{4i-2}\right)^2 + 5\left(x_{ni-1} - x_{ni}\right)^2 \\ + \left(x_{ni-2} - 2x_{ni-1}\right)^4 + 10\left(x_{ni-3} - x_{ni}\right)^4\end{bmatrix}$	[-4,+5]	0.00	Unimodal/ Séparable
F8 (Zakharov)	$\sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^4$	[-5,+10]	0.00	Unimodal/ Non séparable
F9(Rotated Hyper-Ellipsoid)	$\sum_{i=1}^{n}\sum_{j=1}^{i} x_j^2$	[-65.536,+ 65.536]	0.00	Unimodal/ Séparable
F10 (Schweffel)	$\sum_{i=1}^{n}\left(-x_1\sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	[-100,+100]	0.00	Multimodal/ Séparable

Benchmarks	Function	Plage	Min	Propriété
F11 (Sumsquares)	$\sum_{i=1}^{n} ix_i^2$	[-10, 10]	0.00	Unimodal/ Séparable
F12 (Quartic)	$\sum_{i=1}^{n} ix_i^4 + random(0,1)$	[-1.28, 1.28]	0.00	Unimodal/ Séparable
F13 (Brown)	$\sum_{i=1}^{n-1} (x_i^2)^{(x_{i+1}^2+1)} + (x_{i+1}^2)^{(x_i^2+1)}$	[−1, 4]	0.00	Unimodal/ Non séparable
F14 (Xin-She Yang)	$\sum_{i=1}^{n} \epsilon_i \lvert x_i \rvert^i$,where ϵ is random number in [0,1]	[-5, +5]	0.00	Multimodal/ Séparable
F15 (Salomon)	$1 - \cos\left(2\pi \sqrt{\sum_{i=1}^{n} x_i^2}\right) + 0.1 \sqrt{\sum_{i=1}^{n} x_i^2}$	[-100,+100]	0.00	Multimodal/ Non séparable

III.4.1 Réglage des paramètres

Trouver le paramétrage approprié est l'un des enjeux majeurs de l'application des algorithmes métaheuristiques. En général, le choix d'une bonne valeur de paramètre est important pour une bonne performance. Si les réglages des paramètres ne sont pas optimaux, les résultats ne le sont pas optimaux non plus. Afin d'avoir une évaluation équitable entre toutes les méthodes, il est essentiel de définir la valeur de chaque méthode à sa valeur optimale. Un réglage optimal des paramètres implique que le meilleur réglage est appliqué afin d'obtenir le meilleur résultat possible.

L'une des questions concernant l'algorithme proposé est de savoir quelles valeurs doivent être définies pour le paramètre le nombre de cluster M. Il semble qu'un nombre initial de clusters entre 5% et 20% de la population donnera les meilleurs résultats comme mentionné dans (Dumas, et al., 2009). La valeur de 15% sera donc retenue pour toutes les simulations présentées. Les valeurs des paramètres sont sélectionnées soit sur la base des réglages courants dans la littérature, soit déterminées par nos expériences numériques préliminaires.

Deux expériences ont été menées sur des problèmes de test bien connus. La première expérience est la comparaison entre SA-ES (auto-adaptative), ES-NM (sr) hybride (sans réinitialisation) et ES-NM (ar) hybride (avec réinitialisation) pour voir la performance de l'approche proposé en termes de vitesse d'exécution et qualité de solution et au même temps pour évaluer l'influence de stratégie de réinitialisation de notre l'hybridation. Le tableau III.3 nous indique, pour chaque algorithme, la moyenne, la meilleure et la pire valeur sont affichées après 5000 évaluations sur 20 exécutions pour chacune des versions avec D = 20. Les valeurs en gras représentent les meilleurs résultats obtenus. Nous avons également vérifié si les différences entre les solutions trouvées par notre approche ES-NM (ar) et les algorithmes ES-NM (sr) et SA-ES standard (tableau III.4) étaient statistiquement significatives à travers le test statistique Wilcoxon rank-sum test.

La deuxième expérience résume les résultats dans le tableau III.5, le tableau III.7 et le tableau III.9 qui montrent la comparaison de notre approche hybride avec cinq d'autres algorithmes bien connus de la littérature : SA-ES (Meyer-Nieberg & Beyer, 2007), PSO (S. Chen, 2012), DE (Jeyakumar & Shanmugavelayutham, 2011), ICA (Hosseini & Al Khaled, 2014) et GA (Picek et al., 2013) exécutés 20 fois indépendamment dans 10, 30 et 50 dimensions respectivement. Le même critère de terminaison est appliqué à tous les algorithmes, ce qui met fin à la recherche lorsque tous les algorithmes atteignent le même nombre d'itérations. Les valeurs minimale (meilleure), pire et moyenne (moyenne) de chaque algorithme sont imprimées en gras.

Pour décider de la signification des résultats, un test statistique non paramétrique appelé test de Wilcoxon rank-sum à 5% de niveau de signification est également effectué. Ce test est utilisé pour vérifier si une paire de deux solutions est statistiquement différente ou non. Une telle comparaison par paire est constituée par ES-NM vs SA-ES, ES-NM vs PSO, ES-NM vs DE, ES-NM vs ICA, et ES-NM vs GA. Les valeurs de p obtenues à partir de ce test statistique sont rapportées dans les tableaux III.6, III.8 et III.10. Selon (García et al., 2009), ces valeurs de p inférieures à 0,05 peuvent être considérées comme des preuves solides contre l'hypothèse nulle (p > 0,05 rejette l'hypothèse nulle). Notez que les valeurs p supérieures à 0,05 sont soulignées. N/A indique « non applicable », ce qui signifie que l'algorithme proposé ne pouvait être comparé à l'autre algorithme dans le test rank-sum.

De plus, les valeurs spécifiques utilisées sont données dans le tableau III.2. Nous avons exécuté ES-NM, SA-ES, PSO, DE, ICA et GA 20 fois indépendamment avec un maximum de 10000 itérations par cycle pour vérifier les performances de ces algorithmes. Les dimensions des problèmes de référence ont été définies sur 10, 30 et 50, et tous les paramètres associés à l'optimiseur restent les mêmes pendant les tests.

Les expériences ont été menées dans MATLAB R2013a, dont l'environnement d'exécution est Intel® Core ™ i3 2,53 GHz avec une capacité de mémoire de 6 Go.

Tableau IV-2 Valeurs des paramètres utilisées dans cette comparaison (Boukhari et al., 2019)

Algorithme	Paramètres	Nom des Parametres	Valeur Numérique
Evolutionary strategy (SA-ES)	μ	Parents	15
	λ	Descendants	100
	σ	Mutation initiale	N*0.5
	τ	Taux d'apprentissage	$1/\sqrt{N}$
Nelder-Mead (NM)	α	Coefficient de reflection	1
	β	Coefficient de l'expansion	2
	γ	Coefficient de contraction	0.5
	δ	Coefficient de shrinking	0.5
Particle swarm optimization (PSO)	nPar	Particules	15
	C1	Coefficient personnel	2
	C2	Coefficient social	2
	w_{max}	Masse d'inertie max	0.9
	w_{min}	Masse d'inertie min	0.4
Differential evolution (DE)	nPop	Individus	15
	Pcr	Probabilité de croisement	0.2
	F	Facteur d'échelle	[0.2, 0.8]
Imperalist competitive algorithm (ICA)	N	Nombre de pays	15
	β	Coefficient d'assimilation	2
	γ	Coefficient d'angle d'assimilation	π/4
Genetic Algorithm (GA)	nPop	Individus	15
	Pcr	Probabilité de croisement	0.2
	Pm	Probabilité de mutation	0.6

III.4.2 Résultats expérimentaux et discussion

Analyse de l'expérience 01 :

Pour les fonctions unimodales comportant qu'un minimum global, on peut voir (Tableau III.3) que l'approche hybride obtient de meilleurs résultats pour 05 sur les 08 fonctions. ES-NM présente des résultats qui sont bien meilleurs que ceux de SA-ES classique pour les fonctions Sphere (F1), Salomon (F6), Powel (F7), Sumsquares (F11) et Brown (F13). Cependant que SA-

ES montre malgré tous des performances améliorées par rapport à ES-NM et réalise de meilleurs résultats pour 03 fonctions Rosenbrock (F3), Zakharov (F8) et Quartic (F12). En termes de valeurs moyennes de la fonction objectif, ES-NM (sr) obtient un résultat légèrement supérieur pour les fonctions powel (F7) et Brown (F13) par rapport au ES-NM(ar) et SA-ES. Quant aux fonctions multimodales comportant plusieurs minima locaux, nous pouvons voir que notre algorithme obtient de meilleurs résultats que SA-ES classique pour toutes les fonctions. Ces résultats valident davantage la supériorité globale de l'algorithme ES-NM. Pour toutes les fonctions, les résultats obtenus montrent encore une fois la supériorité de notre approche hybride avec réinitialisation ES-NM (ar) et occupe la première place et se montre particulièrement performant pour les fonctions multimodales. Cependant ES-NM sans-réinitialisation (sr) obtient les meilleurs résultats pour 09 fonctions et SA-ES pour 06 fonctions. Ce qui affirme l'effet positive de la réinitialisation sur l'algorithme hybride.

La valeur calculée de la statistique p-valeur (Tableau III.4) est inférieure à la valeur critique (0.05) pour 11 fonctions de test sur un total de 15 fonctions comparant ES-NM (ar) vs. SA-ES, et 14 fonctions sur 15 en comparant ES-NM (ar) vs ES-NM (sr). Nous devons donc accepter l'hypothèse nulle pour ces fonctions et nous ne pouvons donc prétendre qu'il existe une différence statistique significative entre ES-NM (ar) et les deux les algorithmes.

L'algorithme ES-NM (ar) avec le mécanisme de réinitialisation est statistiquement différent et plus performant que la même version hybride sans réinitialisation ES-NM (sr). Ces résultats confirment une fois de plus que la réinitialisation a un effet significatif d'un point de vue statistique sur notre schéma hybride.

Pour conclure, D'après les tableaux de comparaison (III.3, III.4), l'hybridation de l'algorithme de stratégie d'évolution (ES) avec la méthode Nelder-Mead (NM) améliore considérablement les performances de l'algorithme pour toutes les fonctions de test. L'ES-NM a obtenu de meilleures solutions par rapport à l'ES pur sur différents benchmarks à cause du bon équilibre entre la recherche globale (exploration) et la recherche locale (exploitation). Malgré la complexité des fonctions de test, l'algorithme proposé a prouvé son adaptabilité et sa capacité à dépasser les difficultés de convergence piégées par l'ES classique.

Tableau IV-3 Résultats de comparaison obtenus par SA-ES, ES-NM (ar), ES-NM (sr) en 20 D(Boukhari et al., 2019)

Functions		ES-NM (ar)	ES-NM (sr)	SA-ES
F1	Best	**3.97**	6.49	5.54
	Worst	**14.06**	17.26	15.49
	Avg	**9.82**	13.77	8.56
F2	Best	**12.69**	14.98	18,51
	Worst	**33.87**	35.92	64.94
	Avg	**17.43**	21.26	36.31
F3	Best	22.62	25.57	**19.25**
	Worst	36.04	35.76	**33.31**
	Avg	27.40	29.14	**23.38**
F4	Best	**8 .31**	13.55	15.41
	Worst	**16 .95**	18.93	21.64
	Avg	**11.17**	16.53	30.47
F5	Best	13.58	**12.78**	15.48
	Worst	**18.06**	23.92	27.63
	Avg	**14.89**	17.73	21.57
F6	Best	38.22	**37.62**	55.24
	Worst	**43.15**	45.19	66.87
	Avg	**40.75**	42.49	59.83
F7	Best	74.96	75.09	**70.60**
	Worst	**91.85**	99.24	95.19

Functions		ES-NM (ar)	ES-NM (sr)	SA-ES
	Avg	84.71	**80.97**	82.37
	Best	27.12	31.84	**26.33**
F8	Worst	49.55	53.32	**40.22**
	Avg	43.73	42.90	**33.92**
	Best	9.67	11.94	**5.81**
F9	Worst	19.47	23.83	**16.38**
	Avg	**12.28**	16.85	12.44
	Best	**11.57**	13.88	26.08
F10	Worst	**18.43**	25.59	32.77
	Avg	**13.78**	17.36	28.14
	Best	**24.43**	31.63	29.48
F11	Worst	**37.01**	43.37	41.18
	Avg	**26.26**	40.67	38.11
	Best	18.83	19.66	**16.48**
F12	Worst	**26.18**	35.26	29.41
	Avg	22.91	26.56	**20.21**
	Best	10.63	8.09	**7.84**
F13	Worst	19.18	**11.51**	14.94
	Avg	13.91	**9.11**	9.37
	Best	**28.67**	37.13	31.40
F14	Worst	**46.55**	48.76	65.16
	Avg	**43.66**	46.09	59.08
	Best	**6.89**	13.12	18.14
F15	Worst	**17.93**	22.46	46.54
	Avg	**11.79**	19.80	23.61

Tableau IV-4 Comparaison de Wilcoxon-Test pour SA-ES, ES-NM (ar) et ES-NM (sr). Le tableau montre les valeurs (P-valeur) résultantes pour chaque comparaison par paire en 20D(Boukhari et al., 2019)

Fonction	ES-NM (ar) vs SA-ES	ES-NM (ar) vs ES-NM (sr)	ES-NM(sr) vs SA-ES
F1	6.48e-04	2.73e-05	0.08816
F2	0.00337	1.46e-04	3.74e-04
F3	0.06451	5.84e-04	0.56147
F4	0.00836	0.00361	4.26e-05
F5	2.76e-06	4.26e-06	8.41e-04
F6	1.64e-05	8.41e-06	6.68e-05
F7	0.07392	0.05714	0.09484
F8	0.05176	0.00937	0.06488
F9	0.00293	0.03668	0.64012
F10	6.47e-05	6.48e-04	3.04e-06
F11	8.84e-08	0.00351	0.00663
F12	0.07298	0.06293	0.07205
F13	0.00017	6.72e-04	0.08814
F14	3.94e-09	0.61337	6.28e-06
F15	1.78e-06	6.28e-04	1.56 e-05

Analyse de l'expérience 02 :

La comparaison de ces six algorithmes (μ, λ) ES-NM, SA-ES, PSO, DE, ICA et AG les uns avec les autres montre que (μ, λ) ES-NM est supérieur et le plus robuste que les autres. Notre algorithme hybride montre de bonnes performances sur la majorité des problèmes de test comme décrit dans les tableaux de comparaisons. Cependant, la raison de l'infériorité de l'approche hybride sur certains problèmes de test pourrait être à cause à la nature compliquée de ces fonctions et du théorème de No Free Lunch (Joyce & Herrmann, 2018) qui exclut toute

affirmation concernant la supériorité d'une méthode par rapport à une autre. De plus, l'analyse du comportement de l'algorithme proposé en fonction de la nature des Benchmarks est décrite ci-dessous :

Analyse des résultats dans les fonctions unimodales : huit (08) fonctions complexes unimodales bien connues ont utilisé F1, F3, F7, F8, F9, F11, F12 et F13. Les preuves obtenues à partir des résultats de calcul fournissent un solveur d'optimisation avec une bonne fiabilité et une efficacité de qualité pour résoudre les fonctions unimodales. Cependant, sur F3, F7et F9, l'algorithme a du mal à trouver l'optimum, et souvent les meilleures solutions sont restées à des points qui ne sont même pas des optima locaux en raison de la non-séparation et certaines de ces fonctions inhérentes à un certain type de tromperie sont difficiles à optimiser. La tromperie pourrait être que le minimum global soit situé sur une surface plate ou situé très près d'un minimum global. Parfois, la tromperie est due au fait que le minimum global se situe dans des vallées courbes étroites ou montre des propriétés fractales autour du minimum global de En termes de performance globale ES-NM améliore la convergence du ES orignal et nettement meilleur que les autres sur la majorité des fonctions unimodales.

Analyse des résultats dans les fonctions multimodales : Sept (07) autres fonctions de référence complexes et multimodales ont utilisé F2, F4, F5, F6, F10, F14 et F15. Il ressort clairement des tableaux de comparaison que l'approche proposée est compétente pour fournir des solutions efficaces et sauter de nombreux minimums locaux et converger vers un minimum global dans de nombreux cas, comme dans les fonctions F2, F4, F5 en 10D, F5 et F10 en 30D, et F10 en 50D avec un taux de réussite globale grâce au forte capacité d'exploration et de l'efficacité du processus de réinitialisation de surmonter les minimums locaux, Malgré un processus de recherche local NM considérablement intensifié, ES-NM échappe à une convergence prématurée et ne risque pas d'être piégé de manière permanente dans des minima locaux de précision. Cependant, il reste bloqué dans le minimum local pour certaines autres fonctions de test en raison de la complexité de ces fonctions, même si le mécanisme de redémarrage est déclenché. La possibilité de convergence pour l'algorithme ES-NM est également aidée par l'approche de clustering qui affine la recherche au fur et à mesure que la procédure d'optimisation locale progresse afin de réduire la recherche locale dans la région de pic local.

Analyse des résultats dans l'espace de recherche multidimensionnelle : à partir de différentes comparaisons de dimensions, les résultats montrent clairement que l'ES-NM est plus performante et présente une bonne stabilité et une bonne adaptation dans de nombreux tests de calcul tels que F1, F2, F7, F9 et F10. La principale observation des résultats des expériences de différentes dimensions 10, 30 et 50 est que les performances de l'algorithme proposé ne se dégradent pas de manière significative lorsque le nombre de dimensions augmente, ce qui démontre son adaptabilité et sa stabilité grâce au mécanisme en trois étapes et à l'exploration-exploitation efficace équilibre.

L'expérience informatique acquise sur les fonctions de test précédentes et les résultats statistiques confirment à nouveau le riche potentiel de l'approche proposée pour être un solveur efficace, efficace et polyvalent pour les problèmes d'optimisation sans contrainte.

Tableau IV-5 Résultats de comparaison obtenus par ES-NM, SA-ES, PSO, DE, ICA, GA en 10 D(Boukhari et al., 2019)

Function		ES-NM	ES	PSO	DE	ICA	GA
F1	Best	**0**	6.6874e-129	1.4273e-257	**0**	**0**	**0**
	Worst	**0**	3.7487e-128	4.0507e-257	**0**	**0**	1.4516e-226
	Avg	**0**	1.4516e-128	2.4659e-257	**0**	**0**	1.2961e-251
F2	Best	**0**	1.9899	**0**	**0**	**0**	**0**
	Worst	**0**	13.9294	10.9445	**0**	4.7571	12.9406

Function		ES-NM	ES	PSO	DE	ICA	GA
	Avg	**0**	6.2682	3.9301	**0**	1.2368	0.00561
	Best	**1.0455e-15**	0.014557	1.9595e-07	0.0024643	6.1554e-08	0.0393-04
F3	Worst	**1.0455e-05**	15.7736	3.9866	4.8524	90002.2439	9.676297
	Avg	**3.6719e-07**	0.98241	0.19933	1.1672	4808.6223	0.1249
	Best	**0**	3.5527e-15	**0**	3.5527e-15	3.5527e-19	**0**
F4	Worst	**0**	18.9339	1.6462	3.5527e-15	1.1551	2.0629
	Avg	**0**	16.513	0.3711	3.5527e-15	0.11551	0.09305
	Best	**0**	**0**	**0**	**0**	0.032006	**0**
F5	Worst	**0**	0.039382	**4.6501e-15**	**0**	0.39615	22.7439
	Avg	**0**	0.017733	**1.5788e-129**	**0**	0.10259	1.9595e-07
	Best	**0**	7.8662	0.15324	0.099873	**0**	0.099873
F6	Worst	**3.0612e-09**	15.6019	0.70087	0.099873	0.24487	4.05523
	Avg	**0.000375**	12.7649	0.31483	0.099873	**0.003751**	0.11054
	Best	**5.7496e-11**	1.7091e-07	7.9008e-06	9.8381e-11	1.7129e-09	2.9728e-06
F7	Worst	2.2985e-08	1.1024e-06	6.5019e-05	0.0025556	**1.2324e-08**	3.8903e-05
	Avg	**2.1767e-09**	6.1906e-07	1.9376e-05	0.00039841	4.6635e-09	1.6672e-05
	Best	**0**	1.7091e-07	1.1542e-233	**0**	3.943e-241	1.8341e-83
F8	Worst	**0**	1.1024e-06	3.1260e-233	**0**	64.4933	1.9928e-05
	Avg	**0**	6.1906e-07	2.5933e-233	**0**	9.0566	1.6702e-23
	Best	**0**	1.9348e-254	**0**	**0**	**0**	**0**
F9	Worst	**0**	1.1835e-253	**0**	8.265e-304	**0**	**0**
	Avg	**0**	6.2085e-254	**0**	4.1325e-305	**0**	**0**
	Best	**0**	**0**	**0**	7.4452e-19	**0**	**0**
F10	Worst	**0**	**0**	**0**	0.014347	8.3816e-31	13.3556e-23
	Avg	**0**	**0**	**0**	0.00082334	2.5761e-31	9.9061e-23
	Best	**0**	**0**	**0**	**0**	**0**	**0**
F11	Worst	**0**	3.8561e-24	2.1184e-58	3.7052e-116	**0**	3.9036e-110
	Avg	**0**	4.6219e-28	8.7601e-61	1.6530e-121	**0**	7.6582e-119
	Best	**0**	4.9603e-76	**0**	**0**	2.3114e-132	**0**
F12	Worst	**4.6591e-185**	8.5239e-56	1.6524e-171	3.7612e-97	5.7651e-35	0.3664e-107
	Avg	3.7016e-191	2.6512e-67	**5.684e-201**	2.6545e-112	2.3114e-126	1.2905e-136
	Best	**0**	5.6309e-189	**0**	1.7627e-219	**0**	**0**
F13	Worst	2.1986e-181	2.9202e-151	2.4548e-141	6.9022e-101	2.6582e-183	**3.5016e-187**
	Avg	1.8993e-189	0.3291e-157	1.9251e-151	4.7481e-127	3.6737e-185	**2.2518e-193**
	Best	**0**	2.7619e-38	**0**	1.6104e-132	**0**	4.9072e-117
F14	Worst	**6.6538e-95**	4.7036e-45	1.5109e-36	6.9057e-16	7.9001e-87	7.7528e-19
	Avg	**3.6591e-116**	2.6016e-49	6.0447e-38	3.8429e-23	8.3174e-103	3.6501e-34
	Best	**0**	4.3194e-12	7.6408e-76	4.7619e-118	**0**	**0**
F15	Worst	5.5916e-83	3.3176e-02	4.6549e-54	1.9951e-52	6.7116e-56	**1.6295e-89**
	Avg	**2.7761e-97**	9.3831e-04	2.5629e-61	2.7403e-59	1.9838e-79	4.5877e-93

Tableau IV-6 Comparaison de Wilcoxon-Test pour ES-NM, SA-ES, PSO, DE, ICA et GA. Le tableau montre les valeurs (P-valeur) résultantes pour chaque comparaison par paire en 10 D(Boukhari et al., 2019)

Fonction	ES-NM vs ES	ES-NM vs PSO	ES-NM vs DE	ES-NM vs ICA	ES-NM vs GA
F1	4.66e-12	1.02e-11	N/A	N/A	2.62e-11
F2	2.34e-14	4.73e-10	N/A	0.0628	4.96e-13
F3	2.07e-10	3.74e-09	5.53e-07	0.00781	1.21e-09
F4	3.02e-10	0.0561	1.21e-12	8.99e-11	3.02e-11
F5	1.69e-07	4.26e-08	2.39e-07	0.0638	3.76e-11
F6	6.25e-10	8.41e-11	3.06e-09	3.64e-04	3.63e-09
F7	2.74e-09	6.68e09	2.29e-08	0.0425	0.0629
F8	2.95e-13	4.84e-11	N/A	3.94e-13	4.72e-10
F9	6.67e-10	N/A	4.81e-08	N/A	N/A
F10	N/A	N/A	2.76e-13	5.74e-12	8.52e-11
F11	3.56e-11	3.04e-11	8.27e-12	N/A	2.52e-09

Fonction	ES-NM vs ES	ES-NM vs PSO	ES-NM vs DE	ES-NM vs ICA	ES-NM vs GA
F12	5.44e-13	0.0629	7.53e-09	6.04e-12	4.43e-12
F13	2.17e-10	6.72e-05	6.33e-07	3.14e-06	0.0877
F14	7.02e-09	0.0061	1.41e-06	4.39e-08	3.18e-10
F15	1.79e-07	6.28e-08	4.69e-05	2.35e-05	3.34e-11

Tableau IV-7 Résultats de comparaison obtenus par ES-NM, SA-ES, PSO, DE, ICA, GA en 30 D(Boukhari et al., 2019)

Functions		ES-NM	ES	PSO	DE	ICA	GA
F1	Best	**1.0512e-271**	2.8923e-127	1.3888e-171	4.4664e-116	3.5006e-138	1.2208e-235
	Worst	**4.3118e-269**	9.2766e-127	2.9382e-129	1.0815e-98	4.8552e-104	1.5442e-193
	Avg	**3.8642e-259**	5.5317e-127	1.4691e-130	5.4101e-100	1.9456e-123	1.2961e-211
F2	Best	**0**	28.8538	8.5628	**0**	**0**	3.3295
	Worst	**3.8014e-04**	73.6268	24.7175	6.9647	32.7501	12.7763
	Avg	**1.6711e-05**	56.1156	12.5551	2.3382	6.3701	5.9145
F3	Best	7.2169-02	15.3802	0.30151	**6.8261e-06**	5.7491e-05	2.9622e-05
	Worst	**2.1664**	589.4624	94.1068	90.2516	62.2067	32.4614
	Avg	**0.0179**	65.9169	44.1565	13.9315	11.7561	2.1565
F4	Best	7.1054e-16	17.554	3.5527e-16	1.3404	**5.9801e-16**	7.5661e-05
	Worst	4.7794e-15	19.2309	**5.1054e-15**	8.0857	3.3905	3.103745
	Avg	**2.3907e-15**	18.7226	3.9278e-15	4.2614	0.006502	0.003745
F5	Best	**0**	**0**	**0**	**0**	**0**	**0**
	Worst	**0**	0.034467	0.19904	0.007396	0.18701	0.0004798
	Avg	**0**	0.0059122	0.054436	0.0003698	6.9278e-11	2.4652e-101
F6	Best	**0.0009113**	20.8999	0.19905	0.005998	0.007305	6.19002
	Worst	0.05667	29.5999	0.29987	2.9996	**0.17232**	23.1097
	Avg	**0.00467**	25.7799	0.21488	1.3999	0.10694	9.3564
F7	Best	5.6719e-06	0.00010234	9.5761e-05	9.4248e-07	3.6398e-07	**1.5706e-08**
	Worst	0.000235	0.00016768	0.00055752	0.013209	**1.1633e-04**	0.00918661
	Avg	**2.7611e-05**	0.00013884	0.00024545	0.004443	8.9633e-05	2.57655e-04
F8	Best	**3.6392e-64**	4.1325e-08	2.5512e-07	1.5643e-20	2.6205e-18	1.9167e-22
	Worst	**3.1033e-43**	0.00038613	7.4057e-05	36.0239	2.9622e-05	7.8224e-12
	Avg	**6.7743e-53**	4.9825e-05	1.0537e-05	3.7601e-04	3.8501e-04	5.7003e-19
F9	Best	**9.1789e-285**	7.9597e-81	1.8011e-168	1.4122e-110	1.8935e-152	6.3647e-198
	Worst	**3.1874e-235**	3.0054e-73	9.924e-111	5.5132e-69	12.8692	2.3114e-121
	Avg	**6.0791e-257**	2.5463e-74	4.962e-112	2.7566e-70	8.9869	5.4126e-156
F10	Best	**0**	5.1069e-57	2.166e-13	**0**	**0**	6.0951e-59
	Worst	**0**	9.4652e-31	2.0701	7.7334e-24	6.5095e-46	1.2318e-12
	Avg	**0**	4.5103e-32	0.15985	3.9253e-25	2.5374e-49	1.6299e-17
F11	Best	**0**	**0**	**0**	**0**	**0**	**0**
	Worst	**0**	0.0084098	2.7633e-27	1.3884e-33	**0**	3.5491e-31
	Avg	**0**	1.9674e-06	6.6791e-29	1.5007e-39	**0**	2.9081e-37
F12	Best	**0**	1.9057e-22	**0**	4.9451e-156	2.7601e-101	**0**
	Worst	**1.1504e-166**	7.9034e-06	2.2009e-121	0.00934	3.5122e-35	1.6772e-77
	Avg	2.0549e-171	3.8904e-07	**2.1091e-197**	2.1091e-95	2.6441e-37	4.7603e-81
F13	Best	**0**	5.6309e-189	**0**	1.7627e-219	**0**	1.7301e-122
	Worst	**2.1986e-161**	2.9202e-151	2.4548e-141	6.9022e-101	3.8421e-149	4.3044e-54
	Avg	**1.8993e-165**	0.3291e-157	1.9251e-151	4.7481e-127	2.9033e-150	2.3192e-54
F14	Best	**2.9627e-194**	5.8576e-08	2.2833e-127	1.7842e-72	1.7842e-162	3.9855e-97
	Worst	8.8657e-58	0.0009476	0.002104	4.5661e-23	**7.9001e-63**	3.3834e-11
	Avg	3.6519e-63	5.9111e-05	1.0537e-23	1.1052e-34	**1.7662e-82**	7.4511e-19
F15	Best	**0**	4.2977e-04	1.1966e-41	1.4991e-11	**0**	2.9803e-39
	Worst	3.1003e-21	2.0154166	**9.8702e-24**	79.902602	8.8745e-07	1.6769e-11
	Avg	**1.9422e-28**	0.0119845	8.8242e-25	17.17355	3.9022e-09	2.9007e-19

Tableau IV-8 Comparaison de Wilcoxon-Test pour ES-NM, SA-ES, PSO, DE, ICA et GA. Le tableau montre les valeurs (P-valeur) résultantes pour chaque comparaison par paire en 30 D(Boukhari et al., 2019)

P-valeur	ES-NM vs ES	ES-NM vs PSO	ES-NM vs DE	ES-NM vs ICA	ES-NM vs GA
F1	4.66e-12	1.02e-11	4.97e-12	3.37e-12	2.62e-11
F2	2.34e-14	4.73e-10	8.83e-09	7.64e-13	4.96e-13
F3	5.02e-11	9.56e-12	2.19e-10	6.59e-12	3.63e-10
F4	4.69e-10	1.73e-05	7.58e-09	2.93e-07	5.94e-03
F5	5.98e-13	3.94e-08	2.73e-12	5.68e-09	7.31e-13
F6	6.94e-12	4.07e-08	1.64e-05	2.95e-15	4.69e-09
F7	4.85e-10	6.01e-13	3.83e-08	0.0739	2.84e-15
F8	2.95e-13	4.84e-11	8.21e-13	3.94e-13	9.37e-11
F9	6.67e-10	1.74e-09	7.39e-12	5.83e-15	3.82e-09
F10	1.13e-11	4.07e-13	2.76e-13	5.74e-12	8.52e-11
F11	2.07e-10	3.74e-09	5.53e-07	NaN	1.21e-09
F12	3.02e-10	0.0561	1.21e-12	8.99e-11	3.02e-11
F13	1.69e-07	4.26e-08	2.39e-07	8.64e-05	3.76e-11
F14	6.25e-10	8.41e-11	3.06e-09	0.0638	3.63e-09
F15	2.74e-09	6.68e09	3.84e-13	2.29e-08	4.72e-10

Tableau IV-9 Résultats de comparaison obtenus par ES-NM, SA-ES, PSO, DE, ICA, GA en 50 D(Boukhari et al., 2019)

Functions		ES-NM	ES	PSO	DE	ICA	GA
F1	Best	**2.7601e-187**	1.2673e-118	4.0975e-138	8.7761e-114	2.7692e-67	5.1671e-157
	Worst	1.7683e-101	3.3532e-97	1.3034e-101	5.8002e-76	1.7608e-59	**7.7824e-102**
	Avg	**1.0679e-105**	6.3407e-98	2.1744e-103	4.3664e-81	6.3541e-62	1.7074e-103
F2	Best	**0**	97.80733	19.6006	6.8766e-18	2.07919	4.68438
	Worst	**5.1609**	25.01641	42.7693	9.60037	57.16023	9.55183
	Avg	**1.5717**	22.28314	29.9483	6.40117	31.65993	7.40022
F3	Best	0.009671	26.70015	26.99471	38.72309	**0.0076491**	0.029803
	Worst	**27.56597**	311.7601	87.23014	102.8256	48.67272	52.76931
	Avg	25.40377	69.81334	54.80226	71.80722	33.14795	**23.54998**
F4	Best	2.7610-04	20.60934	1.67011	3.90231	3.97821	**0.1764e-04**
	Worst	**0.02776**	29.56429	9.38455	11.19334	16.6190	7.89014
	Avg	**0.00826**	27.44701	5.69429	6.56326	9.76631	0.02896
F5	Best	**0**	2.2204e-16	**0**	**0**	7.8901e-08	7.8541e-06
	Worst	**2.2914e-15**	0.24596	7.7542	0.242446	81.68451	1.30115
	Avg	**1.0034e-15**	0.06579	0.4085	0.061663	11.90043	0.05633
F6	Best	**0.19987**	32.4999	1.2999	0.29987	1.7999	0.5912
	Worst	1.1054	38.0999	3.9999	**0.49987**	8.4999	5.5499
	Avg	0.36997	35.4449	2.4549	**0.33988**	4.8599	3.9141
F7	Best	2.3714e-05	0.000817	0.000194	0.00071381	**8.6737e-05**	1.9194e-02
	Worst	**0.0007318**	0.001161	0.635612	59.79742	15.36078	4.120914
	Avg	**1.6329e-04**	0.001016	0.032597	34.49291	0.02597	0.0005735
F8	Best	**1.3295e-13**	0.69451	2.5512e-07	12.00851	2.6205e-04	7.07143
	Worst	**1.0912e-09**	38.1863	7.4057e-05	63.07294	3.900158	33.05618
	Avg	**5.4191e-11**	9.8033	1.0537e-05	51.21672	0.005651	29.63549
F9	Best	2.5017e-101	5.4203e-25	3.3379e-89	**1.2207e-101**	7.197e-39	9.5402e-89
	Worst	**3.9041e-97**	1.3995e-21	2.4443e-69	7.8982e-09	1675.7245	1.9793e-11
	Avg	**7.3438e-99**	2.6305e-22	2.4546e-70	2.7814e-12	18.35946	2.6943e-19
F10	Best	**0**	1.2572e-30	9.1595e-26	3.857e-13	6.4696e-11	4.7801e-15
	Worst	**0**	4.992e-30	5.5468e-19	1.5821	5.4739e-09	3.86816
	Avg	**0**	3.5314e-30	2.8428e-20	0.11029	1.6816e-09	1.9253e-07
F11	Best	**0**	5.2016e-07	**0**	7.6774e-87	**0**	5.8703e-51
	Worst	2.5693e-59	9.84098	8.5882e-17	3.3623e-03	1.9654e-38	7.1282e-22
	Avg	**7.5571e-97**	1.7011e-04	2.3169e-19	6.9004e-05	4.6585e-41	1.6485e-25
	Best	**0**	2.9827e-12	6.6749e-103	1.5044e-18	5.3439e-87	**0**

Functions		ES-NM	ES	PSO	DE	ICA	GA
F12	Worst	**2.7281e-97**	5.6753e-04	5.2165e-27	0.05842	3.2713e-13	0.3664e-69
	Avg	**1.9441e-121**	1.6779e-05	3.8033e-34	0.00739	6.6739e-19	1.2905e-73
	Best	**0**	8.8729e-106	4.7601e-122	1.7627e-59	**0**	7.7571e-79
F13	Worst	**6.6455e-128**	2.3418e-13	2.6716e-86	5.7471e-14	8.8914e-116	4.6199e-26
	Avg	**3.6449e-132**	1.6916e-49	4.7639e-93	1.3116e-17	**4.1553e-137**	5.1564e-31
	Best	**1.7629e-127**	4.6684e-04	7.6723e-89	8.5503e-22	9.6583e-119	1.6533e-61
F14	Worst	**6.7611e-11**	8.6684	3.1548e-07	5.64228e-04	2.6577e-11	5.8037e-08
	Avg	4.7119e-17	1.5627	2.7149e-11	2.83431e-05	**4.9002e-19**	3.7434e-09
	Best	**0**	0.006501	6.3051e-15	4.6715e-07	**0**	6.7323e-24
F15	Worst	**1.6026e-16**	19.7202	5.6988e-07	101.6716	7.7826e-04	6.4156e-09
	Avg	**3.7705e-17**	12.0119	1.5662e-09	68.6022	2.6711e-06	3.6710e-11

Tableau IV-10 Tableau III.10 Comparaison de Wilcoxon-Test pour ES-NM, SA-ES, PSO, DE, ICA et GA. Le tableau montre les valeurs résultantes pour chaque comparaison par paire en 50 D (Boukhari et al., 2019)

P-valeur	ES-NM vs ES	ES-NM vs PSO	ES-NM vs DE	ES-NM vs ICA	ES-NM vs GA
F1	6.68e-10	3.92e-12	1.93e-08	1.38e-10	7.54e-10
F2	4.36e-12	6.63e-11	5.89e-09	6.64e-06	9.82e-07
F3	4.09e-08	5.64e-08	2.51e-07	7.41e-07	0.0612
F4	5.04e-08	0.0081	8.27e-06	4.19e-05	8.14e-05
F5	3.61e-09	6.16e-09	0.0538	1.35e-07	7.66e-06
F6	8.27e-11	1.31e-10	2.02e-09	0.0638	2.75e-09
F7	4.76e-05	8.58e09	4.81e-10	1.26e-07	0.0716
F8	4.97e-11	6.74e-10	5.26e-11	3.84e-11	4.25e-09
F9	8.69e-10	3.64e-07	4.35e-07	5.03e-09	1.95e-09
F10	3.15e-11	6.97e-10	9.72e-09	5.94e-10	3.41e-10
F11	6.35e-11	4.46e-10	9.06e-05	0.0038	3.75e-09
F12	0.0781	0.0984	5.74e-13	3.29e-09	4.01e-10
F13	1.05e-12	2.86e-11	2.71e-06	2.64e-11	9.34e-10
F14	0.0537	1.71e-04	7.99e-12	7.73e-14	3.97e-09
F15	2.43e-10	5.04e-11	3.16e-10	1.94e-12	8.66e-10

La comparaison de ces six algorithmes (ES-NM, ES, PSO, DE, ICA, GA) les uns avec les autres montre que ES-NM est l'algorithme le plus supérieur et le plus efficace que les autres. L'ES-NM affiche de bonnes performances sur 10 problèmes de test sur 15 (moyenne). Cependant, la raison de l'insuffisance d'ES-NM sur cinq (5) problèmes de test pourrait être la nature compliquée de ces fonctions. Il ressort clairement du tableau 3 que la variante ES-NM proposée est compétente pour fournir des solutions efficaces et sauter de nombreux minimums locaux et converger vers le minimum global (Best) dans 2, 5, 10, 11, 12, 13, 15 fonctions. Ces performances témoignent du meilleur compromis entre ES-NM entre diversification et intensification, ainsi que du schéma décisionnel incorporé aux algorithmes dans une phase de recherche locale adaptative. La dernière ligne du tableau 3 résume les algorithmes de classement en fonction du nombre de fois où chaque algorithme obtient de meilleurs résultats, où un nombre plus élevé représente la supériorité de l'algorithme. Les preuves obtenues à partir des résultats fournissent un solveur d'optimisation avec une bonne fiabilité et une efficacité de qualité pour résoudre les fonctions unimodales et multimodales.

Les tableaux statistiques (III.6, III.8, III.10) présentent les p-valeurs produites par Wilcoxon test pour la comparaison par paire sur deux ensembles indépendants d'échantillons (ES-NM par rapport à ES, PSO, DE, ICA et GA), en considérant un niveau de signification de 5%. Pour le test de Wilcoxon, une hypothèse nulle suppose qu'il existe une différence significative entre les valeurs moyennes de deux algorithmes. En revanche, une hypothèse alternative (rejet de l'hypothèse nulle) suggère que la différence entre les valeurs moyennes des deux méthodes comparées est insignifiante. Comme le montrent toutes les valeurs rapportées, suffisamment de

preuves sont fournies pour rejeter l'hypothèse nulle (toutes les valeurs sont inférieures à 0,05 et, en tant que telles, satisfont aux critères de niveau de signification de 5%), sauf dans quelques cas où PSO surpasse ES- NM dans les fonctions F4 (10D), F12 (10D, 30D, 50D) et ICA dans les fonctions F2, F5 (10D), F7, F14 (30D), F6 (50D) et GA dans les fonctions F7, F13 (10D), F3, F7 (50D). Cela prouve la signification statistique des résultats de la méthode proposée et écarte ainsi la possibilité d'être une coïncidence. L'expérience informatique acquise sur les fonctions de test précédentes confirme à nouveau le riche potentiel de l'algorithme hybride ES-NM proposé pour être un solveur polyvalent efficace et efficace pour les problèmes d'optimisation sans contrainte.

III.5 APPLICATION AU PROBLÈME DU FINANCE : PORTEFEUILLE

L'objectif du problème de portefeuille est d'allouer la richesse entre différents actifs afin de maximiser un ensemble de fonctions de performance. Le problème du portefeuille est formulé comme un problème d'optimisation impliquant deux critères : le rendement du portefeuille qui doit être maximisé et le risque du portefeuille qui doit être minimisé (Di Tollo, 2015). En présence de deux critères, il n'y a pas une seule solution optimale, mais un ensemble de portefeuilles optimaux, les portefeuilles dits efficaces, qui font le compromis entre le risque et le rendement. Une méthode à somme pondérée a été mise en œuvre pour trouver des solutions aux problèmes d'optimisation de portefeuille à deux objectifs (Sawik, 2013). La première fonction objective définit le risque de portefeuille ; cet objectif vise à minimiser les risques soumis à des contraintes spécifiques. Le deuxième objectif est de maximiser le rendement attendu du portefeuille. Dans le cas où un décideur est disposé à minimiser la valeur du pire rendement attendu, un modèle de portefeuille doit être formulé avec une valeur à risque conditionnelle (CVaR) (Z. Wang et al., 2014) (Serraino & Uryasev, 2013) comme mesure du risque. L'optimisation du portefeuille a été utilisée avec plusieurs objectifs et résolue avec de nombreuses métaheuristiques. (Chang et al., 2009) ont résolu le problème de l'algorithme génétique (GA) dans un court délai raisonnable, mais tous les cas à risque ne sont pas utilisés. (Qin et al., 2014) ont proposé une optimisation de l'essaim de particules hybrides (PSO) et une colonie d'abeilles artificielles (ABC) qui introduisent les opérateurs ABC dans PSO. La technique proposée s'avère optimale et meilleure en termes de profit. Dans une autre étude, (Tuba & Bacanin, 2014) ont utilisé un algorithme Firefly amélioré (FA) pour un problème d'optimisation de portefeuille contraint et non contraint et les résultats indiquent que l'algorithme proposé est meilleur que d'autres. (Kamili & Riffi, 2016) ont fait une étude comparative des méta-heuristiques (Cat Swarm Optimization CSO, bat algorithme BA et particules swarm optimisation (PSO). Un état de l'art sur les solveurs évolutionnaires pour le problème d'optimisation de portefeuille est illustré par (Yuen et al., 2016). (Jing Wang, 2019) a utilisé un nouvel approche bio-inspiré pour résoudre le problème d'optimisation de portefeuille, les résultats des tests ont indiqué que par rapport à d'autres algorithmes, l'algorithme FA, PSO, DE, GA est plus robuste et plus précis.

III.5.1 Formulation du problème

Il existe de nombreuses formulations du problème d'optimisation de portefeuille. Parmi les premiers, le modèle de variance moyenne (MV) de (Markowitz, 1952). Ce modèle est un problème d'optimisation avec des variables à valeur réelle. Où N est le nombre d'actifs disponibles, μ_i est le rendement moyen d'un actif i et σ_i, j est la covariance des rendements entre les actifs i et j respectivement ($i = 1, \ldots, N;\ j = 1, \ldots, N$). R^* est le rendement moyen ciblé et ω_i est la variable de décision qui contrôle la proportion du capital total disponible qui est investie dans le titre financière i. Une approche pratique qui utilise le paramètre d'aversion

au risque $\lambda \in [0,1]$ fusionne la variance (risque) et le retour en fonction d'objectif unique définie comme suit :

$$\min \lambda[\sum_{i=1}^{N}\sum_{j=1}^{N}\omega_i\omega_j\sigma_{i,j}]+(1-\lambda)[-\sum_{i=1}^{N}\omega_i\mu_i] \quad \text{(III.8)}$$

Sujet à

$$\sum_{i=1}^{N}\omega_i = 1 \quad \text{(III.9)}$$

$$0 \le \omega_i \le 1, \forall_i \in (1,2,...N) \quad \text{(III.10)}$$

Dans le modèle défini, l'efficacité du portefeuille est basée sur le concept d'optimalité de Pareto. Pour un niveau de risque donné, il ne devrait pas y avoir de portefeuille avec un rendement attendu supérieur à un portefeuille efficace. La courbe continue désignée comme frontière efficace montre visuellement les dépendances entre les changements de λ et les intersections du rendement moyen et de la variance. Cette courbe représente les meilleurs compromis entre le rendement moyen et la variance dans la théorie de Markowitz.

III.5.2 Données et étude expérimentale

Pour l'étude expérimentale, les stocks de six sociétés (IBM, Google, Microsoft, Apple, Yahoo, Amazon) appartenant à différentes industries ont été sélectionnés pour être optimisés. Les rendements mensuels et les prix de clôture des actions sont sélectionnés (Heris, 2014). Un calcul de rendement mensuel est formulé comme suit : $M_t = \ln\frac{P_t}{P_{t-1}}$; Où, $P\ (t)$ est le prix de clôture, $P\ (t-1)$ est le prix de clôture de la veille et $M\ (t)$ est le rendement mensuel.

L'algorithme ES-NM proposé, ainsi que l'optimisation de l'essaim de particules (PSO) (Mohammad-Moradi et al., 2009), l'algorithme de compétition impérialiste (ICA) (Emami et al., 2012) et l'algorithme génétique de tri non-dominé (NSGA-II) (Lin, 2012) pour résoudre l'optimisation de portefeuille sont testés et comparés selon le coefficient de variation métrique et la frontière efficace. Afin d'assurer l'équité de l'algorithme, les paramètres de tous les algorithmes sont définis sur les mêmes que ceux utilisés dans (Heris, 2014) et dans le tableau III.2, à l'exception de certains paramètres supplémentaires de l'algorithme génétique multi-objectif choisis dans la version originale NSGA- II (Deb et al., 2002).

Toutes les expériences ont été effectuées sur Windows 7 avec Intel® Core ™ i3 CPU@2,53 GHz et 6 Go de RAM. Tous les codes de programme ont été écrits en MATLAB R2013a.

III.5.3 Résultats et discussion

Un ensemble de solutions non dominées générées par un optimiseur peut être mesuré avec différentes mesures de performance disponibles dans la littérature. Dans le cas, nous n'avons pas d'informations concernant le vrai front Pareto, puis le coefficient de la mesure de variation *CV* est utilisé, ce qui permet de déterminer combien de volatilité, ou de risque, est supposé par rapport au montant de rendement attendu des investissements selon l'équation III.2. Le tableau III.11 enregistre les résultats de calcul de ES-NM, NSGA-II, PSO et ICA. Les résultats du

risque de l'ES-NM sont meilleurs moins que les autres tandis que les meilleurs, les pires et les rendements moyens sont augmentés, car la valeur du coefficient de variation fournit une bonne estimation du risque et du rendement mesurés par la métrique correspondante. Une meilleure compréhension de la nature des solutions trouvées peut être obtenue en analysant la Figure III.3

Les graphiques présentent les frontières efficaces du compromis entre le Retour attendu vs. Risque pour l'ensemble de données historique et montrer la qualité de la solution, ainsi que la cohérence de notre algorithme hybride pour obtenir un retour en hauteur et une stabilité au risque le plus faible possible par rapport au comportement convergent des autres algorithmes. D'un autre côté, notre approche hybride élimine un grand nombre de solutions qui couvrent tout le front et restent une partie moins découverte, ce qui peut restreindre une petite gamme pour le décideur pour sélectionner un portefeuille efficace. Notre algorithme hybride est nettement meilleur que tous les autres algorithmes.

Tableau IV-11 Résultats informatiques (Les meilleures valeurs sont imprimées en gras) (Boukhari et al., 2021)

		ES-NM	**NSGA-** II	**PSO**	**ICA**
Risk	Best	**0.0127**	0.0330	0.0330	0.0330
	Worst	**0.0069**	0.0142	0.0119	0.0119
	Avg	**0.0085**	0.0094	0.0186	0.0191
Return	Best	**0.0142**	0.0121	**0.0142**	**0.0142**
	Worst	**0.0042**	0.0022	0.0018	0.0018
	Avg	**0.0083**	0.0049	0.0074	0.0073
Coefficient of variation CV		**0.0933**	0.1343	0.2162	0.2235

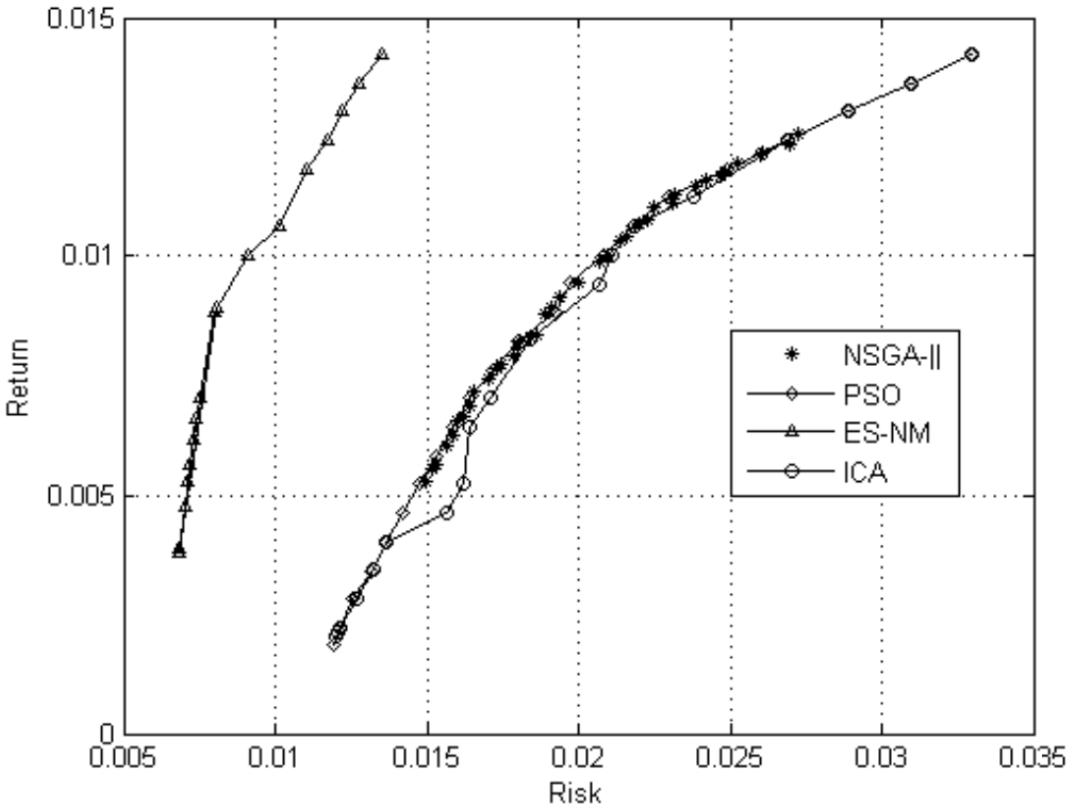

Figure IV-4 Frontière Pareto du portefeuille le mieux généré (Boukhari et al., 2021)

III.6 CONCLUSION

Dans cette étude, nous avons proposé et formulé avec succès une nouvelle approche appelée Evolution Strategy-Nelder-Mead (ES-NM), basée sur le comportement intelligent des algorithmes évolutionnaires et la capacité de convergence de la méthode Nelder-Mead inspirée de l'Amoeba pour résoudre des problèmes d'optimisation. Les stratégies évolutionnaires (ES) a été appliqué pour explorer mieux l'espace de recherche et la méthode Nelder-Mead (NM) pour améliorer le processus de recherche locale. En utilisant un critère de contraction pour basculer entre La recherche globale et locale et une méthode de regroupement a été utiliser pour modérer la recherche locale en appliquant seulement sur les individus prometteurs. Une technique de réinitialisation est adoptée afin d'échapper des points locaux. L'algorithme proposé a été appliqué avec succès à une variété de fonctions de type continu et comparé aux autres algorithmes de la littérature et au problème réel de Portfolio.

La performance des algorithmes est évaluée selon deux critères : la convergence et le coût de calcul. En menant une variété d'études expérimentales et d'après la mise en œuvre et la comparaison, on peut noter qu'il s'agit d'un algorithme très prometteur qui améliore la convergence et le coût de calcul de manière significative sur la plupart des problèmes testés et surpasse d'autres algorithme bio-inspirés autonomes. ES-NM préserve la propriété de convergence globale de ES et maintient une vitesse de convergence rapide. Ainsi, le résultat de cette recherche a ouvert la voie à de nouvelles améliorations pouvant conduire à de futures applications dans les problèmes d'optimisation difficile.

CHAPITRE IV :

CONTRIBUTION 2 : APPROCHE HYBRIDE PROPOSÉE POUR L'OPTIMISATION ÉVOLUTIONNAIRE MULTI-OBJECTIFS

IV.1 INTRODUCTION

L'optimisation multi-objectifs consiste à obtenir simultanément un certain nombre d'objectifs optimisés. Les objectifs sont souvent en conflit les uns avec les autres ; l'amélioration d'une fonction objective entraîne souvent la dégradation d'autres valeurs objectives. La solution à ce type de problème consiste généralement à rechercher l'optimisation simultanée de plusieurs objectifs contradictoires et concurrents. Cependant, il existe un ensemble de solutions qu'aucune meilleure solution unique dans l'espace de recherche n'est meilleure que l'autre avec tous les objectifs considérés. Cet ensemble de solutions est connu sous le nom de solutions Pareto-optimales. En utilisant les solutions optimales Pareto générées, le décideur fera des compromis sur tous les objectifs pris en compte dans la résolution des problèmes du monde réel en fonction de ses connaissances, expériences et expertise (Roijers, 2013).

Il existe un certain nombre de méthodes pour résoudre les problèmes d'optimisation multi-objectifs. Les méthodes classiques de résolution des problèmes d'optimisation à objectifs multiples utilisent l'algorithme pour convertir l'optimisation à objectifs multiples en un problème d'optimisation à objectif unique en mettant l'accent sur une solution Pareto-optimale particulière à la fois, ce qui nécessite plusieurs exécutions pour produire des solutions alternatives pour chaque objectif. Les problèmes d'optimisation multi-objectifs du monde réel sont généralement multidimensionnels et multimodaux, et parfois il n'y a pas de définition claire de certaines fonctions objectives, les méthodes classiques sont généralement inadéquates pour produire les solutions optimales de Pareto pour la résolution de problèmes du monde réel en raison de la grande complexité de calcul. Parce qu'il n'existe pas de solution unique qui optimise simultanément chaque objectif dans un problème non trivial d'optimisation multi-objectif, plusieurs recherches et algorithmes ont été appliqués pour résoudre les problèmes d'optimisation multi-objectif (P. Peng et al., 2014).

Au cours des vingt dernières années, plusieurs algorithmes évolutionnaires multi-objectifs ont été proposés. Bien que ces algorithmes présentent différentes approches de la sélection environnementale, la plupart d'entre eux, y compris le populaire NSGA-II (Deb et al., 2002), SPEA2 (Zitzler et al, 2001), SMS-EMOA (Beume et al., 2007) et MOEA/D (Q. Zhang & Li, 2007) , utilisent un Algorithme génétique (GA) presque identique pour rechercher l'espace de décision. Ce n'est que récemment que des approches hybride s'appuyant sur d'autres algorithmes évolutionnaires, tels que l'évolution différentielle (DE) ont été proposées.

Dans cette recherche, un nouvel algorithme évolutionnaire hybride et multi-objectif unifié est proposé. L'applicabilité et l'efficacité de l'approche proposé sont démontrées en menant des expériences sur trois ensembles de fonctions de référence bien connus, les résultats sont ensuite comparés à d'autres algorithmes évolutionnaires multi-objectifs bien connus en tant qu'études comparatives pour la validation et la vérification. Ensuite, l'algorithme est adapté et appliqué pour résoudre le problème complexe du finance (portfolio optimization).

IV.2 MOTIVATION ET INSPIRATION DE L'APPROCHE PROPOSÉ

Les problèmes d'optimisation multi-objectif MO réels ont généralement des fonctions et des contraintes objectives complexes, et les informations concernant le front optimal de Pareto et son compromis sont généralement limitées ou inconnues a priori. En l'absence de préférence claire de la part du décideur, l'objectif ultime de l'optimisation MO est de découvrir l'intégralité du compromis. Cependant, par définition, cet ensemble de vecteurs objectifs est potentiellement un ensemble infini, et il n'est probablement pas possible de trouver le compromis complet. De plus, la présence d'un trop grand nombre d'alternatives pourrait également être trop écrasante pour le processus décisionnel. En tant que tel, il pourrait être plus pratique de se contenter de la

découverte d'autant de solutions non dominées que possible avec des ressources de calcul limitées. Plus précisément, l'objectif de l'optimisation MO est de trouver un ensemble approximatif de solutions aussi proche que possible du front optimal de Pareto. L'ensemble approximatif doit satisfaire les objectifs d'optimisation suivants (Goh & Tan, 2009).

1. Minimisez la distance entre les solutions obtenues et le front optimal de Pareto.
2. Obtenir une bonne répartition des solutions obtenues le long du front optimal de Pareto.
3. Maximiser la diffusion des solutions obtenues le long du front optimal de Pareto.

Le premier objectif de convergence ou de proximité est avant tout la prise en compte de tous les problèmes d'optimisation, tandis que les deux autres objectifs d'optimisation de maximiser la diversité et la diffusion des solutions sont entièrement uniques à l'optimisation MO. Le deuxième objectif qui est lié à la diversité définit la qualité de la couverture des solutions obtenues dans l'espace optimal, tandis que le troisième objectif qui est lié à l'espacement définit la répartition uniforme des solutions obtenues dans l'espace optimal. La raison de trouver un ensemble approximatif de solutions diverses et uniformément réparties près du front optimal de Pareto est de fournir au décideur suffisamment d'informations sur les compromis entre les multiples solutions avant que la décision finale ne soit prise. Il convient également de noter que les objectifs de convergence et de diversité sont quelque peu contradictoires par nature, ce qui explique pourquoi l'optimisation multi-objectifs est beaucoup plus difficile que l'optimisation mono-objectif.

Les solutions de problèmes d'optimisation multi-objectifs sont considérées comme tout aussi bonnes sans informations de préférence subjective supplémentaires, qui sont également appelées solutions non dominées, optimales de Pareto, efficaces de Pareto ou non inférieures. Parmi les solutions optimales de Pareto, les décideurs déterminent la solution la plus applicable ou la plus favorable en prenant des compromis et des sacrifices. En raison des limites des méthodes classiques dans la recherche des ensembles Pareto-optimaux pour des problèmes difficiles avec des espaces de solutions non convexes, discontinus et multimodaux., Les algorithmes évolutionnaires sont bien adaptés pour résoudre plusieurs problèmes d'optimisation multi-objectifs.

Un algorithme évolutionnaire peut trouver un ensemble de plusieurs solutions non dominées en une seule fois en recherchant simultanément différentes régions d'un espace de solution. Au cours de la dernière décennie, l'application des algorithmes génétique (EA) pour résoudre les problèmes d'optimisation multi-objectifs (MOP) et les processus d'exploration de données pour la découverte des connaissances est devenue un domaine de recherche populaire. Il y avait un certain nombre d'algorithmes développés dans la recherche pour répondre aux problèmes d'optimisation multi-objectifs et souvent contradictoires (A. Zhou et al., 2011).

Comme il existe deux espaces, les algorithmes évolutionnaires multi-objectifs ont tendance à fonctionner à deux niveaux : ils utilisent une procédure de recherche pour explorer l'espace de décision et une procédure dite de sélection environnementale pour sélectionner les meilleures solutions en fonction de leur positionnement dans l'espace objectif. Bien que les deux ne soient pas complètement indépendants (la recherche dans l'espace de décision est généralement guidée par les valeurs objectives des solutions), ils peuvent être considérés comme deux tâches distinctes.

La motivation de ce travail était double. Tout d'abord, nous voulions concevoir un algorithme efficace pour l'optimisation multi-objectifs, qui utiliserait DE pour l'exploration de l'espace de décision d'une manière simple. Bien que des algorithmes basés sur DE pour l'optimisation multi-objectifs aient déjà été proposés dans le passé (voir les travaux connexes présentés dans

la section IV.5), ils ont soit ignoré la caractéristique de base de DE de comparer chaque nouvelle solution à son parent, soit l'ont appliquée trop strictement pour l'optimisation multi-objective. De plus, les approches existantes n'utilisent que la méthode de sélection environnementale de NSGA-II, alors que notre objectif était de permettre des combinaisons d'exploration de l'espace de décision pour booster l'algorithme à sélectionner des solutions meilleures avec une bonne diversité par rapport aux algorithmes proposées de la littérature. Le deuxième objectif est d'appliquer une hybridation efficace de l'algorithme d'optimisation multi-objectif basé sur DE avec d'autre technique d'apprentissage afin de réduire la complexité de calcule et que les résultats soient précis et simples. Notre approche hybride pourrait aider en explorant l'espace.

IV.2 ÉVOLUTION DIFFÉRENTIEL ALGORITHME

Comme mentionné au chapitre I, Pour tenter de trouver l'optimum global des fonctions non linéaires, non convexes, multimodales et non différenciables définies dans l'espace des paramètres continus ($\subseteq \mathbb{R}$), Storn et Price ont proposé l'évolution différentielle (DE) algorithme en 1995. Depuis sa première version, DE et ses variantes ont émergé comme l'une des familles les plus compétitives et les plus polyvalentes des algorithmes évolutionnaires et ont été appliqués avec succès pour résoudre de nombreux problèmes du monde réel issus de divers domaines de la science et de la technologie (S. Das et al., 2016). En commençant par un ensemble uniformément aléatoire de solutions candidates échantillonnées à partir du volume de recherche réalisable, chaque itération (communément appelée génération dans la terminologie informatique évolutive) de DE fonctionne selon les mêmes étapes de calcul que celles employées par un algorithme évolutionnaire standard (AE). Cependant, DE diffère nettement des AE bien connus comme Evolution Strategies (ESs) et Evolutionary Programming (EP) en ce qui concerne le fait qu'il mute les vecteurs de base (parents secondaires) avec des différences échelonnées des membres distincts de la population actuelle. Au fil des itérations, ces différences tendent à s'adapter aux échelles naturelles du paysage objectif. Par exemple, si la population devient compacte dans une variable mais reste largement dispersée dans une autre, les vecteurs de différence échantillonnés seront plus petits dans la première variable, mais plus grands dans la seconde. Cette adaptation automatique améliore considérablement les mouvements de recherche de l'algorithme. Cette propriété est également connue sous le nom de mutation autoréférentielle. En d'autres termes, alors que ES, EP et certains autres algorithmes génétiques codés réels (GA) nécessitent la spécification ou l'adaptation de la taille de pas absolue pour chaque variable au cours des itérations, le DE canonique nécessite uniquement la spécification d'un seul facteur d'échelle relatif F pour toutes les variables. Contrairement à plusieurs autres techniques de calcul évolutionnaire, DE de base se révèle être un algorithme très simple dont la mise en œuvre ne nécessite que quelques lignes de code dans n'importe quel langage de programmation standard **(**voir l'algorithme 5 dans la sous-section I.6.4). De plus, le DE canonique nécessite très peu de paramètres de contrôle (le facteur d'échelle, le taux de croisement et la taille de la population) - une caractéristique qui le rend facile à utiliser pour les praticiens. Néanmoins, DE présente des performances remarquables tout en optimisant une grande variété de fonctions objectives en termes de précision finale, de vitesse de calcul et de robustesse. Il est intéressant de noter que les variantes de DE ont assuré les premiers rangs dans diverses compétitions entre EA organisées dans le cadre de la série de conférences du Congrès de l'$IEEE$ sur le calcul évolutionnaire. Il est évident qu'aucun autre paradigme de recherche unique n'a été en mesure d'obtenir un classement compétitif dans presque toutes les compétitions CEC sur des problèmes d'optimisation à objectif unique, contraint, dynamique, à grande échelle, multi-objectif et multimodal.

L'évolution différentielle (DE) est un AE basé sur la population simple et efficace qui a été rapportée dans plusieurs études sur sa haute robustesse, sa vitesse de convergence rapide et sa

bonne qualité de solution, ce qui en fait une EA très populaire dans la communauté du calcul évolutif. En raison de ces forces observées dans DE, l'utilisation de cette évaluation environnementale peut offrir une réponse à certaines, mais pas à toutes, les limitations susmentionnées. Afin de surmonter ces limites, plusieurs travaux d'amélioration ont été effectués sur DE afin de lui permettre de répondre à l'objectif d'équilibrer la précision de la solution et le taux de convergence tout en maintenant la diversité de la population pour différents types de MOOP. Afin d'atteindre simultanément une vitesse de convergence rapide et une capacité de recherche globale efficace, les chercheurs ont exploré l'utilisation d'hybridation de différentes EA pour les MOOP ainsi que la formulation d'algorithmes mémoriels qui intègrent la recherche locale dans les EA, et ces efforts peuvent également être trouvés dans DE (PANT et al., 2011) (Dong & Wang, 2014). Les chercheurs ont également introduit l'utilisation de variantes de DE adaptatives ou auto-adaptatives (Caponio et al., 2009) (J. Zhang & Sanderson, 2009) qui éliminent la nécessité de subir le processus fastidieux d'essais et d'erreurs de réglage des paramètres de contrôle en DE à un réglage optimal pour les problèmes testés. Cependant, il y a encore beaucoup de place pour l'amélioration de l'aspect de l'amélioration de DE dans son ensemble, et cette thèse vise à explorer des moyens réalisables d'améliorer l'algorithme DE de base pour gérer un large éventail de MOOP.

IV.3 VARIANTES DE IMPORTANTES POUR L'OPTIMISATION GLOBALE

Depuis sa création, DE a été le plus souvent appliqué aux problèmes d'optimisation globale impliquant une seule fonction objective et des contraintes liées aux variables de décision. Après 2010, la plupart des variantes de DE à objectif unique utilisent des schémas d'adaptation de paramètres et des stratégies de mutation et de croisement multiples. Par conséquent, la démarcation entre les algorithmes décrits dans cette section et ceux discutés dans la section précédente peut ne pas être aussi importante. Cependant, dans cette section, nous nous limitons à l'examen de ces méthodes DE qui utilisent des mécanismes heuristiques importants pour améliorer leurs mouvements de recherche, en plus des techniques de sélection de stratégie et d'adaptation des paramètres. Nous passons en revue les algorithmes DE qui ont été publiés dans des revues et conférences de premier plan qui couvrent le domaine du calcul évolutif. Nous discutons de telles variantes de DE sous les sept chapitres suivants, bien qu'un tel regroupement ne soit pas toujours très rigide, car une proposition peut utiliser une combinaison de techniques de différents chapitres. Ces têtes sont DE avec de nouvelles techniques d'initialisation, DE avec de nouvelles ou techniques améliorées de mutation, DE avec des techniques de croisement nouvelles ou améliorées, DE avec des improvisations basées sur des sous-populations et des clusters, topologie de population et diversité de population guidée DE, DE avec représentation d'individus dans l'espace probabiliste, et DE avec cadre de sélection des parents.

IV.3.1 Opt based DE (2-Opt DE)

Pour parvenir à une convergence plus rapide et meilleure que DE classique, (Chiang et al., 2010) ont proposé une variante de DE, où un schéma de mutation influencé par l'algorithme 2-Opt est utilisé à la place des schémas $DE/rand/1$ ou $DE/rand/2$ habituels. L'algorithme 2-Opt a été initialement conçu pour trouver des itinéraires dans le problème des vendeurs ambulants, et a ensuite été appliqué à l'algorithme génétique, au recuit simulé, etc. Cela offre une bonne chance d'échapper aux optima locaux. Suivant l'essence de cette idée, les auteurs conçoivent un schéma de mutation $DE/2-Opt/1$ impliquant trois vecteurs avec les indices $r1$, $r2$ et $r3$ de la population actuelle, et correspondant au $i^{\text{ème}}$ vecteur cible comme suit (pour les problèmes de minimisation) :

$$v_i = \begin{cases} x_{r1} + F(x_{r2} - x_{r3}) \; si \; f(x_{r1}) < f(x_{r2}) \\ sinon \;\; x_{r2} + F(x_{r1} - x_{r3}) \end{cases} \qquad \text{(IV.1)}$$

IV.3.2 Proximity-based DE (ProDE)

(Epitropakis, Li, & Burke, 2013) ont proposé un schéma de mutation induite par la proximité pour DE, où les voisins d'un vecteur parent, plutôt que les aléatoires, seront utilisés pour générer le vecteur donneur. Pour éviter de sacrifier la capacité d'exploration, une approche probabiliste a été suggérée et des résultats empiriques démontrant la dynamique de mutation d'un cycle DE ont été fournis pour justifier l'approche. Le schéma de mutation calcule d'abord la distance par paire entre tous les membres d'une population, et les stocke dans une matrice $R = [r_{ij}]_{NP*NP}$, où r_{ij} est la distance entre le $i^{\text{ème}}$ et le $j^{\text{ème}}$ membre de la population. A partir de ces distances par paire, une probabilité par paire est calculée comme suit, où le voisin éloigné minimum d'un vecteur aura la probabilité la plus élevée d'être sélectionné comme indice r_i. Supposons que la matrice de probabilité soit R_{p}. Alors,

$$R_p(i,j) = \; 1 - \frac{r_{ij}}{\sum_{k=1}^{NP} r_{ik}} \qquad \text{(IV.2)}$$

Maintenant, pour chaque indice i, sélectionnez trois vecteurs avec des indices $k \; = \; r_1, r_2 \; et \; r_3$ dans la population (où, $r_1, r_2, r_3 \in \{1,2, \dots NP\}$) en utilisant une stratégie de roue de roulette sans remplacement basée sur la probabilité de $R_P(i,:)$. Le vecteur donneur v'est alors généré comme suit :

$$v' = v_{r1} + F(v_{r2} - v_{r3}) \qquad \text{(IV.3)}$$

Une mise à jour complète de R et R_p n'est pas requise à chaque génération, car seules les distances et les probabilités pour la progéniture nouvellement sélectionnée doivent être mises à jour. La stratégie a montré expérimentalement d'améliorer les schémas de mutation exploratoires tels que $DE/rand/1$, mais ne parvient pas à montrer une amélioration significative lorsqu'elle est utilisée sur des fonctions hautement multimodales ou hybrides ou avec des schémas de mutation gourmands.

IV.3.3 DE avec différentiels généralisés (DEGD)

(M. M. Ali, 2011) a proposé une nouvelle variante de DE, avec un schéma de mutation modifié. Le schéma de mutation, contrairement à DE/rand/1, n'utilise pas la différence d'échelle des vecteurs, il utilise plutôt la différence des vecteurs d'échelle, définie comme suit.

$$v = x_{r1} + F_1 x_{r2} - F_2 x_{r3} \qquad \text{(IV.4)}$$

Notez que lorsque, $F_1 + F_2 = F$, alors l'équation proposée correspond à celle de DE/rand/1. L'auteur a également affirmé qu'au lieu de générer des vecteurs d'essai pour l'ensemble de la population, il vaut mieux ne le faire que pour les pires solutions. L'auteur a donné des résultats expérimentaux montrant que le meilleur étant remplacé dans une sélection moins souvent que les pires solutions (disons les m solutions du bas de la liste triée des individus de la population, basées sur la fonction objective et quand le meilleur est en haut), générer un essai n'est pas une tâche très bénéfique, lorsque des besoins élevés en puissance de calcul sont présents. Pour augmenter les chances de générer une progéniture de meilleure qualité, le processus de génération de vecteur d'essai peut être répété au plus dire q fois, pour chaque cible. Si dans les premières $q \; - \; 1$ mutations, un essai réussi n'est pas généré, une projection vectorielle est

utilisée à la place de la mutation (deux vecteurs seront choisis au hasard dans la population, et le pire sera projeté sur le meilleur) pour la quatrième fois comme suit :

$$v = \left(\frac{x_{r1}^T . x_{r2}}{x_{r2}^T . x_{r2}}\right) x_{r2} \qquad \text{(IV.5)}$$

IV.3.4 DE avec des opérateurs de mutation basés sur le classement

(Gong & Cai, 2013) suggèrent qu'une mutation peut être plus fructueuse si le vecteur de base et l'un des vecteurs terminaux du vecteur de différence peuvent être sélectionnés en fonction de leur fonction objective, tandis que le troisième est sélectionné au hasard. Cependant, pour rendre le processus moins gourmand, un vecteur n'aura qu'une probabilité d'être sélectionné comme base ou terminal. Les auteurs ont proposé qu'au lieu d'approches aléatoires ou basées sur la proximité, la probabilité qu'un vecteur soit sélectionné dans la mutation peut être calculée à partir de leur rang de fitness dans la population. La stratégie proposée est donc plus rapide et moins coûteuse en termes de calcul. La population doit être triée en fonction de ses valeurs de fitness dans l'ordre croissant (le meilleur est au sommet de la liste). Ensuite, pour la $i^{\text{ème}}$ solution, la valeur de rang R_i est définie comme de $NP - 1$. La probabilité de sélection de la $i^{\text{ème}}$ solution est alors définie comme $p_i = R_i/Np$. Au moment de sélectionner trois vecteurs pour la stratégie $DE/rand/1$, les deux premiers vecteurs sont choisis proportionnellement à leur probabilité de sélection.

IV.3.5 Intersection Mutation DE (IMDE)

Pour améliorer la capacité de recherche et la convergence des DE classiques, (Y. Zhou et al., 2013) ont proposé deux nouvelles variantes (la première axée sur l'exploration, tandis que la seconde sur l'exploitation) avec des schémas de mutation et de croisement modifiés. À chaque génération, la population est d'abord triée. Les meilleurs vecteurs M sont conservés dans un ensemble B et les membres restants sont inclus dans un ensemble W, indiquant des solutions meilleures et pires. Dans la première variante de DE, différentes stratégies de mutation et de croisement sont prescrites pour l'ensemble B et W. Pour tout individu de l'ensemble B, la mutation se fait d'une manière similaire à $rand/1$; seul le vecteur de base est sélectionné au hasard dans l'ensemble W, tandis que les deux vecteurs nécessaires pour calculer la différence sont sélectionnés dans l'ensemble B. Le croisement est similaire au croisement binomial dans la nature ; cependant, un donneur ne peut participer au croisement que s'il est plus en forme que la cible, sinon l'essai sera le même vecteur que la cible. Pour un vecteur de l'ensemble W, la mutation est à nouveau effectuée de la même manière que $rand/1$, seule la base est prise à partir de l'ensemble B et les deux autres vecteurs sont sélectionnés à partir de W. Un croisement binomial régulier est appliqué dans ce scénario. Dans la deuxième proposition, seule une nouvelle mutation est conseillée pour les membres de l'ensemble B, tandis que le reste de l'algorithme est similaire à la première variante. Ici, la base et la borne gauche des vecteurs de différence sont sélectionnées dans l'ensemble W, tandis que le troisième vecteur est choisi dans l'ensemble B.

IV.3.6 DE multi-population avec ensembles équilibrés

(M. Z. Ali et al., 2015) ont proposé une DE multi-population avec recherche équilibrée (mDE-bES) pour augmenter la diversité de la population de DE afin de gérer des problèmes d'optimisation à grande échelle. La population est divisée en sous-groupes où chacun de ces sous-groupes suit des stratégies de mutation et de mise à jour différentes. Les auteurs ont

introduit une nouvelle stratégie de mutation qui soutient le processus de recherche en utilisant des informations provenant soit du meilleur individu, soit d'un individu aléatoire, de manière probabiliste. Il utilise une combinaison linéaire de vecteurs pour produire un vecteur de base. Toutes les stratégies sélectionnent les individus en utilisant une stratégie qui classe les individus en fonction de leur fonction objective. Des individus choisis au hasard sont utilisés pour migrer périodiquement entre les sous-groupes pour soutenir le processus d'évolution.

IV.3.7 Compact DE

La difficulté d'utiliser des AE dans des dispositifs intelligents utilisés dans la vie courante provient de l'exigence de mémoire pour stocker la population et de la puissance de calcul pour la traiter. Pour résoudre ce problème, une classe des AE connue sous le nom AE compactes a été développée avec une représentation réduite de la population par ses propriétés statistiques plutôt que comme un ensemble d'individus. (Mininno et al., 2011) ont développé une variante compacte de l'algorithme DE, appelée cDE, en adoptant les opérations générales et le workflow du DE commun. Le cDE au moment de l'optimisation d'un problème d-dimensionnel, représente une population par une distribution gaussienne multidimensionnelle, tronquée et normalisée, ne stockant que la moyenne et les variances de chaque dimension. cDE initialise d'abord les moyennes à zéro et les variances à être élevées, pour imiter le comportement d'une distribution uniforme et en échantillonne de manière pseudo-aléatoire une solution initiale «élite». Dans une itération de l'algorithme, cDE échantillonne un certain nombre de vecteurs de solution de la distribution comme requis par la stratégie de mutation (par exemple trois vecteurs pour le schéma de mutation $DE/rand/1$) et génère le vecteur donneur. Le croisement est effectué entre l'élite et le donneur pour produire un vecteur d'essai. La sélection se fait entre l'essai et l'élite en fonction de leurs valeurs de fitness et le vecteur sélectionné remplace l'élite. L'élite nouvellement sélectionnée et le perdant du processus de sélection seront tous deux utilisés pour mettre à jour la moyenne et les variances, pour la prochaine itération. Les principaux avantages du cDE sont d'une part le faible encombrement d'au plus $6d$ unités ($2d$ pour la distribution, $3d$ pour la mutation et d pour l'élite), et d'autre part les calculs de mutation, de croisement et de fitness coûteux en calculs ne sont effectués qu'une seule fois pour générer ou sélectionner un seul vecteur par itération. Cependant, les dimensions sont supposées indépendantes, ce qui simplifie le problème en représentant la matrice de covariance en diagonale. En réalité, cela peut devenir une contrainte très dure et l'algorithme peut ne pas identifier complètement les pics mondiaux dans un paysage fonctionnel multimodal avec des liens solides entre les groupes de variables.

IV.3.8 Gaussian Bare-bones DE

Pour réduire les effets des paramètres de contrôle et réduire l'importance de la sélection ou de l'adaptation, (H. Wang et al., 2013) ont proposé une double variation de l'ED, influencée par le concept de l'algorithme d'essaim de particules nues, et les ont appelées DE gaussiennes Bare-bones DE (GBDE) et GBDE modifié (MGBDE). Dans GBDE, la mutation et la sélection de Cr ont été mises à jour, tandis que dans MGBDE, l'étape de mutation est encore modifiée pour inclure l'avantage de la stratégie $DE/best/1$. Dans GBDE, au lieu de régulier schémas de mutation, les donneurs sont générés en les sélectionnant au hasard dans une distribution gaussienne avec une moyenne μ et un écart type σ. Comme les μ et σ sont calculés à partir de la population actuelle elle-même, le besoin de F est éliminé. Pour générer le donneur du $i^{\text{ème}}$ vecteur, μ est calculé comme la moyenne du meilleur et le vecteur cible, tandis que σ est le module de la différence entre eux. La mutation proposée est exploratoire au début, mais à

mesure que la génération augmente, la différence entre le meilleur et tout individu diminuera et la moyenne ira vers le meilleur, encourageant l'exploitation. Une technique de croisement binomial régulière sera utilisée, mais la valeur de Cr sera initialement choisie parmi une distribution gaussienne de moyenne 0,5 et un écart type de 0,1. Dans les générations ultérieures, la valeur Cr dépendra de l'individu et conservera sa valeur actuelle pour un individu si cette solution a été mise à jour dans la génération actuelle, sinon elle sera réinitialisée. Dans le MBGDE, la stratégie de mutation est considérée comme étant $DE/best/1$ ou gaussienne avec une probabilité égale. Selon les résultats expérimentaux, MBGDE a surpassé GBDE. En raison de la nature exploratoire de la mutation, les deux ont réussi plusieurs fonctions multimodales, tout en ne dépassant pas les autres variantes de DE dans certains des cas multimodaux ou unimodaux simples.

IV.3.9 Neighborhood and Direction Information based DE (NDi-DE)

Comprendre et utiliser les informations sur le quartier et la direction est important pour une population DE pour rechercher efficacement. Suivant cette ligne de pensée, Cai et Wang (2013) ont suggéré une amélioration de la stratégie de mutation DE en introduisant le guidage des informations de voisinage et directionnelles. Ils ont d'abord défini un schéma de sélection guidée de quartier (NGS) pour sélectionner les vecteurs de base et de différence pour la mutation. Pour générer le donneur de la $i^{\text{ème}}$ cible, le NGS attribue d'abord une probabilité à chaque vecteur de la population. La probabilité pour tout vecteur d'indice j est calculée comme suit :

$$p_j = 1 - \frac{d(x_i,x_j)}{\sum_{j=1}^{NP} d(x_i,x_j)} \qquad \text{(IV.6)}$$

Où $d(x_i, x_j)$ est la distance euclidienne entre les vecteurs x_i et x_j. L'algorithme utilise une méthode de roulette pour sélectionner trois vecteurs proportionnellement à leurs probabilités dans la population. Le vecteur de base est considéré comme le vainqueur du tournoi entre les vecteurs choisis ; les deux autres sont soustraits pour former le vecteur de différence ($x_{\text{difference}}$). Par ces techniques, le NGS exploite les informations de voisinage et sélectionne également le meilleur vecteur comme vecteur de base. Les informations de direction sont incluses à l'aide d'une stratégie de mutation induite par la direction (DIM). DIM introduit un nouveau vecteur connu sous le nom de vecteur de direction et l'ajoute au vecteur de base et au vecteur de différence à l'échelle obtenus par le processus de sélection guidé par le voisinage. Pour la mutation de la $i^{\text{ème}}$ cible, DIM identifie d'abord la pire solution la plus proche et la meilleure solution la plus proche du vecteur cible. Le meilleur voisin proche est défini comme étant le jème vecteur, pour lequel le rapport de la différence de fitness entre x_i et x_j et la distance entre x_i et x_j est le maximum. Le pire voisin le plus proche est défini comme étant le $j^{\text{ème}}$ vecteur, pour lequel le rapport de la différence de fitness entre x_j et $x_i$et la distance entre x_i et x_j est le maximum. Soit le meilleur voisin le plus proche appelé $x_{\text{i\_best}}$ et le pire le plus proche $x_{\text{i\_worst}}$ .Trois types de propriétés directionnelles, à savoir l'attraction directionnelle (DA), la répulsion directionnelle (DR) et la convergence directionnelle (DC), peuvent être calculés à partir de ces vecteurs, et ils sont définis comme suit :

$$DA_i = I_{DA}\left(x_{i_{best}} - x_i\right), \qquad \text{(IV.7)}$$

$$DR_i = I_{DR}\left(x_{i_{worst}} - x_i\right), \qquad \text{(IV.8)}$$

$$DC_i = I_{DC1}\left(x_{i_{best}} - x_i\right) - I_{DC2}\left(x_{i_{worst}} - x_i\right) \qquad \text{(IV.9)}$$

I_{DA}, I_{DR}, I_{DC1} et I_{DC2} sont des facteurs d'échelle qui peuvent être contrôlés pour limiter l'influence du vecteur directionnel. Le schéma de mutation rand/1 avec DIM est le suivant :

$$v = x_{base} - Fx_{difference} + DF \qquad \text{(IV.10)}$$

Ici, DT peut être DA, DC ou DR. L'algorithme est pourtant simple et empiriquement prouvé efficace. Le choix du paramètre d'échelle et des informations directionnelles dépendent respectivement du problème et de la stratégie de mutation.

IV.3.10 DE with Parent Selection Framework

DE classique n'a pas de critère de sélection parent pour la mutation ou le croisement, car chaque individu doit subir ces deux étapes. (Guo et al., 2015) ont proposé un Framework de sélection des parents intéressant pour DE. Ils considéraient la stagnation comme une situation dans une itération, où l'algorithme ne converge pas vers un point fixe et ne trouve pas de meilleure solution. Ils ont également essayé de quantifier la stagnation. En surveillant simplement la somme des distances des vecteurs individuels à partir de la moyenne de la population, on peut identifier si l'algorithme converge vers un point fixe. Cependant, pour les fonctions multimodales de grande dimension et complexes, il est très difficile pour DE d'atteindre une population convergente. Néanmoins, il est possible de savoir si DE n'a pas réussi à générer une meilleure solution pour un certain nombre d'itérations successives. Si tel est le cas, la population peut être améliorée en ajoutant des descendants des parents qui ont réussi auparavant. Pour chaque individu de la population, un compteur est maintenu qui augmente d'une unité s'il n'est pas remplacé dans la sélection, sinon remis à 0, pour être utilisé dans l'itération suivante. Au moment de la mutation $rand/1$, si le compteur d'un vecteur cible franchit un seuil prédéfini, trois vecteurs sont choisis au hasard dans l'archive de taille Np contenant le dernier solutions mises à jour ; sinon, les vecteurs sont choisis au hasard dans la population actuelle. L'archive est mise à jour en ajoutant les solutions nouvellement sélectionnées en remplaçant les plus anciennes, après chaque génération.

IV.4 DE HYBRIDE

L'hybridation de DE avec d'autres méthodes de recherche a été un domaine de recherche populaire. Certaines tentatives récentes et notables d'hybridation de DE avec PSO et BA peuvent être trouvé dans (Tang et al., 2016) (Jadon et al., 2017). (Anupam Trivedi et al., 2016) ont proposé un hybride de DE et GA pour résoudre les problèmes de planification d'engagement d'unité. GA a été utilisé pour gérer les variables d'engagement d'unité binaire tandis que DE a été utilisé pour optimiser les variables liées à la répartition continue de la puissance. (Hasan et al., 2014) a proposé deux variantes hybrides de DE, l'une avec une recherche locale pour améliorer la capacité d'exploitation et l'autre avec l'algorithme Harmony Search (HS) inspiré par la musique pour trouver en coopération l'optimum global. Pour gérer la situation de problèmes dimensionnels mixtes discrets et à valeur réelle, des auteurs a suggéré une gestion généralisée des valeurs discrètes basée sur l'idée de recherche de table (A. K. Qin & Forbes, 2011). (Boussaïd et al., 2011) ont utilisé un hybride de DE et d'optimisation basée sur la biogéographie (BBO) pour fournir une solution numérique au schéma d'allocation de puissance optimale dans un réseau de capteurs sans fil. (He & Zhou, 2018) ont étudié différents modes de génération de population par DE et $CMA-ES$. L'avantage de la génération de style DE est la couverture de l'espace de recherche distribué non paramétrique, avec une relation directe de progéniture parent dans les générations successives. D'autre part, la génération de style CMA utilise une distribution intermédiaire paramétrique pour générer une progéniture et modifie la

distribution avec des informations cumulatives recueillies auprès de toutes les générations précédentes. Le tableau ci-dessous illustre l'hybridation de DE assisté par d'autre optimiseur.

Tableau 0-1 Exemples d'algorithmes hybrides DE récemment développés

Méthode hybride	Référence	Applications
Differential evolution + Invasive Weed Optimization (IWO)	(Basak et al., 2013)	Fonction contrainte limitée à un seul objectif
Differential evolution + Firefly Algorithm (FFA)	(Abdullah et al., 2013)	Estimation des paramètres du modèle biologique
Differential evolution + Gravitational Search Algorithm (GSA)	(Chakraborti et al., 2015)	Problème d'optimisation binaire impliquant une sélection hautement discriminante de sous-ensembles de fonctionnalités
Differential Evolution + local neighborhood search	(Shao & Pi, 2016)	Planification de l'atelier de flux
Differential evolution + fuzzy clustering chaotic	(Tran et al., 2016)	Probleme de nivellement de ressources multiples
Differential evolution + opposition based learning + Greywolf optimizer	(Cheng & Tran, 2015)	Planification des horaires de travail
Differential evolution + k-means	(Mustafa et al., 2019)	Problème de regroupement de données
Differential evolution + surrogate model	(Cai et al., 2019)	Optimisation coûteuse de la conception
Differential evolution + Neuron network	(L. Wang et al., 2015)	Prévision de séries chronologiques
Differential evolution + BFGS search	(L. Peng et al., 2017)	Benchmarking problèmes
Differential evolution + Taguchi method	(Guo et al., 2015)	Conception technique et modélisation de machines et de composants

IV.5 ÉVOLUTION DIFFÉRENTIELLE POUR L'OPTIMISATION MULTI-OBJECTIVE

Plusieurs chercheurs ont appliqué DE pour résoudre des problèmes multi-objectifs (A. Zhou et al., 2011), avec des performances supérieures et impressionnantes à d'autres algorithmes à

objectifs multiples dans des fonctions de référence (Benchmarks) et des applications pratiques largement rapportées et vérifiées (M. Ali et al., 2012) (Reddy & Kumar, 2007). La procédure générale de DEMO est montrée dans l'algorithme 11**.** Comme DE, l'algorithme commence par une population P de p solutions créées aléatoirement. À chaque génération, les étapes suivantes sont répétées. Un candidat est construit à partir de son parent (et d'autres solutions de P) en utilisant la stratégie $DE/rand/1/bin$ décrite précédemment **.** Après cela, le candidat est évalué et comparé à son parent. À ce stade, DEMO diffère de DE (voir l'étape 2.1 (c) dans l'algorithmes 11). Dans DEMO, le candidat ne remplace le parent que s'il le domine. Si le parent domine le candidat, le candidat est écarté. Sinon (lorsque le candidat et le parent sont incomparables), le candidat est ajouté à la population. Après avoir répété cette étape p fois, nous obtenons une population de taille comprise entre p et $2p$. Si la population s'est agrandie, elle doit être tronquée à la taille p en utilisant l'une des approches de sélection environnementale. En fin de génération, les solutions de P sont énumérées au hasard (comme dans DE). Le résultat final de DEMO consiste en des solutions non dominées de P.

Algorithme 11 : procédure générale de MODE(Robič & Filipič, 2005)

Entrée : paramètres de l'algorithme.

Sortie : Solutions non dominées de P.

1. Évaluer la population initiale P de p solutions aléatoires.

2. Tant que le critère d'arrêt n'est pas atteint, procédez comme suit :

2.1. Pour chaque solution x i (i = 1, ..., p) de P répéter :

(a) Créez le candidat c à partir du parent x i (voir algorithme 3.2).

(b) Calculer les objectifs du candidat.

(c) Si le candidat domine le parent, le candidat remplace le parent.

Si le parent domine le candidat, le candidat est écarté.

Sinon, le candidat est ajouté à la population.

2.2. Si la population a plus de p solutions, appliquer un environnement

Procédure de sélection pour obtenir les meilleures solutions p.

2.3. Énumérer au hasard les solutions dans P.

Notez que les candidats nouvellement créés qui entrent dans la population (soit par remplacement soit par ajout) participent instantanément à la création des candidats suivants. Cela permet d'atteindre une convergence rapide vers le front optimal de Pareto.

L'extension du DE aux MOP a également donné des résultats prometteurs (S. Das et al., 2016). Dans ce qui suit, nous passerons en revue le multi-objectif DE (MODE) sous cinq aspects, à savoir les stratégies adaptatives, l'opérateur de sélection, le contrôle de la diversité, la gestion des contraintes et les applications pratiques.

Pour les stratégies adaptatives de type de mutation et de paramétrage de MODE, (Huang et al., 2007) ont proposé un MODE auto-adaptatif (MOSaDE) capable de sélectionner de manière adaptative les stratégies de mutation appropriées. Plus tard, une nouvelle version de l'hybride MOSaDE avec des stratégies d'apprentissage par objectifs (OWMOSaDE) a été développée par (Huang et al., 2009), qui peuvent sélectionner de manière adaptative des stratégies de mutation et des paramètres de croisement appropriés pour chaque objectif séparément. (J. Zhang & Sanderson, 2008) ont présenté un MODE auto-adaptatif dans lequel les informations de direction des solutions inférieures archivées à la population actuelle sont utilisées dans le processus de mutation. (L. Tang et al., 2019) ont utilisé les informations d'évolution des solutions le long de chaque direction de recherche pour adapter les paramètres de contrôle de MODE. (Li et al., 2014) ont proposé une variante de MOEA/D avec des opérateurs DE

(MOEA/DFRRMAB) où les taux d'opérateurs de mutation ont été déterminés de manière adaptative par un schéma de bandit à plusieurs bras.

Pour l'opérateur de sélection dans MODE, les modes traditionnels (par exemple, (Huang et al., 2009), et évolution différentielle de Pareto (Abbass, et al., 2001)) ont utilisé le schéma de sélection un-à-un, c'est-à-dire que le vecteur d'essai a été autorisé à entrer dans le prochaine génération si et seulement si elle pouvait dominer sa solution cible. Différent de ce schéma, certains MODE ont préféré stocker les nouveaux vecteurs d'essai puis sélectionner de bonnes solutions à partir de l'union de vecteurs d'essai et la population d'origine pour construire la population suivante. L'application d'un tel schéma se retrouve dans le MODE nommé PDEA proposé par (Madavan, 2002), DEMO proposé par (Robič & Filipič, 2005), ADEMO proposé par (Qian & li, 2008), MODEA proposé par (M. Ali et al., 2012), et le DEMON proposé par (Rakshit et al., 2014). Un hybride des deux schémas ci-dessus a été adopté par le MODE nommé GDE3 (Kukkonen & Lampinen, 2005), qui était une extension du DE généralisé (Kukkonen & Lampinen, 2004).

En ce qui concerne le contrôle de la diversité dans MODE, de nombreuses recherches ont préféré adopter le classement Pareto et la distance de surpopulation de NSGA-II (Coello Coello et al., 2004) pour tronquer l'archive élitiste. Au lieu de surpasser la distance, (Y. N. Wang et al., 2010) ont utilisé une mesure de diversité d'entropie d'encombrement dans le MODE auto-adaptatif pour préserver la diversité des archives élitistes. Huang et al. [31] ont développé une mesure de diversité basée sur la distance harmonique pour maintenir la diversité des archives externes. La -dominance a été intégrée dans MODE pour garantir la diversité des archives externes (V. L. Huang et al., 2005). Outre les métriques de mesure de la diversité, la stratégie des populations multiples a également été adoptée dans MODE pour maintenir la diversité. Un MODE coopératif (CMODE) avec plusieurs populations a été présenté par Wang et al. (Jiahai Wang et al., 2016), dans lequel chaque population a traité un seul objectif. Une autre version du mode de multi-population a été proposée dans (X. Wang & Tang, 2016), dans laquelle chaque sous-population a adopté sa propre stratégie de mutation et a concouru pour les ressources d'évolution.

Pour le MOP avec contraintes, (Santana-Quintero et al, 2010) a incorporé la recherche locale basée sur la théorie des ensembles approximatifs dans MODE. (Liang et al., 2014) ont développé un nouveau MODE en concevant une nouvelle méthode de gestion des contraintes pour guider la recherche d'individus en utilisant un certain nombre de bonnes solutions infaisables.

Pour l'application pratique du MODE, (X. Wang & Tang, 2013) ont présenté un MODE pour l'optimisation du fonctionnement de la pyrolyse du naphta. (Sharma & Rangaiah, 2013) ont présenté un MODE incorporant une liste taboue pour les applications du génie chimique. (Xu, Qi et al., 2013) ont étudié l'optimisation du fonctionnement du processus de réaction d'oxydation du p-xylène et développé un MODE avec une stratégie auto-adaptative pour la génération de vecteurs d'essai et le contrôle des paramètres. (Baatar et al., 2014) ont proposé un MODE avec une stratégie de contrôle adaptatif des paramètres et un classement non dominé pour la MOP des problèmes électromagnétiques. (Shen & Wang, 2017) ont proposé un MODE adaptatif pour le problème de définition du point de fonctionnement avec quatre objectifs axés sur le coût de la consommation de carburant et la réduction des émissions.

(Denysiuk et al., 2013) ont proposé une variante DE pour résoudre les problèmes d'optimisation à objectifs multiples appelée MyODEMR (DE à objectifs multiples avec restriction de mutation). L'algorithme utilise le concept de dominance de Pareto couplé à la métrique de distance générationnelle inversée pour sélectionner la population pour la prochaine génération

à partir d'une combinaison des populations parent et progéniture. L'algorithme utilise également une stratégie de restriction du vecteur de différence dans la mutation DE pour améliorer les caractéristiques de convergence sur des paysages de remise en forme multimodaux. Récemment, (Bandyopadhyay & Mukherjee, 2015) ont proposé un algorithme d'optimisation à plusieurs objectifs qui réorganise périodiquement les objectifs en fonction de leur statut de conflit et sélectionne un sous-ensemble d'objectifs conflictuels pour un traitement ultérieur. Les auteurs ont utilisé DEMO comme évolution métaheuristique sous-jacente algorithme, et a mis en œuvre la technique de sélection d'un sous-ensemble d'objectifs en conflit en utilisant un ordre basé sur la corrélation des objectifs. La méthode résultante est appelée α-DEMO, où α est un paramètre déterminant le nombre d'objectifs contradictoires à sélectionner. Le DE a également été utilisé comme optimiseur de base dans les MOEA basés sur la décomposition récemment développés pour une optimisation à plusieurs objectifs comme (S. Yang et al, 2017).

IV.6 ÉVOLUTION DIFFÉRENTIELLE MULTI-OBJECTIFS AVEC OPÉRATEUR DE MUTATION BASÉ SUR LE CLASSEMENT MODE-RMO

Dans cette section, l'algorithme MODE-RMO proposé est présenté en détail. Dans MODE-RMO, l'opérateur de mutation basé sur le classement est introduit dans MODE pour améliorer ses performances en accélérant la vitesse de convergence. Par conséquent, dans la partie suivante, nous allons d'abord décrire l'opérateur de mutation basé sur le classement pour l'optimisation multi-objectif ; puis MODE-RMO est minutieusement élucidé.

Dans la nature, les bons individus contiennent toujours de bonnes informations et elles sont plus susceptibles d'être utilisées pour guider d'autres personnes. Sur la base de ces considérations, Gong et Cai ont proposé un opérateur de mutation basé sur le classement pour DE dans l'optimisation à objectif unique (Gong & Cai, 2013), dans lequel les parents sont sélectionnés proportionnellement en fonction de leur classement dans la population actuelle. Plus un parent obtient un rang élevé, plus il sera sélectionné. Les auteurs ont intégré l'opérateur de mutation basé sur le classement proposé dans certains algorithmes DE, et les résultats expérimentaux ont indiqué que l'opérateur de mutation basé sur le classement est capable d'améliorer les performances des algorithmes DE dans l'optimisation d'un seul objectif. L'opérateur de mutation basé sur le classement a été introduit dans MODE pour MOP. Par conséquent, nous souhaitons étendre cet opérateur pour MODE. Cependant, le défi de cette extension est que, la population peut être directement triée du meilleur au pire dans l'optimisation à objectif unique, alors qu'il existe de nombreuses solutions qui ne sont pas dominées entre elles dans l'optimisation à objectifs multiples. Considérant que l'algorithme MODE doit générer des solutions de Pareto approximatives avec à la fois une bonne convergence et une bonne propagation, par conséquent, un tri rapide non dominé et une distance de saturation sont incorporés dans l'opérateur de mutation basé sur le classement pour traiter les MOP.

IV.6.1 Tri rapide non dominé et distance de rassemblement

La distance de tri et de rassemblement rapide non exprimée est proposée par Deb et al. dans l'algorithme NSGA-II (Deb et al., 2002). Dans la procédure de tri rapide non majoré, pour chaque solution, deux entités sont calculées :

1) n_p : le compte de domination, c.-à-d. le nombre de solutions qui dominent la solution p;

2) S_p: un ensemble de solutions que la solution domine. Toutes les solutions du premier front non dominé auront leur compte de domination à zéro. Maintenant, pour chaque solution p avec $n_p = 0$, nous visitons chaque membre (q) de son ensemble S_p et réduisons son compte de

domination par 1. Si pour un membre le compte de domination devient nul, nous le mettons dans une liste séparée Q. Ces membres appartiennent au deuxième front non dominé. Maintenant, la procédure ci-dessus se poursuit avec chaque membre et le troisième front est identifié. Ce processus se poursuit jusqu'à ce que tous les fronts soient identifiés.

La distance de surpeuplement est utilisée pour obtenir une estimation de la densité des solutions entourant un individu particulier i dans la population, elle calcule la distance moyenne de deux solutions de chaque côté de la solution i ($ie\ i+1\ \ et\ \ i-1$) le long de chacun des objectifs. Le calcul de la distance de rassemblement nécessite de trier la population en fonction de chaque valeur de fonction objective dans un ordre de grandeur croissant. Ensuite, pour chaque fonction objective, les solutions aux limites (solutions avec les valeurs de fonction les plus petites et les plus grandes) se voient attribuer une valeur de distance infinie. Toutes les autres solutions intermédiaires se voient attribuer une valeur de distance égale à la différence normalisée absolue des valeurs de fonction de deux solutions adjacentes. Ce calcul est poursuivi avec d'autres fonctions objectives. La valeur globale de la distance de rassemblement est calculée comme la somme des valeurs de distance individuelles correspondant à chaque objectif. Chaque fonction objective est normalisée avant de calculer la distance de surpeuplement.

IV.6.2 Attribution de classement et probabilité de sélection

Après avoir obtenu le nombre de front de non-dénomination i_{front} et la distance de rassemblement i_{cd} pour chaque solution i, nous pouvons définir un ordre partiel pour l'ensemble de la population. La solution i est meilleure que la solution j, si l'une des deux conditions est remplie : $(i) i_{front} < j_{front}$; $(ii) i_{front} = j_{front}\ et\ i_{cd} > j_{cd}$

Maintenant, la population peut être trier par ordre croissant en fonction de l'ordre partiel défini ci-dessus. Le classement d'un individu est attribué comme suit :

$$R_i = Np - 1, i = 1,2, \dots, Np \qquad \text{(IV.11)}$$

Où Np est la taille de la population. Selon l'équation. (IV.1), le meilleur individu de la population actuelle obtiendra le rang le plus élevé. Après avoir attribué le classement pour chaque individu, la probabilité de sélection pi du $i^{\text{ème}}$ individu est calculée comme suit :

$$p_i = \frac{R_i}{Np}, i = 1,2, \dots, Np \qquad \text{(IV.12)}$$

Après avoir calculé la probabilité de sélection de chaque individu par l'équation. (IV.12), l'opérateur de mutation basé sur le classement de « $DE/rand/1$ » pour l'optimisation multi objectif peut être décrit sur l'algorithme 13. À partir de l'approche de l'algorithme 13, l'individu avec un classement plus élevé aura une plus grande probabilité d'être sélectionné comme vecteur de base ou vecteur terminal dans l'opérateur de mutation ; par conséquent, il est bénéfique pour la propagation de bonnes informations dans la population à la progéniture. Dans l'opérateur de mutation basé sur le classement pour l'optimisation multi-objectifs, seul le vecteur de base et le vecteur terminal sont sélectionnés en fonction de leurs probabilités de sélection, tandis que le vecteur de départ est sélectionné au hasard. En effet, si les deux vecteurs du vecteur de différence sont tous deux choisis parmi de meilleurs vecteurs, la taille de pas de recherche du vecteur de différence peut diminuer rapidement et conduire à une convergence prématurée (Gong et al., 2014).

Algorithme 13 Opérateur de mutation basé sur le classement de «DE/rand/1» pour l'optimisation multi-objectif

Entrée : l'indice de vecteur cible i et les valeurs des fonctions objectives de la population

Sortie : les indices vectoriels sélectionnés r_1, r_2, r_3

Étape 1 : calcul du nombre de fronts de non dominé de chaque individu par une procédure de tri rapide non dominé

Étape 2 : calculer la distance de surpeuplement de chaque individu ;

Étape 3 : trier la population par ordre croissant et attribuer le classement et la probabilité de sélection p_i pour chaque individu ;

Étape 4 : Sélectionnez aléatoirement $r_1 \in \{1, Np\}$ {indice de vecteur de base}

Tant que rand > p_{r1} ou r_1 == i

Sélectionnez aléatoirement $r_1 \in \{1, Np\}$

Fin Tant que

Étape 5 : Sélectionnez aléatoirement $r_2 \in \{1, Np\}$ {*indice de vecteur terminal*}

Tant que $rand > p_{r2}$ *ou* $r_2 == r_1$ *ou* $r_2 == i$

Sélectionnez aléatoirement $r_2 \in \{1, Np\}$

Fin Tant que

Étape 6 : Sélectionnez aléatoirement $r3 \in \{1, Np\}$ {indice de vecteur de départ}

Tant que $r_3 == r_2$ *ou* $r_3 == r_1$ *ou* $r_3 == i$

Sélectionnez aléatoirement $r_3 \in \{1, Np\}$

Fin Tant que

IV.6.3 Opérateur de sélection

En MODE, le croisement est effectué de la même manière que dans l'optimisation à objectif unique. Cependant, la sélection doit être repensée, car le vecteur d'essai et le vecteur cible sont souvent non dominés l'un par rapport à l'autre. Dans MODE-RMO, nous utilisons l'opérateur de sélection suivant, qui comporte trois étapes :

(1) Si le vecteur d'essai domine le vecteur cible, utilisez le vecteur d'essai pour remplacer le vecteur cible.

(2) Si le vecteur cible domine le vecteur d'essai, le vecteur d'essai est mis au rebut.

(3) Sinon, le vecteur d'essai est ajouté à la population.

Ainsi, à la fin d'une génération, la taille totale de la population se situe entre NP et $2NP$. Cette population est tronquée pour la prochaine étape de l'algorithme. Le processus de troncature consiste à trier et à évaluer les individus d'un même front avec la distance de surpopulation. La procédure de troncature ne conserve que les meilleurs vecteurs NP de la population.

Par intégration avec l'opérateur de mutation basé sur le classement, l'algorithme MODE-RMO peut être décrit comme suit :

Algorithme 12 procédure générale de l'algorithme MODE-RMO

Étape 1 : définir le nombre de génération $g = 0$,

Initialiser au hasard la population $P^g = \{x_1^g, x_2^g, \dots, x_{Np}^g\}$

avec $x_i^g = \{x_{i1}^g, x_{i2}^g, \dots, x_{iD}^g\}$ $(i = 1, \dots, Np)$ uniformément répartis dans l'espace de recherche

Définissez le facteur d'échelle de mutation F, la constante de croisement CR et le nombre maximal de générations Maxgen.

Étape 2 : évaluer la valeur de fitness de chaque vecteur cible x_i^g.

Étape 3 : Pour chaque vecteur individuel, effectuez les étapes suivantes de l'étape 4 à l'étape 7.

Étape 4 : déterminer les indices vectoriels sélectionnés r_1, r_2 et r_3 à la méthode décrite à l'algorithme 13.

Étape 5 : Utiliser l'opérateur de mutation basé sur le classement pour générer un vecteur m_i^{g+1} correspondant au vecteur cible x_i^g selon l'équation.(5).

Étape 6 : Utilisez l'opération de croisement pour générer un vecteur d'essai v_i^{g+1} pour chaque vecteur cible x_i^g selon l'équation.(6).

Étape 7 : Évaluez le vecteur d'essai v_i^{g+1} et utilisez l'opération de sélection suivante :

(i) Si v_i^{g+1} domine $x_i^g, x_i^{g+1} = v_i^{g+1}$.

(ii) Si x_i^g domine v_i^{g+1}, v_i^{g+1} est rejeté.

(iii) Si x_i^g et v_i^{g+1} ne sont pas exprimés l'un avec l'autre, ajoutez v_i^{g+1} à la population.

Étape 8 : Après l'étape 7, la taille de la population varie de Np à 2Np. Tri de la population basé sur un tri non dominé et une distance De surpopulation, et les meilleurs individus Np survivent à la génération suivante.

Étape 9 : Réglez g = g + 1, revenez à l'étape 3 jusqu'à ce que le $Maxgen$ maximum soit atteint.

Il est à noter que la population actuelle a été triée en fonction du tri et de la distance d'encombrement rapides de l'étape 8 de la dernière génération ; et donc, à partir de la deuxième génération, il n'a pas besoin de trier à plusieurs reprises la population actuelle dans Step 4.

Les avantages de MODE-RMO peuvent être énumérés comme suit (X. Chen et al., 2014) :

- Étant donné que les bons parents sont plus susceptibles d'être choisis dans l'opérateur de mutation basé sur le classement, il est utile d'améliorer les performances de l'algorithme MODE.
- Par rapport à l'algorithme MODE original, MODE-RMO n'a besoin que d'un petit coût de calcul supplémentaire pour trier la population et sélectionner les individus. MODE-RMO n'augmente pas la complexité globale de l'algorithme MODE d'origine.
- Différent de la plupart des variantes de l'état de l'art MODE qui ont une structure très complexe, MODE-RMO a une structure très simple ; par conséquent, il peut être facilement mis en œuvre.

IV.7 APPRENTISSAGE PAR OPPOSITION

L'apprentissage par opposition (OBL) est un nouveau concept d'apprentissage automatique, inspiré de la relation opposée entre les entités. En 2005, pour la première fois, le concept d'opposition a été introduit, ce qui a attiré beaucoup d'efforts de recherche au cours de la dernière décennie. Une variété d'algorithmes informatiques doux tels que les méthodes d'optimisation, l'apprentissage par renforcement, les réseaux de neurones artificiels et les systèmes flous ont déjà utilisé le concept d'OBL pour améliorer leurs performances. De janvier 2005 à 2018, plus de 600 publications ont été publiées sur le concept OBL. Ces travaux de recherche ont été publiés dans des conférences, des revues et des livres sur l'apprentissage automatique ou l'informatique douce. Le concept d'opposition calculatoire (Tizhoosh, 2005) était inspiré du concept d'opposition dans le monde réel et les nombres opposés étaient simplement définis dans comme suit.

Définition 1 (numéro opposé). (Tizhoosh, 2005) Soit $x \in [a, b]$ un nombre réel. Son opposé, $\breve{x}$, est défini comme suit :

$$\breve{x} = a + b - x, \quad \text{(IV.13)}$$

Définition 2 (point opposé dans l'espace D). (Tizhoosh, 2005) Soit x $(x_1, \ldots, x_D)$ un point dans l'espace dimensionnel D et $x_i \in [a_i, b_i]$, $i = 1, 2, \ldots, D$. L'opposé de x est défini par $\breve{x}$ $(\breve{x}_i, \ldots, \breve{x}_D)$ comme suit :

$$\breve{x}_i = a_i + b_i - x_i, \quad \text{(IV.14)}$$

En fait, ils indiquent que pour trouver la solution optimale inconnue, la recherche simultanée d'une direction aléatoire et de son opposé donne une chance plus élevée de trouver les régions prometteuses. Il est raisonnable que si les estimations actuelles (supposer) sont loin de la solution optimale inconnue, le calcul de leurs opposés mène à la direction opposée vers la solution optimale inconnue. Notez que le point opposé de base est calculé de la même manière qu'un point réfléchi lorsqu'il est calculé via le point central $((x_1 + x_2 + \cdots + x_D)/2)$.

IV.8 ALGORITHME HYBRIDE PROPOSÉ MODE-OBL

Cette section décrit l'évolution différentielle multi-objectif basée sur l'opposition MODE-OBL développé dans cette étude sur la base de l'algorithme MODE-RMO (X. Chen et al., 2014) (voir l'algorithme 12). Dans MODE-OBL, les solutions potentielles candidates sont traitées à l'initialisation et les capacités d'exploration et d'exploitation au cours des différents processus d'optimisation sont principalement concernées. OBL peut être utilisé en deux étapes de MODE-OBL. Premièrement, au stade de l'initialisation afin de parvenir à des solutions candidates à l'ajustement plus approprié dans des conditions où il n'y a pas de connaissances a priori sur les individus initiaux ; Deuxièmement, lors de la mise en œuvre de MODE-OBL afin de forcer la population actuelle à se lancer dans de nouvelles solutions candidates plus adaptées que celles actuelles. Ces deux étapes sont appelées respectivement initialisation de la population basée sur l'opposition et saut de génération basé sur l'opposition. De cette façon, l'algorithme proposé peut converger plus rapidement tout en conservant une bonne diversité. La figure 0-1 montre l'architecture opérationnelle globale de l'algorithme proposé. La majorité des travaux ont utilisé OBL dans le cadre de l'optimisation mono-objective et récemment des tentatives très peu sont investi pour hybrider OBL avec les algorithmes évolutionnaires multi-objectives (Ahandani & Alavi-Rad, 2012) (Luong et al, 2018). D'où notre approche est inspirée pour incorporer une technique d'apprentissage en deux phase (série) dans MODE.

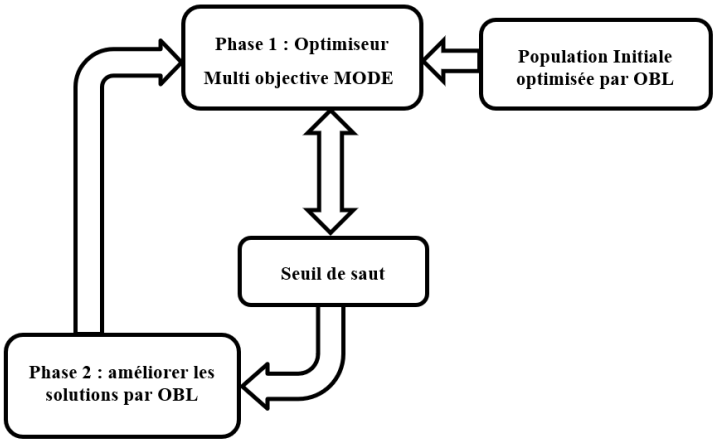

Figure 0-1 l'architecture globale de l'algorithme proposé

IV.8.1 Initialisation basé sur l'apprentissage par opposition

L'initialisation aléatoire, en l'absence de connaissances a priori, réduit les chances d'échantillonner de meilleures régions dans des algorithmes basés sur la population. Cependant, l'utilisation d'OBL peut obtenir des candidats de départ plus adaptés même en l'absence de connaissances a priori et augmenter la probabilité de détecter de meilleures régions. OBL est un mécanisme efficace dans le domaine de l'optimisation en raison du potentiel prometteur d'améliorer les performances de divers algorithmes d'optimisation, notamment KH (G. G. Wang et al., 2016) , DE (Park & Lee, 2016) et PSO (H. Wang et al., 2011). L'Optimisation basée sur l'opposition : Soit $P\ (x_1, x_2, \ldots, x_D)$, un point dans un espace de dimensions D avec $xi \in [a_i, b_i]\ (i = 1, 2, \ldots, D)$, soit une solution candidate. Supposons que $f\ (x)$ est une fonction de fitness utilisée pour mesurer l'optimalité candidate. Selon la définition du point opposé, $\breve{P}\ (\breve{x}_1, \breve{x}_2, \ldots, \breve{x}_D)$ est l'opposé de $P\ (x_1, x_2, \ldots, x_D)$,. Si $f\ (\breve{P})$ est meilleur que $f\ (P)$, alors le point P peut être remplacé par $\breve{P}$; sinon, nous continuons avec P. Par conséquent, le point et son point opposé sont évalués simultanément pour continuer avec l'ajusteur (Rahnamayan et al., 2008). Inspiré de l'idée de (G. G. Wang et al., 2016), nous menons ici une nouvelle méthode d'initialisation de MODE avec OBL, qui est différente de la sélection de survie dans les précédents algorithmes basés sur OBL (dont les stratégies OBL choisissent les meilleurs individus Np parmi les individus Np originaux et les individus Np opposés dans initialisation). Les principales étapes pour expliquer cette procédure sont données comme le montre la Figure 0-1 :

Étape 1 : Diviser la population $P\ (Np)$ en deux parties, la première moitié de la population P_1 est générée par une distribution aléatoire

Étape 2 : La moitié restante de la population P_2 est initialisée en termes d'OBL, comme indiqué dans la section : IV.7

$$P_2 = a_i + b_i - P_1 \qquad \text{(IV.15)}$$

Étape 3 : L'ensemble $P_1 \cup P_2$ est restructuré en tant que population initiale Np.

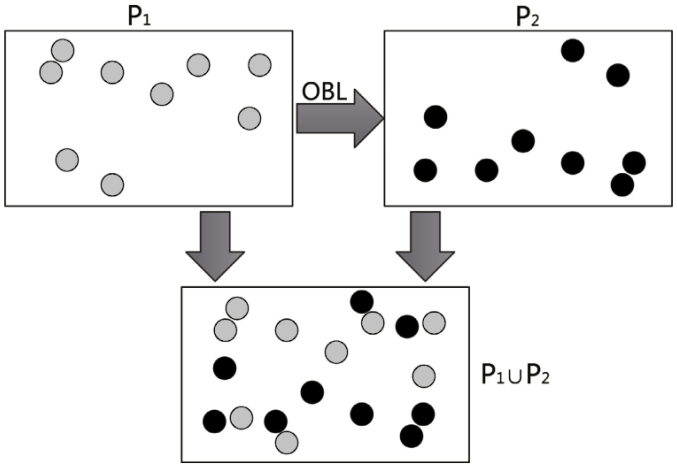

Figure 0-2 stratégie d'initialisation basée sur OBL dans MODE

La stratégie OBL utilisée dans notre algorithme proposé est différente de l'algorithme traditionnel basé sur OBL. En ce qui concerne l'initialisation, la stratégie OBL traditionnelle initialise d'abord la population au hasard, puis calcule la population opposée. Alors que l'opération OBL dans notre algorithme proposé divise d'abord la population en deux parties, puis génère de manière aléatoire et calcule l'inverse respectivement, ce qui peut obtenir des

candidats de départ plus adaptés lorsqu'il n'y a aucune connaissance a priori de la solution. De plus, après les étapes ci-dessus dans la génération et le calcul opposé, les deux sous-populations sont composées d'une population, ce qui peut rendre la taille de la population inchangée dans le processus d'optimisation et aider l'algorithme à fonctionner efficacement.

Quant à l'utilisation de l'OBL dans la phase d'évolution, les algorithmes traditionnels basés sur l'OBL peuvent appliquer l'OBL dans la phase d'évolution avec un taux de saut ou une probabilité de saut. Cependant, l'OBL du MODE proposé calcule directement le contraire sans le taux de saut, ce qui peut augmenter la probabilité de détecter efficacement de meilleures régions et réduire le paramétrage complexe, en particulier pour le problème du monde réel. Cette stratégie est employée dans le but d'accélérer la convergence lorsqu'il n'y a pas de connaissance a priori sur les solutions, atteignant ainsi de meilleures solutions plus rapidement.

IV.8.2 Saut de génération basé sur l'opposition

Afin d'améliorer la convergence globale et d'éviter les solutions sous-optimales, la technique OBL est réappliquée à la population actuelle. À ce stade, si la condition de saut est satisfaite :

$$j_r\ rand() \leq -\left(\frac{g}{g_{max}}\right)^2 + 2\left(\frac{g}{g_{max}}\right) \qquad \text{(IV.16)}$$

Où Gen et G_{max} sont respectivement les générations actuelle et maximale (Luong et al., 2018). La population d'opposition correspondante est calculée en forçant le courant un à passer à une nouvelle solution. Après cela, les individus NP les plus aptes sont choisis parmi la population combinée de la population actuelle et de l'opposition comme population actuelle pour la prochaine génération. Contrairement au processus de la phase d'initialisation basée sur l'opposition, le saut de génération calcule la population d'opposition de manière dynamique. Au lieu d'utiliser les limites d'intervalle prédéfinies des variables ($[LB_j, UB_j]$), le saut de génération calcule l'opposé de chaque variable en fonction des valeurs minimum (Min_j^p) et maximum (Max_j^P) pour cette variable dans la population actuelle.

$$X_{i,j}^{0,courant} = Min_j^p + Max_j^P - x_{i,j}^{courant} (j = 1, \dots, D; i = 1, \dots, NP) \qquad \text{(IV.17)}$$

En restant dans les limites statiques de l'intervalle des variables, nous sautons en dehors de l'espace de recherche déjà réduit et perdons la connaissance de l'espace réduit actuel (population convergée). Par conséquent, nous calculons des points opposés en utilisant l'intervalle actuel de variables dans la population qui est, au fur et à mesure de la recherche, de plus en plus petite que la plage initiale correspondante.

L'organigramme ci-dessous (figure 0.3) represente en detail les deux phases de l'algorithme hybride proposé MODE-OBL.

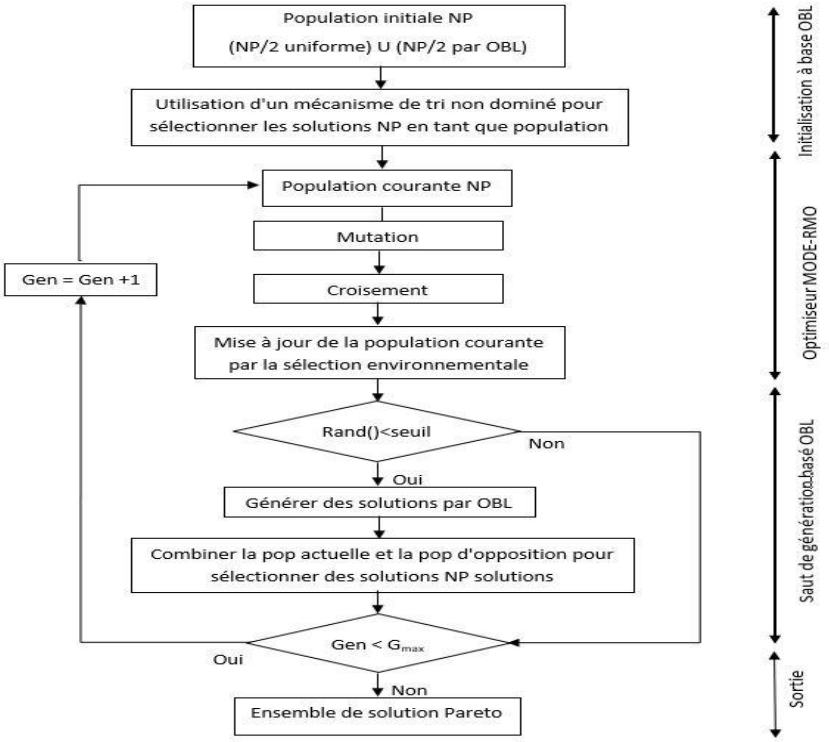

Figure 0-3 Organigramme de l'algorithme hybride proposé MODE-OBL

IV.9 RÉSULTATS EXPÉRIMENTAUX ET COMPARAISONS

Cinq algorithmes de pointe, à savoir NSGA-II, MOPSO, NSDE, MOEA/D-DE et MODE-RMO ont été choisis pour la comparaison des performances avec MODE-OBL proposé. NSGA-II (Deb, Pratap, et al., 2002b) est un algorithme populaire dans l'optimisation évolutionnaire multi-objectif car il a la capacité de réaliser des solutions prometteuses pour la plupart des MOOP. Cet algorithme utilise le classement Pareto et la distance de surpopulation comme opérateurs d'affectation de fitness, sélection de tournois binaires, croisement uniforme, mutation bit-flip et archivage parent-progéniture. L'évolution différentielle de tri non dominée (NSDE) (Angira & Babu, 2005) est une extension de l'évolution différentielle de base pour répondre à l'optimisation multi-objective. Il adopte les techniques de tri, de classement et d'élitisme non dominées trouvées dans NSGA-II, mais la principale différence entre elles est que le NSDE utilise l'opérateur de mutation d'évolution différentielle au lieu de l'opérateur SBX. Pour MOEA/D-DE (H. Li & Zhang, 2009), c'est un algorithme évolutionnaire qui décomposent n'importe quel MOP donné en un certain nombre de sous-problèmes à objectif unique. Chaque sous-problème est optimisé simultanément pendant le processus de recherche évolutive. Pour la décomposition du MOOP, l'approche de Tchebycheff est utilisée dans cet algorithme et la différence entre MOEA/D-SBX et MOEA/D-DE réside dans leurs opérateurs génétiques par lesquels l'opérateur de croisement SBX est utilisé avec une mutation polynomiale pour MOEA/D-SBX tandis que MOEA/D-DE utilise le croisement DE/rand/1 avec mutation polynomiale. Enfin pour le MOPSO (Carlos A. Coello Coello & Lechuga, 2002), il est étendu à partir de l'algorithme d'optimisation des essaims de particules (PSO), et il combine essentiellement les caractéristiques fortes de PSO. Pour des comparaisons justes, tous les paramètres des algorithmes comparés sont définis sur les valeurs recommandées comme dans leurs articles originaux.

Dans les sous-sections suivantes, les problèmes de test et les indicateurs de qualité utilisés dans nos expériences comparatives sont d'abord présentés. Ensuite, les paramètres expérimentaux adoptés dans cette étude sont fournis. De plus, trente essais indépendants sont exécutés pour chaque problème de test afin d'éviter le phénomène stochastique, et le test de somme de rang de Wilcoxon est adopté avec niveau de signification de 0.05 pour comparer les résultats obtenus par MODE-OBL et les cinq algorithmes comparés pour déterminer si l'algorithme le plus performant diffère des résultats des concurrents d'une manière statistiquement significative. Où les symboles «+», «-» et «≈» indiquent que le résultat est significativement meilleur, significativement pire et statistiquement similaire à celui obtenu par le MODE-OBL, respectivement.

IV.9.1 les problèmes de tests

Dans cette section, Un total de 12 problèmes de tests de références ont été choisis pour tester les performances d'optimisation de l'algorithme hybride proposé MODE-OBL en termes de convergence vers le vrai front de Pareto ainsi que la capacité à maintenir un ensemble de solutions diverses. Les problèmes de test utilisés comprenaient des problèmes ZDT (E. Zitzler, et al 2000) et DTLZ (Deb, et al 2005). Pour les problèmes de test, ils peuvent posséder deux, trois ou cinq fonctions objectives et avoir un nombre évolutif de variables de décision. Ces problèmes ont été choisis car ils couvrent différentes caractéristiques de l'optimisation multi-objectifs, à savoir le front de Pareto convexe, le front de Pareto non convexe, le front de Pareto discret, la multimodalité et la non-uniformité de la distribution des solutions. La présence de ces caractéristiques pourra poser des défis à un algorithme d'optimisation multi-objectif.

Tableau 0-2 Problèmes de test multi-objectifs. S (scalabilité), M (le nombre de fonctions objectives), K (paramètre scalaire), N (le nombre de variables de décision), SP (séparable), NS (non-séparable). (E. Zitzler, et al 2000) (Deb, et al 2005)

Instance	M	N	Intervalle	Géométrie	SP/NS	U/M
ZDT1	2	30(S)	$[0,1]^n$	Convexe	SP	U
ZDT2	2	30(S)	$[0,1]^n$	Concave	SP	U
ZDT3	2	30(S)	$[0,1]^n$	Déconnecté	SP	M
ZDT4	2	30(S)	$[0,1]^n \times [-5,5]^{n-1}$	Convexe	SP	M
ZDT6	2	30(S)	$[0,1]^n$	Concave	SP	M
DTLZ1	3(S)	$\boldsymbol{m} + \boldsymbol{K} - 1(S)$	$[0,1]^n$	Linéaire	SP	M
DTLZ2	3(S)	$\boldsymbol{m} + \boldsymbol{K} - 1(S)$	$[0,1]^n$	Concave	SP	U
DTLZ3	3(S)	$\boldsymbol{m} + \boldsymbol{K} - 1(S)$	$[0,1]^n$	Concave	SP	M
DTLZ4	3(S)	$\boldsymbol{m} + \boldsymbol{K} - 1(S)$	$[0,1]^n$	Concave	SP	U
DTLZ5	3(S)	$\boldsymbol{m} + \boldsymbol{K} - 1(S)$	$[0,1]^n$	Dégénéré	NS	U
DTLZ6	3(S)	$\boldsymbol{m} + \boldsymbol{K} - 1(S)$	$[0,1]^n$	Dégénéré	NS	U
DTLZ7	3(S)	$\boldsymbol{m} + \boldsymbol{K} - 1(S)$	$[0,1]^n$	Déconnecté	SP	M

IV.9.2 Indicateurs de performance

Dans notre étude expérimentale, Afin de faire une comparaison équitable des différents algorithmes d'optimisation, des mesures de performance pertinentes et applicables aux objectifs d'optimisation de proximité et de distribution doivent être utilisées. Deux mesures largement utilisées sont choisies pour évaluer les performances de chaque algorithme, qui sont nommées la distance générationnelle (Van Veldhuizen & Lamont, 1998), et la distance générationnelle inversée (IGD) (Coello & Cortés, 2005). GD et IGD peuvent mesurer efficacement la convergence et la diversité des solutions obtenues respectivement. La convergence décrit le degré d'approximation du résultat obtenu par l'algorithme au vrai front de Pareto (PF). Plus la convergence de l'algorithme est forte, plus l'ensemble de solutions est proche de la vraie solution optimale et plus le résultat est précis. La distribution décrit les caractéristiques de distribution du résultat obtenu dans l'espace objectif. D'une part, les résultats devraient être distribués autant que possible sur l'ensemble du PF, et d'autre part, les résultats devraient être distribués aussi uniformément que possible. Plus la distribution de l'algorithme est forte, meilleure est la capacité d'exploration globale de l'algorithme.

Distance générationnelle (GD). La distance générationnelle (GD) est un indicateur de performance unaire qui est défini comme :

$$GD = \sqrt{\frac{\sum_{i=1}^{N} d(p^*,p)_i^2}{N}} \qquad \text{(IV.16)}$$

Où N est le nombre de solutions dans PF^*, $p \in PF$, $p^* \in PF^*$ et $d(p^*,p)_i$ est la distance euclidienne minimale dans l'espace objectif entre p^* et p pour chaque membre i. GD illustre la capacité de convergence de l'algorithme en mesurant la proximité entre le front optimal de Pareto et le front de Pareto évolué. Ainsi, une valeur inférieure de GD montre que le front de Pareto évolué est plus proche du front optimal de Pareto. Cet indicateur est une métrique représentative qui fournit une mesure quantitative de l'objectif de proximité d'optimisation multi-objectif

Distance générationnelle inversée (IGD). IGD est un indicateur unaire par lequel la distance de chaque solution dans le front de Pareto optimal au front de Pareto obtenu est calculée. Soit P $*$ un ensemble de solutions uniformément réparties dans l'espace objectif le long du front de Pareto. P est une approximation du PF, qui est obtenue par l'algorithme. IGD est décrit comme :

$$IGD(P,P^*) = \frac{\sum_{|P^*|}^{i=1} dist(P_i^*,P)}{|P^*|}, \qquad \text{(IV.17)}$$

Où $dist\ (P_i^*,P)$ est la distance euclidienne entre un point $x^* \in P^*$ et son plus proche voisin dans P, et $|P^*|$ est la cardinalité de P^*. On peut voir d'après la définition de l'IGD que, pour un grand $|P^*|$, il peut couvrir approximativement tout le front de Pareto, qui est un autre aspect de la métrique en termes de diversité.

IV.9.3 Résultats

Pour les autres algorithmes comparés, leurs réglages de paramètres utilisés dans cette étude ont suivi ceux utilisés dans leurs études originales. Les réglages expérimentaux et paramètres globaux sont résumés dans le tableau 0-3. La comparaison a été effectuée pour examiner leurs performances d'optimisation dans les problèmes de tests décrits précédemment. Tous les

algorithmes ont été implémentés en MATLAB et exécutés sur un ordinateur Intel® Core ™ i3 2,53 GHz avec une capacité de mémoire de 6 Go.

Tableau 0-3 valeurs des paramètres utilisées dans cette comparaison

Réglages des paramètres de comparaison	Valeurs des paramètres
Taille de la population pour tous les algorithmes	100 pour des problèmes avec 2 objectifs 300 pour des problèmes avec 3 objectifs 500 pour des problèmes avec 5 objectifs
Critère d'arrêt	50000 évaluations de problèmes avec 2 objectifs 150000 évaluations de problèmes avec 3 objectifs 250000 évaluations de problèmes avec 5 objectifs
Nombre de variables de décision pour les problèmes ZDT	300 pour ZDT1, ZDT2 et ZDT3, et 100 pour ZDT4 et ZDT6
Nombre de variables de décision pour les problèmes DTLZ	12 pour DTLZ1 et DTLZ3, et 120 pour tous les autres problèmes DTLZ
Nombres d'éxécutions indepandantes	30 pour chaque algorithme
Taux de mutation	1/n (où n est le nombre de variables de décision)
Taux de croisement pour NSGA-II et NSDE	0.8
Facteur d'échelle de mutation pour NSDE	0.5
Taille du voisinages pour les algorithmes MOEA/D	20
Poids d'inertie pour l'algorithme MOPSO	0.5
(C_1, C_2) coéfficients pour l'algorithme MOPSO	C_1 = 1 ; C_2=2

Des études comparatives ont été menées pour l'évaluation des performances des six algorithmes dans le cadre d'une suite complète de fonctions de test de référence. Les résultats de simulation en termes de mesure des valeurs moyennes et écart type de la distance générationnelle (GD) et la distance générationnelle inversée (IGD) sur 30 cycles de simulation sont présentés dans les tableaux IV.4, IV.5 et IV.6. Les parenthèses à côté des problèmes de test indiquent le nombre d'objectifs (M) et de variables de décision (D) pour les problèmes.

Les problèmes de test ZDT (E. Zitzler et al., 2000) sont un ensemble de problèmes d'optimisation bi-objectifs simples qui sont évolutifs dans le nombre de variables de décision et ont différentes caractéristiques dans le front optimal de Pareto telles que la convexité, la concavité, la discontinuité, l'optimalité locale et la non-uniformité. Comme la plupart des algorithmes évolutionnaires sont capables de résoudre les problèmes ZDT sans difficultés, le nombre de variables de décision a été fixé à dix fois ses paramètres d'origine dans cette étude. Cela poserait alors de plus grands défis aux algorithmes en raison d'un plus grand espace de recherche. Les résultats indiquent que MODE-OBL a les meilleures performances globales. Une réalisation notable de MODE-OBL est sa capacité à atteindre la convergence pour ZDT4 alors que tous les autres algorithmes ne peuvent pas le faire pour ce problème. Le problème ZDT4 est un problème extrêmement multimodal avec la présence de nombreux optima locaux,

et il est donc probable que les autres algorithmes aient rencontré des difficultés en étant piégés dans les optima locaux. La bonne performance globale obtenue par MODE-OBL peut être attribuée aux effets complémentaires du DE basé sur l'opposition.

Tableau 0-4 Résultats obtenus par les algorithmes pour les problèmes ZDT

Problème de test	Algorithme	IGD	GD
ZDT1(2,300)	NSGA-II	0.118 (7.5e-03) -	0.066 (3.1e-03) -
	MOPSO	0.072 (8.9e-03) -	0.488 (1.5e-01) -
	NSDE	0.716 (1.9e-01) -	0.286 (4.9e-02) -
	MOEA/D-DE	0.339 (4.4e-02) -	0.189 (4.5e-02) -
	MODE-RMO	0.048 (1.1e-03) -	0.054 (3.3e-03) -
	MODE-OBL	**0.047 (6.8e-04)**	**0.052 (3.1e-04)**
ZDT2(2,300)	NSGA-II	0.079 (1.5e-02) -	0.366 (1.9e-02) -
	MOPSO	0.073 (1.8e-03) -	0.652 (7.1e-02) -
	NSDE	0.060 (2.2e-03) ≈	0.400 (2.2e-02) -
	MOEA/D-DE	0.099 (2.9e-03) -	2.889 (1.1e-01) -
	MODE-RMO	0.232 (6.8e-02) -	1.594 (2.1e-01) -
	MODE-OBL	**0.059 (1.0e-02)**	**0.339 (1.8e-02)**
ZDT3(2,300)	NSGA-II	0.447 (2.3e-02) +	0.507 (7.5e-01) -
	MOPSO	0.587 (7.1e-02) -	0.714 (4.9e-01) -
	NSDE	0.810 (1.2e-01) -	0.870 (1.2e-01) -
	MOEA/D-DE	0.669 (2.6e-01) -	0.402 (3.8e-01) -
	MODE-RMO	0.454 (8.2e-02) -	0.859 (1.4e-01) -
	MODE-OBL	**0.429 (5.4e-02)**	**0.283 (3.8e-03)**
ZDT4(2,100)	NSGA-II	0.053 (4.2e-03) -	0.011 (1.3e-03) -
	MOPSO	0.076 (4.6e-04) -	0.347 (2.0e-02) -
	NSDE	0.067 (1.2e-03) -	0.021 (2.6e-03) -
	MOEA/D-DE	0.399 (2.0e-02) -	0.742 (1.0e-05) -
	MODE-RMO	0.067 (7.3e-03) -	0.024 (5.0e-03) -
	MODE-OBL	**0.048 (7.0e-04)**	**0.006 (3.8e-05)**
ZDT6(2,100)	NSGA-II	0.034 (1.1e-03) ≈	0.101 (1.2e-02) -
	MOPSO	0.047 (4.1e-04) -	0.143 (6.4e-03) -
	NSDE	0.090 (1.9e-02) -	0.087 (2.9e-03) -
	MOEA/D-DE	0.321 (5.5e-02) -	1.156 (3.3e-01) -
	MODE-RMO	**0.034 (6.3e-04)** +	**0.068 (1.5e-03)** +
	MODE-OBL	0.042 (6.4e-04)	0.108 (9.8e-03)

La suite de problèmes DTLZ créés par (Deb et al., 2002) est extensible à n'importe quel nombre d'objectifs et de variables de décision. Par conséquent, pour cette étude, les problèmes DTLZ consistaient en trois ou cinq fonctions objectives. Le nombre de variables de décision dans DTLZ1 et DTLZ3 a été fixé à 12 car elles sont des problèmes hautement multimodaux et donc plus difficiles. Pour les autres problèmes DTLZ, ils sont généralement plus faciles à résoudre et le nombre de variables de décision a donc été fixé à 120 à la place. Pour le cas de problèmes DTLZ avec trois fonctions objectives, MODE-OBL atteint des performances compétitives par rapport aux autres algorithmes de cette étude. D'après les résultats de la simulation, MODE-OBL atteint les valeurs IGD et GD les plus faibles ou se rapproche des meilleures valeurs pour la plupart des problèmes DTLZ. Cependant, pour le cas de DTLZ5 et DTLZ6, MODE-OBL ne s'est pas aussi bien comporté par rapport aux autres algorithmes. Pour DTLZ5, on observe que l'algorithme NSGA-II qui incorpore l'utilisation de l'opérateur SBX ainsi que MOPSO donnent de meilleurs résultats par rapport aux algorithmes avec l'opérateur DE. Cela suggère que l'utilisation de l'opérateur DE peut ne pas être aussi puissante pour résoudre les problèmes de front de Pareto dégénéré.

Dans les problèmes DTLZ avec cinq fonctions objectives, les résultats démontrent que MODE-OBL atteint les meilleures performances globales pour DTLZ1, DTLZ3 et DTLZ7 lorsqu'il est opposé à tous les algorithmes comparés. Pour DTLZ2, MODE-OBL atteint la valeur IGD la plus basse mais pas pour la métrique GD. Pour DTLZ4, DTLZ5 et DTLZ6, on observe que l'algorithme basés sur la décomposition affichent généralement de meilleures performances en termes de meilleures valeurs IGD et GD que les autres pour ces trois problèmes. Cela démontre la meilleure capacité des algorithmes basés sur la décomposition à résoudre des problèmes multi-objectifs par rapport aux algorithmes basés sur la domination. Cela est attribué au fait que les algorithmes basés sur la décomposition permettent une meilleure sélection de solutions prometteuses en utilisant des valeurs de fitness agrégées. Pour les autres algorithmes basés sur la domination de cette étude, le comportement de domination entre les solutions doit être déterminé avant de décider quelles sont les solutions supérieures. Cependant, à mesure que le nombre de fonctions objectives augmentera, le comportement de domination sera affaibli. Par conséquent, il sera plus difficile pour les algorithmes basés sur la domination de sélectionner les meilleures solutions dans un espace objectif supérieur.

On observe que MODE-OBL affiche généralement de meilleures performances pour DTLZ1, DTLZ3 et DTLZ7, et des performances compétitives pour DTLZ2, avec trois et cinq fonctions objectives. DTLZ1 et DTLZ3 sont des problèmes hautement multimodaux, et le succès de MODE-OBL dans la gestion de ces problèmes est probablement attribué aux fortes capacités d'exploration inhérentes à son opérateur DE qui permettent à l'algorithme de s'échapper de l'optimal local. Comme pour DTLZ2, la phase de saut de génération basé sur l'opposition dans MODE-OBL aide à produire une pression de sélection adéquate vers le grand front de Pareto sphérique dans le domaine objectif de grande dimension. La bonne performance montrée par MODE-OBL pour DTLZ7 pourrait également être attribuée à la forte nature exploratoire de son opérateur DE complétée par l'opérateur de mutation basé sur le classement (RMO) car cela aide l'algorithme à découvrir les sous-populations distribuées dans toutes les régions Pareto-optimales déconnectées. De plus, la méthode de sélection environnementale utilisée est également efficace pour maintenir les solutions trouvées dans les régions Pareto-optimales déconnectées. Ces facteurs peuvent expliquer pourquoi MODE-OBL est capable de bien gérer le problème DTLZ7.

Tableau 0-5 Résultats obtenus par les algorithmes pour les problèmes DTLZ (3 objectifs)

Problème de test	Algorithme	IGD	GD
DTLZ1(3,12)	NSGA-II	0.200 (2.3e-02) -	2.802 (9.8e-01) -
	MOPSO	0.198 (3.0e-02) -	2.711 (9.2e-02) +
	NSDE	0.254 (7.1e-02) -	4.069 (3.0e-01) -
	MOEA/D-DE	0.764 (9.1e-02) -	3.147 (1.3e-01) -
	MODE-RMO	0.439 (2.8e-02) -	3.716 (1.6e+00) -
	MODE-OBL	**0.151 (4.9e-03)**	**2.431 (5.6e-02)**
DTLZ2(3,120)	NSGA-II	0.201 (1.5e-01) -	0.012 (1.9e-03) -
	MOPSO	0.178 (8.4e-04) -	**0.006 (1.4e-05)** +
	NSDE	0.2719 (5.1e-01) -	2.130 (5.7e+00) -
	MOEA/D-DE	**0.153 (1.1e-03)** +	0.049 (1.4e-02) -
	MODE-RMO	0.254 (4.3e-02) -	0.742 (5.8e-05) -
	MODE-OBL	0.154 (8.8e-03)	0.111 (2.4e-02)
DTLZ3(3,12)	NSGA-II	0.766 (4.0e-02) -	0.3013 (2.1e-01) -
	MOPSO	1.288 (4.1e-01) -	**0.2931 (3.7e-02)** +
	NSDE	0.948 (2.1e-02) -	1.0623 (6.5e-01) -
	MOEA/D-DE	**0.519 (1.0e-01)** +	0.4865 (2.4e-02) -
	MODE-RMO	0.877 (3.5e-01) -	0.8574 (9.2e-01) -
	MODE-OBL	0.565 (1.2e-01)	0.3021 (5.8e-01)

DTLZ4(3,120)	NSGA-II	**0.146 (1.5e-03)** +	0.387 (5.0e-04) -
	MOPSO	0.394 (2.9e-02) -	0.368 (1.1e-02) ≈
	NSDE	0.463 (8.5e-02) -	2.648 (4.1e-01) -
	MOEA/D-DE	0.180 (3.7e-02) -	0.391 (2.1e-02) -
	MODE-RMO	0.185 (8.2e-05) -	0.431 (5.2e-02) -
	MODE-OBL	0.158 (6.9e-03)	**0.368 (8.2e-03)**
DTLZ5(3,120)	NSGA-II	0.100 (1.5e-03) +	0.253 (5.3e-03) -
	MOPSO	0.177 (5.7e-03) -	0.046 (3.2e-03) +
	NSDE	0.537 (6.8e-03) -	0.134 (6.6e-01) -
	MOEA/D-DE	**0.099 (4.6e-06)** +	**0.026 (1.5e-02)** +
	MODE-RMO	0.231 (1.6e-02) -	0.668 (8.1e-01) -
	MODE-OBL	0.124 (2.7e-03)	0.211 (4.7e-03)
DTLZ6(3,120)	NSGA-II	0.413 (7.1e-02) -	1.874 (6.3e-01) -
	MOPSO	9.086 (4.8e+00) -	1.272 (3.6e-02) -
	NSDE	0.201 (4.1e-02) -	2.063 (6.1e-02) -
	MOEA/D-DE	0.112 (3.1e-03) -	**0.246 (1.3e-03)** +
	MODE-RMO	0.113 (2.5e-03) -	1.346 (5.5e-02) -
	MODE-OBL	**0.099 (2.9e-03)**	1.080 (7.3e-03)
DTLZ7(3,120)	NSGA-II	**2.547 (2.1e-01)** +	0.813 (3.8e-01) -
	MOPSO	11.18 (9.6e+00) -	0.791 (4.2e-02) -
	NSDE	18.07 (2.5e+02) -	0.955 (2.1e-01) -
	MOEA/D-DE	6.061 (6.1e-01) -	0.744 (6.4e-01) -
	MODE-RMO	7.645 (1.1e+00) -	0.669 (8.6e-02) -
	MODE-OBL	2.614 (7.7e-02)	**0.668 (2.2e-03)**

Tableau 0-6 Résultats obtenus par les algorithmes pour les problèmes DTLZ (5 objectifs)

Problème de test	Algorithme	IGD	GD
DTLZ1(5,12)	NSGA-II	1.355 (2.6e-02) -	1.814 (1.6e-02) -
	MOPSO	8.413 (3.7e+00) -	3.297 (1.0e+00) -
	NSDE	1.686 (1.1e-01) -	2.278 (3.2e-01) -
	MOEA/D-DE	1.344 (5.0e-06) -	1.745 (1.4e-02) -
	MODE-RMO	1.567 (5.7e-02) -	1.793 (2.2e-02) -
	MODE-OBL	**1.227 (3.8e-03)**	**1.726 (2.5e-02)** +
DTLZ2(5,120)	NSGA-II	0.410 (5.9e-02) -	0.194 (7.2e-01) -
	MOPSO	1.397 (5.0e-05) -	0.438 (3.2e-02) -
	NSDE	3.561 (3.2e-02) -	4.152 (4.2e-01) -
	MOEA/D-DE	0.257 (1.1e-03) -	**0.156 (1.2e-03)** +
	MODE-RMO	0.658 (7.4e-02) -	0.306 (4.6e-02) -
	MODE-OBL	**0.224 (4.7e-03)**	0.218 (2.1e-02) -
DTLZ3(5,12)	NSGA-II	1.548 (5.9e-02) -	2.852 (4.4e-03) -
	MOPSO	3.157 (2.5e+00) -	4.043 (8.7e-02) -
	NSDE	4.699 (2.1e+00) -	2.973 (2.2e+00) -
	MOEA/D-DE	0.881 (1.1e-01) -	**1.663 (2.1e-02)** +
	MODE-RMO	1.978 (4.1e-01) -	2.528 (5.8e-02) -
	MODE-OBL	**0.682 (2.1e-01)**	1.906 (4.5e-03) -
DTLZ4(5,120)	NSGA-II	2.245 (8.9e-03) -	0.956 (1.4e-02) -
	MOPSO	4.398 (2.2e-02) -	1.030 (8.0e-02) -
	NSDE	6.544 (3.9e-02) -	3.680 (8.7e-01) -
	MOEA/D-DE	**2.224 (1.4e-03)** +	**0.899 (2.0e-02)** +
	MODE-RMO	2.365 (1.1e-02) -	3.258 (3.9e-01) -
	MODE-OBL	2.330 (1.6e-02) -	1.269 (4.4e-02) -
DTLZ5(5,120)	NSGA-II	1.267 (1.4e-02) -	0.475 (1.2e-01) -
	MOPSO	3.441 (4.7e-02) -	8.762 (5e+05) -
	NSDE	2.811 (4.3e-02) -	4.75 (1.4e-01) -
	MOEA/D-DE	**1.169 (5.0e-03)** +	**0.097 (5.7e-02)** +
	MODE-RMO	1.345 (2.8e-02) -	1.682 (5.7e+01) -

	MODE-OBL	1.219 (1.0e-02) -	1.104 (2.0e-03) -
DTLZ6(5,120)	NSGA-II	2.2e+09 (6e+09) -	15.735 (1.1e+00) -
	MOPSO	5.9e+10 (3e+10) -	17.512 (3.7e+01) -
	NSDE	2.946 (2.7e+01) -	17.23 (1.3e+02) -
	MOEA/D-DE	1.061 (5.5e-02) -	**4.98 (2.1e+00) +**
	MODE-RMO	1.402 (8.5e-02) -	12.887 (3.2e+00) -
	MODE-OBL	**0.927 (2.8e-03)**	9.216 (6.2e+00) -
DTLZ7(5,120)	NSGA-II	1.498 (1.7e-03) -	2.565 (4.6e-01) -
	MOPSO	1.463 (4.3e-01) -	7.127 (3.8e+02) -
	NSDE	12.02 (2.7e+01) -	7.982 (2.2e-03) -
	MOEA/D-DE	1.320 (2.3e-02) -	3.287 (6.8e-00) -
	MODE-RMO	1.946 (4.3e-01) -	3.889 (1.4e+01) -
	MODE-OBL	**1.053 (4.0e-03)**	**1.996 (5.2e-01)**

IV.10 OPTIMISATION DE PORTEFEUILLE À GRANDE ÉCHELLE

La finance computationnelle est un domaine d'application émergent des algorithmes métaheuristiques. Ces méthodes d'optimisation deviennent l'alternative d'approche de résolution lorsqu'il s'agit de versions réalistes de plusieurs problèmes de prise de décision en finance. Le problème de sélection de portefeuille peut être défini comme la répartition optimale de la richesse entre un nombre fini d'actifs qui fait suite à un traitement minutieux de toutes les informations disponibles sur les investisseurs et les marchés. Le modèle de variance moyenne de Markowitz est de loin la procédure la plus répandue en matière d'allocation d'actifs (Guerard, 2016).

Il existe quelques concepts clés dans l'optimisation de portefeuille. Premièrement, la récompense et le risque sont mesurés par le rendement attendu et la variance d'un portefeuille. Le rendement attendu est calculé sur la base du rendement historique d'un actif et la variance est une mesure de la dispersion des rendements. Deuxièmement, les investisseurs sont exposés à deux types de risques : le risque non systématique et le risque systématique. Le risque non systématique est le risque intrinsèque d'un actif qui peut être diversifié en détenant un grand nombre d'actifs. Ces risques ne présentent pas suffisamment d'informations sur le risque global de l'ensemble du portefeuille. Le risque systématique, ou risque de portefeuille, est le risque généralement associé au marché qui ne peut être éliminé. Troisièmement, la covariance entre les différents rendements des actifs donne la variabilité ou le risque d'un portefeuille. Par conséquent, un portefeuille bien diversifié contient des actifs qui ont des corrélations faibles ou négatives (Duan, 2004).

Les AE fonctionnent avec un ensemble de solutions, appelé population. Cette fonction est particulièrement adaptée à la résolution de problèmes multi-objectifs, car elle permet d'approcher la frontière efficace en un seul passage. En conséquence, les algorithmes évolutifs multi-objectifs (MOEA) ont reçu une attention croissante pour les applications financières (Ponsich et al, 2013). En fait, l'optimisation de portefeuille a été l'une des premières applications réussies des MOEA en économie et en finance.

La complexité croissante des applications pratiques a conduit les chercheurs à développer des procédures heuristiques pour résoudre leurs problèmes d'optimisation de portefeuille. Ces techniques nécessitent moins d'informations de domaine à prendre en compte que les méthodes de programmation mathématique standard basées sur le gradient. De plus, ils garantissent des approximations satisfaisantes des solutions dans un temps de calcul juste même lorsqu'ils traitent des variables de non-convexité, de discontinuité et de décision entière. Les approches qui ont été proposées dans la littérature sur l'informatique douce peuvent être classées dans les

deux groupes suivants. D'une part, les méthodes à objectif unique optimisent une somme pondérée des objectifs du portefeuille. D'autre part, les algorithmes évolutifs multi-objectifs (MOEA) tentent de s'attaquer directement au problème d'allocation sous sa forme multi-objectifs en optimisant simultanément le risque et la récompense. Dans le premier cas, l'ensemble complet des profils risque-rendement est obtenu en faisant varier un paramètre qui représente l'aversion au risque de l'investisseur (Cura, 2009) (Woodside-Oriakhi et al, 2011). Dans le second cas, la frontière efficace complète est représentée en un seul passage (Meghwani & Thakur, 2017) (Mishra et al, 2014). Les deux catégories accordent une grande attention aux types de codage et aux techniques de gestion des contraintes (Metaxiotis & Liagkouras, 2015).

Alors que les méthodes d'optimisation à objectif unique considèrent soit un risque minimal pour un rendement donné, soit un risque maximal pour un rendement attendu donné ou une fonction objectif qui pondère les deux objectifs et qui doivent donc être exécutées plusieurs fois avec les pondérations respectives (Pai, 2019), Les méthodes d'optimisation multi-objectifs trouvent un ensemble de solutions Pareto, tout en équilibrant simultanément deux ou plusieurs fonctions objectives.

IV.10.1 Formulation du problème

La clé pour atteindre les objectifs des investisseurs est de fournir une stratégie de portefeuille optimale qui montre aux investisseurs combien investir dans chaque actif d'un portefeuille donné. Par conséquent, la variable de décision des problèmes d'optimisation de portefeuille est le vecteur de pondération des actifs $\bar{x} = [x_1, x_2, ...x_n]^T$ avec comme poids de l'actif dans le portefeuille. Le rendement attendu de chaque actif du portefeuille est exprimé sous forme vectorielle $\bar{p} = [p_1, p_2, ...p_n]^T$ avec p_i comme rendement moyen de l'actif. Le rendement attendu du portefeuille est la moyenne pondérée du rendement de l'actif individuel (Eq IV.19) La variance et la covariance de l'actif individuel sont caractérisées par une matrice de variance-covariance $v = \begin{bmatrix} \sigma_{11} & \cdots & \sigma_{1n} \\ \vdots & \ddots & \vdots \\ \sigma_{1n} & \cdots & \sigma_{nn} \end{bmatrix}$, où $\sigma_{i,i}$ est la variance de l'actif et la covariance entre l'actif i et $\sigma_{i,j}$ est la covariance entre l'actif i et l'actif j. La variance du portefeuille est définie dans l'équation (IV.18).

La représentation mathématique de l'optimisation de portefeuille a été introduite par Markowitz dans les années 50 et il a été récompensé par un prix Nobel d'économie en 1990 (Markowitz, 2014). Le modèle de Markowitz suppose que les investisseurs souhaitent maximiser le rendement sous un certain niveau de risque ou minimiser le risque avec un certain niveau de rendement et ce modèle utilise la moyenne et la variance du prix historique normalisé des actifs pour mesurer le rendement et le risque attendus du portefeuille (R. Kumar, 2016). Le modèle peut être exprimé comme un problème bi-objectif et formulé comme suit :

$$MIN : \text{Risk } \sigma^2 = \sum_{i+1}^{n} \sum_{j+1}^{n} w_i w_j \sigma_{ij} \qquad \text{(IV.18)}$$

$$Max : \text{Return } r_p = \sum_{i+1}^{n} \sum_{j+1}^{n} w_i r_i \qquad \text{(IV.19)}$$

$$\text{Subject to}: \sum_{i+1}^{n} w_i = 1; w_i \geq 0 \qquad \text{(IV.20)}$$

Où n est le nombre d'actifs dans le portefeuille et la dimension du problème, w_i est le poids de l'actif $i^{éme}$. σ^2 représente le risque de portefeuille et σ_{ij} représente la covariance entre l'actif i et l'actif j. Si $i = j$, σ_{ij} est seulement la variance de cet actif particulier. r_p est le rendement du portefeuille, tandis que r_i est le rendement individuel de l'actif i.

IV.10.2 Paramètres expérimentaux et dataset

Toutes les expériences sont menées à l'aide du modèle standard de Markowitz (moyenne-variance). 100 jours de cours de clôture de 100 et de 500 actions ont été téléchargés et utilisés comme données de stock historiques et aussi utilisés comme données de test dans la simulation. Les rendements mensuels et les prix de clôture des actions sont sélectionnés à partir de (P-N-Suganthan, 2017). Un calcul de rendement mensuel est formulé comme suit : $M_t = \ln \frac{P_t}{P_{t-1}}$; Où $P(t)$ est le cours de clôture, $P(t-1)$ le cours de clôture de la veille et $M(t)$ le rendement mensuel. Des expériences sont menées sur 100 et 500 stocks. Ainsi, la taille des chromosomes est de 100 et 500, respectivement. Et le nombre d'évaluation des fonctions est respectivement de 100 000 et 300 000 pour les trois algorithmes et tous les algorithmes sont exécutés 25 fois avec une initialisation aléatoire.

À l'exception de MODE-OBL, deux autres algorithmes évolutifs multi-objectifs sont également testés sur ce problème d'optimisation de portefeuille pour comparaison en raison de leur supériorité respectée par rapport aux métriques GD et IGD dans différents problèmes de benchmarks testés précédemment (section IV.9). Pour NSGA-II codé en réel, une taille de population de 100 est utilisée, une probabilité de croisement de 0,9 et une probabilité de mutation de 1/n, où *n* est le nombre de variables de décision, les indices de distribution pour les opérateurs de croisement et de mutation comme présenté dans (Lin, 2012). MOEA/D-DE utilise une taille de population de 100, *F* est réglé sur 0,5 et *CR* est réglé sur 0,1.

Pour vérifier la robustesse des résultats, 25 simulations pour chaque algorithme et pour chaque problème de test sont utilisées. Les algorithmes sont implémentés dans MATLAB R2019b et les expériences sont réalisées sur un ordinateur portable Intel Core i5 7300U 2,6 GHz avec 6 Go de RAM.

IV.10.3 Résultats expérimentaux et analyse

Un ensemble de solutions non dominées générées par des optimiseurs peut être mesuré avec différentes mesures de performance disponibles dans la littérature. Dans le cas où nous n'avons pas d'informations sur le vrai front de Pareto, alors la mesure du coefficient de variation CV qui permet de déterminer combien de volatilité, ou de risque, est supposé par rapport au montant de rendement attendu des investissements selon l'équation (III.2) et la métrique d'espacement (SP) suggérée par (Schott, 1995) est calculée avec une mesure de distance relative entre solutions consécutives dans l'ensemble non dominé obtenu, comme suit :

$$SP = \sqrt{\frac{1}{|S|-1}} \sum_{i=1}^{|S|} \left(\bar{d} - d_i\right)^2 \qquad \text{(IV.21)}$$

Où $d_i = min_{(s_i,s_j)\in S,\ s_i \neq s_j} \left\|F(s_i) - F(s_j)\right\|_1$ est la distance $l_1$entre un point $s_i \in S$ et le point le plus proche de l'approximation du front de Pareto produite par le même algorithme, et $\bar{d}$ est la moyenne du d_i.

D'après les résultats obtenus dans le tableau IV.7, on observe que MODE-OBL obtient de meilleurs fronts optimaux de pareto avec une meilleure convergence et diversité par rapport à NSGA-II et MOEA/D-DE sur le problème des 100 stocks. Les différences entre les différents algorithmes sont très proches par rapport aux résultats obtenus en termes de métrique de distribution avec une légère supériorité de MODE-OBL. Notez que MOEA/D-DE a montré de bonnes performances lors de l'optimisation du portefeuille de 500 actions, indiquant que les MOEA basé sur l'approche de décomposition a une plus grande capacité à résoudre les problèmes d'optimisation à grande échelle.

Tableau 0-7 Les résultats de comparaison des algorithmes MODE-OBL, MOEA / D-DE, NSGA-II

Dataset	*Metrique*	*Statistique*	*MODE-OBL*	*MOEA/D-DE*	*NSGA-II*
100 stocks	Risk	Best Avg	**0.0281e-04(1.0019)** 0.0742e-04**(1.0093)**	0.0697e-04(1.0017) **0.0665e-04**(1.0087)	0.1351e-04(1.0016) 0.6894e-04(1.0083)
	Return	Best Avg	**1.1056(2.0036e-04)** **1.0093** (0.0742e-04)	1.1036(2.004e-04) 1.0087(**0.0665e-04**)	1.1036(2.004e-04) 1.0083(0.0689e-04)
	Coefficient de variation (CV)	Best Worst Avg	**6.7795e-03** **1.4993e-02** **1.1725e-02**	2.5005e-01 2.9526e-01 2.7482e-01	5.2053e-01 5.6592e-01 5.4216e-01
	Métrique D'espacement (SP)	Best Worst Avg	**6.9662e-06** **1.4376e-05** **1.0603e-05**	8.9935e-06 1.9869e-05 1.2816e-05	6.6883e-05 1.4525e-04 9.8367e-05
500 stocks	Risk	Best Avg	0.2107e-04(1.0023) **0.4890**e-**04(**1.0053**)**	**0.1890e04(1.0025)** 0.5156e-04(**1.0057)**	0.3772e-04(1.0024) 0.6135e-04 (1.0043)
	Return	Best Avg	1.0067(1.0956e-04) 1.0053**(0.4890e-04)**	**1.0071(1.0253**e-04**)** **1.0057(**0.5156e-04)	1.0051(1.2715 e-04) 1.0043(0.6135e-04)
	Coefficient de variation (CV)	Best Worst Avg	**1.3851e-02** 3.5104e-01 2.8757e-01	1.5036e-02 **3.4263e-01** **2.6981e-01**	5.6942e-01 8.9172e-01 6.7819e-01
	Métrique D'espacement (SP)	Best Worst Avg	2.1776e-04 **2.5627e-04** **2.3606e-04**	**1.4915e-04** 5.1583e-04 2.7524e-04	1.3038e-02 4.1017e-02 2.9133e-02

Une meilleure compréhension de la nature des solutions trouvées peut être obtenue en analysant les figures IV.4 et IV.5. Les graphiques présentés dans ces figures montrent les frontières efficaces. Comme on peut le voir sur ces graphiques, les meilleurs résultats (Risk/Return) sont générés par MOEA/D-DE sur 500 titres et quasi-cohérents avec les résultats obtenus par MODE-OBL sur 100 titres. Bien que NSGAII soit le moins performant, il peut toujours générer des fronts distribués et satisfaisants. Les expériences menées sur les problèmes d'optimisation de portefeuille avec 100 et 500 actions montrent que la mutation de classement basée sur l'algorithme différentiel multi-objectif combiné à l'apprentissage basé sur l'opposition a présenté une excellente performance par rapport à deux autres algorithmes évolutifs multi-objectifs et constitue une solution potentielle pour ce type de problèmes.

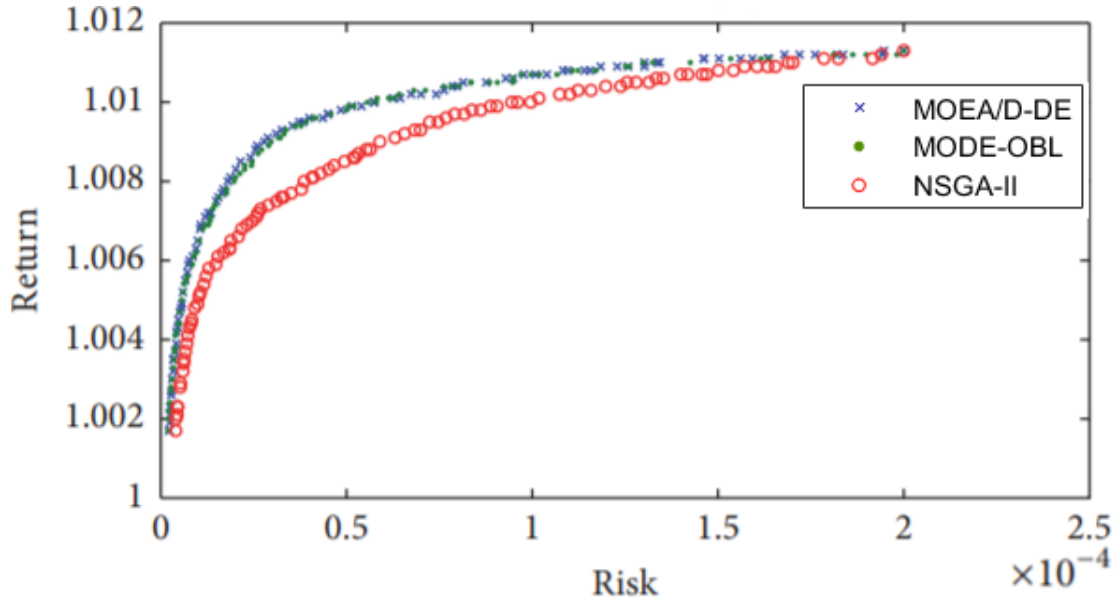

Figure 0-4 frontière de pareto générée par un problème de 100 stocks

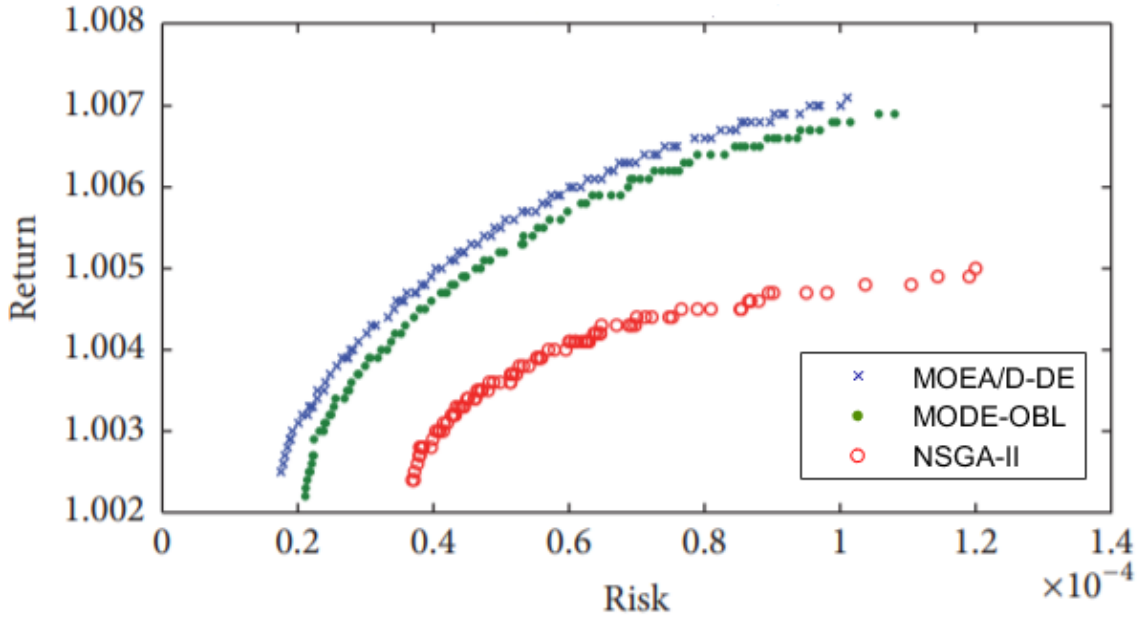

Figure 0-5 frontière de pareto générée par un problème de 500 stocks

IV.11 CONCLUSION

En vue de la complexité et de la dimensionnalité croissantes observées aujourd'hui dans plusieurs problèmes d'optimisation, il existe un fort besoin de trouver des moyens d'améliorer l'efficience et l'efficacité des algorithmes évolutionnaires. Avec des algorithmes hybride comme solution possible pour permettre une convergence rapide et une forte capacité de recherche globale avec robustesse, cela a motivé la proposition d'un nouvel algorithme hybride basé sur l'évolution différentielle dans le contexte de l'optimisation multi-objectif à présenter dans ce chapitre. Contrairement à la plupart des autres variantes d'évolution différentielle, MODE-OBL commence par une population initiale optimisé au début de l'évolution grâce à une stratégie d'apprentissage à base de la technique OBL qui réduit la complexité et guide la recherche vers des solutions efficace de l'espace objective. MODE-FMO basé sur une stratégie

de mutation de classement est également hybridé avec la recherche par opposition pour agir comme une forme de recherche qui contribue à améliorer les capacités d'exploitation de l'algorithme global. L'apprentissage basé sur l'opposition étant incorporé dans le processus de l’évolution, cela peut donner une probabilité plus élevée de trouver des solutions presque optimales et booster la recherche de MODE dans l'algorithme proposé tout au long du processus évolutionnaire. MODE-OBL sera en mesure de compléter les forces d'un nouvel algorithme d'évolution différentielle multi-objectif basé sur l'opposition, qui équilibre les capacités d'exploration et d'exploitation telles que trouvées dans le DE original, ainsi que OBL qui apporte une pression de sélection plus élevée. Grâce à la validation de MODE-OBL à l'aide d'une suite de problèmes de référence de test soigneusement sélectionnés pour une optimisation continue multi-objectif, il est observé que MODE-OBL obtient globalement de meilleures performances en termes de convergence et de diversité par rapport aux autres algorithmes comparés dans ce chapitre. De plus, MODE-OBL peut être recommandé pour résoudre des problèmes multimodaux ainsi que des problèmes avec des ensembles de Pareto complexes, comme en témoignent ses performances d'optimisation supérieures dans ces types de problèmes par rapport aux autres algorithmes de cette étude.

CONCLUSION GÉNÉRALE

Conclusion

Généralement, un problème complexe peut être exprimé sous la forme d'un problème d'optimisation, dans lequel on définit une fonction objective que l'on recherche à minimiser ou à maximiser selon le contexte. Différentes techniques de résolution ont été développées pour résoudre ces problèmes complexes où la nature et la théorie de l'évolution ont été toujours une source d'inspiration de nouvelles méthodes de calcul.

Les méthodes bio-inspirées ont prouvées leur efficacité pour résoudre des problèmes complexes divers, dans plusieurs domaines de recherche et d'ingénierie. Mais, malgré leur grand succès connu ces dernières décennies, plusieurs problèmes sont rencontrés : la convergence prématurée des solutions, le conflit des objectifs lors de l'évaluation des solutions, le réglage des paramètres et enfin la distinction entre une solution optimale locale et une autre globale qui est parfois difficile. D'autre part, la mise au point d'une méthode de recherche doit trouver un équilibre entre deux tendances opposées lors de la recherche dans l'espace des solutions : l'intensification et la diversification. L'intensification permet de se concentrer autour des meilleures solutions rencontrées, considérées comme prometteuses ; alors que la diversification incite la recherche à explorer des nouvelles zones de l'espace de recherche en vue d'y trouver de bonnes solutions.

L'apport de l'hybridation est totalement bénéfique : solutions améliorées et convergence plus rapide. C'est le résultat d'un équilibre entre l'exploration et l'exploitation de l'espace de recherche. En revanche, il ne faut pas ignorer le fait que l'hybridation nécessite des ressources en plus pour le calcul et par conséquent la complexité algorithmique peut être augmentée. Un autre problème pouvant aussi survenir est celui de la multiplication du nombre de paramètres à régler.

Les travaux de recherche présentés concernent le développement de nouvelles méthodes d'optimisation globale qui s'appuient sur les métaheuristiques. Nous avons focalisé nos recherches sur les algorithmes évolutionnaires dits stochastiques destinés à résoudre des problèmes d'optimisation difficiles mono et multi-objectifs.

Le travail de cette thèse a mis l'accent sur l'utilité de l'hybridation des méthodes évolutionnaires devant la complexité grandissante des problèmes actuels. Nous avons abordé en premier lieu, les motivations qui sont derrière ce choix, ainsi les différentes stratégies d'hybridation et les architectures correspondantes.

Notre première contribution est le couplage de l'algorithme des stratégies évolutionnaires auto-adaptative SA-ES avec la méthode directe du simplex. Le couplage de ces deux algorithmes est fait en hybridation série collaborative. Cette technique hybride intègre des concepts de SA-ES, du Simplex non-linéaire, du regroupement « clustering » et un critère de contraction pour trouver un équilibre entre exploration et exploitation. La recherche locale par la méthode directe « Nelder-Mead » n'est déclencher que si le critère de contraction est atteint grâce une mesure statistique de variation sur les espaces de décision et d'objective. Pour réduire le nombre de fonction d'évaluation, un nombre limité d'individu est choisi pour être intensifier après avoir regrouper les individus du même centre de masse.

L'approche hybride proposée est considérée comme un Framework adaptatif, efficace et flexible quel que soit l'algorithme évolutionnaire implémenter, il est capable à surmonter les points locaux et converge vers des solutions globales efficaces.

Pour sortir du cadre mono objectif et affronter les problèmes multi-objectifs la performance de l'algorithme développé a été valider par son adaptation au problème multi-objective de

finance « portfolio » en utilisant une approche classique d'agrégation à deux objectives, ce qui à donner des résultats meilleurs par rapport à des algorithmes évolutionnaires multi-objectifs en termes de qualité de solutions générer dans le front de Pareto.

La deuxième contribution concerne le couplage d'algorithme d'évolution différentielle multi-objectif MODE avec la méthode d'apprentissage par opposition OBL, cette nouvelle méthode a un avantage par rapport à la première contribution en raison des limites des méthodes classiques basé sur l'agrégation des objectives dans la recherche des ensembles Pareto-optimaux pour des problèmes difficiles avec des espaces de solutions non convexes et discontinus, ce qui nécessite plusieurs exécutions pour produire des solutions alternatives pour chaque objectif.

Cet avantage réside aussi au niveau d'une énorme minimisation de temps de calcul, du fait que la deuxième contribution utilisée qui appartient à la classe de l'approche postériori « MOEA » basé sur la dominance de Pareto pour la résolution de problèmes du monde réel en raison de la haute complexité informatique. L'approche posteriori permet de générer l'ensemble de front optimal par une seule exécution en recherchant simultanément différentes régions d'un espace de solution pour plus de deux objectifs.

Le concept de l'apprentissage par opposition OBL est intégrée d'abord à l'initialisation de la population de MODE afin d'orienter la recherche vers des zones de solutions non-dominées et prometteuses. La deuxième phase de l'approche proposée est également hybridée avec la recherche par opposition pour agir comme une forme de recherche locale qui contribue à améliorer les capacités d'exploitation de l'algorithme global.

Les résultats expérimentaux montrent que notre algorithme MODE-OBL est capable d'améliorer la vitesse de convergence et de générer des solutions qui se rapprochent du meilleur front de Pareto pour la majorité des Benchmarks en comparaison avec les algorithmes de l'état de l'art en dépit de l'augmentation de dimensionnalité et du nombre de fonction d'objectif.

En outre, MODE-OBL peut être recommandé pour résoudre des problèmes multimodaux ainsi que des problèmes avec des ensembles compliqués Pareto comme vu de ses performances d'optimisation supérieure dans de tels types de problèmes par rapport aux autres algorithmes dans cette étude.

Perspectives

Certes les méthodes présentées dans cette thèse ont donné des résultats remarquables et pertinent mais la recherche continue !

Sur la base des travaux de recherche menés dans cette thèse, certaines directions de recherche possibles qui méritent des investigations futures sont recommandées ici. Premièrement, des travaux supplémentaires peuvent être effectués pour examiner l'auto-adaptation de la taille de la population dans les algorithmes proposés dans cette thèse. De plus, si la population peut être adaptée à des paramètres optimaux tout au long du processus évolutif, cela peut conduire à la génération de meilleurs individus qui, à leur tour, entraîneront de meilleures performances d'optimisation.

Nous proposons d'hybrider dans le futur d'autres algorithmes heuristiques notamment les algorithmes de l'intelligence en essaim. Nous comptons travailler sur des problèmes multi-objectifs plus complexes pour bien améliorer les limites de nos méthodes.

Nous envisageons généraliser l'approche proposée pour d'autres problèmes académiques et réels avec introduction du parallélisme à plusieurs niveaux. Ceci va améliorer considérablement le temps de calcule et la qualité des solutions pour les décideurs.

BIBLIOGRAPHIE

Abbass, H. A., Sarker, R., & Newton, C. (2001). PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems. *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*. https://doi.org/10.1109/cec.2001.934295

Abdullah, A., Deris, S., Anwar, S., & Arjunan, S. N. V. (2013). An Evolutionary Firefly Algorithm for the Estimation of Nonlinear Biological Model Parameters. *PLoS ONE*. https://doi.org/10.1371/journal.pone.0056310

Abouhawwash, M., Seada, H., & Deb, K. (2017). Towards faster convergence of evolutionary multi-criterion optimization algorithms using Karush Kuhn Tucker optimality based local search. *Computers and Operations Research*. https://doi.org/10.1016/j.cor.2016.04.026

Ahandani, M. A., & Alavi-Rad, H. (2012). Opposition-based learning in the shuffled differential evolution algorithm. *Soft Computing*. https://doi.org/10.1007/s00500-012-0813-9

Ahandani, M. A., Vakil-Baghmisheh, M. T., & Talebi, M. (2014). Hybridizing local search algorithms for global optimization. *Computational Optimization and Applications*. https://doi.org/10.1007/s10589-014-9652-1

Ali, M. M. (2011). Differential evolution with generalized differentials. *Journal of Computational and Applied Mathematics*. https://doi.org/10.1016/j.cam.2010.10.018

Ali, M., Siarry, P., & Pant, M. (2012). An efficient Differential Evolution based algorithm for solving multi-objective optimization problems. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2011.09.025

Ali, M. Z., Awad, N. H., & Suganthan, P. N. (2015). Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2015.04.019

Andrew, A. M. (1998). Evolution and Optimum Seeking982Hans-Paul Schwefel. Evolution and Optimum Seeking . New York: Wiley-Interscience 1995. ix + 444 pp, ISBN: 0-471-57148-2 £70 hardback with disk Sixth-Generation Computer Technology Series . *Kybernetes*. https://doi.org/10.1108/k.1998.27.8.975.2

Angira, R., & Babu, B. V. (2005). Non-dominated sorting differential evolution (NSDE): An extension of differential evolution for multi-objective optimization. *Proceedings of the 2nd Indian International Conference on Artificial Intelligence, IICAI 2005*.

Antoniou, A., & Lu, W. S. (2007). Practical optimization: Algorithms and engineering applications. In *Practical Optimization: Algorithms and Engineering Applications*. https://doi.org/10.1007/978-0-387-71107-2

Applegate, R. Bixby, V. C. & W. C. (1998). On the Solution of Travelling Salesman Problems. *Documenta Mathematica*, *ICM III*, 645–656.

Argüello, M. F., Bard, J. F., & Yu, G. (1997). A GRASP for aircraft routing in response to groundings and delays. *Journal of Combinatorial Optimization*. https://doi.org/10.1023/A:1009772208981

Asafuddoula, M., Ray, T., & Sarker, R. (2014). An adaptive hybrid differential evolution algorithm for single objective optimization. *Applied Mathematics and Computation*. https://doi.org/10.1016/j.amc.2014.01.041

Astolfi, A. (2008). Optimization: An introduction. *Optimization in Food Engineering*.

Baatar, N., Jeong, K. Y., & Koh, C. S. (2014). Adaptive parameter controlling non-dominated

ranking differential evolution for multi-objective optimization of electromagnetic problems. *IEEE Transactions on Magnetics*. https://doi.org/10.1109/TMAG.2013.2282395

Bader, J., & Zitzler, E. (2011). HypE: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*. https://doi.org/10.1162/EVCO_a_00009

Bambha, N. K., Bhattacharyya, S. S., Teich, J., & Zitzler, E. (2004). Systematic integration of parameterized local search techniques in evolutionary algorithms. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1109/TEVC.2004.823471

Bandyopadhyay, S., & Mukherjee, A. (2015). An algorithm for many-objective optimization with reduced objective computations: A study in differential evolution. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2014.2332878

Basak, A., Maity, D., & Das, S. (2013). A differential invasive weed optimization algorithm for improved global numerical optimization. *Applied Mathematics and Computation*. https://doi.org/10.1016/j.amc.2012.12.057

Bashir, H. A. (2014). global optimization with hybrid evolutionary computation. the University of Manchester.

Bellman, R. (1954). The Theory of Dynamic Programming. *Bulletin of the American Mathematical Society*. https://doi.org/10.1090/S0002-9904-1954-09848-8

Beume, N., Naujoks, B., & Emmerich, M. (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2006.08.008

Beyer, H. G., & Sendhoff, B. (2017). Simplify your covariance matrix adaptation evolution strategy. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2017.2680320

Blum, C., Puchinger, J., Raidl, G., & Roli, A. (2011). Hybrid metaheuristics. In *Springer Optimization and Its Applications*. https://doi.org/10.1007/978-1-4419-1644-0_9

Bochinski, E., Senst, T., & Sikora, T. (2018). Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. *Proceedings - International Conference on Image Processing, ICIP*. https://doi.org/10.1109/ICIP.2017.8297018

Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. (2018). A Study on Self-adaptation in the Evolutionary Strategy Algorithm. *IFIP Advances in Information and Communication Technology*. https://doi.org/10.1007/978-3-319-89743-1_14

Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. (2019). An Efficient Hybrid Evolution Strategy Algorithm with Direct Search Method for Global Optimization. *International Journal of Organizational and Collective Intelligence*. https://doi.org/10.4018/ijoci.2019070104

Boussaïd, I., Chatterjee, A., Siarry, P., & Ahmed-Nacer, M. (2011). Hybridizing biogeography-based optimization with differential evolution for optimal power allocation in wireless sensor networks. *IEEE Transactions on Vehicular Technology*. https://doi.org/10.1109/TVT.2011.2151215

Cai, X., Gao, L., Li, X., & Qiu, H. (2019). Surrogate-guided differential evolution algorithm for high dimensional expensive problems. *Swarm and Evolutionary Computation*.

https://doi.org/10.1016/j.swevo.2019.04.009

Caponio, A., Neri, F., & Tirronen, V. (2009). Super-fit control adaptation in memetic differential evolution frameworks. *Soft Computing*. https://doi.org/10.1007/s00500-008-0357-1

Carlos A. Coello Coello, Gary B. Lamont, and D. A. V. V. (2007). Evolutionary Algorithms for Solving Multi-Objective Problems. In *Evolutionary Algorithms for Solving Multi-Objective Problems*. https://doi.org/10.1007/978-0-387-36797-2

Chakraborti, T., Chatterjee, A., Halder, A., & Konar, A. (2015). Automated emotion recognition employing a novel modified binary quantum-behaved gravitational search algorithm with differential mutation. *Expert Systems*. https://doi.org/10.1111/exsy.12104

Chambers, L. G., & Fletcher, R. (2001). Practical Methods of Optimization. *The Mathematical Gazette*. https://doi.org/10.2307/3621816

Chang, T. J., Yang, S. C., & Chang, K. J. (2009). Portfolio optimization problems in different risk measures using genetic algorithm. *Expert Systems with Applications*. https://doi.org/10.1016/j.eswa.2009.02.062

Chelouah, R., & Siarry, P. (2003). Genetic and Nelder-Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research*. https://doi.org/10.1016/S0377-2217(02)00401-0

Chen, S. (2012). Particle swarm optimization with pbest crossover. *2012 IEEE Congress on Evolutionary Computation, CEC 2012*. https://doi.org/10.1109/CEC.2012.6256497

Chen, X., Du, W., & Qian, F. (2014). Multi-objective differential evolution with ranking-based mutation operator and its application in chemical process optimization. *Chemometrics and Intelligent Laboratory Systems*. https://doi.org/10.1016/j.chemolab.2014.05.007

Chen, Y., Zhong, J., & Tan, M. (2018). A Fast Memetic Multi-Objective Differential Evolution for Multi-Tasking Optimization. *2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings*. https://doi.org/10.1109/CEC.2018.8477722

Cheng, M. Y., & Tran, D. H. (2015). Opposition-based Multiple Objective Differential Evolution (OMODE) for optimizing work shift schedules. *Automation in Construction*. https://doi.org/10.1016/j.autcon.2015.03.021

Chiang, C. W., Lee, W. P., & Heh, J. S. (2010). A 2-Opt based differential evolution for global optimization. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2010.05.012

Coelho, V. N., Coelho, I. M., Souza, M. J. F., Oliveira, T. A., Cota, L. P., Haddad, M. N., … Guimarães, F. G. (2016). Hybrid self-adaptive evolution strategies guided by neighborhood structures for combinatorial optimization problems. *Evolutionary Computation*. https://doi.org/10.1162/EVCO_a_00187

Coello, C. A. C., & Cortés, N. C. (2005). Solving multiobjective optimization problems using an artificial immune system. *Genetic Programming and Evolvable Machines*. https://doi.org/10.1007/s10710-005-6164-x

Coello Coello, C. A. (2006). Evolutionary multi-objective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*. https://doi.org/10.1109/MCI.2006.1597059

Coello Coello, Carlos A., & Lechuga, M. S. (2002). MOPSO: A proposal for multiple objective particle swarm optimization. *Proceedings of the 2002 Congress on Evolutionary*

Computation, CEC 2002. https://doi.org/10.1109/CEC.2002.1004388

Coello Coello, Carlos A., & Pulido, G. T. (2001). A micro-genetic algorithm for multiobjective optimization. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/3-540-44719-9_9

Coello Coello, Carlos A., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2004.826067

Conn, A. R., Scheinberg, K., & Vicente, L. N. (2009). Global convergence of general derivative-free trust-region algorithms to first- and second-order critical points. *SIAM Journal on Optimization*. https://doi.org/10.1137/060673424

Corne, D., Jerram, N., Knowles, J., Oates, M., & Martin, J. (2001). PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*.

Cotta, C., Mathieson, L., & Moscato, P. (2018). Memetic algorithms. In *Handbook of Heuristics*. https://doi.org/10.1007/978-3-319-07124-4_29

Cura, T. (2009). Particle swarm optimization approach to portfolio optimization. *Nonlinear Analysis: Real World Applications*. https://doi.org/10.1016/j.nonrwa.2008.04.023

Dan, S. (2013). Evolutionary Optimization Algorithm : Biologically Inspired and Population-Based Approaches to Computer Intelligence. In *Wiley*. https://doi.org/10.1007/s13398-014-0173-7.2

Darwin, C. (1860). The Origin of Species, by Means of Natural Selection, or the Preservation of Favored Races in the Struggle for Life. *The Crayon*. https://doi.org/10.2307/25528056

Das, I., & Dennis, J. E. (1998). Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*. https://doi.org/10.1137/S1052623496307510

Das, S., Biswas, A., Dasgupta, S., & Abraham, A. (2009). Bacterial foraging optimization algorithm: Theoretical foundations, analysis, and applications. *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-642-01085-9_2

Das, S., Mullick, S. S., & Suganthan, P. N. (2016). Recent advances in differential evolution-An updated survey. *Swarm and Evolutionary Computation*. https://doi.org/10.1016/j.swevo.2016.01.004

de Freitas, A. R. R., Guimarães, F. G., Pedrosa Silva, R. C., & Souza, M. J. F. (2014). Memetic self-adaptive evolution strategies applied to the maximum diversity problem. *Optimization Letters*. https://doi.org/10.1007/s11590-013-0610-0

De Jong, K. (2007). Parameter setting in EAs: A 30 year perspective. *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-540-69432-8_1

Deb, K. (2011). Multi-objective Optimisation Using Evolutionary Algorithms: An Introduction. In *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*. https://doi.org/10.1007/978-0-85729-652-8_1

Deb, K., & Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*.

https://doi.org/10.1109/TEVC.2013.2281535

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002a). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/4235.996017

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002b). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/4235.996017

Deb, K., Sindhya, K., & Hakanen, J. (2016). Multi-objective optimization. In *Decision Sciences: Theory and Practice*. https://doi.org/10.1201/9781315183176

Deb, K., Thiele, L., Laumanns, M., & Zitzler, E. (2002). Scalable multi-objective optimization test problems. *Proceedings of the 2002 Congress on Evolutionary Computation, CEC 2002*. https://doi.org/10.1109/CEC.2002.1007032

Deb, K., Thiele, L., Laumanns, M., & Zitzler, E. (2005). Scalable Test Problems for Evolutionary Multiobjective Optimization. In *Evolutionary Multiobjective Optimization*. https://doi.org/10.1007/1-84628-137-7_6

Delvecchio, G., Lofrumento, C., Neri, F., & Labini, M. S. (2006). A fast evolutionary-deterministic algorithm to study multimodal current fields under safety level constraints. *COMPEL - The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*. https://doi.org/10.1108/03321640610666754

Denysiuk, R., Costa, L., & Santo, I. E. (2013). Many-objective optimization using differential evolution with variable-wise mutation restriction. *GECCO 2013 - Proceedings of the 2013 Genetic and Evolutionary Computation Conference*. https://doi.org/10.1145/2463372.2463445

Di Tollo, G. (2015). Hybrid metaheuristic for portfolio selection: Comparison with an exact solver and search space analysis. *Proceedings of the 2015 Federated Conference on Computer Science and Information Systems, FedCSIS 2015*. https://doi.org/10.15439/2015F7

Dong, N., & Wang, Y. (2014). A memetic differential evolution algorithm based on dynamic preference for constrained optimization problems. *Journal of Applied Mathematics*. https://doi.org/10.1155/2014/606019

Du, H., Wang, Z., Zhan, W., & Guo, J. (2018). Elitism and distance strategy for selection of evolutionary algorithms. *IEEE Access*. https://doi.org/10.1109/ACCESS.2018.2861760

Duan, Y. C. (2004). A Multi-Objective Approach to Portfolio Optimization. *Rose-Hulman Undergraduate Mathematics Journal*. https://doi.org/10.1007/s00780-003-0114-3

Dumas, L., Druez, B., & Lecerf, N. (2009). A fully adaptive hybrid optimization of aircraft engine blades. *Journal of Computational and Applied Mathematics*. https://doi.org/10.1016/j.cam.2008.10.041

Duvidier, D. (2000). *Etude de l'hybridation des méta-heuristiques, application à un problème d'ordonnancement de type jobshop*. université du littoral France.

Eberhart, R., & Kennedy, J. (1995). New optimizer using particle swarm theory. *Proceedings of the International Symposium on Micro Machine and Human Science*. https://doi.org/10.1109/mhs.1995.494215

Emami, M., Amini, A., & Emami, A. (2012). Stock Portfolio Optimization with Using a New

Hybrid Evolutionary Algorithm Based on ICA and GA: Recursive-ICA-GA (Case Study of Tehran Stock Exchange). *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.2067126

Emel Soylu, & Uysal, A. (2017). A Hybrid Genetic-Ant Colony Algorithm for Travelling Salesman Problem. *INTERNATIONAL JOURNAL of ENGINEERING SCIENCE AND APPLICATION, 01*(03).

Emmerich, M. T. M., & Deutz, A. H. (2018). A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural Computing*. https://doi.org/10.1007/s11047-018-9685-y

Epitropakis, M. G., Li, X., & Burke, E. K. (2013). A dynamic archive niching differential evolution algorithm for multimodal optimization. *2013 IEEE Congress on Evolutionary Computation, CEC 2013*. https://doi.org/10.1109/CEC.2013.6557556

Erickson, M., Mayer, A., & Horn, J. (2001). The niched pareto genetic algorithm 2 applied to the design of groundwater remediation systems. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/3-540-44719-9_48

Fan, Q., Yan, X., & Xue, Y. (2017). Prior knowledge guided differential evolution. *Soft Computing*. https://doi.org/10.1007/s00500-016-2235-6

Festa, P. (2014). A brief introduction to exact, approximation, and heuristic algorithms for solving hard combinatorial optimization problems. *International Conference on Transparent Optical Networks*. https://doi.org/10.1109/ICTON.2014.6876285

Floreano, D., & Mattiussi, C. (2008). Bio-inspired artificial intelligence Theories, Methods, and Technologies. *2012 3rd National Conference on Emerging Trends and Applications in Computer Science*. https://doi.org/10.1007/s10710-010-9104-3

Fogel, D. B., & Ghozeil, A. (1997). A note on representations and variation operators. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/4235.687882

Fonseca, C. M., & Fleming, P. J. (1993). Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. *Icga*. https://doi.org/citeulike-article-id:2361311

Fu, M. (2015). Handbook of Simulation Optimization. In *International Series in Operations Research and Management Science*. https://doi.org/10.1007/978-1-4939-1384-8_9

García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*. https://doi.org/10.1007/s10732-008-9080-4

Genetic algorithms in search, optimization, and machine learning. (1989). *Choice Reviews Online*. https://doi.org/10.5860/choice.27-0936

Gheraibia, Y., & Moussaoui, A. (2013). Penguins Search Optimization Algorithm (PeSOA). *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-642-38577-3_23

Glover, F., Laguna, M., & Martí, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*.

Goel, T., & Deb, K. (2002). Hybrid methods for multi-objective evolutionary algorithms. *Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02)*.

Goh, C.-K., & Tan, K. C. (2009). Evolutionary Multi-objective Optimization in Uncertain Environments: Issues and Algorithms. In *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-540-95976-2_7

Goldberg, D. (1991). Real-coded genetic algorithms, virtual alphabets, and blocking. *Complex Systems*.

Gong, W., & Cai, Z. (2013). Differential evolution with ranking-based mutation operators. *IEEE Transactions on Cybernetics*. https://doi.org/10.1109/TCYB.2013.2239988

Gong, W., Cai, Z., & Liang, D. (2014). Engineering optimization by means of an improved constrained differential evolution. *Computer Methods in Applied Mechanics and Engineering*. https://doi.org/10.1016/j.cma.2013.10.019

Gonzalez-Fernandez, Y., & Chen, S. (2014). Identifying and exploiting the scale of a search space in particle swarm optimization. *GECCO 2014 - Proceedings of the 2014 Genetic and Evolutionary Computation Conference*. https://doi.org/10.1145/2576768.2598280

Gonzalez, O. M., Segura, C., Pena, S. I. V., & Leon, C. (2017). A memetic algorithm for the Capacitated Vehicle Routing Problem with Time Windows. *2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings*. https://doi.org/10.1109/CEC.2017.7969619

Grefenstette, J. J. (1987). Incorporating Problem Specific Knowledge into Genetic Algorithms. In *Genetic Algorithms and Simulated Annealing*.

Grosan, C., & Abraham, A. (2007). Hybrid evolutionary algorithms: Methodologies, architectures, and reviews. *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-540-73297-6_1

Guerard, J. (2016). The theory of risk, return, and performance measurement. In *Portfolio Construction, Measurement, and Efficiency: Essays in Honor of Jack Treynor*. https://doi.org/10.1007/978-3-319-33976-4_1

Guo, S. M., Yang, C. C., Hsu, P. H., & Tsai, J. S. H. (2015). Improving differential evolution with a successful-parent-selecting framework. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2014.2375933

Hachimi, H., Makhloufi, A., Ellaia, R., & El Hami, A. (2013). Optimization of engineering structures by heuristic algorithms. *Proceedings of 2013 International Conference on Industrial Engineering and Systems Management, IEEE - IESM 2013*.

Han, F., Mo, C., & Gao, H. (2018). An adaptive hybrid differential evolutionary algorithm for the parameter identification of rotating machinery. *JVC/Journal of Vibration and Control*. https://doi.org/10.1177/1077546317743890

Hao, J. K., Galinier, P., & Habib, M. (1999). Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes. *Revue d'Intelligence Artificielle*.

Harada, K., Ikeda, K., & Kobayashi, S. (2006). Hybridization of genetic algorithm and local search in multiobjective function optimization: Recommendation of GA then LS. *GECCO 2006 - Genetic and Evolutionary Computation Conference*. https://doi.org/10.1145/1143997.1144116

Hasan, B. H. F., Abu Doush, I., Al Maghayreh, E., Alkhateeb, F., & Hamdan, M. (2014). Hybridizing harmony search algorithm with different mutation operators for continuous problems. *Applied Mathematics and Computation*. https://doi.org/10.1016/j.amc.2013.12.139

Hassanat, A., Almohammadi, K., Alkafaween, E., Abunawas, E., Hammouri, A., & Prasath, V. B. S. (2019). Choosing mutation and crossover ratios for genetic algorithms-a review with a new dynamic approach. *Information (Switzerland)*. https://doi.org/10.3390/info10120390

He, X., & Zhou, Y. (2018). Enhancing the performance of differential evolution with covariance matrix self-adaptation. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2017.11.050

Helbig, M., & Engelbrecht, A. P. (2013). Performance measures for dynamic multi-objective optimisation algorithms. *Information Sciences*. https://doi.org/10.1016/j.ins.2013.06.051

Heris, M. K. (2014). Yarpiz Academic Dataset. Retrieved from yarpiz website: http://yarpiz.com/391/ypap112-portfolio-optimization.dataset

Hindi, M. M., & Yampolskiy, R. V. (2012). Genetic algorithm applied to the graph coloring problem. *CEUR Workshop Proceedings*.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). Niched Pareto genetic algorithm for multiobjective optimization. *IEEE Conference on Evolutionary Computation - Proceedings*. https://doi.org/10.1109/icec.1994.350037

Hosseini, S., & Al Khaled, A. (2014). A survey on the Imperialist Competitive Algorithm metaheuristic: Implementation in engineering domain and directions for future research. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2014.08.024

Houck, C. R., Kay, M. G., Joines, J. A., & Wilson, J. R. (1997). Empirical investigation of the benefits of partial lamarckianism. *Evolutionary Computation*. https://doi.org/10.1162/evco.1997.5.1.31

Huang, V. L., Qin, A. K., Suganthan, P. N., & Tasgetiren, M. F. (2007). Multi-objective optimization based on self-adaptive differential evolution algorithm. *2007 IEEE Congress on Evolutionary Computation, CEC 2007*. https://doi.org/10.1109/CEC.2007.4424939

Huang, V. L., Zhao, S. Z., Mallipeddi, R., & Suganthan, P. N. (2009). Multi-objective optimization using self-adaptive differential evolution algorithm. *2009 IEEE Congress on Evolutionary Computation, CEC 2009*. https://doi.org/10.1109/CEC.2009.4982947

Huang, W., Zhang, Y., & Li, L. (2019). Survey on Multi-Objective Evolutionary Algorithms. *Journal of Physics: Conference Series*. https://doi.org/10.1088/1742-6596/1288/1/012057

Huband, S., Hingston, P., Barone, L., & While, L. (2006). A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2005.861417

Humeau, J., Liefooghe, A., Talbi, E. G., & Verel, S. (2013). ParadisEO-MO: From fitness landscape analysis to efficient local search algorithms. *Journal of Heuristics*. https://doi.org/10.1007/s10732-013-9228-8

IEEE-Xplore. (n.d.). Retrieved from http://ieeexplore.ieee.org

Imene Cherki , Abdelkader Chaker , Zohra Djidar, N. K. and, & Benzergua, F. (2019). A Sequential Hybridization of Genetic Algorithm and Particle Swarm Optimization for the

Optimal Reactive Power Flow. *Sustainability*.

Inthachot, M., Boonjing, V., & Intakosum, S. (2016). Artificial Neural Network and Genetic Algorithm Hybrid Intelligence for Predicting Thai Stock Price Index Trend. *Computational Intelligence and Neuroscience*. https://doi.org/10.1155/2016/3045254

Ishibuchi, H., & Murata, T. (1998). A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*. https://doi.org/10.1109/5326.704576

Jadon, S. S., Tiwari, R., Sharma, H., & Bansal, J. C. (2017). Hybrid Artificial Bee Colony algorithm with Differential Evolution. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2017.04.018

Jaimes, A. L. (2011). *Techniques to Deal with Many-objective Optimization Problems Using Evolutionary Algorithms*. National Politechnic of Mexico.

Jain, H., & Deb, K. (2014). An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, Part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2013.2281534

Janga Reddy, M., & Nagesh Kumar, D. (2012). Computational algorithms inspired by biological processes and evolution. *Current Science*.

Jaszkiewicz, A. (2002). Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*. https://doi.org/10.1016/S0377-2217(01)00104-7

Jeyakumar, G., & Shanmugavelayutham, C. (2011). Convergence Analysis of Differential Evolution Variants on Unconstrained Global Optimization Functions. *International Journal of Artificial Intelligence & Applications*. https://doi.org/10.5121/ijaia.2011.2209

Jiang, S., & Yang, S. (2017). A Steady-State and Generational Evolutionary Algorithm for Dynamic Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2016.2574621

Joyce, T., & Herrmann, J. M. (2018). A review of no free lunch theorems, and their implications for metaheuristic optimisation. In *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-319-67669-2_2

K. V. Price, Rainer Martin Storn, J. L. (2005). *Differential Evolution-A Practical Approach to Global Optimization* (Natural Co). springer.

Kaikai, P. X. ; Z. J. ; C. H. ; C. X. ; H. (2015). A differential evolution-based hybrid NSGA-II for multi-objective optimization. *2015 IEEE 7th International Conference on Cybernetics and Intelligent Systems (CIS)*. IEEE Conference on Robotics, Automation and Mechatronics (RAM).

Kamili, H., & Riffi, M. E. (2016). A comparative study on portfolio optimization problem. *Proceedings - 2016 International Conference on Engineering and MIS, ICEMIS 2016*. https://doi.org/10.1109/ICEMIS.2016.7745339

Kämpf, J. H., & Robinson, D. (2009). A hybrid CMA-ES and HDE optimisation algorithm with application to solar energy potential. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2008.09.009

Kelner, V., Capitanescu, F., Léonard, O., & Wehenkel, L. (2008). A hybrid optimization

technique coupling an evolutionary and a local search algorithm. *Journal of Computational and Applied Mathematics*. https://doi.org/10.1016/j.cam.2006.03.048

Knowles, J., & Corne, D. (1999). The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimisation. *Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999*. https://doi.org/10.1109/CEC.1999.781913

Koduru, P., Das, S., & Welch, S. M. (2007). Multi-objective hybrid PSO using ν-fuzzy dominance. *Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference*. https://doi.org/10.1145/1276958.1277125

Koza, J. (1992). Genetic programming: on the programming of computers by means of natural selection. In *Biosystems*.

Kramer, O. (2016). Benchmark Functions. *Studies in Big Data: Machine Learning for Evolution Strategies*. https://doi.org/10.1007/978-3-319-33383-0

Kramer, O. (2018). Evolution of Convolutional Highway Networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-77538-8_27

Krasnogor, N., & Smith, J. (2008). Memetic algorithms: The polynomial local search complexity theory perspective. *Journal of Mathematical Modelling and Algorithms*. https://doi.org/10.1007/s10852-007-9070-9

Krause, O., Arbonès, D. R., & Igel, C. (2016). CMA-ES with optimal covariance update and storage complexity. *Advances in Neural Information Processing Systems*.

Kuhn, H. W., & Tucker, A. W. (2014). Nonlinear programming. In *Traces and Emergence of Nonlinear Programming*. https://doi.org/10.1007/978-3-0348-0439-4_11

Kukkonen, S., & Lampinen, J. (2004). An extension of generalized differential evolution for multi-objective optimization with constraints. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-540-30217-9_76

Kukkonen, S., & Lampinen, J. (2005). GDE3: The third evolution step of generalized differential evolution. *2005 IEEE Congress on Evolutionary Computation, IEEE CEC 2005. Proceedings*. https://doi.org/10.1109/cec.2005.1554717

Kumar, M., Husain, M., Upreti, N., & Gupta, D. (2020). Genetic Algorithm: Review and Application. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3529843

Kumar, R. (2016). Risk and return. In *Valuation*. https://doi.org/10.1016/b978-0-12-802303-7.00002-4

Kumar, R. G. G. R. K. (2013). Hybridization in Genetic Algorithms. *International Journal of Advanced Research in Computer Science and Software Engineering*, *3*(4).

Lara, A., Sanchez, G., Coello, C. A. C., & Schütze, O. (2010). HCS: A new local search strategy for memetic multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2009.2024143

Laumanns, M., Thiele, L., & Zitzler, E. (2005). An adaptive scheme to generate the pareto front based on the epsilon-constraint method. *Practical Approaches to Multi-Objective Optimization*.

Lawrence Fogel-Alvin Owens-Michael Walsh. (1966). *Artificial Intelligence through Simulated Evolution* (John Wiley; J. Wiley, Ed.). Retrieved from John Wiley

Le Riche, R., Schoenauer, M., & Sebag, M. (2007). Un état des lieux de l'optimisation évolutionnaire et de ses implications en sciences pour l'ingénieur. *Modélisation Numérique: Défis et Perspectives, Vol. 2, Traité Mécanique et Ingénierie Des Matériaux*.

Lenoir, A., & Monmarché, N. (2009). Des fourmis réelles aux fourmis artificielles. In *Fourmis artificielles 1-Des bases de l'optimisation aux applications industrielles*.

Li, H., & Zhang, Q. (2009). Multiobjective optimization problems with complicated pareto sets, MOEA/ D and NSGA-II. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2008.925798

Li, K., Fialho, A., Kwong, S., & Zhang, Q. (2014). Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2013.2239648

Liang, J. J., Zheng, B., Xu, F. Y., Qu, B. Y., & Song, H. (2014). Multi-objective differential evolution algorithm based on fast sorting and a novel constraints handling technique. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*. https://doi.org/10.1109/CEC.2014.6900525

Liao, Y. F., Yau, D. H., & Chen, C. L. (2012). Evolutionary algorithm to traveling salesman problems. *Computers and Mathematics with Applications*. https://doi.org/10.1016/j.camwa.2011.12.018

Lin, P. C. (2012). Portfolio optimization and risk measurement based on non-dominated sorting genetic algorithm. *Journal of Industrial and Management Optimization*. https://doi.org/10.3934/jimo.2012.8.549

Lotfy, M. E., Senjyu, T., Farahat, M. A. F., Abdel-Gawad, A. F., Lei, L., & Datta, M. (2018). Hybrid genetic algorithm fuzzy-based control schemes for small power system with high-penetration wind farms. *Applied Sciences (Switzerland)*. https://doi.org/10.3390/app8030373

Lou, Y., Yuen, S. Y., Chen, G., & Zhang, X. (2019). On-line Search History-assisted Restart Strategy for Covariance Matrix Adaptation Evolution Strategy. *2019 IEEE Congress on Evolutionary Computation, CEC 2019 - Proceedings*. https://doi.org/10.1109/CEC.2019.8790368

Lozano, M., & García-Martínez, C. (2010). Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report. *Computers and Operations Research*. https://doi.org/10.1016/j.cor.2009.02.010

Lozano, Manuel, Herrera, F., Krasnogor, N., & Molina, D. (2004). Real-coded memetic algorithms with crossover hill-climbing. *Evolutionary Computation*. https://doi.org/10.1162/1063656041774983

Lshibuchi, H., Sakane, Y., Tsukamoto, N., & Nojima, Y. (2010). Simultaneous use of different scalarizing functions in MOEA/D. *Proceedings of the 12th Annual Genetic and Evolutionary Computation Conference, GECCO '10*. https://doi.org/10.1145/1830483.1830577

Luong, D. L., Tran, D. H., & Nguyen, P. T. (2018). Optimizing multi-mode time-cost-quality trade-off of construction project using opposition multiple objective difference evolution. *International Journal of Construction Management*. https://doi.org/10.1080/15623599.2018.1526630

Lwin, K., Qu, R., & Kendall, G. (2014). A learning-guided multi-objective evolutionary

algorithm for constrained portfolio optimization. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2014.08.026

Ma, H., Simon, D., Siarry, P., Yang, Z., & Fei, M. (2017). Biogeography-Based Optimization: A 10-Year Review. *IEEE Transactions on Emerging Topics in Computational Intelligence*. https://doi.org/10.1109/tetci.2017.2739124

Ma, X., Liu, F., Qi, Y., Gong, M., Yin, M., Li, L., … Wu, J. (2014). MOEA/D with opposition-based learning for multiobjective optimization problem. *Neurocomputing*. https://doi.org/10.1016/j.neucom.2014.04.068

Madani BEZOUI. (2011). Méthode Adaptée de programmation quadratique convexe: Théorie et Applications. *Les Editions Universitaires Européenes*.

Madavan, N. K. (2002). Multiobjective optimization using a Pareto differential evolution approach. *Proceedings of the 2002 Congress on Evolutionary Computation, CEC 2002*. https://doi.org/10.1109/CEC.2002.1004404

Mahdavi, S., Rahnamayan, S., & Deb, K. (2018). Opposition based learning: A literature review. *Swarm and Evolutionary Computation*. https://doi.org/10.1016/j.swevo.2017.09.010

Manikandan, P., & Ramyachitra, D. (2017). Bacterial Foraging Optimization-Genetic Algorithm for Multiple Sequence Alignment with Multi-Objectives. *Scientific Reports*. https://doi.org/10.1038/s41598-017-09499-1

Mao, B., Xie, Z., Wang, Y., Handroos, H., & Wu, H. (2018). A Hybrid Strategy of Differential Evolution and Modified Particle Swarm Optimization for Numerical Solution of a Parallel Manipulator. *Mathematical Problems in Engineering*. https://doi.org/10.1155/2018/9815469

Markowitz, H. (1952). PORTFOLIO SELECTION. *The Journal of Finance*. https://doi.org/10.1111/j.1540-6261.1952.tb01525.x

Markowitz, H. (2014). Mean-variance approximations to expected utility. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2012.08.023

Martinez, S. Z., & Coello Coello, C. A. (2013). A hybridization of MOEA/D with the nonlinear simplex search algorithm. *Proceedings of the 2013 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making, MCDM 2013 - 2013 IEEE Symposium Series on Computational Intelligence, SSCI 2013*. https://doi.org/10.1109/MCDM.2013.6595443

Martínez, S. Z., & Coello Coello, C. A. (2012). A direct local search mechanism for decomposition-based multi-objective evolutionary algorithms. *2012 IEEE Congress on Evolutionary Computation, CEC 2012*. https://doi.org/10.1109/CEC.2012.6252990

McClymont, K., & Keedwell, E. (2012). Deductive sort and climbing sort: New methods for non-dominated sorting. *Evolutionary Computation*. https://doi.org/10.1162/EVCO_a_00041

Meghwani, S. S., & Thakur, M. (2017). Multi-criteria algorithms for portfolio optimization under practical constraints. *Swarm and Evolutionary Computation*. https://doi.org/10.1016/j.swevo.2017.06.005

Metaxiotis, K., & Liagkouras, K. (2015). The solution of the 0-1 multi-objective knapsack problem with the assistance of multi-objective evolutionary algorithms based on

decomposition: A comparative study. *2015 5th International Workshop on Computer Science and Engineering: Information Processing and Control Engineering, WCSE 2015-IPCE*. https://doi.org/10.18178/wcse.2015.04.001

Meyer-Nieberg, S., & Beyer, H. G. (2007). Self-adaptation in evolutionary algorithms. *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-540-69432-8_3

Miettinen, K. (2008). Introduction to multiobjective optimization: Noninteractive approaches. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-540-88908-3-1

Mininno, E., Neri, F., Cupertino, F., & Naso, D. (2011). Compact differential evolution. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2010.2058120

Mishra, S. K., Panda, G., & Majhi, R. (2014). A comparative performance assessment of a set of multiobjective algorithms for constrained portfolio assets selection. *Swarm and Evolutionary Computation*. https://doi.org/10.1016/j.swevo.2014.01.001

Mnasri, S., Nasri, N., Van Den Bossche, A., & Val, T. (2017). A hybrid ant-genetic algorithm to solve a real deployment problem: A case study with experimental validation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-67910-5_30

Mohammad-Moradi, S., Khaloozadeh, H., Forouzanfar, M., Paravi, R., & Forghani, N. (2009). Swarm Intelligence in Portfolio Selection. In *Particle Swarm Optimization*. https://doi.org/10.5772/6750

Molina, D., Lozano, M., García-Martínez, C., & Herrera, F. (2010). Memetic algorithms for continuous optimisation based on local search chains. *Evolutionary Computation*. https://doi.org/10.1162/evco.2010.18.1.18102

Mühlenbein, H., Gorges-Schleuter, M., & Krämer, O. (1988). Evolution algorithms in combinatorial optimization. *Parallel Computing*. https://doi.org/10.1016/0167-8191(88)90098-1

Mustafa, H. M. J., Ayob, M., Nazri, M. Z. A., & Kendall, G. (2019). An improved adaptive memetic differential evolution optimization algorithms for data clustering problems. *PLoS ONE*. https://doi.org/10.1371/journal.pone.0216906

Nelder, J. A., & Mead, R. (1965). A Simplex Method for Function Minimization. *The Computer Journal*. https://doi.org/10.1093/comjnl/7.4.308

Neri, F., Cotta, C., & Moscato, P. (2012). Handbook of Memetic Algorithms. In *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-642-23247-3

Nguyen, Q. H., Ong, Y. S., & Krasnogor, N. (2007). A study on the design issues of Memetic Algorithm. *2007 IEEE Congress on Evolutionary Computation, CEC 2007*. https://doi.org/10.1109/CEC.2007.4424770

Nocedal, J., & Wright, S. J. (2000). Numerical optimization 2nd edition. In *International ADAMS user conference*.

Oliveira, A. C. M., & Lorena, L. A. N. (2007). Hybrid evolutionary algorithms and clustering search. *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-540-73297-6_4

P-N-Suganthan. (2017). Large-Scale Portfolio Optimization Using Multiobjective

Evolutionary Algorithms and Preselection Methods. Retrieved from https://github.com/P-N-Suganthan/CODES/blob/master/2017-MPE-Portfolio.rar

Pai, G. A. V. (2019). Multi-objective Metaheuristics for Managing Futures Portfolio Risk. *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*. https://doi.org/10.1109/SSCI.2018.8628879

PANT, M., THANGARAJ, R., & ABRAHAM, A. (2011). DE-PSO: A NEW HYBRID META-HEURISTIC FOR SOLVING GLOBAL OPTIMIZATION PROBLEMS. *New Mathematics and Natural Computation*. https://doi.org/10.1142/s1793005711001986

Park, S. Y., & Lee, J. J. (2016). Stochastic Opposition-Based Learning Using a Beta Distribution in Differential Evolution. *IEEE Transactions on Cybernetics*. https://doi.org/10.1109/TCYB.2015.2469722

Peng, L., Zhang, Y., Dai, G., & Wang, M. (2017). Memetic differential evolution with an improved contraction criterion. *Computational Intelligence and Neuroscience*. https://doi.org/10.1155/2017/1395025

Peng, P., Addam, O., Elzohbi, M., Özyer, S. T., Elhajj, A., Gao, S., … Alhajj, R. (2014). Reporting and analyzing alternative clustering solutions by employing multi-objective genetic algorithm and conducting experiments on cancer data. *Knowledge-Based Systems*. https://doi.org/10.1016/j.knosys.2013.11.003

Petrowski, A., & Ben Hamida, S. (2016). Evolutionary algorithms. In *Metaheuristics*. https://doi.org/10.1007/978-3-319-45403-0_6

Piad-Morffis, A., Estevez-Velarde, S., Bolufe-Rohler, A., Montgomery, J., & Chen, S. (2015). Evolution strategies with thresheld convergence. *2015 IEEE Congress on Evolutionary Computation, CEC 2015 - Proceedings*. https://doi.org/10.1109/CEC.2015.7257143

Picek, S., Jakobovic, D., & Golub, M. (2013). On the recombination operator in the real-coded genetic algorithms. *2013 IEEE Congress on Evolutionary Computation, CEC 2013*. https://doi.org/10.1109/CEC.2013.6557948

Ponsich, A., Jaimes, A. L., & Coello, C. A. C. (2013). A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2012.2196800

Qi, Y., Ma, X., Liu, F., Jiao, L., Sun, J., & Wu, J. (2014). MOEA/D with Adaptive Weight Adjustment. *Evolutionary Computation*. https://doi.org/10.1162/EVCO_a_00109

Qian, W., & li, A. (2008). Adaptive differential evolution algorithm for multiobjective optimization problems. *Applied Mathematics and Computation*. https://doi.org/10.1016/j.amc.2007.12.052

Qin, A. K., & Forbes, F. (2011). Harmony search with differential mutation based pitch adjustment. *Genetic and Evolutionary Computation Conference, GECCO'11*. https://doi.org/10.1145/2001576.2001651

Qin, Q., Li, L., & Cheng, S. (2014). A novel hybrid algorithm for mean-CVaR portfolio selection with real-world constraints. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*.

Qu, B. Y., Suganthan, P. N., & Liang, J. J. (2012). Differential evolution with neighborhood mutation for multimodal optimization. *IEEE Transactions on Evolutionary Computation*.

https://doi.org/10.1109/TEVC.2011.2161873

Qu, X., Ong, Y. S., Hou, Y., & Shen, X. (2019). Memetic Evolution Strategy for Reinforcement Learning. *2019 IEEE Congress on Evolutionary Computation, CEC 2019 - Proceedings*. https://doi.org/10.1109/CEC.2019.8789935

Rahmat, N. A., Musirin, I., & Abidin, A. F. (2014). Differential Evolution Immunized Ant Colony Optimization Technique (DEIANT) in solving economic dispatch by considering prohibited operating zones. *Proceedings of the 2014 IEEE 8th International Power Engineering and Optimization Conference, PEOCO 2014*. https://doi.org/10.1109/PEOCO.2014.6814472

Rahnamayan, R. S., Tizhoosh, H. R., & Salama, M. M. A. (2008). Opposition-based differential evolution. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2007.894200

Rai, D., & Tyagi, K. (2013). Bio-inspired optimization techniques. *ACM SIGSOFT Software Engineering Notes*. https://doi.org/10.1145/2492248.2492271

Raidl, J. P. & G. R. (2005). Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. In Springer (Ed.), *Proceedings of the First International Work-Conference on the Interplay Between Natural and Artificial Computation* (pp. 41–53). Springer.

Rakshit, P., Konar, A., Das, S., Jain, L. C., & Nagar, A. K. (2014). Uncertainty management in differential evolution induced multiobjective optimization in presence of measurement noise. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. https://doi.org/10.1109/TSMC.2013.2282118

Ravindran, A., Ragsdell, K. M., & Reklaitis, G. V. (2007). Engineering Optimization: Methods and Applications: Second Edition. In *Engineering Optimization: Methods and Applications: Second Edition*. https://doi.org/10.1002/9780470117811

Reddy, M. J., & Kumar, D. N. (2007). Multiobjective differential evolution with application to reservoir system optimization. *Journal of Computing in Civil Engineering*. https://doi.org/10.1061/(ASCE)0887-3801(2007)21:2(136)

Robič, T., & Filipič, B. (2005). DEMO: Differential Evolution for Multiobjective Optimization. *Lecture Notes in Computer Science*. https://doi.org/10.1007/978-3-540-31880-4_36

Rochenberg, I. (1965). *Cybernetic solution path of an experimental problem.* Ministry of Aviation.

Roijers, D. M., Vamplew, P., Whiteson, S., & Dazeley, R. (2013). A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*. https://doi.org/10.1613/jair.3987

Rudolph, G. (1994). Convergence Analysis of Canonical Genetic Algorithms. *IEEE Transactions on Neural Networks*. https://doi.org/10.1109/72.265964

Saha, S., & Bandyopadhyay, S. (2013). A generalized automatic clustering algorithm in a multiobjective framework. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2012.08.005

Salehpour, M., Jamali, A., Bagheri, A., & Nariman-zadeh, N. (2017). A new adaptive differential evolution optimization algorithm based on fuzzy inference system. *Engineering Science and Technology, an International Journal*.

https://doi.org/10.1016/j.jestch.2017.01.004

Sampson, J. R. (1976). Adaptation in Natural and Artificial Systems (John H. Holland). *SIAM Review*. https://doi.org/10.1137/1018105

Santana-Quintero, L. V., Hernández-Díaz, A. G., Molina, J., Coello Coello, C. A., & Caballero, R. (2010). DEMORS: A hybrid multi-objective optimization algorithm using differential evolution and rough set theory for constrained problems. *Computers and Operations Research*. https://doi.org/10.1016/j.cor.2009.02.006

Saputra, A. A., Takeda, T., & Kubota, N. (2015). Efficiency energy on humanoid robot walking using evolutionary algorithm. *2015 IEEE Congress on Evolutionary Computation, CEC 2015 - Proceedings*. https://doi.org/10.1109/CEC.2015.7256941

Sawik, B. (2013). Survey of multi-objective portfolio optimization by linear and mixed integer programming. *Applications of Management Science*. https://doi.org/10.1108/S0276-8976(2013)0000016007

Schaffer, J. D. (James D., & Spears, W. M. (1989). Proceedings of the Third International Conference on Genetic Algorithms. In *Proceedings of the third international conference on Genetic algorithms*.

Schaffer, J. D., & Morishima, A. (1987). An Adaptive Crossover Distribution Mechanism for Genetic Algorithms. *Proceedings of the Second International Conference on Genetic Algorithms*.

Schott, J. R. (1995). Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. In *Massachusetts Institute of Technology, Cambridge, Massachusetts*.

Schwefel, H.-P., & Schwefel, H.-P. (1977). Optimierungsaufgaben und Optimiermethoden. In *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*. https://doi.org/10.1007/978-3-0348-5927-1_2

sciencedirect. (n.d.). Retrieved from http://www.sciencedirect.com

Sekanina, L. (2010). *Evolutionary circuit design: Tutorial*. https://doi.org/10.1109/ddecs.2010.5491830

Serraino, G., & Uryasev, S. (2013). Conditional Value-at-Risk (CVaR). In *Encyclopedia of Operations Research and Management Science*. https://doi.org/10.1007/978-1-4419-1153-7_1232

Shao, W., & Pi, D. (2016). A self-guided differential evolution with neighborhood search for permutation flow shop scheduling. *Expert Systems with Applications*. https://doi.org/10.1016/j.eswa.2015.12.001

Sharma, S., & Rangaiah, G. P. (2013). An improved multi-objective differential evolution with a termination criterion for optimizing chemical processes. *Computers and Chemical Engineering*. https://doi.org/10.1016/j.compchemeng.2013.05.004

Shen, Y., & Wang, Y. (2017). Operating Point Optimization of Auxiliary Power Unit Using Adaptive Multi-Objective Differential Evolution Algorithm. *IEEE Transactions on Industrial Electronics*. https://doi.org/10.1109/TIE.2016.2598674

Sierra, M. R., & Coello, C. A. C. (2005). Improving PSO-Based multi-objective optimization using crowding, mutation and epsilon-dominance. In *Evolutionary Multi-Criterion Optimization*.

Simon, D., Omran, M. G. H., & Clerc, M. (2014). Linearized biogeography-based optimization

with re-initialization and local search. *Information Sciences*. https://doi.org/10.1016/j.ins.2013.12.048

Sindhya, K., Deb, K., & Miettinen, K. (2011). Improving convergence of evolutionary multi-objective optimization with local search: A concurrent-hybrid algorithm. *Natural Computing*. https://doi.org/10.1007/s11047-011-9250-4

Sindhya, K., Miettinen, K., & Deb, K. (2013). A hybrid framework for evolutionary multi-objective optimization. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2012.2204403

Sinha, A., Chen, Y., & Goldberg, D. E. (2006). Designing Efficient Genetic and Evolutionary Algorithm Hybrids. In *Recent Advances in Memetic Algorithms*. https://doi.org/10.1007/3-540-32363-5_12

Snášel, V., Abraham, A., Krömer, P., Pant, M., & Muda, A. K. (2016). Innovations in bio-inspired computing and applications: Proceedings of the 6th international conference on innovations in bio-inspired computing and applications (IBICA 2015) held in Kochi, India during december 16-18, 2015. *Advances in Intelligent Systems and Computing*. https://doi.org/10.1007/978-3-319-28031-8

SpringerLink. (n.d.). Retrieved from http://www.springerlink.com

Srinivas, N., & Deb, K. (1994). Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*. https://doi.org/10.1162/evco.1994.2.3.221

Stanojević, P., Marić, M., & Stanimirović, Z. (2015). A hybridization of an evolutionary algorithm and a parallel branch and bound for solving the capacitated single allocation hub location problem. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2015.04.018

Storn, R., & Price, K. (1997). Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*. https://doi.org/10.1023/A:1008202821328

Suganuma, M., Shirakawa, S., & Nagao, T. (2017). A genetic programming approach to designing convolutional neural network architectures. *GECCO 2017 - Proceedings of the 2017 Genetic and Evolutionary Computation Conference*. https://doi.org/10.1145/3071178.3071229

Sun, J., Garibaldi, J. M., Krasnogor, N., & Zhang, Q. (2013). An intelligent multi-restart memetic algorithm for box constrained global optimisation. *Evolutionary Computation*. https://doi.org/10.1162/EVCO_a_00068

Talbi, E.-G. (2009). Metaheuristics for Multiobjective Optimization. In *Metaheuristics*. https://doi.org/10.1002/9780470496916.ch4

Talbi, E. G., Cahon, S., & Melab, N. (2007). Designing cellular networks using a parallel hybrid metaheuristic on the computational grid. *Computer Communications*. https://doi.org/10.1016/j.comcom.2006.08.017

Talbi, El Ghazali, Rahoual, M., Mabed, M. H., & Dhaenens, C. (2001). A hybrid evolutionary approach for multicriteria optimization problems: Application to the flow shop. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/3-540-44719-9_29

Tang, B., Zhu, Z., & Luo, J. (2016). Hybridizing Particle Swarm Optimization and Differential Evolution for the Mobile Robot Global Path Planning. *International Journal of Advanced Robotic Systems*. https://doi.org/10.5772/63812

Tang, L., Wang, X., & Dong, Z. (2019). Adaptive Multiobjective Differential Evolution with Reference Axis Vicinity Mechanism. *IEEE Transactions on Cybernetics*. https://doi.org/10.1109/TCYB.2018.2849343

Tian, G., Ren, Y., & Zhou, M. (2016). Dual-Objective Scheduling of Rescue Vehicles to Distinguish Forest Fires via Differential Evolution and Particle Swarm Optimization Combined Algorithm. *IEEE Transactions on Intelligent Transportation Systems*. https://doi.org/10.1109/TITS.2015.2505323

Tian, Y., Cheng, R., Zhang, X., Cheng, F., & Jin, Y. (2018). An Indicator-Based Multiobjective Evolutionary Algorithm with Reference Point Adaptation for Better Versatility. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2017.2749619

Tizhoosh, H. R. (2005). Opposition-based learning: A new scheme for machine intelligence. *Proceedings - International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA 2005 and International Conference on Intelligent Agents, Web Technologies and Internet*. https://doi.org/10.1109/cimca.2005.1631345

Törn, A., Ali, M. M., & Viitanen, S. (1999). Stochastic Global Optimization: Problem Classes and Solution Techniques. *Journal of Global Optimization*. https://doi.org/10.1023/A:1008395408187

Tran, D. H., Cheng, M. Y., & Pham, A. D. (2016). Using Fuzzy Clustering Chaotic-based Differential Evolution to solve multiple resources leveling in the multiple projects scheduling prob lem. *Alexandria Engineering Journal*. https://doi.org/10.1016/j.aej.2016.03.038

Trautmann, H., Wagner, T., & Brockhoff, D. (2013). R2-EMOA: Focused multiobjective search using R2-indicator-based selection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-642-44973-4_8

Tripathy, A., & Schwefel, H.-P. (1982). Numerical Optimization of Computer Models. *The Journal of the Operational Research Society*. https://doi.org/10.2307/2581158

Trivedi, Adarsh, Srivastava, S., Mishra, A., Shukla, A., & Tiwari, R. (2018). Hybrid evolutionary approach for Devanagari handwritten numeral recognition using Convolutional Neural Network. *Procedia Computer Science*. https://doi.org/10.1016/j.procs.2017.12.068

Trivedi, Anupam, Srinivasan, D., Biswas, S., & Reindl, T. (2016). A genetic algorithm - Differential evolution based hybrid framework: Case study on unit commitment scheduling problem. *Information Sciences*. https://doi.org/10.1016/j.ins.2016.03.023

Tseng, L. Y., & Liang, S. C. (2006). A hybrid metaheuristic for the quadratic assignment problem. *Computational Optimization and Applications*, *34*(1), 85–113. https://doi.org/10.1007/s10589-005-3069-9

Tuba, M., & Bacanin, N. (2014). Upgraded firefly algorithm for portfolio optimization problem. *Proceedings - UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, UKSim 2014*. https://doi.org/10.1109/UKSim.2014.25

Tvrdík, J., & Křivý, I. (2015). Hybrid differential evolution algorithm for optimal clustering.

Applied Soft Computing Journal. https://doi.org/10.1016/j.asoc.2015.06.032

Uslu, M. F., Uslu, S., & Bulut, F. (2019). An adaptive hybrid approach: Combining genetic algorithm and ant colony optimization for integrated process planning and scheduling. *Applied Computing and Informatics*. https://doi.org/10.1016/j.aci.2018.12.002

V. L. Huang, P. N. Suganthan, A. K. Qin, and S. B. (2005). Multiobjective differential evolution with external archive and harmonic distance-based diversity measure. *School Elect. Electron.*

Vaisakh, K., Praveena, P., & Sujatah, K. N. (2013). Differential evolution and bacterial foraging optimization based dynamic economic dispatch with non-smooth fuel cost functions. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-03756-1_52

Vakil-Baghmisheh, M. T., & Ahandani, M. A. (2014). A differential memetic algorithm. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-011-9302-2

Van Veldhuizen, D. a. (1999). Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations. *IRE Transactions on Education*. https://doi.org/10.1109/TE.1962.4322266

Van Veldhuizen, D. a., & Lamont, G. B. (1998). Evolutionary Computation and Convergence to a Pareto Front. *Late Breaking Papers at the Genetic Programming 1998 Conference*.

Vasant, P. (2015). Hybrid Evolutionary Optimization Algorithms. In *Research Methods*. https://doi.org/10.4018/978-1-4666-7456-1.ch047

Wan Liang Wang , Weikun Li, Y. L. W. (2019). An Opposition-Based Evolutionary Algorithm for Many-Objective Optimization with Adaptive Clustering Mechanism. *Computational Intelligence and Neuroscience*.

Wang, G. G., Deb, S., Gandomi, A. H., & Alavi, A. H. (2016). Opposition-based krill herd algorithm with Cauchy mutation and position clamping. *Neurocomputing*. https://doi.org/10.1016/j.neucom.2015.11.018

Wang, H., Rahnamayan, S., Sun, H., & Omran, M. G. H. (2013). Gaussian bare-bones differential evolution. *IEEE Transactions on Cybernetics*. https://doi.org/10.1109/TSMCB.2012.2213808

Wang, H., Wu, Z., Rahnamayan, S., Liu, Y., & Ventresca, M. (2011). Enhancing particle swarm optimization using generalized opposition-based learning. *Information Sciences*. https://doi.org/10.1016/j.ins.2011.03.016

Wang, Jiahai, Zhang, W., & Zhang, J. (2016). Cooperative Differential Evolution with Multiple Populations for Multiobjective Optimization. *IEEE Transactions on Cybernetics*. https://doi.org/10.1109/TCYB.2015.2490669

Wang, Jing. (2019). A novel firefly algorithm for portfolio optimization problem. *IAENG International Journal of Applied Mathematics*.

Wang, L., Zeng, Y., & Chen, T. (2015). Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. *Expert Systems with Applications*. https://doi.org/10.1016/j.eswa.2014.08.018

Wang, X., & Tang, L. (2013). Multiobjective operation optimization of naphtha pyrolysis process using parallel differential evolution. *Industrial and Engineering Chemistry*

Research. https://doi.org/10.1021/ie401954d

Wang, X., & Tang, L. (2016). An adaptive multi-population differential evolution algorithm for continuous multi-objective optimization. *Information Sciences*. https://doi.org/10.1016/j.ins.2016.01.068

Wang, Y., Cai, Z., & Zhang, Q. (2011). Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2010.2087271

Wang, Y. N., Wu, L. H., & Yuan, X. F. (2010). Multi-objective self-adaptive differential evolution with elitist archive and crowding entropy-based diversity measure. *Soft Computing*. https://doi.org/10.1007/s00500-008-0394-9

Wang, Z., Kuang, J. C., & Zhang, S. N. (2014). Portfolio optimization of China's generation technology based on conditional value-at-risk (CVaR) in plant level. *WIT Transactions on Information and Communication Technologies*. https://doi.org/10.2495/ISME20133433

Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., Peters, J., & Schmidhuber, J. (2014). Natural evolution strategies. *Journal of Machine Learning Research*.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/4235.585893

Woodside-Oriakhi, M., Lucas, C., & Beasley, J. E. (2011). Heuristic algorithms for the cardinality constrained efficient frontier. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2011.03.030

Xie, C. W., Xu, L., Wang, S. W., Xiao, C., & Xia, X. W. (2017). An Enhanced Multi-objective Fireworks Explosion Optimization Algorithm. *Tien Tzu Hsueh Pao/Acta Electronica Sinica*. https://doi.org/10.3969/j.issn.0372-2112.2017.10.002

Xu, B., Qi, R., Zhong, W., Du, W., & Qian, F. (2013). Optimization of p-xylene oxidation reaction process based on self-adaptive multi-objective differential evolution. *Chemometrics and Intelligent Laboratory Systems*. https://doi.org/10.1016/j.chemolab.2013.04.013

Yang, D., Jiao, L., & Gong, M. (2009). Adaptive multi-objective optimization based on nondominated solutions. *Computational Intelligence*. https://doi.org/10.1111/j.1467-8640.2009.00332.x

Yang, S., Jiang, S., & Jiang, Y. (2017). Improving the multiobjective evolutionary algorithm based on decomposition with new penalty schemes. *Soft Computing*. https://doi.org/10.1007/s00500-016-2076-3

Yen, J., Liao, J. C., Lee, B., & Randolph, D. (1998). A hybrid approach to modeling metabolic systems using a genetic algorithm and simplex method. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*. https://doi.org/10.1109/3477.662758

Yildiz, Y. E., Altun, O., & Topal, O. A. (2015). International Conference on Computing and Informatics. In U. U. Malaysia (Ed.), *ICOCI 2015*. Istambul: Universiti Utara Malaysia.

Yuen, S. Y., Chow, C. K., Zhang, X., & Lou, Y. (2016). Which algorithm should i choose: An evolutionary algorithm portfolio approach. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2015.12.021

Zhang, J., & Sanderson, A. C. (2008). Self-adaptive multi-objective differential evolution with direction information provided by archived inferior solutions. *2008 IEEE Congress on*

Evolutionary Computation, CEC 2008. https://doi.org/10.1109/CEC.2008.4631174

Zhang, J., & Sanderson, A. C. (2009). JADE: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2009.2014613

Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2007.892759

Zhang, Q., Liu, W., & Li, H. (2009). The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances. *2009 IEEE Congress on Evolutionary Computation, CEC 2009*. https://doi.org/10.1109/CEC.2009.4982949

Zhang, Q., Zhou, A., & Jin, Y. (2008). RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2007.894202

Zhang, X., Tian, Y., Cheng, R., & Jin, Y. (2015). An Efficient Approach to Nondominated Sorting for Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2014.2308305

Zhou, A., Qu, B. Y., Li, H., Zhao, S. Z., Suganthan, P. N., & Zhangd, Q. (2011). Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*. https://doi.org/10.1016/j.swevo.2011.03.001

Zhou, Y., Li, X., & Gao, L. (2013). A differential evolution algorithm with intersect mutation operator. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2012.08.014

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: empirical results. *Evolutionary Computation*. https://doi.org/10.1162/106365600568202

Zitzler, Eckart, Laumanns, M., & Thiele, L. (2001). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*. https://doi.org/10.1.1.28.7571

ANNEXE – LISTES DES ABRÉVIATIONS ET DES ALGORITHMES

ABC	artificial bees colony
ACO	Ant colony optimization
ANN	Artificial neurol network
BA	Bat algorithm
BBO	biogeographic optimization
BFO	Bacterial Foraging Optimization
CEC	Congress On Evolutionary Computation
CMA-ES	Covariance matrix-evolution strategy
CNN	Convolution neurol network
CSO	Cat swarm optimization
CV	Coefficient of variation
DE	Differential evolution
ED	Euclidian distance
EP	Evolutionary programming
ES	Evoluon strategey
FA	Firefly algorithm
FLC	Fuzzy logic controller
GA	Genetic algorithm
GD	Generational Distance
GP	Genetic algorithm
HV	Hypervolume
HypE	Hypervolume estimation algorithm
IBEA	Indicator based evolutionary algorithm
ICA	Imperialist competitive Algorithm
IEEE	Institute of Electrical and Electronics Engineers
IGD	Inverted generational distance
LS	local search
MA	Memetic algorithm
MODE	Multi-objective differential evolution
MOEA	Multi-objective evolutionary algorithm
MOEA/D	Multi-objective evolutionary algorithm/Decomposition
MOGA	Multi-objective genetic algorithm
MOOP	Multi-objective optimization problem
MOP	Multiobjective problem
MOPSO	Multi-objective particle swarm optimization
NM	Nelder-Mead
NN	Neurol network
NPGA	Niched Pareto genetic algorithm
NSDE	Non-dominated sorting differential evolution
NSGA	Non-dominated sorting genetic algorithm
OBL	Opposition based learning
PAES	Pareto archive evolution strategy
PESA	Pareto envelope selection algorithm
PSO	Particle swarm optimization
Rand	Random
RMEDA	Regularity model-based multi-objective estimation of distribution algorithm
RMO	Ranking based mutation
SA	Simulated annealing
SA-ES	Self-adaptive evolution strategy
SBX	Simulated binary crossover
SMS-EMOA	S-Metric Selection Evolutionary Multiobjective Optimization Algorithm
SP	Spacing metric
SPEA	Strength Pareto evolutionary algorithm
SS	Scatter search
TS	Tabu search
TSP	Travelling salesman problem
VEGA	Vector evaluated genetic algorithm
VNS	Variable neighborhood search
WSN	Wireless sensor network

ANNEXE – PUBLICATIONS SCIENTIFIQUES

Nous présentons ci-dessous les différentes publications scientifiques qui ont découlé de ce travail de thèse.

Revues scientifiques internationales

- Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. (2019). An Efficient Hybrid Evolution Strategy Algorithm with Direct Search Method for Global Optimization. *International Journal of Organizational and Collective Intelligence, 9*(3), 63-78. doi:10.4018/ijoci.2019070104 (Indexed In: INSPEC)
- Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. (in press). Solving Mono and Multi-objective Problems Using Hybrid Evolutionary algorithm and Nelder-Mead Method. *International Journal of Applied Metaheuristic Computing (IJAMC), 12*(3). (Indexed In: SCOPUS, Web of Science Emerging Sources Citation Index (ESCI), INSPEC)

Conférences internationales

- Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. (2018). A Study on Self-adaptation in the Evolutionary Strategy Algorithm. 6th IFIP International Conference on Computational Intelligence and Its Applications (CIIA), May 2018, Oran, Algeria. pp.150-160, ⟨10.1007/978-3-319-89743-1_14⟩.
- Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. "An Adaptive Hybrid Evolution Strategy Algorithm with Nelder-Mead Method for Global Optimization". In Proceedings of the third International Conference on Multimedia Information Processing (CITIM'2018). October 09-10, 2018, Mascara, Algeria
- Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. (2018d). An Adaptive Hybrid Algorithm for solving unconstrained Optimization problems. In Proceedings of the third International Symposium on Informatics and its Applications (ISIA'2018). November 6-7, 2018, M'sila, Algeria

Chapitres de Livres

- Boukhari, N., Debbat, F., Monmarché, N., & Slimane, M. (2018). A Study on Self-adaptation in the Evolutionary Strategy Algorithm. *IFIP Advances in Information and Communication Technology*. https://doi.org/10.1007/978-3-319-89743-1_14

En soumission

- An Adaptive Evolutionary Algorithm with Simplex Method and Opposition-based Learning for Solving Portfolio Optimization Problem. (Informatica journal).
- An Adaptive Hybrid Multi-Objective Differential Evolution based ranking mutation with opposing-based learning for solving multi-objective problems. (ASSA journal).