

Listes de contenus disponibles sur [ScienceDirect](https://www.sciencedirect.com)

Essaim et calcul évolutif

page d'accueil de la revue : www.elsevier.com/locate/swevo

Document d'enquête

Une prescription de lignes directrices méthodologiques pour comparer les algorithmes d'optimisation bio-inspirés



Antonio LaTorre a, , Daniel Molina b,b, Eneko Osabac, Javier Poyatos^b, Javier Del Ser cd, François Herrera b

^a Centre de simulation informatique, Université polytechnique de Madrid, Espagne b

DaSCI : Institut andalou de science des données et d'intelligence computationnelle, Université de Grenade, Espagne

^c TECNALIA, Alliance basque de recherche et de technologie (BRTA), Espagne^d Université du Pays Basque (UPV/EHU), Espagne

informations sur l'article

Mots clés:

Optimisation bio-inspirée
Analyse comparative
Réglage des paramètres
Méthodologies de comparaison
analyses statistiques
Examen des recommandations
Des lignes directrices

abstrait

L'optimisation bio-inspirée (y compris le calcul évolutif et l'intelligence en essaim) est un sujet de recherche en pleine croissance avec de nombreux algorithmes bio-inspirés compétitifs proposés chaque année. Dans un domaine aussi actif, préparer une proposition réussie d'un nouvel algorithme bio-inspiré n'est pas une tâche facile. Compte tenu de la maturité de ce domaine de recherche, proposer une nouvelle technique d'optimisation avec des éléments innovants ne suffit plus. Outre la nouveauté, les résultats rapportés par les auteurs doivent prouver qu'ils réalisent une avancée significative par rapport aux résultats précédents de l'état de l'art. Malheureusement, toutes les nouvelles propositions ne répondent pas correctement à cette exigence. Certains d'entre eux ne parviennent pas à sélectionner des repères ou des algorithmes de référence appropriés avec lesquels comparer. Dans d'autres cas, le processus de validation effectué n'est pas défini de manière principielle (ou n'est même pas fait du tout). Par conséquent, la signification des résultats présentés dans de telles études ne peut être garantie. Dans ce travail, nous passons en revue plusieurs recommandations dans la littérature et proposons des lignes directrices méthodologiques pour préparer une proposition réussie, en tenant compte de toutes ces questions. Nous nous attendons à ce que ces lignes directrices soient utiles non seulement aux auteurs, mais aussi aux examinateurs et aux éditeurs tout au long de leur évaluation des nouvelles contributions dans le domaine.

1. Introduction

Les algorithmes bio-inspirés dans le domaine de l'optimisation sont un domaine de recherche mature. Le nombre de contributions soumises aux conférences et revues de ce domaine augmente fortement chaque année [1]. Cependant, une grande partie de ces propositions ne prouvent pas de manière appropriée la qualité des nouveaux algorithmes. Il arrive souvent qu'un travail présentant un nouvel algorithme bio-inspiré soulève des doutes quant à la véritable contribution de la nouvelle proposition. Par conséquent, ces préoccupations compromettent son acceptation par la communauté des chercheurs ou, alternativement, sa capacité à évaluer la véritable contribution et l'importance de la recherche proposée.

Il y a un certain nombre de raisons à ce fait noté, allant des papiers de mauvaise qualité aux œuvres manquant d'originalité [2]. Dans ces cas, il n'y a pas grand-chose à faire si ce n'est continuer à enquêter pour obtenir de meilleurs résultats. D'un autre côté, il existe un certain nombre d'œuvres dont la contribution semble significative, mais qui ne peut être acceptée pour plusieurs raisons. Ceux-ci incluent, mais sans s'y limiter, des défauts expérimentaux, des efforts de validation douteux/insuffisants ou une discussion faible des résultats. Bien que ces pratiques puissent être facilement évitées, leur répétition

leur occurrence en fait un problème crucial : des pratiques expérimentales rigoureuses sont nécessaires pour que la communauté puisse adhérer aux conclusions tirées d'un travail de recherche, conduisant éventuellement à des avancées significatives dans ce domaine de recherche.

Plusieurs articles peuvent être trouvés avec des suggestions sur les questions importantes trouvées dans les benchmarks expérimentaux et la comparaison entre les méta-heuristiques. Chacun d'eux se concentre sur un aspect spécifique, comme la manière de concevoir les expériences [3] ou la manière de sélectionner et d'interpréter des tests statistiques pour évaluer les différences relatives entre les algorithmes [4]. Cependant, à notre connaissance, il n'existe aucun travail antérieur qui traite, en même temps, de différentes questions pertinentes qui pourraient finalement compromettre l'équité supposée dans les comparaisons de performances entre les techniques. Lorsque l'objectif est de discriminer quel algorithme fonctionne le mieux parmi un ensemble de choix possibles, l'équité doit être un principe inébranlable. Cela comprend la conception de l'indice de référence, la sélection des mesures de performance, ainsi que l'analyse et la discussion des résultats. Sans que ce principe soit garanti tout au long du déroulement expérimental, les conclusions extraites de ces études resteront incertaines.

Auteurs correspondants.

Adresses e-mail : a.latorre@upm.es (A. LaTorre), dmolina@decsai.ugr.es (D. Molina), eneko.osaba@tecnalia.com (E. Osaba), jpyatosamador@ugr.es (J. Poyatos), javier.delser@tecnalia.com (J. Del Ser), herrera@decsai.ugr.es (F. Herrera).

<https://doi.org/10.1016/j.swevo.2021.100973>

Reçu le 18 avril 2020 ; Reçu sous forme révisée le 23 juin 2021 ; Accepté le 13 août 2021

Disponible en ligne le 20 août 2021 2210-6502/© 2021 Les auteurs. Publié par Elsevier

BV Ceci est un article en libre accès sous licence CC BY-NC-ND (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

L'objectif principal de ce manuscrit s'aligne sur les remarques ci-dessus.

Plus précisément, nous passons en revue le contexte de la littérature et fournissons un ensemble de lignes directrices utiles destinées aux chercheurs pour éviter les erreurs courantes dans les expériences avec des méta-heuristiques bio-inspirées. Ces mauvaises pratiques pourraient éventuellement générer des doutes sur l'équité des comparaisons qui y sont rapportées. Notre approche méthodologique est pragmatique, visant principalement à permettre aux nouveaux chercheurs entrant dans ce domaine de préparer plus facilement une section expérimentale selon des normes de qualité élevées. Nous commençons notre étude en explorant les approches actuelles d'analyse des algorithmes. Nous portons une attention particulière aux mauvaises pratiques, identifiées non seulement dans ce travail mais aussi dans la littérature antérieure. Toutes ces informations se trouvent dans la [section 2](#).

Ensuite, nous proposons 4 lignes directrices que les auteurs introduisant une nouvelle proposition algorithmique devraient prendre en compte pour augmenter leurs chances de faire adopter leur travail par la communauté. Nous fournissons une brève motivation pour chacun d'eux dans les paragraphes suivants, tandis que les [sections 3 à 6](#) présentent des directives méthodologiques détaillées pour l'élaboration de propositions réussies :

- Ligne directrice #1 : Repères. Parfois, la référence est un problème du monde réel.

Dans ces cas, le benchmark mesure comment l'algorithme proposé s'attaque au problème posé. En revanche, dans d'autres cas, la proposition est comparée à d'autres algorithmes de référence en utilisant un benchmark spécialement conçu pour tester leurs performances. Dans tous les cas, la sélection de la référence appropriée est une question importante, car les conclusions qui peuvent être extraites de l'étude dépendent profondément du banc d'essai. Malheureusement, les benchmarks choisis présentent fréquemment certaines caractéristiques qui pourraient favoriser les algorithmes avec un biais particulier. Ceci n'est bien sûr pas souhaitable pour des raisons d'équité dans les comparaisons ultérieures. Ainsi, les résultats obtenus par le solveur nouvellement proposé doivent être analysés en prenant en compte les différentes caractéristiques des problèmes de test couverts par le benchmark en question.

- Ligne directrice #2 : Validation des résultats. La présentation des résultats bruts disposés dans des tableaux pleine page ne suffit plus aujourd'hui. Une bonne validation des résultats d'un point de vue statistique doit toujours être fournie avec les tableaux susmentionnés. En ce sens, il est important que non seulement des tests statistiques soient utilisés, mais aussi que les bons soient appliqués. Il est assez courant de trouver des tests paramétriques qui sont utilisés sans s'assurer que les hypothèses requises pour ces tests sont satisfaites par les résultats obtenus. De plus, nous recommandons également des techniques de visualisation pour l'analyse comparative. Ils peuvent résumer une quantité importante d'informations dans une représentation condensée qui peut être rapidement saisie et interprétée par le lecteur.

- Directive #3 : Analyse des composantes et ajustement des paramètres de la proposition. Les hypothèses de la proposition doivent être clairement énoncées en début d'article, et discutées une fois les résultats validés. De plus, les auteurs doivent mener une analyse approfondie des résultats en considérant, au moins, les aspects suivants : identification des phases de recherche (équilibre entre exploration et exploitation), analyse des composantes et de la complexité (analyse individuelle de la contribution de chacune des composantes de la méthode globale et leur complexité), ajustement des paramètres de l'algorithme et comparaison statistique avec des algorithmes de pointe (tel que décrit dans la ligne directrice n° 2).
- Directive #4 : Pourquoi mon algorithme est-il utile ? Enfin, les contributeurs potentiels doivent clairement indiquer pourquoi leur algorithme proposé devrait être considéré comme pertinent par le reste de la communauté. Dans cette ligne directrice, nous discutons de cette question en profondeur à partir de différents points de vue.

Nous suggérons également plusieurs raisons pour lesquelles une nouvelle proposition pose une avancée dans les connaissances (c'est-à-dire qu'elle est jugée compétitive par rapport aux méthodes de l'état de l'art, qu'elle présente des apports méthodologiques qui stimulent d'autres recherches, ou d'autres raisons élaborées ultérieurement).

Afin d'illustrer chacun des problèmes abordés dans cette contribution, nous recourons à différents cas d'utilisation issus de notre expérience antérieure ou spécialement adaptés aux besoins de cette étude. Les données utilisées pour chacun de ces exemples de problèmes peuvent varier, car tous

les situations peuvent être clairement expliquées avec un seul exemple. Nous fournissons également plusieurs études de cas dans la [section 7](#) qui décrivent le processus de conception et d'évaluation de nouveaux algorithmes selon la méthodologie proposée ici. Chacun d'eux reprend l'ensemble des lignes directrices méthodologiques en sélectionnant dans un premier temps un benchmark standard, une mesure de performance (benchmark). Ensuite, nous procédons à une bonne validation statistique des résultats par rapport à ceux des algorithmes de référence. Nous utilisons également des techniques de visualisation pour offrir une vision plus claire des résultats. Pour continuer, la contribution de chaque composant du nouvel algorithme est analysée pour s'assurer que tous contribuent aux résultats de la méthode globale. Chaque étude de cas se termine par une discussion sur l'utilité de la nouvelle proposition. Comme on peut le constater, il couvre bien l'ensemble des orientations méthodologiques proposées dans cette contribution.

En résumé, les principaux éléments clés de ce travail sont :

1. Une revue de la littérature, en mettant l'accent sur l'identification des mauvaises pratiques dans l'analyse de nouvelles propositions algorithmiques.
2. Quatre lignes directrices méthodologiques pour aider les auteurs à réaliser des contributions adoptées par la communauté.
3. Plusieurs études de cas, telles que décrites dans le paragraphe précédent, qui simulent le processus de proposition d'un nouvel algorithme en suivant les quatre lignes directrices méthodologiques susmentionnées.

Le reste de cet article est organisé comme suit. La [section 2](#) traite de plusieurs lignes directrices et recommandations utiles antérieures dans la littérature. Les [sections 3 à 6](#) présentent et discutent des lignes directrices proposées dans ce travail, tandis que dans la [section 7](#), nous fournissons plusieurs études de cas couvrant certaines de ces lignes directrices. Enfin, la [section 8](#) conclut l'étude.

2. Questions pertinentes pour la proposition de lignes directrices méthodologiques

Dans tout domaine scientifique, il est crucial de travailler dans des conditions expérimentales correctes et impartiales, et de procéder à une analyse rigoureuse et adéquate des résultats obtenus. Cependant, il existe parfois de petits aléas qui peuvent conduire à des comparaisons biaisées par inadvertance, profitant partiellement à un certain type d'algorithmes par rapport aux autres.

Dans cette section, nous passons en revue les travaux antérieurs dans la littérature, conseillant de manière constructive contre les problèmes qui pourraient générer des doutes objectifs sur la force des affirmations expérimentales. Plus précisément, nous revisitons plusieurs sujets particuliers pertinents pour la présente étude : une description inadéquate ou incomplète d'un algorithme ([Section 2.1](#)), la présence de biais dans le processus de recherche ([Section 2.2](#)), des caractéristiques pertinentes à prendre en compte lors de la sélection des repères ([Section 2.3](#)), les études antérieures se sont concentrées sur la validation des résultats expérimentaux ([Section 2.4](#)) et les travaux existants sur l'analyse des composants et le réglage des paramètres ([Section 2.5](#)). Les sujets abordés dans cette première analyse contextuelle ont un lien direct avec nos orientations méthodologiques données dans les [sections 3 à 6](#).

2.1. Description inappropriée de l'algorithme

Il s'agit d'un problème commun à de nombreuses propositions, en particulier dans celles de méthodes plus avancées. La reproductibilité des résultats scientifiques devrait toujours être exigée et cela n'est pas possible si certains détails de mise en œuvre manquent [[5,6](#)]. Cela comprend non seulement une description de haut niveau de l'algorithme, mais également des détails de mise en œuvre, des dépendances, des valeurs de paramètres, etc. De plus, si la proposition est basée sur une méthode préexistante, les différences par rapport à l'algorithme de base doivent être stressés. Enfin, et bien que cela ne soit pas encore obligatoire dans la plupart des revues et conférences, nous encourageons les auteurs à diffuser librement le code source de leurs algorithmes, pour permettre à d'autres utilisateurs de mieux répliquer leurs résultats.

2.2. Biais dans le processus de recherche

L'une des décisions les plus critiques lors de l'évaluation d'une proposition algorithmique est la sélection de la référence utilisée pour montrer sa qualité.

Malheureusement, pour de nombreux articles, le banc d'essai a été proposé par le même

auteurs, et est normalement une combinaison de fonctions théoriques synthétiques bien connues. De plus, la seule mesure de la performance est souvent le benchmark proposé. La conception d'un bon benchmark n'est pas une tâche facile, et ils peuvent être utilisés pour bénéficier des méthodes nouvellement proposées en exploitant tout biais dans l'algorithme de recherche [7].

- Optima proches du centre du domaine de recherche : une des sources de biais les plus typiques est la tendance de certains algorithmes à explorer avec plus d'intensité les environs du centre du domaine de recherche. Ce problème a été discuté pour la première fois et appelé biais structurel dans [8]. De nombreuses versions des algorithmes génétiques (GA), de l'optimisation des essais de particules (PSO) ou de l'évolution différentielle (DE) [9], ont exploité cette caractéristique, car c'est traditionnellement là où se situait l'optimum du problème analysé. Ces algorithmes, en revanche, ont tendance à présenter de mauvaises performances près des limites de la recherche de domaine [10]. Dans [11] et [12] une expérimentation détaillée sur le biais structurel dans les algorithmes de recherche est donnée. Éviter ce genre de biais dans la conception des algorithmes n'est pas aisé, mais au moins ils ne devraient pas être évalués sur des benchmarks favorisés par ces biais lors de l'exploration. Une approche populaire pour éviter d'avoir l'optimum au centre de la recherche de domaine est le déplacement.
- Sensibilité au repère : une autre source possible de biais apparaît lorsque l'exploration du domaine de recherche se fait principalement en se déplaçant le long des directions du repère. A cet égard, certains algorithmes se sont révélés très sensibles au système de coordonnées [13]. Certaines fonctions de référence sont tournées pour tester l'invariance des algorithmes à de telles transformations. Idéalement, l'algorithme devrait être invariant à ces rotations, comme Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [14]) et Black-Box DE [13].

Afin de comparer des algorithmes conçus pour éviter ces sources de biais, plusieurs benchmarks prenant en compte ces enjeux ont été proposés. Cela permet une comparaison plus juste entre les propositions algorithmiques, qui peuvent être comparées sur le même banc de test. En particulier, en optimisation des paramètres réels, plusieurs benchmarks ont été proposés depuis 2005 à ce jour [15–19]. Tous ces repères tentent d'éviter la première source de biais en déplaçant l'emplacement de l'optimum global. De plus, dans les bancs d'essai les plus récents, un nombre croissant de fonctions ont été tournées, présentant des paysages plus complexes. Une vue plus détaillée des différents benchmarks proposés et de leur évolution peut être trouvée dans [20].

Pour terminer sur cette question, les benchmarks devraient certainement évaluer la sensibilité des algorithmes à différentes sources de biais comme celles mentionnées précédemment. Cependant, lorsque l'on passe du domaine académique à des scénarios réels, il est important de garder à l'esprit que l'objectif n'est pas de trouver une bonne approche méta-heuristique, mais plutôt de résoudre efficacement un problème donné. Ceci dit, la disponibilité d'informations a priori sur tout biais du problème à résoudre doit être exploitée. En effet, si un solveur exploite une certaine source de biais connue pour exister dans le problème, alors ce solveur doit être préféré. En fait, les benchmarks sont utiles pour identifier les algorithmes avec une bonne performance sur des problèmes avec des sources de biais similaires. Ces réflexions concordent avec les travaux récents autour de l'exploitation des connaissances spécifiques à un problème lors de la conception d'un algorithme d'optimisation visant à le résoudre efficacement (optimisation de la boîte grise, voir par exemple [21] et ses références). En résumé, si l'objectif de l'étude est de résoudre un problème donné, tirer parti des sources possibles de biais lors de la conception de l'algorithme est pratique et conseillé.

2.3. Caractéristiques pertinentes des benchmarks

Différentes propriétés liées au paysage des fonctions composant le benchmark doivent être abordées pour une analyse juste du comportement du ou des solveurs proposés. Parmi eux :

- Séparabilité des composants : certaines fonctions peuvent être facilement résolues en optimisant chaque dimension individuellement, il est donc crucial

de ne pas utiliser uniquement des fonctions séparables dans le cas-test. C'est le cas, par exemple, du benchmark CEC'2008 LSGO proposé dans [22], dans lequel de nombreuses fonctions sont de ce type. Dans des benchmarks plus récents, en particulier dans le domaine de l'optimisation à grande échelle, l'accent est mis sur l'évaluation de la capacité des algorithmes à identifier les sous-composants existants dans les fonctions. Si la nouvelle proposition traite de ce type de problème, elle devrait être testée sur un benchmark permettant d'évaluer cette caractéristique, comme par exemple celui proposé dans [23].

- Dimensionnalité des problèmes : une autre question importante concerne les dimensions du benchmark, car certains algorithmes sont conçus pour fonctionner correctement uniquement pour des problèmes de très faible dimension. Comme la taille du domaine de recherche augmente de façon exponentielle avec la dimensionnalité du problème, la soi-disant malédiction de la dimensionnalité [24] pose un défi informatique important. C'est notamment le cas pour des algorithmes bien connus tels que PSO [25] et DE [26]. En effet, comme les performances de la plupart des algorithmes se dégradent lorsque la dimension grandit, la tendance actuelle est de développer des algorithmes spécifiques pour les problèmes de plus grande dimensionnalité. Néanmoins, il est de plus en plus important d'offrir des performances robustes pour une gamme moyenne de valeurs de dimension. Certains benchmarks sont conçus pour évaluer les performances des algorithmes dans des problèmes de taille petite à modérée [15–19], tandis que d'autres visent des problèmes de taille beaucoup plus grande. Ce sont des problèmes d'une nature très différente et, normalement, les algorithmes avec une performance exceptionnelle sur un type de problèmes ne fonctionnent pas comme tels lorsqu'ils sont appliqués à d'autres types de problèmes. Par exemple, des stratégies telles que le calcul de la matrice de covariance des solutions comme le fait [14] ne s'adaptent pas bien aux problèmes de plus grande taille. De plus, ces derniers problèmes nécessitent des capacités d'exploration accrues de l'algorithme pour couvrir un espace de recherche beaucoup plus large, ce qui a un coût important en termes d'évaluations de fitness.
- Nombre d'optima des problèmes : Il y a des fonctions qui n'ont qu'un seul optimum, et d'autres qui ont plusieurs optima appelées fonctions multimodales. Les fonctions multimodales peuvent avoir plusieurs optima globaux, avec la même valeur de fitness, et également plusieurs optima locaux, avec des valeurs de fitness différentes (pires). La présence d'optimums locaux augmente la difficulté du processus d'optimisation car l'algorithme peut y être bloqué. C'est le cas des algorithmes à fort comportement élitiste [28]. L'augmentation de la difficulté est due au fait que les algorithmes d'optimisation en boîte noire ne connaissent pas le nombre d'optimums locaux, bien que plusieurs travaux aient proposé des méthodes pour estimer leur nombre [29].
- Problèmes de conditionnement élevé : une fonction de conditionnement élevé ou mal conditionnée est une fonction dans laquelle un petit changement dans les variables de la solution implique un grand changement dans sa valeur de fitness.

Cela signifie que la solution/réponse correcte au problème devient difficile à trouver pour les algorithmes d'optimisation [30,31].

- Fonctions bruitées : enfin, le bruit est un autre facteur important peu pris en compte dans la littérature. Cependant, cela change récemment dans plusieurs benchmarks récents qui considèrent également que cela est sué. Ces benchmarks, tels que les benchmarks BBOB [32] ou Nevergrad [33], incluent explicitement des fonctions avec différents degrés de bruit qui ressemblent à des scénarios réels dans lesquels le bruit peut être un enjeu très important. Malgré cet intérêt récent, très peu d'études ont jusqu'ici traité des fonctions bruitées [34,35].

2.4. Validation des résultats

La sélection des algorithmes concurrents à inclure dans la comparaison est un autre aspect crucial du benchmarking. Dans la littérature actuelle, de nombreux algorithmes différents peuvent être trouvés et choisis comme algorithmes de référence pour un benchmark donné. Malheureusement, il n'y a pas de critère clair pour faire une telle sélection. Bien que les bonnes pratiques suggèrent généralement de comparer avec les algorithmes de pointe les plus récents, il arrive souvent que les auteurs ne comparent leur proposition qu'avec des versions de base ou très similaires d'autres algorithmes, comme cela a été repéré dans [36].

Dans ce contexte, les études comparant différents algorithmes sont rares, et les résultats qui y sont rapportés dépendent fortement du ou des problèmes. Dans [37], un benchmark des fonctions classiques a été utilisé pour comparer entre Cuckoo Search (CS), PSO, DE et Artificial Bee Colony (ABC). L'étude conclut que les meilleurs résultats ont été obtenus par CS et DE, et les pires ont été ceux rendus par ABC. D'autre part, pour un problème différent [38], ABC s'est avéré le plus performant, suivi de DE et PSO.

Pour la partie méthodologique des comparaisons, il existe beaucoup plus d'études. Les tests statistiques, par exemple, sont au cœur des contributions antérieures sur ce sujet. Cependant, de telles contributions sont fréquemment rédigées d'un point de vue statistique – comme celle de Demšar [39] – ce qui rend difficile pour les chercheurs dans ce domaine d'adopter leurs recommandations méthodologiques. Plus récemment, des tutoriels ont tenté de rapprocher les domaines des méta-heuristiques et des statistiques inférentielles [4].

Quelques exemples peuvent être trouvés dans [40], dans lequel un traitement statistique est proposé pour distinguer les mesures de performance dans les algorithmes évolutionnaires adaptatifs. Un autre bon exemple est [41], qui montre que dans un benchmark à paramètres réels populaire (CEC'2005), les conditions nécessaires pour exécuter des tests d'hypothèses paramétriques n'étaient pas remplies, et des tests non paramétriques ont donc été recommandés. Plus récemment, dans [42], des recommandations pour la comparaison d'algorithmes évolutionnaires sont fournies, qui peuvent même être extrapolées à des benchmarks d'apprentissage automatique.

Une autre question importante d'un point de vue méthodologique est l'évaluation de la performance des algorithmes bio-inspirés du point de vue du design expérimental. Certaines études [5] fournissent des recommandations générales pour concevoir des expériences de comparaison d'algorithmes d'une manière similaire à ce que nous faisons dans cette contribution. Cependant, ces recommandations sont beaucoup plus générales car elles visent un périmètre plus large (la conception d'algorithmes et non de méthodes d'optimisation bio-inspirées, notamment). Cette différence dans la cible des lignes directrices proposées fait passer à côté de certains enjeux importants inhérents aux méthodes d'optimisation bio-inspirées que nous couvrons dans cette contribution. D'autres travaux, bien que focalisés sur les méthodes d'optimisation, sont plus précis dans leurs recommandations, ciblant des problématiques spécifiques telles que la sélection des problèmes et les mesures de performance [43–45]. Enfin, d'autres études sont plus orientées vers l'analyse et la définition de cadres expérimentaux tels que ceux utilisés dans les sessions spéciales de la CEC et le cadre COCO [6]. Bien qu'il s'agisse également d'une contribution très pertinente au domaine de la conception expérimentale dans le contexte des algorithmes évolutionnaires, il traite le problème sous un angle différent et doit être considéré comme un complément aux lignes directrices présentées ici.

2.5. Analyse des composants et réglage des paramètres

Les nouvelles propositions sont fréquemment basées sur des algorithmes préexistants auxquels certains composants ont été ajoutés/remplacés. Cependant, l'ajout/le remplacement de nouveaux composants n'est pas toujours suffisamment justifié. Il s'agit d'un défaut de conception fondamental qui peut contribuer à rendre les algorithmes de plus en plus complexes dans lesquels les nouveaux ajouts ne rapportent qu'une contribution marginale à la performance globale de la méthode. Les auteurs doivent clairement discuter de la contribution de chaque nouveau composant afin que la nouvelle proposition soit considérée comme significative.

Un autre sujet de recherche important dans la conception d'algorithmes méta-heuristiques est la sélection des valeurs de leurs paramètres. En effet, les paramètres peuvent être une épée à double tranchant. D'une part, ils accordent une flexibilité pour contrôler le comportement de recherche de l'algorithme. D'un autre côté, trouver les valeurs des paramètres qui conduisent aux meilleures performances de recherche est un autre problème d'optimisation lui-même [46]. Pour cette raison, il existe une longue littérature d'études traitant des meilleures valeurs de paramètres pour différents algorithmes méta-heuristiques, tels que GA [47], PSO [48] ou DE [26,49].

Le réglage des paramètres peut être effectué au moyen de différents outils de réglage automatique. Il existe plusieurs outils consolidés de cette nature,

avec des caractéristiques différentes [50]. F-RACE [51] et I-RACE [52] sont des modèles itératifs qui, à chaque étape, évaluent un ensemble de configurations de paramètres candidats, en écartant plusieurs d'entre eux tout au long de la recherche. Ces méthodes éliminent les candidats sur le résultat de comparaisons statistiques, par exemple une analyse de variance à deux facteurs. I-RACE est une implémentation de l'itérative F-RACE qui comprend plusieurs extensions, comme un redémarrage à l'aide de distributions normales. REVAC [53], au contraire, s'appuie sur un algorithme d'estimation de distribution (EDA). Pour chaque paramètre, REVAC commence par échantillonner une distribution uniforme de valeurs. Ensuite, à chaque étape, il réduit la plage de valeurs de chaque paramètre en utilisant des opérateurs de transformation spécialement conçus, en considérant une mesure d'entropie. ParamILS [54] est un algorithme itératif de recherche locale qui, à partir d'une configuration de paramètres par défaut, applique une recherche locale avec des perturbations aléatoires pour améliorer les configurations.

3. Ligne directrice n° 1 : Repères

La première décision à laquelle un chercheur doit faire face lorsqu'il prépare une nouvelle contribution dans le domaine de l'optimisation est la sélection de la référence pour tester le ou les algorithmes nouvellement proposés. Une fois cela fait, les auteurs doivent identifier des algorithmes pertinents pour comparer les résultats obtenus et garantir la pertinence des conclusions qui en sont tirées.

Cette première ligne directrice traite de ces deux aspects cruciaux : la sélection du benchmark (Section 3.1), et le ou les algorithmes de référence auxquels la proposition est comparée (Section 3.2).

3.1. Sélection du benchmark

Comme mentionné précédemment, cette première décision est l'un des facteurs les plus importants pour prouver la qualité et les performances d'un algorithme. Dans les problèmes du monde réel, ce n'est pas du tout une décision, car la référence est le problème à résoudre. En revanche, lors de la conception et de l'amélioration de techniques méta-heuristiques, la sélection d'un benchmark est une décision importante à prendre. Ce problème est commun à tous les types d'optimisation. Au cours des dernières années, des benchmarks ont été proposés pour plusieurs types d'optimisation (telles que l'optimisation combinatoire et numérique) dans le but principal de devenir un standard pour les comparaisons futures. Sans perte de généralité, dans la section suivante nous nous concentrerons sur l'optimisation numérique, cependant toutes les conclusions et recommandations données ci-après sont applicables quel que soit le domaine.

Dans le domaine de l'optimisation numérique, des sessions spéciales consacrées au benchmarking ont eu lieu dans des événements de référence tels que le IEEE Congress on Evolutionary Computation ou la Genetic and Evolutionary Computation Conference. Dans ces événements, les participants pouvaient comparer leurs algorithmes dans un environnement contrôlé avec un ensemble homogène de fonctions [20, 55]. Néanmoins, les efforts visant à fournir des références et des outils standard pour la comparaison des méthodes d'optimisation bio-inspirées ne se limitent pas aux sessions spéciales et aux compétitions, comme indiqué ci-dessous :

- Sessions spéciales et ateliers :
 - Session spéciale IEEE Real-Coding (depuis 2005) • Ateliers Black-Box Optimization Benchmarking (BBOB) à GECCO et PPSN (depuis 2009)
 - Black-Box Optimization Competition (BBComp) (depuis 2015) • IEEE Large-scale Global Optimization Special Session (depuis 2008)
 - Atelier Benchmarking au GECCO et PPSN (depuis 2020) • Autres sessions spéciales (plus spécifiques) : contraintes, multimodales, monde réel, etc. • Outils :
- Cadre de comparaison des optimiseurs continus (COCO)¹ • IOHProfiler²

¹ <https://github.com/tusar/coco/> [https://](https://github.com/tusar/coco/)

² [iohprofiler.github.io/](https://github.com/tusar/coco/)

- Nevergrad³
- TACO : boîte à outils pour la comparaison automatique des optimiseurs⁴
- Benchmarks CEC de PN Suganthan⁵ • Benchmark IEEE CEC LSGO⁶

• Autres initiatives :

- Réseau d'analyse comparative⁷
- Cost Action CA15140 (ImAppNio)⁸

Cependant, certains travaux ignorent tout simplement ces repères et outils standards. Au lieu de cela, ils choisissent plutôt leur propre sous-ensemble de fonctions pour évaluer leur proposition. Ceci est problématique pour plusieurs raisons. Premièrement, comme il ressort de la section 2.2, il est très difficile de savoir s'il existe un biais dans la sélection des fonctions du point de vue des performances de l'algorithme considéré. Deuxièmement, de nombreuses fonctions de référence différentes existent (ou peuvent être définies). Par conséquent, il devient très difficile d'évaluer les différentes caractéristiques de toutes ces fonctions de référence. Enfin, les comparaisons avec d'autres méthodes de référence impliquent généralement de les exécuter par les mêmes auteurs de la nouvelle méthode, car il est très peu probable que plusieurs études se soient concentrées sur les mêmes fonctions sélectionnées ou conditions expérimentales. En conséquence, évaluer la qualité d'une nouvelle contribution qui n'utilise pas de critères de référence standard devient presque impossible à réaliser, et ne devrait donc pas être crédité par la communauté.

Cependant, il existe deux situations particulières dans lesquelles les chercheurs n'ont pas d'autre choix que d'utiliser des instances de problème générées ad hoc : i) lorsque le problème à résoudre n'a jamais été abordé dans la littérature précédente, et donc qu'aucune référence ne peut être trouvée ; ou ii) lorsqu'un problème du monde réel est à l'étude, avec des exigences et des contraintes spécifiques. Dans ces cas, le processus de génération d'instance doit être profondément détaillé, et toutes les instances générées doivent être partagées pour que d'autres chercheurs puissent reproduire et améliorer les résultats présentés. De plus, pour chacune de ces deux alternatives, les praticiens doivent générer une référence aussi réaliste et générale que possible.

D'autre part, il convient également d'être très prudent dans le choix d'un référentiel pour l'expérimentation à mener. Les benchmarks de la littérature ont été conçus avec certains objectifs en tête, et sont appropriés pour tester certaines caractéristiques des algorithmes évalués. Une liste non exhaustive de ces caractéristiques suit :

- Évitements des biais de l'algorithme de recherche : Afin d'éviter les problèmes décrits dans la section 2.2, il est fortement conseillé que l'optimum ne soit pas situé au centre du domaine de recherche (par exemple par décalage). De plus, la rotation doit être appliquée pour tester la sensibilité de l'algorithme au système de coordonnées.
- Sensibilité au nombre d'optima locaux : Le nombre d'optima locaux d'une fonction est une autre caractéristique importante d'un problème de test. À moins d'être correctement pris en compte dans la conception algorithmique, les espaces de recherche avec plusieurs optima locaux peuvent affecter négativement la convergence des métaheuristiques vers l'optimum global. Dans ce genre de problèmes, de multiples optima locaux peuvent agir comme des bassins d'attraction, empêchant l'algorithme d'atteindre les optima globaux s'il n'équilibre pas correctement son rapport exploration/exploitation.

En outre, les benchmarks établissent normalement les conditions expérimentales dans lesquelles les algorithmes doivent être évalués. En particulier, il est très fréquent qu'un benchmark sélectionne un point commun :

- Mesure de performance : Dans les algorithmes évolutionnaires, il est courant d'utiliser l'erreur moyenne obtenue pour les différentes exécutions. Cependant, il existe des mesures et des indicateurs de performance plus adéquats pour l'optimisation dynamique [56] et l'optimisation multi-objectifs.

[57]. De plus, d'autres indicateurs de performance, tels que le temps d'exécution de l'empreinte mémoire, pourraient également fournir des informations intéressantes sur le comportement des algorithmes (en particulier dans des scénarios réels). Néanmoins, cela doit être soigneusement considéré car ces mesures peuvent être biaisées en fonction de facteurs externes tels que le langage de programmation choisi et non l'algorithme lui-même, ce qui peut entraver la comparaison.

- Critère d'arrêt : Pour qu'une comparaison de plusieurs algorithmes soit équitable, tous doivent fournir un effort similaire pour trouver une solution. Ceci est normalement réalisé en établissant un critère d'arrêt commun. Si ce n'était pas le cas, un algorithme pourrait s'arrêter lorsqu'une précision prédéfinie est atteinte alors que d'autres algorithmes pourraient fonctionner jusqu'à ce qu'un budget maximum d'évaluations de fitness soit épuisé. Les résultats des deux algorithmes ne sont pas comparables et c'est pourquoi la plupart des benchmarks définissent un critère d'arrêt commun. Cela permet également de saisir une image complète des performances des algorithmes, car différentes méthodes peuvent produire des taux de convergence différents, et les résultats de la comparaison peuvent différer selon le point de contrôle auquel ils sont évalués. Ce problème se posera et sera discuté dans les cas d'utilisation présentés dans la section 7.

Enfin, il existe une autre caractéristique pertinente du point de vue de la conception qui n'est pas spécifiquement liée aux fonctions elles-mêmes, mais aux algorithmes qui résolvent ces fonctions. Les algorithmes utilisent la fonction de fitness pour guider le processus de recherche, mais parfois seul le classement des solutions calculé à partir des valeurs de fitness est réellement utilisé. En ce sens, certains algorithmes récents comme l'algorithme Firefly [58] ou l'algorithme d'optimisation Grasshopper [59] utilisent les informations quantitatives fournies par la fonction de fitness pour guider le processus de recherche, tandis que d'autres proposent des méthodes de mutation et/ou de sélection [60] qui demandent seulement de savoir si une solution est meilleure que l'autre. Dans certains scénarios réels, la dernière approche peut être très bénéfique car elle simplifie le processus de définition de la fonction de fitness.

3.2. Sélection des algorithmes de référence

Une autre question importante, qui est en fait liée à la directive précédente, est la sélection des algorithmes de référence à inclure dans la comparaison. D'une part, si l'algorithme proposé s'appuie sur d'autres algorithmes de base, ceux-ci doivent être inclus dans la comparaison pour vérifier la contribution individuelle de chacun d'eux, comme nous le verrons en détail dans la section 5. D'autre part, une fois que le benchmark a été sélectionné, les meilleures méthodes à ce jour pour ce benchmark particulier doivent également être prises en compte dans la comparaison. Malheureusement, de nombreux articles ne parviennent pas à comparer leur algorithme proposé avec des homologues concurrents [36].

Une expérimentation bien informée devrait, au moins, inclure les meilleurs algorithmes dans la session spéciale où le benchmark a été proposé à l'origine (comme c'est généralement le cas). Nous nous référons à [20] pour une revue mise à jour sur les sessions spéciales et les compétitions sur l'optimisation continue.

Enfin, les auteurs doivent également considérer des algorithmes similaires, non seulement de la même famille (par exemple, PSO, DE, GA ...) mais aussi l'algorithme de base, si la proposition est une amélioration par rapport à un algorithme précédent, ou d'autres approches similaires (par exemple exemple, différents algorithmes mémétiques avec une recherche locale commune). Notre revendication à cet égard est d'arrêter de comparer de nouvelles méthodes avec des algorithmes classiques qui ont été clairement dépassés par les plus récents. Les comparaisons adoptant cette stratégie trompeuse sont à éviter pour l'apport scientifique discutable de leur proposition.

4. Ligne directrice #2 : Validation des résultats

Une procédure de validation fondée sur des principes pour le benchmark est tout aussi importante qu'une conception d'expérimentation correcte (voir la ligne directrice n° 1). Pour cela, nous mettons l'accent sur deux outils différents : l'analyse statistique et l'analyse visuelle comparative. Les deux approches sont couvertes dans les deux sous-sections suivantes : l'analyse statistique (Section 4.1) et les techniques de visualisation pour comparer les méta-heuristiques (Section 4.2).

³ <https://github.com/facebookresearch/nevergrad>

⁴ <https://tacolab.org/> <https://github.com/PN-Suganthan>

⁵ Suganthan http://www.tlsgo.org/special_sessions/

⁶ cec2019.html#original-code <https://sites.google.com/view/benchmarking-network/>

⁷ <https://imappnio.dcs.aber.ac.uk/>

⁸

Tableau
1 Tests recommandés selon les caractéristiques des données.

Conditions	Variances égales	Écarts inégaux
Normalement distribué	Test t de Student apparié Test t de Welch apparié	
Pas normalement distribué	Test de rang signé de Wilcoxon	

4.1. Analyse statistique : tests nonparamétriques et au-delà

La comparaison statistique des résultats devrait être considérée comme obligatoire dans les benchmarks actuels entre algorithmes bio-inspirés. Cependant, même si des comparaisons statistiques sont faites dans les études rapportées de nos jours, elles ne sont pas toujours effectuées correctement. Dans les tests d'hypothèses inférentielles, il existe certaines méthodes populaires, telles que le test t ou la famille de tests ANOVA. Cependant, ces tests sont qualifiés de tests paramétriques car ils supposent une série d'hypothèses sur les données auxquelles ils sont appliqués (c'est-à-dire sur les paramètres de la distribution sous-jacente des données). Si de telles hypothèses ne tiennent pas (par exemple, l'hypothèse de normalité des résultats), la fiabilité des tests n'est pas garantie et des approches alternatives doivent être envisagées. Ainsi, soit ces conditions sont vérifiées pour être vraies (c'est-à-dire en utilisant un test de normalité pour prouver la normalité dans la distribution), soit un autre type de test qui ne fait pas ces hypothèses doit être utilisé à la place. C'est le cas des tests non paramétriques, qui ne supposent pas de caractéristiques particulières pour la distribution sous-jacente des données. Ce caractère non paramétrique doit être vu comme un avantage par rapport aux tests paramétriques précités (pour leur indépendance vis-à-vis des données) mais aussi comme une limitation (les tests non paramétriques sont moins puissants) imposée par la nature des données sous-jacentes, que ne satisfont pas aux exigences d'utilisation des méthodes paramétriques plus puissantes.

En conséquence, les tests paramétriques doivent être privilégiés chaque fois qu'ils peuvent être utilisés en toute sécurité (c'est-à-dire chaque fois que les hypothèses sur la distribution sous-jacente des données sont satisfaites). Malheureusement, cela n'est souvent pas le cas lorsque l'on compare les résultats d'algorithmes bio-inspirés. Par conséquent, des tests non paramétriques devraient être utilisés à la place [4]. Une erreur courante (moins fréquente dans les recherches actuelles) consiste à appliquer des tests paramétriques sans vérifier si les hypothèses requises sont satisfaites.

Dans le paragraphe suivant, nous fournissons un flux de travail pour décider du type de test à choisir. Il se compose des étapes suivantes :

- Vérifier les conditions requises pour l'application de la paramétrique test de choix (normalement, le test t de Student).
 - Normalité : test de Shapiro-Wilk [61] ou Kolmogorov-Smirnov.
Shapiro-Wilk doit être utilisé avec des échantillons de plus petite taille [62].
 - Homocédasticité (variances égales) : test de Levene [63].
- Si les deux conditions sont satisfaites, appliquer le test t de Student [64].
- Si seule la normalité peut être garantie, alors l'alternative du test t de Welch est envisagée [65].
- Si aucune des hypothèses sur la distribution sous-jacente n'est vérifiée, le test non paramétrique des rangs signés de Wilcoxon est utilisé [66].

Une fois le test approprié sélectionné, la comparaison peut être effectuée. Tout d'abord, le classement de chaque algorithme sur l'ensemble du benchmark doit être calculé, et la signification des différences dans les valeurs de classement doit être testée. Le test de somme des rangs de Friedman peut servir à cette fin [67]. Si les différences sont déclarées statistiquement significatives par le test de Friedman, nous procédons aux comparaisons par paires avec le test sélectionné dans le tableau 1. Dans ces comparaisons par paires, un algorithme de référence est comparé à toutes les autres méthodes sélectionnées pour validation. Normalement, l'algorithme de référence est choisi pour être celui avec le meilleur classement moyen ou, alternativement, la nouvelle proposition présentée dans l'ouvrage.

Un autre oubli typique noté dans la littérature est de négliger l'erreur accumulée. Un test statistique pour deux échantillons, comme le test de Wilcoxon, a une erreur estimée, mais cette erreur augmente avec chaque paire de comparaisons. Ainsi, en comparant simultanément les résultats de notre proposition avec ceux obtenus par plusieurs autres algorithmes, l'application

du test de Wilcoxon (ou d'autres tels que le test t) est totalement déconseillé, car il ne peut garantir que la proposition est statistiquement meilleure que tous les autres algorithmes de référence. Ainsi, une fois les p-values par paires calculées, une méthode de correction doit être utilisée pour contrecarrer l'effet des comparaisons multiples, en contrôlant soit le taux d'erreur par famille, soit le taux d'erreur de fausse découverte [68]. Plusieurs procédures ont été proposées à cette fin, parmi lesquelles Bonferroni-Dunn [69], Holm-Bonferroni [70], Hochberg [71] et Hommel [72] sont les plus utilisées [4].

Également liée à la validation statistique, une autre recommandation est de fournir les p-values des tests effectués dans l'expérimentation. Cependant, nous notons que la valeur p, en tant que telle, n'est pas une mesure totalement objective, car elle dépend fortement de la taille de l'échantillon [73]. La section 7 présente plusieurs exemples de comparaisons qui passent par les étapes méthodologiques prescrites dans cette deuxième ligne directrice.

Traditionnellement, les méthodes standard de test d'hypothèse nulle que nous venons de décrire ont été utilisées pour comparer les performances de différents algorithmes métaheuristiques. Cela a produit au fil des ans un grand nombre de tests post-hoc différents et de représentations graphiques qui facilitent le processus d'évaluation de l'algorithme qui fonctionne le mieux en moyenne, avec une signification statistique (par exemple, des tracés de distance critique). Cependant, de nombreuses critiques ont surgi ces derniers temps sur différents aspects de ces tests qui suggèrent qu'un pas de plus devrait être fait vers des moyens alternatifs pour évaluer la pertinence statistique dans les études de comparaison de performances. Pour commencer, l'utilisation du paramètre dit de significativité (souvent noté α) se heurte à son manque d'interprétabilité et n'est pas directement lié aux différences de performances observées entre les homologues du benchmark. De plus, les conclusions tirées des tests d'hypothèses non statistiques sont largement sensibles au nombre d'échantillons utilisés pour leur calcul, ainsi qu'au nombre d'algorithmes et de problèmes sur lesquels porte l'étude. Les travaux de Benavoli et al dans [74] ont éclairé ce sujet et ont proposé l'utilisation de l'analyse bayésienne pour l'analyse comparative. Le paradigme bayésien fait des déclarations sur la distribution de la différence entre les deux algorithmes comparés, ce qui peut être utile lorsque le test de signification de l'hypothèse nulle (NHST) ne trouve pas de différences significatives entre eux. Le package rNPBST [75] et le framework jMetalPy [76] sont des outils utiles pour appliquer ces tests.

L'utilisation de différents tests peut aider à mettre les résultats en contexte. Comme il est mentionné dans [77], les auteurs encouragent l'utilisation conjointe de tests non paramétriques et bayésiens afin d'obtenir une perspective complète de la comparaison des résultats des algorithmes : « Alors que les tests non paramétriques peuvent fournir des résultats significatifs lorsque il y a une différence entre les algorithmes comparés, dans certaines circonstances ces tests ne fournissent aucune information valable et les tests bayésiens peuvent aider à élucider la vraie différence entre eux » [77]. Les praticiens doivent envisager cette possibilité pour compléter les tests non paramétriques bien connus lorsqu'ils ne fournissent pas une différence complète entre les algorithmes.

4.2. Techniques de visualisation pour l'analyse comparative

Les techniques de visualisation sont d'autres méthodes utiles pour rendre compte des résultats lors de la comparaison de plusieurs algorithmes bio-inspirés. Le principal avantage de ces approches par rapport à la présentation de données brutes dans des tableaux est qu'elles peuvent être très facilement interprétées par le lecteur. Ils ont également la capacité de résumer les informations couvertes par un ou même plusieurs tableaux.

Dans la figure 1, nous fournissons un exemple de quelques visualisations qui illustrent les performances de plusieurs algorithmes sur le benchmark CEC'2013 LSGO. La figure 1(d) utilise un diagramme radar pour visualiser le classement moyen de chaque algorithme bio-inspiré sur différents groupes de fonctions. Chaque groupe a été défini selon certaines caractéristiques communes présentes dans de nombreux benchmarks de pointe : degré de séparabilité, modalité, etc. Figures. 1(c)-1(b) fournissent une vue alternative sur les mêmes données : elles ne décrivent pas leur comportement moyen, mais plutôt le nombre de fois où un algorithme a obtenu les meilleurs résultats globaux pour les problèmes liés à chacun des précédents catégories définies. Dans la Fig. 1(c), le

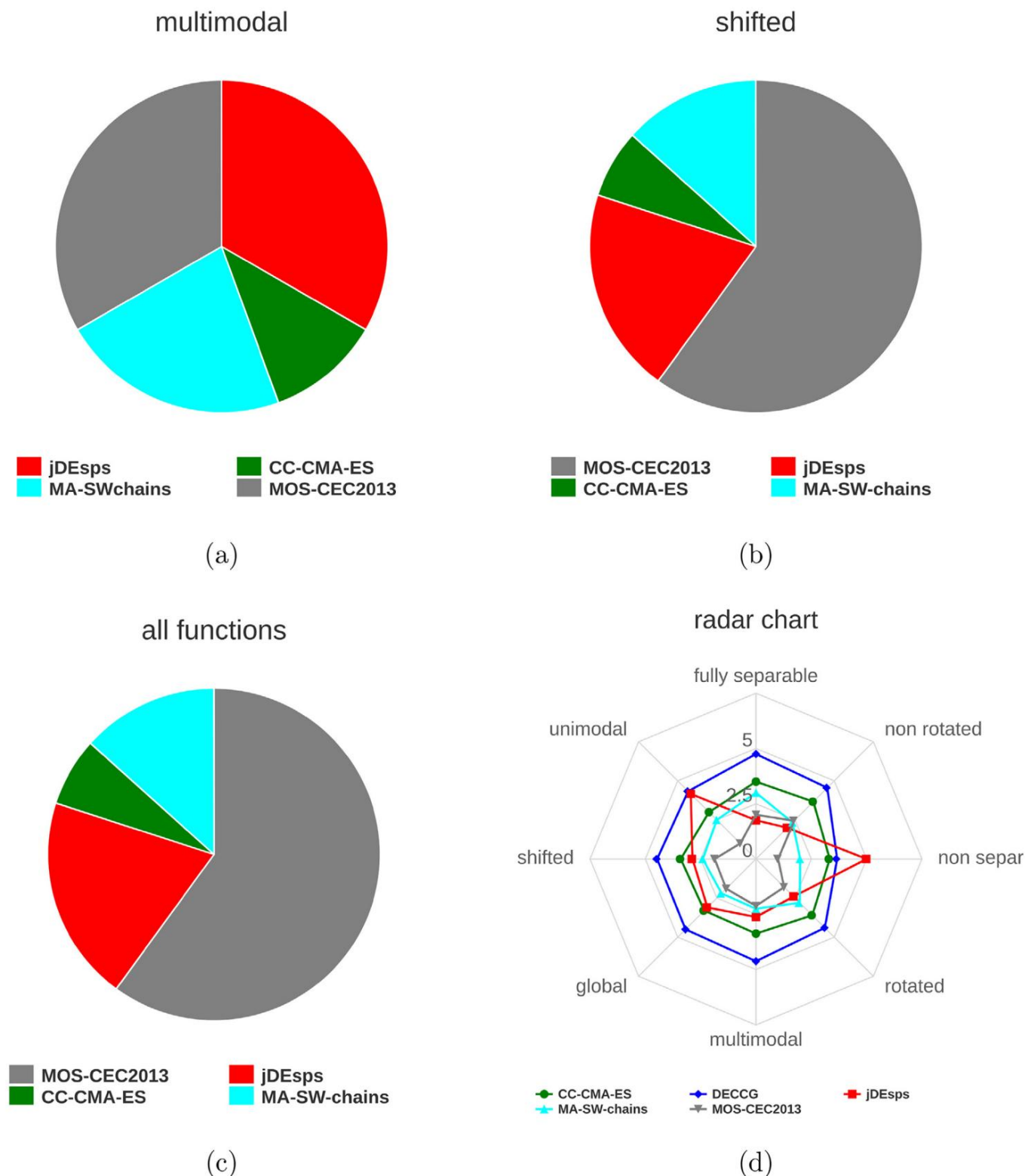
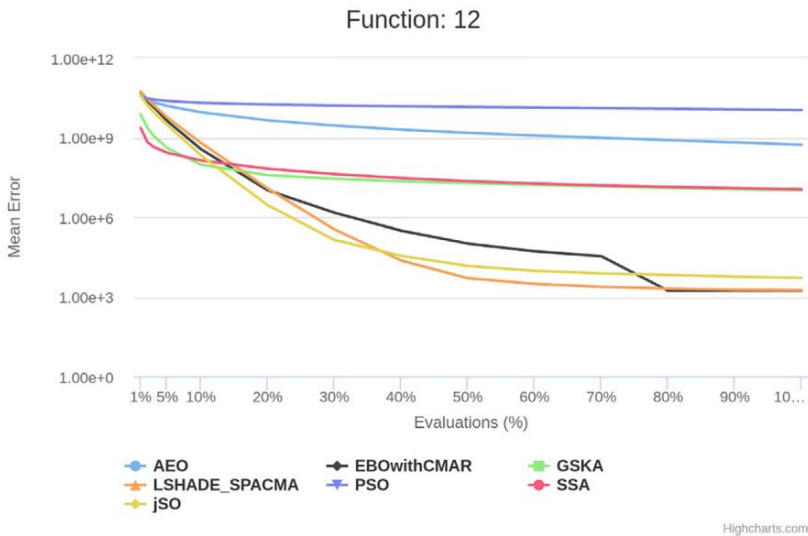


Fig. 1. Différentes visualisations pour la comparaison des performances de plusieurs algorithmes sur le benchmark CEC'2013 LSGO : (a) Classement moyen des algorithmes sur différents types de fonctions ; (b) Fraction de fonctions pour lesquelles un algorithme spécifique a obtenu les meilleurs résultats ; (c) Fraction de fonctions multimodales pour lesquelles un algorithme spécifique a obtenu les meilleurs résultats ; (d) Fraction de fonctions décalées pour lesquelles un algorithme spécifique a obtenu les meilleurs résultats.

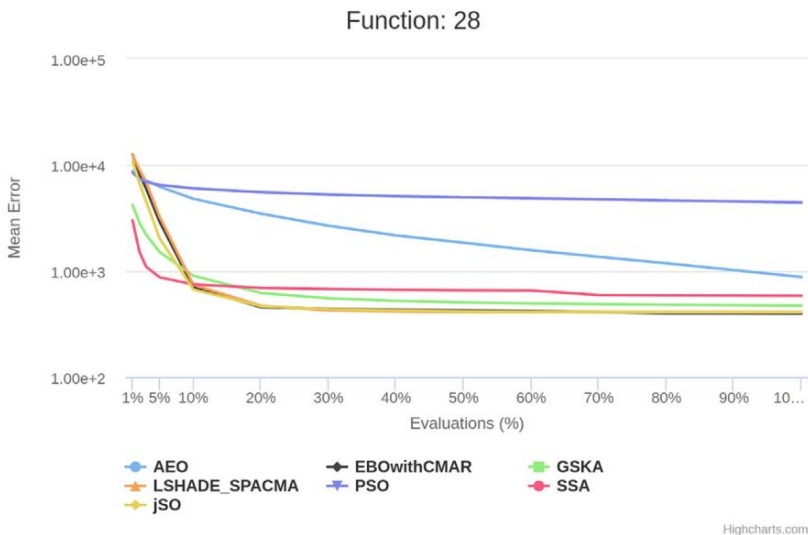
l'ensemble du benchmark est considéré, alors que les Figs. 1(a) et 1(b) montrent les résultats pour les fonctions multimodales et décalées, respectivement.

Une autre représentation visuelle typique est celle de la convergence d'un algorithme. Dans ce cas, la variable discutée est la vitesse de convergence des méthodes impliquées dans la comparaison. Un exemple

de ce type de tracé est fourni à la Fig. 2. Comme on peut le voir sur la Fig. 2(a), qui illustre la vitesse de convergence de plusieurs algorithmes sur F11 du benchmark CEC'2017, bien qu'EBOWithCMAR soit capable d'atteindre l'erreur minimale parmi les algorithmes concurrents, ce n'est qu'au bout



(a)



(b)

de la course qu'il surmonte d'autres méthodes. La plupart du temps, jSO donne de meilleurs résultats.

Avec un langage visuel similaire, nous pouvons représenter des données très différentes. Par exemple, il pourrait être intéressant de mesurer le ratio de problèmes résolus à mesure que le nombre d'évaluations de condition physique augmente. Ces tracés montrent non seulement quel algorithme est capable d'atteindre l'optimum dans le plus de problèmes, mais aussi combien d'efforts sont nécessaires pour l'accomplir. Un exemple de cette visualisation est fourni à la Fig. 6, qui se trouve dans Article 7.1.

Enfin, même les représentations visuelles courantes telles que les boîtes à moustaches peuvent être très utiles. Dans le cadre de l'optimisation bio-inspirée, c'est un bon moyen de montrer non seulement l'erreur moyenne, mais aussi l'erreur pour les différents runs. La figure 3 fournit deux exemples comparant les résultats de différents algorithmes pour l'étude de cas de la section 7.1.

Les idiomes visuels appropriés pour représenter les informations dans ce type de comparaisons ne sont pas limités aux deux exemples donnés dans cette section. Il existe de nombreuses autres représentations alternatives qui peuvent aider à mieux comprendre les résultats en discussion.

Ce qui doit être clair, c'est que différents idiomes prennent en charge différents types d'analyses, et que les techniques de visualisation doivent être soigneusement sélectionnées.

afin de présenter les résultats de manière résumée mais perspicace. Dans l'ensemble, notre recommandation à ce stade est non seulement de visualiser les résultats de la comparaison, mais aussi d'utiliser ces techniques pour compléter et/ou résumer les informations fournies par d'autres moyens.

5. Ligne directrice n° 3 : analyse des composants et ajustement des paramètres de la proposition

Cette troisième ligne directrice pourrait être considérée comme une liste de contrôle pour la section de discussion. Il couvre l'ensemble de l'analyse de la proposition, depuis l'énoncé des hypothèses à prouver par l'expérimentation jusqu'à la présentation des résultats des différentes comparaisons nécessaires pour apprécier l'apport des travaux en cours. Cette section développe cette liste en s'arrêtant sur les aspects suivants : l'origine et les hypothèses de travail qui motivent la proposition (Section 5.1), l'identification des phases de recherche, avec des affirmations à leur sujet solidement étayées par des preuves empiriques (Section 5.2), les indices analyse individuelle des composants algorithmiques de la proposition (section 5.3) et réglage et analyse des paramètres (section 5.4).

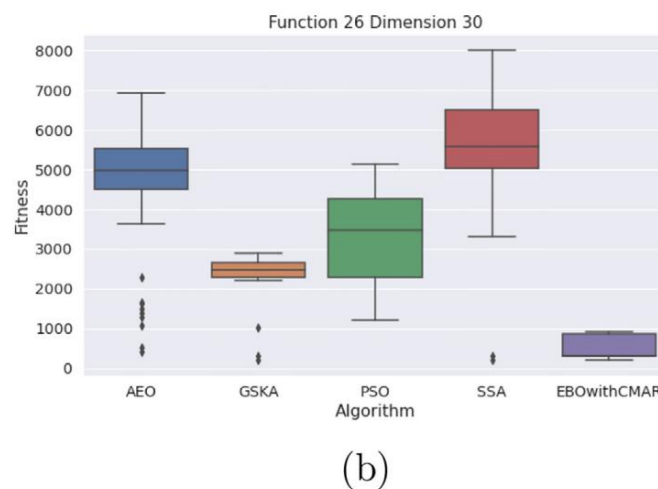
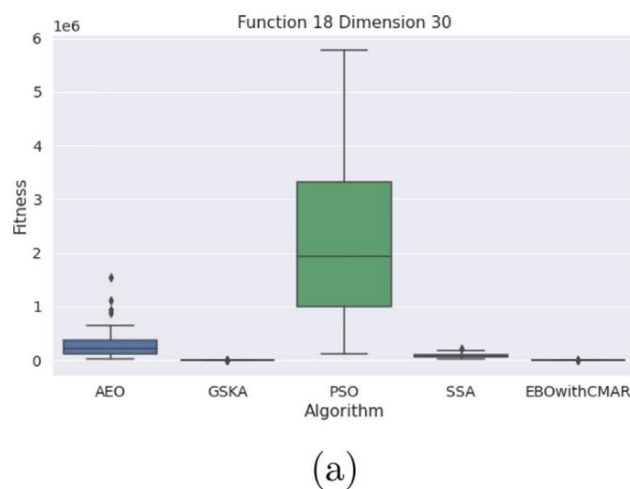


Fig. 3. Boîtes à moustaches pour le benchmark CEC'2017 et la dimension 30 : (a) fonction 18 ; (b) fonction 26..

5.1. Origine, hypothèses et proposition

Avant de commencer à discuter des avantages de la ou des nouvelles approches proposées dans le travail, il est nécessaire d'avoir une perspective claire des résultats attendus, c'est-à-dire les hypothèses de travail de la présente étude. De plus, les auteurs doivent décrire clairement comment la proposition contribue à atteindre les objectifs visés. La plupart du temps, nous supposons que la contribution principale d'un travail est un nouvel algorithme ou une nouvelle modification capable d'améliorer l'état de l'art dans un benchmark particulier (correctement choisi comme déjà discuté dans la section 3.1). Si cette hypothèse tient souvent dans la pratique, toute autre contribution à prendre en compte doit être clairement mise en évidence à ce stade pour que le travail soit considéré comme pertinent. Nous discuterons plus en détail de cette question dans la section 6.

5.2. Identification des phases de recherche

Un problème clé lors de la résolution de problèmes d'optimisation avec des paysages d'adaptation complexes est de maintenir un équilibre approprié entre l'exploration et l'exploitation [78,79]. Il s'agit d'une affirmation récurrente dans de nombreuses contributions, en particulier lorsque les résultats obtenus semblent étayer cette hypothèse (au moins en termes de précision globale). Cependant, la plupart de ces études ne fournissent pas de preuves sur la façon dont l'équilibre exploration/exploitation est maintenu. Il ne suffit normalement pas d'affirmer que « l'algorithme A est meilleur que l'algorithme B car il maintient bien l'équilibre exploration/exploitation ». Ce type d'affirmation nécessite une analyse empirique pour vérifier dans quelle mesure cette affirmation est étayée par des preuves.

Le travail rapporté dans [80] analyse cette question d'un double point de vue. Dans un premier temps, ils inspectent la façon dont différents auteurs mesurent l'équilibre exploration/exploitation, pour conclure que celle-ci est principalement réalisée au moyen de mesures indirectes (par exemple, la diversité des solutions). Deuxièmement, ils proposent une taxonomie de méthodes qui visent à promouvoir la diversité des populations. Les auteurs désireux d'inclure une analyse de ce type dans leurs travaux de recherche devraient mener une étude expérimentale quantitative pour justifier le type d'énoncés que nous avons déjà mentionnés. Ceci peut être réalisé soit en utilisant des opérateurs évolutifs qui améliorent explicitement cet équilibre [81-83] ou en mettant à jour l'algorithme ou les opérateurs pour prendre en compte une sorte de mesure indirecte [84, 85].

5.3. Analyse des composants et simplicité/complexité

Il est courant dans la littérature récente, en particulier dans les articles où la proposition est évaluée sur une référence bien connue, que les nouvelles propositions soient construites sur des algorithmes existants antérieurs. Ces nouvelles méthodes i) améliorent normalement les algorithmes précédents en mettant à jour ou en ajoutant de nouvelles caractéristiques à leur procédure de recherche de base ; ou ii) combiner existent

ing méthodes pour créer un algorithme hybride quelconque. Cependant, peu de ces travaux analysent individuellement chacune des améliorations/composantes de la nouvelle proposition. C'est une question importante du point de vue de la conception algorithmique. Il est vrai que des algorithmes puissants sont généralement recherchés (par leur capacité à trouver des solutions les plus proches possible de l'optimum global). Mais il n'en est pas moins vrai que la simplicité doit être considérée comme un autre aspect préférentiel dans la conception de nouvelles techniques d'optimisation. La simplicité dans la conception algorithmique présente un certain nombre d'avantages :

- Les algorithmes simples ont normalement moins de paramètres à ajuster (ou, à les moins sensibles).
- Leur comportement est plus prévisible, car il y a moins de composants dans impliqué.
- Ils peuvent être décrits et mis en œuvre plus facilement. • Il est moins probable que l'algorithme dépasse un repère particulier.

Toutes ces raisons sont suffisamment importantes pour prêter attention à la complexité du nouvel algorithme. Pour cette raison, il est obligatoire de fournir une analyse approfondie de la contribution de chacune des composantes de la nouvelle méthode à sa performance globale. Chaque modification ou ajout par-dessus l'algorithme d'origine doit être soutenu par une contribution significative au comportement amélioré de la nouvelle méthode. De plus, si cette contribution s'avère faible, cette amélioration doit être envisagée pour être supprimée dans un souci de simplicité. En ce sens, nous encourageons les auteurs à utiliser les mêmes méthodes de validation statistique décrites dans

Section 4.1 pour comparer chacun des composants individuels de l'algorithme. C'est la même procédure que nous recommandons pour comparer la proposition avec d'autres algorithmes de l'état de l'art (voir section 7).

Un exemple illustratif dans ce sens est [86]. Ce travail analyse l'un des algorithmes les plus performants du concours IEEE CEC'2016 sur l'optimisation à objectif unique à paramètres réels, à savoir L-SHADE EpSin [87]. L'une des conclusions de cette analyse est qu'un seul des multiples ajouts à l'algorithme de base L-SHADE apporte une amélioration significative des résultats (l'initialisation du paramètre F à 0,5 pendant la première moitié de la recherche). Les autres modifications (matérialisées par l'inclusion de plusieurs stratégies de recherche locale) n'ont pas été jugées significativement meilleures. De plus, ils ont favorisé un biais dans la recherche vers des solutions autour de l'origine de l'espace de recherche, comme également étayé par [86]. Cela signifie que, même pour les algorithmes concurrents, la contribution de chaque composant doit être soigneusement évaluée. Il en est ainsi, car une version simplifiée de l'algorithme sera toujours plus facile à maintenir, et peut même conduire à de meilleurs résultats.

Un deuxième exemple aligné avec nos recommandations à ce stade émerge des résultats de l'algorithme MOS-SOCO2011 présenté dans [79]. Grâce au cadre Multiple Offspring Sampling (MOS), cette technique d'optimisation combine deux algorithmes bien connus : DE

Tableau
2 Résultats pour MOS-SOCO2011, DE et MTS-LS1 sur différentes fonctions 1000-D [79].

Fonction de référence	MOS-SOCO2011 DE	MTS-LS1
Sphère	0.00e+00	3.71e+01 1.15e-11
Soufre 2.21	4.25e-01	1.63e+02 2.25e-02
Rosenbrock	6.15e+01	1.59e+05 2.10e+02
Rastrigin	0.00e+00	3.47e+01 1.15e-11
Griewank	0.00e+00	7.36e-01 3.55e-03
Ackley Soufre	0.00e+00	8.70e-01 1.24e-11
2.22 Soufre 1.2	0.00e+00	0.00e+00 0.00e+00
Étendu f10	1.94e+05	3.15e+05 1.23e+05
Bohachevsky	0.00e+00	6.26e-02 1.99e+03
Schaffer f12	0.00e+00	1.67e-01 0.00e+00
	0.00e+00	4.42e-02 1.99e+03
	0.00e+00	2.58e+01 5.02e+02
f13	8.80e+01	8.24e+04 8.87e+02
f14	0.00e+00	2.39e+01 2.23e+03
f15	0.00e+00	2.11e-01 0.00e+00
f16	0.00e+00	1.83e+01 1.00e+03
f17	2.25e+01	1.76e+05 1.56e+03
f18	0.00e+00	7.55e+00 1.21e+03
f19	0.00e+00	2.51e-01 0.00e+00
Fonctions résolues	14	1 4

et la première méthode de recherche locale (MTS-LS1) de l'algorithme de recherche de trajectoires multiples (MTS) [88]. L'algorithme hybride MOS-SOCO2011 a été évalué sur le benchmark proposé pour le Soft Computing Special Issue sur la scalabilité des algorithmes évolutionnaires et autres méta heuristiques pour les problèmes d'optimisation continue à grande échelle [89]. MOS SOCO2011 a obtenu les meilleurs résultats globaux parmi tous les participants au numéro spécial. Dans [79], les auteurs ont rapporté non seulement les résultats de l'algorithme MOS-SOCO2011 proposé, mais aussi ceux de chacun des composants indépendants, DE et MTS-LS1, qui sont présentés dans le tableau 2 . Cette triple compilation des résultats permet une comparaison directe sur le nombre de fonctions résolues avec une précision maximale (14, 1 et 4 pour MOS-SOCO2011, DE et MTS-LS1, respectivement), et met en lumière la synergie des deux algorithmes : à l'exception des fonctions Schwefel 2.21 et Schwefel 1.2, pour lesquelles l'algorithme MTS-LS1 a obtenu les meilleurs résultats, la méthode hybride a pu atteindre les performances du meilleur de ses algorithmes de composition, les surpassant normalement.

De plus, une comparaison statistique a également été effectuée, rapportant des valeurs de p significatives pour les deux comparaisons (MOS-SOCO2011 contre DE, et MOS-SOCO2011 contre MTS-LS1). Ces deux comparaisons fournissent ensemble suffisamment de confiance pour soutenir la supériorité de la méthode hybride sur chacun de ses algorithmes de composition. Ce n'est pas le seul exemple d'une telle comparaison. Les auteurs de l'algorithme L-SHADE suivent une approche similaire dans [90], comparant la nouvelle version de leur algorithme aux précédentes, et évaluant l'ajout de nouveaux composants pour prouver leurs avantages.

5.4. Réglage et analyse des paramètres

Un problème important dans la conception d'un algorithme est le nombre de paramètres libres qui peuvent être ajustés pour modifier son comportement. En général, plus il y a de flexibilité, plus il y a de paramètres de contrôle à régler. Très souvent, les valeurs sélectionnées pour ces paramètres sont si déterminantes dans la recherche que même un algorithme bien conçu peut donner de mauvais résultats avec les mauvaises valeurs de paramètres. Par conséquent, la sélection des valeurs de ces paramètres internes est une décision critique qui ne doit être ni sous-estimée ni négligée.

Malheureusement, la sélection des bonnes valeurs de paramètres ou du mécanisme d'adaptation n'est pas une tâche facile, car la plupart du temps, il n'y a pas de critère clair pour guider cette sélection, et cela pourrait être considéré comme un problème d'optimisation en soi. Le réglage des paramètres peut être abordé de différentes manières :

- Réglage hors ligne ou en ligne. Les approches de réglage hors ligne recherchent les réglages optimaux des paramètres EA pour plusieurs problèmes représentatifs. Ensuite, ces valeurs sont appliquées à de nouveaux problèmes [91], ce qui signifie que le réglage est effectué avant que l'algorithme ne soit exécuté sur le problème à résoudre. Le principal inconvénient de cette approche est que les valeurs obtenues ne doivent pas nécessairement être optimales pour de nouveaux problèmes, mais, en pratique, des configurations de paramètres robustes peuvent être obtenues. Une approche alternative consisterait à effectuer le réglage des paramètres pendant l'exécution. Cette stratégie, en théorie, permet d'obtenir des valeurs mieux adaptées à chaque problème, mais c'est une méthode plus complexe à mettre en œuvre. • Contrôle des paramètres statique, adaptatif et auto-adaptatif. Dans [92], les approches de contrôle des paramètres (celles effectuées pendant l'exécution de l'algorithme) sont réparties dans l'une des trois catégories suivantes : déterministe, adaptative et auto-adaptative. Le contrôle déterministe des paramètres signifie que la valeur du paramètre de stratégie est mise à jour selon certaines règles déterministes, sans rétroaction de la recherche. Le contrôle adaptatif des paramètres a lieu lorsqu'une certaine rétroaction du processus de recherche est utilisée pour déterminer la direction et/ou l'ampleur de l'ajustement des paramètres. Enfin, le contrôle des paramètres auto-adaptatif se produit lorsque les valeurs des paramètres sont intégrées dans la solution à optimiser. Ainsi le paramétrage est implicite lors de la recherche. Plusieurs exemples réussis d'auto-adaptation sont discutés ci-dessous.

L'approche la plus courante pour traiter le problème de réglage est de le faire hors ligne et expérimentalement, en comparant les résultats obtenus sur plusieurs combinaisons des valeurs des paramètres. Cependant, il existe plusieurs fosses dans lesquelles un chercheur peut tomber lors d'un réglage manuel :

- Tester chaque paramètre sur un petit nombre de valeurs prédéfinies, sans tenir compte d'aucun retour d'expérience : certains travaux ne testent que des valeurs extrêmes du ou des paramètres à régler (minimum ou maximum sur sa plage). Dans ces circonstances, le nombre de valeurs à tester doit être étendu pour vérifier si un meilleur résultat peut être obtenu. obtenu avec d'autres valeurs sur la plage du ou des paramètres. • Ajuster seulement une petite partie de leurs paramètres : dans ce cas, les valeurs des paramètres restants sont devinées ou initialisées à des valeurs fixes, sans analyser quels paramètres influencent le plus les performances de l'algorithme. Cette analyse justifierait éventuellement les paramètres à sélectionner et à régler avec soin, mais elle est rarement fait dans la littérature.
- Essayez de régler chaque paramètre indépendamment, en gardant les autres fixes : cette approche largement adoptée pose de nombreux problèmes. Les résultats obtenus en faisant varier un seul paramètre dépendent largement des valeurs données aux autres, car il n'est pas rare que plusieurs paramètres s'influencent les uns les autres. Par conséquent, toutes les combinaisons de valeurs de paramètres doivent être optimisées dans leur ensemble. S'il est vrai qu'explorer toutes les combinaisons possibles peut conduire à une explosion combinatoire, il existe plusieurs techniques dans le domaine de la conception expérimentale [3] qui peuvent pallier ce problème, comme la conception fractionnaire [93] ou les méthodes alternatives de Taguchi [94].
- Utiliser des valeurs de paramètres réglées par d'autres auteurs lors d'expériences antérieures : ces paramètres ont généralement été réglés pour un problème différent, et ainsi leurs valeurs peuvent ne pas être appropriées pour le problème/ la référence à l'étude. Bien sûr, ces valeurs peuvent toujours être utilisées, mais elles ne doivent jamais être considérées comme optimales, car leur compétitivité sur un référentiel différent ne peut être garantie [95,96]. • Absence de tests statistiques dans les comparaisons pour ajuster les valeurs des paramètres de la proposition : en raison de la nature stochastique des algorithmes méta-heuristiques, la sélection des valeurs des paramètres en fonction des valeurs de fitness moyennes n'est pas suffisante. Les mêmes procédures statistiques utilisées pour comparer plusieurs techniques doivent être utilisées lors de la comparaison de différentes configurations d'un algorithme. Le choix des valeurs les plus prometteuses est laissé à la discrétion des auteurs potentiels, mais l'utilisation de tests statistiques doit toujours être appliquée, quelle que soit l'approche de réglage des paramètres utilisée.

effectuée. En fait, comme le montre [97], l'utilisation de tests statistiques pour ce type de comparaisons peut être simple.

Une approche alternative au réglage des paramètres est l'auto-adaptation des paramètres [46]. Dans cette approche, les paramètres ne reçoivent pas de valeur fixe. Au lieu de cela, un processus est conçu pour ajuster automatiquement leur valeur en fonction du retour d'information obtenu à partir du processus d'optimisation. Ce type de mécanisme d'adaptation a été appliqué avec succès à DE, qui est très sensible à ses paramètres [26, 98]. Les travaux de [99] ont montré la commodité des valeurs auto-adaptatives par rapport aux réglages de paramètres fixes. Dans [49], le premier DE avec des paramètres auto-adaptatifs a été présenté, dessinant des valeurs de paramètres en échantillonnant une distribution dont la moyenne est mise à jour en considérant de nouvelles solutions entrant dans la population. Depuis, de nouveaux algorithmes améliorant le mécanisme d'auto-adaptation [100, 101] ont été proposés. Dans [90], les paramètres auto-adaptatifs précédents ont été complétés par une taille de population adaptative qui diminue au début du processus de recherche. Dans [102], un réglage des paramètres a été appliqué, améliorant encore les résultats de l'algorithme global.

Bien que les paramètres adaptatifs soient une nette amélioration par rapport aux paramètres fixes, tant en termes de facilité d'utilisation que de robustesse, tous les paramètres ne peuvent pas être auto-ajustés de cette manière (en particulier certains paramètres plus internes). Ainsi, même dans les algorithmes auto-adaptatifs, il existe des paramètres fixes qui doivent être ajustés pour améliorer encore plus les résultats (c'est-à-dire [102]).

Heureusement, le réglage des paramètres peut être effectué automatiquement. Il existe plusieurs outils utiles et consolidés de ce type avec différentes fonctionnalités [50]. Dans les paragraphes suivants, nous décrivons brièvement certaines des méthodes les plus avancées :

- L'optimisation séquentielle des paramètres, SPO [103], est une heuristique qui utilise l'échantillonnage latin hypercube pour déterminer les valeurs des paramètres maintenant le coût de calcul bas.
- Iterated Racing for Automatic Algorithm Configuration, IRACE [52], est un algorithme d'optimisation qui utilise différentes exécutions (courses) et tests statistiques pour identifier des combinaisons de paramètres qui sont pires que d'autres, fournissant un processus automatique pour optimiser une variété de paramètres.
- L'estimation de la pertinence et le calibrage de la valeur des paramètres de l'algorithme évolutionnaire, REVAC [53], est un algorithme évolutionnaire qui utilise des mesures multiparentales de croisement et d'entropie pour estimer la pertinence des paramètres.
- ParamILS [54], est une approche de recherche locale stochastique polyvalente pour configuration automatisée de l'algorithme.

Bien que I-RACE ait donné de meilleurs résultats pour nous dans le passé, toutes les alternatives susmentionnées sont des outils robustes et consolidés, de sorte que la sélection d'un outil parmi les autres dépendra des caractéristiques du problème et des préférences personnelles.

Pour conclure sur ce sujet, nous voudrions faire une remarque supplémentaire. Le réglage des paramètres d'une nouvelle proposition peut avoir un impact important sur l'objectivité de la comparaison. Cette situation peut se produire lorsque la nouvelle proposition est le seul algorithme dont les paramètres sont réglés pour la référence particulière utilisée dans l'expérimentation, alors que les algorithmes de référence utilisent des valeurs de paramètres proposées par leurs auteurs respectifs dans des conditions expérimentales différentes. Cela signifiera probablement que ces algorithmes ne devraient pas rapporter de très bons résultats pour le nouveau scénario expérimental. Dans ce cas, avoir les paramètres de la nouvelle proposition réglés pour le benchmark considéré dans le travail pourrait donner à notre proposition un avantage injuste sur les autres, générant un biais en faveur de l'algorithme proposé dans la comparaison. Idéalement, la solution devrait être de comparer les versions optimisées de tous les algorithmes [96], mais le coût de cette opération pourrait devenir trop coûteux et inabordable en termes de calcul. Cependant, lorsque les algorithmes sont comparés à des benchmarks standards (non définis ad-hoc pour chaque article), ce risque est minimisé, car tous les algorithmes ont été testés dans les mêmes conditions expérimentales.

6. Directive #4 : Pourquoi mon algorithme est-il utile ?

La dernière étape d'une proposition réussie est une discussion approfondie des résultats. Cette discussion doit répondre à une question cruciale : pourquoi mon algorithme est-il utile ?

La réponse la plus évidente à cette question est "parce qu'elle surpasse les méthodes de pointe actuelles". Si l'algorithme entre dans cette première catégorie de propositions, et si ce comportement surperformant est validé par des moyens raisonnés (comme ceux présentés dans ce manuscrit), la contribution a une valeur scientifique claire et peut être apportée à la communauté sous la forme d'une publication. Cependant, ce n'est pas toujours le cas, mais cela ne signifie pas que la contribution n'est pas significative.

Il existe un certain nombre de raisons d'accepter une nouvelle proposition même si elle est incapable de surpasser les meilleurs algorithmes à ce jour. Néanmoins, dans ces circonstances, il est encore plus important de discuter des résultats. Les avantages de l'adoption de la méthode proposée dans une telle contribution doivent être clairement énoncés et mis en évidence en conséquence. Nous discutons ensuite de certaines des raisons qui peuvent être considérées comme suffisantes pour qu'une nouvelle proposition soit acceptée :

- La première de ces raisons est la qualité des résultats. Si, comme mentionné précédemment, les résultats surpassent clairement les méthodes actuelles de pointe, les auteurs ont un argument solide pour que leur article soit accepté. Parfois, il peut suffire que les résultats soient particulièrement bons pour un sous-ensemble de problèmes, étant donné que ce comportement peut être identifié et caractérisé. Cela ne signifie pas que le reste des lignes directrices fournies dans notre article peut être négligé. La discussion des résultats doit être rigoureuse et les conclusions doivent être clairement présentées, sans ambiguïté ni imprécision.

• La seconde de ces raisons est la nouveauté : si un algorithme nouvellement proposé a le potentiel d'évoluer et de devenir compétitif avec les méthodes de pointe actuelles, il devrait être présenté à la communauté. Néanmoins, une attention particulière doit être portée à ce stade pour éviter les problèmes décrits par [1, 104] : il est absolument obligatoire que, outre la métaphore bio-inspirée, la nouvelle proposition algorithmique soit suffisamment compétitive pour un ensemble de problèmes. De plus, nous préconisons fermement le développement d'un langage de description unifié pour les algorithmes méta-heuristiques, capable de décrire sans ambiguïté chacune des étapes algorithmiques des nouvelles propositions, en laissant de côté tout langage métaphorique. Nous croyons fermement que les efforts dans ce sens devraient être intensifiés, en s'appuyant sur les postulats initiaux établis dans certains travaux récents [105, 106]. Plus précisément, les composants méta-heuristiques (y compris les opérateurs de recherche et les modèles comportementaux algorithmiques) et les interfaces entre eux doivent être normalisés afin d'évaluer objectivement les similitudes et les différences entre les solveurs basés sur les métaphores [107]. Une nouvelle métaphore n'est en aucun cas une garantie suffisante pour une contribution scientifique significative. • La troisième de ces raisons est d'ordre méthodologique, c'est-à-dire la pertinence de certaines briques de l'algorithme global. Un algorithme particulier peut inclure un composant donné (par exemple, un optimiseur local) qui peut être pertinent même si l'algorithme dans son ensemble n'est pas totalement compétitif par rapport à la littérature dominante. Un bon exemple à l'appui de cette affirmation peut être observé dans les cadres co-évolutifs, qui incluent généralement une procédure pour identifier les sous-composants qui seront individuellement co-évolus. Dans ces cas, même si l'optimiseur de sous-composants n'est pas très sophistiqué, le cadre co-évolutif peut être pertinent en lui-même. En ce sens, il est important de sélectionner le cadre approprié pour mettre en évidence la caractéristique souhaitée de l'algorithme proposé, comme discuté dans la section 3.1. Suivant le même exemple d'identification de sous-composants, un chercheur focalisé sur l'optimisation globale à grande échelle devrait considérer le benchmark CEC'2013 qui étudie explicitement cette question [23]. Néanmoins, il s'agit d'une considération assez subjective, les auteurs doivent donc clairement mettre en évidence ces avantages pour éviter les affirmations discutables.

7. Études de cas

Afin d'illustrer l'application des lignes directrices précédentes, cette section développe plusieurs études de cas qui suivent notre méthodologie proposée. En particulier, nous considérons les deux scénarios suivants :

- L'étude de différents algorithmes existants, dont le but est d'analyser les avantages et les inconvénients de chacun d'eux. Dans ce premier scénario, la priorité est de comparer toutes les méthodes selon différents critères, afin de déterminer dans quelles circonstances chacune d'entre elles peut être recommandée.
- La proposition d'un nouvel algorithme, dont la priorité est de fournir des preuves éclairées de la compétitivité de la nouvelle proposition, ainsi que de ses avantages par rapport aux algorithmes précédents.

La structure de cette section est conforme aux scénarios ci-dessus.

Tout d'abord, dans la sous-section 7.1, nous présentons plusieurs algorithmes bio-inspirés récents pour l'optimisation des paramètres réels, puis nous recourons à nos lignes directrices proposées afin de les comparer équitablement. Ensuite, dans la sous-section 7.2, nous discutons du deuxième cas d'étude, où un nouvel algorithme est proposé pour un type spécifique de problème d'optimisation : l'optimisation globale à grande échelle.

7.1. Plusieurs algorithmes bio-inspirés modernes pour l'optimisation des paramètres réels

Dans cette première étude de cas, nous simulons le scénario dans lequel une comparaison de plusieurs algorithmes bio-inspirés est conçue pour l'optimisation des paramètres réels. Il y a deux raisons possibles pour lesquelles cette comparaison peut être abordée. D'une part, nous pourrions être disposés à proposer un nouveau solveur, pour lequel nous devons évaluer les performances des algorithmes existants pour les utiliser comme référence pour notre proposition. En revanche, on pourrait être intéressé par la résolution d'un problème particulier similaire à ceux considérés dans la comparaison, pour lequel on analyse plusieurs options algorithmiques afin de déterminer laquelle utiliser.

Les algorithmes considérés dans cette première étude de cas sont des méthodes récentes présentées dans des revues de premier plan :

Algorithme de recherche d'écureuils (SSA): Il s'agit d'un algorithme bio-inspiré qui imite le comportement de recherche de nourriture des écureuils. Il divise les solutions en trois groupes en fonction de leur aptitude (la meilleure, les trois meilleures suivantes et les autres) et adopte différents critères pour les combiner en tenant compte de cet arrangement de regroupement, combinant chaque solution avec une solution dans une catégorie supérieure par une combinaison linéaire aléatoire [108].

Algorithme basé sur le partage des connaissances (GSKA): Il s'agit d'un algorithme inspiré du comportement humain lors du partage des connaissances. Il a été observé qu'il y a plus d'acceptation dans les premières phases et que les gens sont plus questionnés dans les phases ultérieures. D'un point de vue algorithmique, il utilise deux critères différents pour optimiser chaque dimension : dans le premier, appelé gain-partage junior, la variable considérée est mise à jour en tenant compte des variables de ses meilleurs et moins bons individus immédiats dans la population. Dans le deuxième critère, senior gain-partage, la variable est mise à jour en considérant les meilleurs et les pires individus de la population. Initialement, toutes les variables sont mises à jour par le mécanisme junior gain-partage, et pendant la recherche, un nombre croissant de variables sont mises à jour par le critère senior gain-partage [109].

Optimisation basée sur les écosystèmes artificiels (AEO) : il s'agit d'une méta-heuristique inspirée de la nature qui trouve sa motivation dans le flux d'énergie à travers un écosystème. Dans cette proposition, la population est mise à jour au moyen d'un processus itératif composé de différentes phases. La première phase est la production, dans laquelle une solution subit une petite mise à jour aléatoire décroissante. La seconde est la consommation, dans laquelle chaque individu est classé au hasard comme herbivore, omnivore ou carnivore.

Selon sa catégorie, l'individu est mis à jour selon une règle différente. Les solutions nouvellement produites sont insérées dans la population si elles

améliorer les précédents (en les remplaçant). Dans la phase finale (décomposition), chaque individu est muté par une équation de dispersion [110].

Enhanced LSHADE-SPACMA Algorithm (ELSHADE-SPACMA) : Il s'agit d'un nouvel algorithme hybride qui alterne, à chaque itération, i) l'application d'une stratégie DE qui considère, dans son opérateur de mutation, non seulement les meilleurs individus mais aussi les pires ceux [111], ii) avec un LSHADE-SPACMA amélioré [112] qui adapte son paramètre p pour imposer l'exploitation dans ses étapes finales [113]. LSHADE-SPACMA est un algorithme hybride qui applique, avec une certaine probabilité, l'un des algorithmes bien connus L-SHADE [90] et CMA-ES [14]. La probabilité d'application de chaque algorithme est adaptée en considérant les améliorations obtenues par chacun d'eux au fil de la recherche. Dans son article, ELSHADE-SPACMA est déclaré surpasser les précédents gagnants des compétitions d'optimisation des paramètres réels.

Suite à nos suggestions, nous avons utilisé des implémentations existantes des algorithmes, en particulier, pour AEO et GSKA, nous avons utilisé le logiciel Mealpy 9. Pour SSA, nous avons utilisé l'implémentation du projet Indago 10. Enfin, pour ELSHADE-SPACMA, nous avons utilisé le code source fourni par les auteurs de l'algorithme 11.

Dans l'étude de comparaison discutée tout au long de cette section, nous suivons les lignes directrices proposées dans ce travail afin de mener correctement les expérimentations avec ces algorithmes et d'autres méthodes de référence. Au cours de cette étude, nous discutons également des avantages de notre proposition méthodologique.

Nous commençons ensuite par la première directive.

7.1.1. Sélection du benchmark selon la ligne directrice #1

Premièrement, nous devons choisir un benchmark adéquat pour mesurer la performance des algorithmes considérés. Suivant les recommandations de la ligne directrice #1 décrites dans la section 3 :

- Il faut bien sélectionner le benchmark : sans biais inattendu, avec le bon niveau de complexité, et pour le type de problème adressé par l'algorithme.
- Nous devons imposer l'utilisation d'un benchmark standard qui satisfait aux exigences précédentes, compte tenu des caractéristiques des problèmes pour lesquels les algorithmes ont été initialement conçus.

Dans ce cas, tous ces algorithmes ont été spécialement conçus pour l'optimisation des paramètres réels. Heureusement, il existe plusieurs benchmarks bien conçus pour ce type de problèmes, connus pour éviter les biais inattendus, et dotés de différents niveaux de complexité. En particulier, plusieurs benchmarks ont été proposés dans le cadre de l'optimisation des paramètres réels, à la fois dans les compétitions IEEE CEC et GECCO.

Pour notre étude de cas, nous adoptons deux de ces référentiels : les référentiels CEC'2017 [19] et COCO [55], pour les raisons suivantes :

- Il existe plusieurs repères, et tous sont également de bons choix. Cependant, certains des référentiels sont plutôt complémentaires car ils suivent des approches différentes. Les benchmarks CEC à paramètres réels sont plus axés sur la proposition de fonctions objectives difficiles et la mesure de l'erreur finale obtenue par chaque algorithme. D'autre part, le benchmark COCO est composé de fonctions plus simples pour permettre de mesurer les performances d'un algorithme sous différentes perspectives : combien de problèmes chaque algorithme est capable de résoudre, combien d'évaluations de fonctions objectives chaque solveur comparé nécessite pour résoudre chaque problème, et d'autres aspects similaires.
- Lorsque l'objectif est de comparer plusieurs algorithmes et de les analyser équitablement, il convient de disposer de plusieurs sources d'informations qui permettent de confirmer nos intuitions et de comparer les algorithmes de différents points de vue.

7.1.2. Sélection de la mesure du rendement conformément à la ligne directrice no 1

Une autre décision importante à prendre est le choix d'une mesure de performance adéquate. Comme discuté dans la Ligne directrice n° 1, bien que la version finale

⁹ <https://github.com/thieu1995/mealpy> <https://pypi.org/project/Indago/> <https://sites.google.com/view/optimization-project/files>

valeur de l'erreur de fitness est un choix populaire, ce n'est pas la seule alternative pour comparer les performances de plusieurs algorithmes. Une autre option intéressante est, par exemple, l'analyse de l'évolution de la valeur de fitness au fur et à mesure que les algorithmes du benchmark itèrent pour résoudre chaque problème.

Dans notre cas, les deux benchmarks sélectionnés se concentrent sur des encore des mesures complémentaires :

- Dans le benchmark CEC'2017, l'erreur est rapportée pour différents jalons, avec différents ratios d'évaluations : 1 %, 2 %, 3 %, 5 %, 10 %, 20 %, 30 %, 40 %, 50 %, 60 %, 70 %, 80 %, 90 % et 100 %. Dans notre étude, nous visons à montrer l'évolution des algorithmes à mesure que le nombre d'évaluations de fonctions augmente. Par conséquent, nous sélectionnons plusieurs de ces jalons lors du rapport des résultats atteints par chaque algorithme dans la comparaison.
- En outre, nous accordons une attention particulière à l'étape de 100 % pour fournir un score de comparaison final des résultats, sans minimiser l'importance d'atteindre des valeurs de fitness compétitives avec moins d'évaluations.
- Dans le benchmark COCO, la performance est mesurée de manière différente. En particulier, le rapport des fonctions pour lesquelles l'optimum a été atteint est calculé lorsque le nombre d'évaluations de fonctions objectives augmente.

Nous considérons qu'avec ces trois mesures nous obtiendrons un rel une vue complète des performances des différents algorithmes inclus dans notre étude.

Enfin, les résultats sont rassemblés pour chaque fonction, indépendamment. Cependant, pour une analyse plus générale, nous allons également utiliser l'agrégation, en considérant le classement moyen, qui est calculé en triant les algorithmes pour chaque fonction en fonction de leur erreur (attribuant des classements inférieurs aux meilleurs algorithmes). Ensuite, le classement moyen est calculé de sorte qu'un algorithme avec une valeur de classement moyenne inférieure soit déclaré plus performant, en moyenne, qu'un autre avec une valeur de classement plus élevée.

7.1.3. Sélection des algorithmes de référence conformément à la directive n° 1

Afin de procéder à une comparaison équitable, un critère clair est nécessaire pour sélectionner d'autres algorithmes à inclure dans le benchmark, fondé sur la nécessité d'évaluer la commodité des solveurs considérés en ce qui concerne ses performances concurrentielles par rapport aux méthodes dominantes. En suivant nos directives, nous devrions :

- Comparer avec des algorithmes de référence : L'idée est de sélectionner une méthode bien connue pour comparer si les algorithmes sont compétitifs par rapport à elle. Pour cela nous avons considéré un algorithme classique : Particle Swarm Optimization (PSO).
- Comparer avec des algorithmes similaires : cette décision est particulièrement pertinente lorsqu'un nouvel algorithme est proposé basé sur des méthodes particulières ou avec des caractéristiques qui ressemblent à celles d'autres algorithmes existants. Cependant, dans ce cas d'utilisation, les algorithmes à évaluer ont des sources d'inspiration très différentes et des comportements algorithmiques différents. Ainsi, nous considérons qu'il n'est pas nécessaire de sélectionner d'autres algorithmes similaires en plus de PSO. • Comparer avec des algorithmes concurrents : c'est une décision difficile à prendre, car il est souvent difficile d'examiner l'ensemble de l'état de l'art lié au problème d'optimisation/algorithme/ benchmark considéré. Cependant, étant donné que les benchmarks considérés ont été utilisés dans plusieurs compétitions, nous pouvons facilement trouver des algorithmes compétitifs pour chacun d'eux. En particulier, les outils de référence COCO12 se comparent toujours à un précédent lauréat du concours. Dans le cas du benchmark CEC'2017, nous comparons aux deux meilleurs algorithmes de la concurrence : jSO [114] et EBOwithCMAR [115].

Pour résumer, dans cette étude de cas, nous comparons chaque algorithme : (1) entre eux ; (2) à un algorithme de référence bien connu (PSO); et (3) à

Tableau

3 Classement moyen des algorithmes pour chaque dimension.

Algorithme	D10	D30	D50	D100	Moyenne
ELSHADE-SPACMA 2.667		2,133	1,783	1,383	1,9915
EBO avec CMAR	1.933	1.917	2.167	2.367	2,0960
jSO	2.317	2.050	2.117	2.583	2,2667
GSKA	3.883	4.200	4.433	4.533	4,2623
OSP	5.367	5.867	6.000	6.400	5,9085
ASS	5.733	5.667	5.333	4.733	5,3665
OEA	6.100	6.167	6.167	6.000	6,1085

trois algorithmes concurrents (jSO et EBOwithCMAR pour le benchmark CEC'2017, et le vainqueur de 2009 pour le benchmark COCO).

7.1.4. Expérimenter et valider les résultats conformément à la ligne directrice #2

Une fois le plan d'expérimentation mis en place, les expérimentations proprement dites sont réalisées et les résultats sont validés. Conformément aux recommandations sur l'utilisation de la validation statistique fournies dans la ligne directrice n° 2 (voir la section 4.1), la normalité et l'homocédasticité doivent être vérifiées avant que le test statistique approprié puisse être sélectionné. Cependant, dans [41], les résultats d'un précédent benchmark similaire ont été analysés pour ces deux hypothèses. De tels résultats indiquaient clairement qu'aucune des hypothèses n'était satisfaite. Pour cette raison, nous avons décidé d'utiliser des tests non paramétriques. De plus, comme le suggère également la directive n° 2, nous avons également appliqué des tests bayésiens pour comparer les approches les plus performantes entre elles.

Benchmark CEC'2017

La première étape suggérée dans la directive n° 2 consiste à calculer le classement moyen des algorithmes, suivi d'un test d'hypothèse non paramétrique. Afin de réaliser cette comparaison, nous avons recours à Tacolab [116]13, un outil web qui facilite la comparaison d'algorithmes avec différents critères.

Le tableau 3 montre le classement moyen du benchmark CEC'2017. Plusieurs observations peuvent être faites sur les résultats de ce tableau :

- Seul ELSHADE-SPACMA est plus performant que les lauréats précédents : jSO et EBOwithCMAR. C'est notamment le cas pour les problèmes de dimension supérieure (D50 et D100). Aucun des autres algorithmes récemment proposés n'est compétitif par rapport à ces anciens algorithmes de référence. Il s'agit d'un résultat intéressant, puisque les articles dans lesquels ces nouveaux algorithmes ont été proposés pour la première fois ne se comparent pas aux méthodes de l'état de l'art. Au lieu de cela, les algorithmes classiques sont simplement considérés. C'est même le cas de GSKA, qui a été testé sur le benchmark CEC'2017, mais n'a pas inclus jSO ni EBOwithCMAR dans les expérimentations. • Parmi les propositions récentes, ELSHADE-SPACMA est clairement la plus performante, voire meilleure que les précédents lauréats : jSO et EBOwithCMAR. • En dehors des propositions précitées (ELSHADE-SPACMA, EBOavec CMAR et jSO), la proposition avec la meilleure performance moyenne dans ce benchmark est GSKA, suivie de SSA. Il faut souligner que ces algorithmes ne résultent pas de l'hybridation des précédents ceux.
- L'algorithme de référence PSO obtient de moins bons résultats que la plupart des algorithmes, à l'exception d'AEO.

Une image complète de ces résultats peut être affichée si nous mesurons également comment les performances des différents algorithmes évoluent au cours de la recherche. Heureusement, les conditions expérimentales du benchmark imposent que l'erreur soit mesurée à différents jalons : 1%, 2%, 3%, 5%, 10%, 20%, 30%, 40%, 50%, 60%, 70 %, 80 %, 90 % et 100 % des évaluations de condition physique.

Les tableaux 4, 5, 6 et 7 montrent le classement moyen de chaque méthode à ces différentes étapes pour les dimensions 10, 30, 50 et 100, respectivement. De ces tableaux, on peut tirer les conclusions suivantes :

- Pour les dimensions 30 et 50, GSKA est un algorithme beaucoup plus rapide et il est nettement meilleur que toutes les autres propositions jusqu'à 10% de la

12 Disponible sur : <https://github.com/numbbo/coco>

13 Site Web de Tacolab : <https://tacolab.org/>

Tableau 4

Evolution du classement moyen par rapport aux évaluations fitness dans le benchmark CEC'2017 (dimension 10).

Algorithme	1	2	3	5	dix	20	30
OEA	1,70	1,97	2,60	3,80	4,97	5,47	5,80
EBO avec CMAR	4,50	4,50	4,03	3,13	2,30	1,90	2,05
ELSHADE-SPACMA 5.87		5,60	5,20	4,13	3,23	2,27	1,75
GSKA	2,83	2,20	2,20	2,37	2,53	3,27	3,70
OSP	6,20	6h40	6,47	6,47	6,27	6,17	5,93
ASS	1,87	3,17	3,97	4,67	5,20	5,43	5,47
jSO	5,03	4,17	3,53	3,43	3,50	3,50	3h30
(a) 1 à 30 % d'évaluations							
Algorithme	40	50	60	70	80	90	100
AEO	5,93	6,03	6,03	6,07	6,13	6,10	6,10
EBOavec CMAR	1,88	1,80	1,85	1,85	1,73	1,88	1,93
ELSHADE-SPACMA 1.88		2,17	2,25	2,37	2,58	2,65	2,67
GSKA	4,00	3,97	4,07	3,98	3,93	3,93	3,88
OSP	5,77	5,70	5,57	5,47	5,47	5,43	5,37
ASS	5,60	5,63	5,63	5,70	5,67	5,70	5,73
jSO	2,93	2,70	2,60	2,57	2,48	2h30	2,32
(b) 40 à 100 % d'évaluations							

Tableau

5 Evolution du classement moyen par rapport aux évaluations fitness dans le benchmark CEC'2017 (dimension 30).

Algorithme	1	2	3	5	dix	20		
OEA	2.33	3,70	4.53	5.10	5.57	6.00	6.10	
EBO avec CMAR	5.27	4,67	4.03	3.47	2,77	1,90	2.07	
ELSHADE-SPACMA 6.53		5.63	5.27	4.17	2,83	1,93	1,70	
GSKA	1,63	1,40	1,40	1,43	2,10	3,20	3,97	
OSP	5.33	6.07	6.37	6,50	18h30	6.17	6.07	
ASS	2,30	2,87	3.20	3,57	4,57	5,07	5.20	
jSO	4,60	3,67	3,20	3,77	(a) 1–30 %	3,87	3,73	2.90
évaluations 40 60 70								
Algorithme	40				80	90	100	
OEA	6.20	6.27	6.27	6.27	6.20	6.17	6.17	
EBO avec CMAR	2.18	2,37	2,43	2.32	1,92	1,92	1,92	
ELSHADE-SPACMA 1.52		1,40	1,38	1,45	1,72	1,83	2.13	
GSKA	4.10	4.20	4.17	4.17	4.17	4.17	4.20	
OSP	6.03	5,97	5,97	5,97	5,93	5,93	5,87	
ASS	5.27	5.40	5.43	5.47	5,60	5,63	5,67	
jSO	2,70	2,40	2,35	2,37	(b) 40–100 %	2,47	2,35	2.05
Évaluations								

évaluations de la condition physique, y compris les précédents gagnants des compétitions CEC (EBOwithCMAR et jSO) et l'algorithme récent le plus compétitif (ELSHADE-SPACMA). • EBOwithCMAR est l'algorithme avec les meilleurs résultats pour les dimensions 10 et 30 (très proche de ELSHADE-SPACMA dans ces dimensions), alors que ELSHADE-SPACMA est l'algorithme le plus performant pour les dimensions 50 et 100, suivi de près par EBOwithCMAR et jSO. • Pour les dimensions 30, 50 et 100, ELSHADE-SPACMA est le meilleur depuis les 40% du budget des évaluations. Bien que le [tableau 3](#) indique que EBOwithCMAR obtient les meilleurs résultats finaux, ELSHADE SPACMA obtient de meilleurs résultats pendant la majeure partie de la recherche, n'étant amélioré par le premier qu'à la fin.

- Décider quel algorithme doit être appliqué à un problème spécifique dépend fortement de l'effort qui peut être consacré à la recherche. Dans ce benchmark, GSKA est meilleur lorsque moins d'évaluations sont autorisées, tandis que ELSHADE-SPACMA est préféré lorsqu'un plus grand nombre d'évaluations peut être autorisé.

De plus, conformément à la directive #2 ([section 4](#)), nous avons procédé à une validation statistique des résultats pour rejeter l'hypothèse selon laquelle les différences observées dans les performances des algorithmes sont dues à leur nature stochastique et non à des différences réelles dans leurs performances. Tout d'abord, nous utilisons le test de somme des rangs de Friedman pour savoir si des différences significatives peuvent être trouvées entre tous les algorithmes. Les valeurs de p rapportées par ce test sont 3,61e-08, 1,51e-07, 7,58e-7 et 9,62e-10 pour les dimensions

Tableau 6

Evolution du classement moyen par rapport aux évaluations fitness dans le benchmark CEC'2017 (dimension 50).

Algorithme	1	2	3	5	dix	20	
AEO	2,97	4.20	4,90	5.47	5,77	5,97	6.03
EBO avec CMAR	5,77	4.57	4.17	3,67	2,77	2.27	2.40
ELSHADE-SPACMA 6.57		5,90	5.43	4.37	3.10	2.07	1,93
GSKA	1,67	1,83	1,83	1,80	2,60	3,70	4.17
OSP	4.70	5,90	6.17	6h40	6h40	6.23	6.23
ASS	1,90	2.17	2.37	2,73	3.63	4.40	4.73
jSO	4.43	3.43	3.13	3,57	3,73	3.37	2,50
(a) 1 à 30 % d'évaluations							
Algorithme	40	50	60	70	80	90	100
AEO EBO	6.07	6.10	6.10	6.17	6.23	6.20	6.17
avec CMAR	2.47	2,50	2,57	2,50	2.23	2.27	2.17
ELSHADE-SPACMA 1.77		1.37	1,23	1,20	1,30	1,38	1,78
GSKA	4h30	4.33	4.33	4.40	4.40	4.40	4.43
OSP	6.13	6.10	6.07	6.03	6.03	6.00	6.00
ASS	4.93	5.03	5.07	5.13	5,27	5.33	5.33
jSO	2.33	2,57	2.63	2,57	2.53	2.42	2.12
(b) 40 à 100 % d'évaluations							

Tableau

7 Evolution du classement moyen par rapport aux évaluations fitness dans le benchmark CEC'2017 (dimension 100).

Algorithme	1	2	3	5	dix	20	30	
OEA	3.40	4.57	5.13	5.50	5.73	5.83	5.80	
EBO avec CMAR	5.83	4.40	4.10	3.90	3.10	2.53	2.70	
ELSHADE-SPACMA 6.40		6.10	5.33	4.47	3.70	2.93	2.43	
GSKA	1.83	1.93	1.97	2.17	2.77	4.07	4.13	
OSP	4.43	5.77	18h30	6h40	6h40	6h40	6.43	
ASS	1.63	1.97	2.07	2.50	2.97	3.77	4.17	
jSO	4.47	3.27	3.10	3.07	3.33	2.47	2.33	
(a) 1 à 30 % d'évaluations								
Algorithme	40	50	70	60		80	90	100
OEA	5.90	5.97	6.00	6.03	6.07	6.07	6.07	6.00
EBO avec CMAR	2.67	2.50	2.53	2.43	2.13	2.23	2.37	
ELSHADE-SPACMA 2,00	1,43		1,33	1,40	1,47	1,43		1,38
GSKA	4.33	4.43	4.43	4.47	4.50	4.50	4.53	
OSP	6.43	6h40	6h40	6h40	6.37	6.37	6h40	
ASS	4.33	4.53	4.63	4.63	4.67	4.73	4.73	
jSO	2.33	2.73	2.67	2.63	2.80	2.67	2.58	
(b) 40 à 100 % d'évaluations								

Tableau

8 Validation statistique pour le benchmark CEC'2017 et la dimension 10 (EBOwithCMAR est l'algorithme de contrôle).

EBOwithCMAR versus Wilcoxon p-value Wilcoxon p-value		
OEA	2.702e-06	1.621e-05 √
ASS	3.703e-06	1.852e-05 √
OSP	4.110e-06	1.852e-05 √
GSKA	4.374e-05	1.312e-04 √
ELSHADE-SPACMA	0,027	0,055
jSO	0,225	0,225

√ : des différences statistiques existent avec un niveau de signification = 0,05.
: valeur de p corrigée avec la procédure de Holm.

10, 30, 50 et 100, respectivement. Puisque toutes les valeurs des p-values sont nettement inférieures au niveau de confiance = 0,05, nous pouvons affirmer que des différences entre les algorithmes existent et qu'elles sont significatives.

Une fois que des différences significatives sont détectées, nous procédons à une comparaison multiple, car l'utilisation de la procédure de Holm permet de contrôler le taux d'erreur par famille. [Les tableaux 8, 9, 10, 11](#) résument les résultats pour les dimensions 10, 30, 50 et 100, respectivement. En examinant ces résultats, nous arrivons aux idées suivantes :

- EBOwithCMAR est le meilleur algorithme pour les dimensions 10 et 50, alors que ELSHADE-SPACMA est le meilleur algorithme pour les dimensions 50 et 100.

Tableau

9 Validation statistique pour le benchmark CEC'2017 et la dimension 30 (EBOwithCMAR est l'algorithme de contrôle).

EBOwithCMAR versus Wilcoxon p-value Wilcoxon p-value		
OEA	2.702e-06	1.621e-06 ✓
OSP	2.702e-06	1.621e-06 ✓
ASS	2.702e-06	1.621e-06 ✓
GSKA	4.110e-06	1.621e-06 ✓
ELSHADE-SPACMA	0,226	0,452
jSO	0,431	0,452

✓ : des différences statistiques existent avec un niveau de signification = 0,05.
: valeur de p corrigée avec la procédure de Holm.

Tableau

10 Validation statistique pour le benchmark CEC'2017 et la dimension 50 (ELSHADE-SPACMA est l'algorithme de contrôle).

ELSHADE-SPACMA versus Wilcoxon p-value Wilcoxon p-value		
OEA	1.863e-09	1.118e-09 ✓
GSKA	1.863e-09	1.118e-09 ✓
OSP	1.863e-09	1.118e-09 ✓
ASS	1.863e-09	1.118e-09 ✓
jSO	0,054	0,107
EBO avec CMAR	0,509	0,509

✓ : des différences statistiques existent avec un niveau de signification = 0,05.
: valeur de p corrigée avec la procédure de Holm.

Tableau

11 Validation statistique pour le benchmark CEC'2017 et la dimension 100 (ELSHADE-SPACMA est l'algorithme de contrôle).

ELSHADE-SPACMA versus Wilcoxon p-value Wilcoxon p-value		
OEA	3.725e-09	2.235e-08 ✓
GSKA	3.725e-09	2.235e-08 ✓
OSP	3.725e-09	2.235e-08 ✓
ASS	5.588e-09	2.235e-08 ✓
jSO	2.254e-05	4.507e-05 ✓
EBO avec CMAR	3.128e-04	3.128e-04 ✓

✓ : des différences statistiques existent avec un niveau de signification = 0,05.
: valeur de p corrigée avec la procédure de Holm.

- La plupart des nouvelles propositions (AEO, GSKA et SSA) sont statistiquement pires que les algorithmes concurrents proposés en 2017 (EBOwithCMAR) et la dernière proposition ELSHADE-SPACMA. • Pour les dimensions 10, 30 et 50, il n'y a pas de différences significatives entre EBOwithCMAR, ELSHADE-SPACMA et jSO. Cependant, pour la dimension 100, le récent algorithme ELSHADE-SPACMA est statistiquement meilleur qu'eux.

Comme indiqué dans la directive n° 2, d'autres tests statistiques que ceux utilisés jusqu'à présent peuvent être appliqués pour renforcer nos conclusions sur l'étude de cas. Nous discutons ensuite des résultats de deux nouveaux tests utilisés à cette fin : la distance critique et les tests bayésiens.

La figure 4 montre la pertinence statistique des différences entre les classements moyens par rapport au benchmark CEC'2017. La valeur de distance critique (CD) (donnée par un test post-hoc de Nemenyi à une signification = 0,05) indique la distance absolue minimale entre deux classements moyens à déclarer statistiquement différents l'un de l'autre. On peut observer que parmi les nouvelles propositions comparées, seule ELSHADE SPACMA est compétitive par rapport aux approches les plus performantes, bien que GSKA soit également compétitive pour les petits problèmes. Lorsque la dimensionnalité augmente, la distance entre GSKA et les meilleurs algorithmes (ELSHADE-SPACMA, EBOwithCMAR et jSO) augmente, et la différence de classement avec SSA se réduit.

En dimension 10, les écarts entre les rangs des différents algorithmes se réduisent. Afin d'arriver à des conclusions plus perspicaces, nous appliquons plusieurs tests bayésiens sur les résultats obtenus pour cette valeur de dimensionnalité, visualisant la probabilité bayésienne ajustée dans

Tableau

12 Classement moyen des algorithmes les plus compétitifs pour chaque dimension.

Algorithme	D10	D30	D50	D100	Moyenne
EBO avec CMAR	1.683	1.817	2.100	2.300	1,9750
ELSHADE-SPACMA 2.267	2.133	1.783	1.250	1.858	
jSO	2.050	2.050	2.117	2.450	2,1667

coordonnées barycentriques. L'analyse bayésienne effectuée sur les résultats de paires d'algorithmes sélectionnées donne la probabilité qu'un solveur en exécute un autre, sur la base des valeurs de fonction objectif obtenues par chacun d'eux sur toutes les exécutions et tous les problèmes du benchmark. La distribution de probabilité calculée affichée en coordonnées barycentriques après échantillonnage de Monte Carlo, représentant trois régions : une où le premier algorithme surpasse le second ; un second avec le cas inverse (le second surpasse le premier) ; et une région d'équivalence pratique où les résultats obtenus par chaque algorithme peuvent être considérés comme statistiquement équivalents les uns aux autres. Pour statuer sur cette équivalence, un paramètre appelé corde indique la différence minimale entre les scores des deux méthodes pour qu'elles soient considérées comme significativement différentes l'un à l'autre.

À ce stade, il est important de souligner le fait que corde désigne un seuil imposé sur la différence absolue des valeurs de fitness entre les deux algorithmes comparés. Par conséquent, la corde est interprétable et surmonte les problèmes reconnus identifiés autour de l'utilisation des valeurs de p et des niveaux de signification dans les études recourant au NHST pour l'évaluation statistique [74].

Pour revenir à notre étude de cas, nous considérons plusieurs paires d'algorithmes spécifiques : (EBOwithCMAR, jSO), (jSO, GSKA) et (SSA, GSKA). Nous n'avons considéré que les 20 premières exécutions pour chaque fonction afin de rendre le temps de traitement du test abordable en termes de calcul. La valeur de la corde est fixée à 20. La figure 5 représente les tracés bayésiens pour chacune des paires considérées, à partir desquels nous confirmons les faits suivants :

- Il n'y a presque pas de différences pertinentes entre les résultats obtenus par ELSHADE-SPACMA, EBOwithCMAR et jSO. • Pour les dimensions 30, 50 et 100, il y a clairement deux groupes : l'un constitué par EBOwithCMAR, ELSHADE-SPACMA et jSO ; et un autre composé de GSKA, SSA, PSO et AEO. • L'absence de point échantillonné dans les graphiques comparant jSO et GSKA révèle que la probabilité que GSKA obtienne de meilleurs résultats que jSO est exactement nulle. Ceci est d'autant plus pertinent que la valeur seuil utilisée est relativement élevée compte tenu de l'éventail des fonctions objectifs caractérisant le benchmark.
- Les différences entre SSA et GSKA peuvent être considérées comme significatives pour la dimension 10, GSKA apparaissant comme l'approche la plus performante.

Les résultats montrent que ELSHADE-SPACMA surpasse les algorithmes gagnants précédents EBOwithCMAR et jSO. Ainsi, il pourrait être considéré comme la nouvelle méthode de pointe pour ce benchmark. Afin de mener une analyse plus approfondie, notre étude comprend des comparaisons directes de ELSHADE SPACMA contre EBOwithCMAR et jSO. Les résultats de cette comparaison sont présentés dans le tableau 12, où l'on peut observer que parmi les algorithmes les plus compétitifs, ELSHADE-SPACMA continue d'obtenir les meilleurs résultats globaux.

Le tableau 13 présente les résultats des comparaisons par paires entre ELSHADE_SPACMA, EBOwithCMAR et jSO avec le test de Wilcoxon. Ces résultats révèlent qu'il n'y a des différences statistiques qu'en dimension 100. Ce résultat est confirmé dans le tableau 14, dans lequel nous fournissons les résultats corrigés par la méthode de Holm pour tenir compte de l'erreur familiale en dimension 100.

Benchmark COCO

Nous poursuivons nos discussions sur cette première étude de cas avec les résultats sur le benchmark COCO. À cette fin, nous utilisons d'abord le code source et les outils disponibles sur <https://github.com/numbbo/coco> faire l'expérience

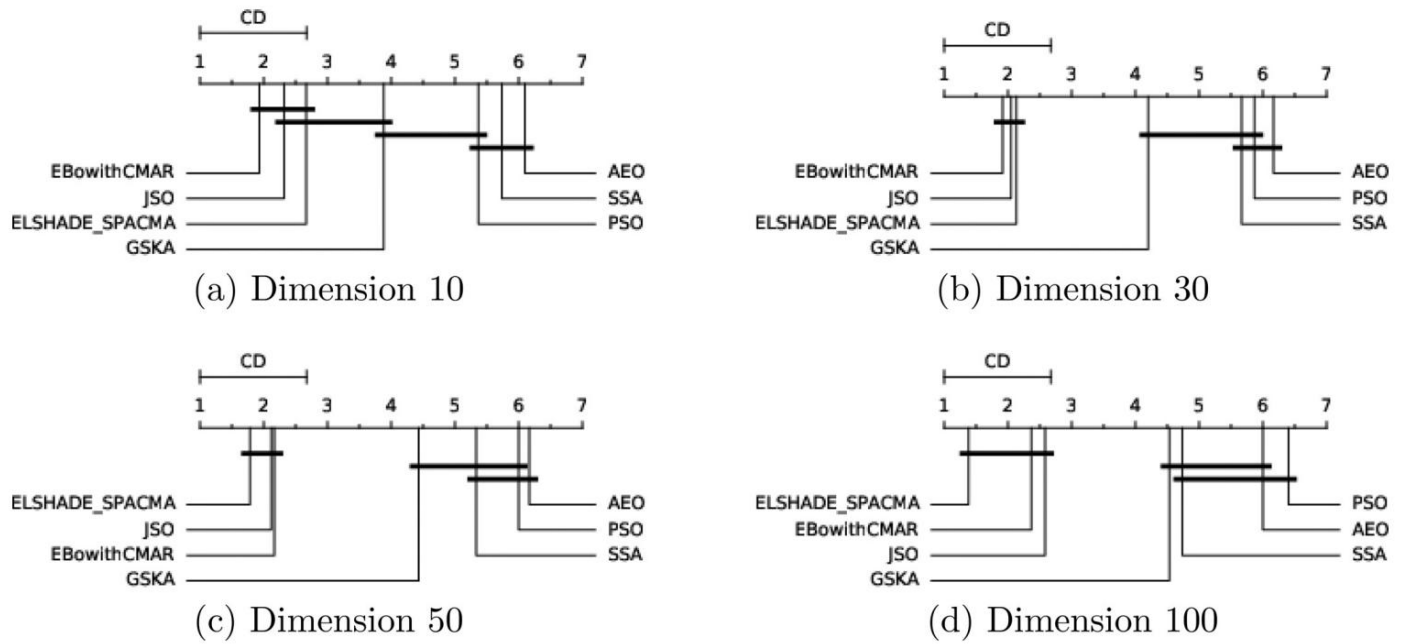


Fig. 4. Tracés de distance critique pour le benchmark CEC'2017.

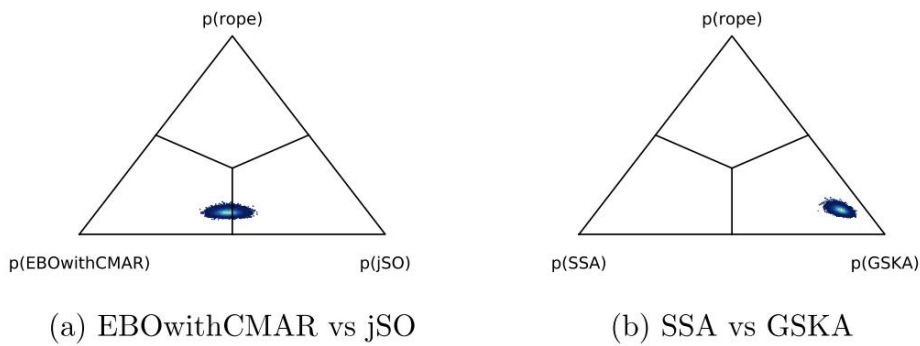
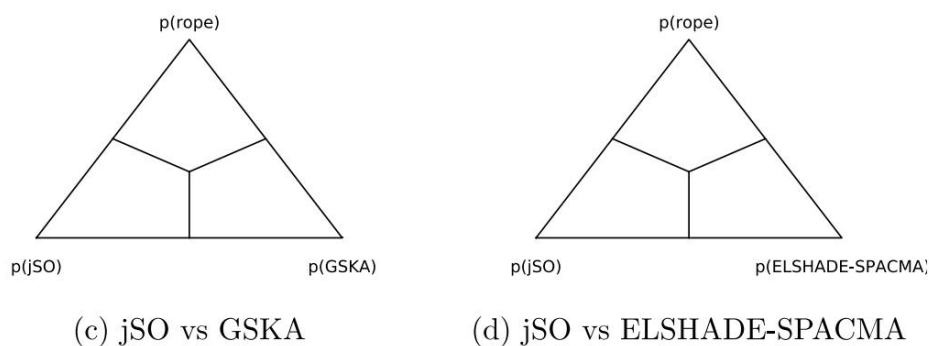


Fig. 5. Tracés bayésiens pour le benchmark CEC'2017 et la dimension 10..



ments et obtenir des résultats de la performance de chaque algorithme. De plus, ces outils permettent d'inclure dans le benchmark l'algorithme concurrentiel proposé en 2009 pour ce benchmark.

La figure 6 visualise, pour chaque algorithme, le ratio de problèmes résolus (problèmes pour lesquels l'erreur obtenue est inférieure à un seuil) lorsque le nombre d'évaluations augmente. Le benchmark est conçu pour différentes valeurs de dimension, de sorte que la Fig. 6 inclut un sous-graphe pour chacune d'entre elles. On peut observer que :

- Pour les faibles valeurs de dimensionnalité (ex. 2 et 5), les résultats obtenus pour les autres propositions sont très similaires et compétitifs, sauf pour l'algorithme de référence (PSO). Cependant, ce comportement change lorsque la dimensionnalité augmente.
- Pour une dimensionnalité égale à 5, ELSHADE-SPACMA (l'acronyme est abrégé sur la Fig. 6) et GSKA sont nettement meilleurs que SSA et AEO.
- Pour la dimension 10, cette amélioration notée augmente encore.

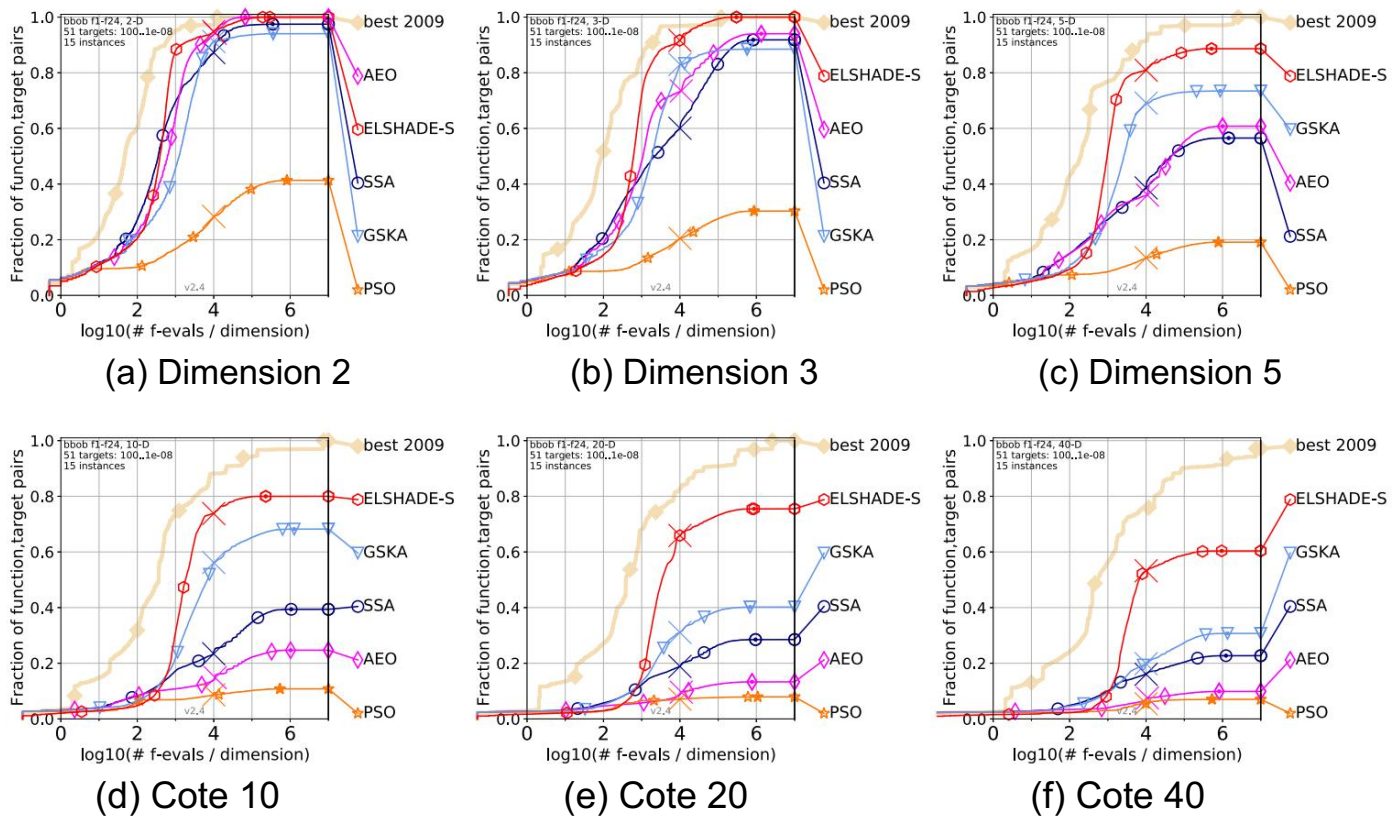


Fig. 6. Rapport entre les problèmes résolus et l'évaluation de la condition physique pour le benchmark COCO.

Tableau 13

Résultats du test de Wilcoxon pour les algorithmes les plus compétitifs dans le benchmark CEC'2017 pour différentes valeurs de dimension.

Algorithmes	D10	D30	D50	D100
ELSHADE-SPACMA contre EBOavecCMAR	0,027	0,226	0,509	3.128e-04
EBOavecCMAR contre jSO	0,431	0,054	0,477	0,796
ELSHADE-SPACMA contre jSO	0,796	0,706	0,556	2.254e-05

Tableau 14

Validation statistique d'algorithmes plus compétitifs pour le benchmark CEC'2017 et la dimension 100 (ELSHADE-SPACMA est l'algorithme de contrôle).

ELSHADE-SPACMA versus Wilcoxon p-value		
	Wilcoxon p-value	Wilcoxon p-value
jSO	2.254e-05	4.507e-05 ✓
EBO avec CMAR	3.128e-04	3.128e-04 ✓

✓ : des différences statistiques existent avec un niveau de signification = 0,05. : valeur de p corrigée avec la procédure de Holm.

- Pour les valeurs de dimensionnalité 20 et 40, GSKA fonctionne toujours mieux que AEO, SSA et GSKA, mais la différence diminue considérablement. GSKA présente un meilleur comportement pour des valeurs de dimensionnalité inférieures. • Pour les valeurs de dimensionnalité 20 et 40, ELSHADE-SPACMA est l'algorithme qui obtient le meilleur score par rapport aux autres propositions avec un grand écart. Cependant, il ne s'approche pas des résultats du meilleur algorithme rapporté en 2009.

Suivant la même procédure que dans le benchmark CEC'2017, nous avons également réfléchi à l'opportunité d'appliquer des tests statistiques. Malheureusement, le nombre réduit d'exécutions pour chaque fonction (15) est trop faible pour permettre des tests statistiques avec des garanties de signification minimales.

Conclusions tenant compte des deux critères de référence

Après avoir analysé ensemble tous les résultats expérimentaux discutés précédemment, nos conclusions peuvent être résumées comme suit :

- GSKA obtient de meilleurs résultats que AEO et SSA. • Pour un budget inférieur en termes de nombre d'évaluations de fonctions objectives, GSKA est meilleur que les autres algorithmes. Au contraire, lorsque le nombre d'évaluations est augmenté, EBOwithCMAR et jSO obtiennent de meilleurs résultats.
- Les nouvelles propositions (AEO, GSKA et SSA) sont compétitives par rapport aux algorithmes d'optimisation classiques (par exemple PSO), mais aucune d'entre elles ne peut rivaliser avec les solveurs modernes comme les gagnants permanents de concours renommés.
- GSKA, pour les petites valeurs de dimensionnalité, est meilleur que les autres algorithmes comparés, mais son avantage est réduit lorsque la dimension augmente.
- ELSHADE-SPACMA a montré un comportement plutôt opposé : il s'améliore à mesure que la valeur de la dimension augmente. C'est en fait le seul algorithme capable de surpasser les algorithmes concurrents existants, en particulier dans les valeurs de dimensionnalité plus élevées. Par exemple, pour la dimension 100, il est même statistiquement meilleur que les algorithmes gagnants précédents. Dans le benchmark COCO en revanche, bien qu'il soit très compétitif, il ne fait pas mieux que le précédent lauréat du concours 2009. Cela pourrait être dû aux valeurs de dimensionnalité inférieures utilisées dans cette compétition.

Au total, les performances de ces algorithmes méta-heuristiques récents peuvent être intéressantes par rapport à celles offertes par les approches classiques, mais n'atteignent pas les niveaux de performance atteints par les méthodes modernes d'optimisation compétitive.

7.1.5. Analyse et réglage des composants conformément à la directive #3

Il ne suffit pas de comparer avec des algorithmes de référence et/ou de pointe. Conformément à la directive n° 3 (section 5), il est important pour les nouveaux

Tableau
15 Classement moyen pour différentes valeurs d'attraction et de taille d'essaim pour l'algorithme et la dimension SSA.

		taille de l'essaim				
		dix	25	50	75	100
attraction	0,1	20h40	18.05	17.82	17h35	17.52
	0,3	19.87	15.58	17.02	14.05	13.58
	0,5	19h30	18h38	13.88	12.65	10h25
	0,7	17.80	17.25	13.08	12.55	8.22
	0,9	18.87	18.15	14.75	10.82	10.35
	1.0	18.83	18.78	14.82	13h48	11h55
(b) Dimension 10						
		taille de l'essaim				
		dix	25	50	75	100
attraction	0,1	19.75	19.68	14.72	13.72	12h55
	0,3	18.75	17h32	15.92	11,95	12.82
	0,5	18h15	19.28	16.48	9h25	10.68
	0,7	19.88	15.88	14h32	12.65	9h35
	0,9	18.78	18h48	15.22	11.85	12h38
	1.0	20h35	18.68	16h45	15.28	14h42
(b) Cote 30						
		taille de l'essaim				
		dix	25	50	75	100
attraction	0,1	20.12	17h42	16.68	15.08	16.75
	0,3	18.58	15.82	12.02	10.75	13.12
	0,5	19h38	17h65	15.05	13.82	11h25
	0,7	17h55	16.48	12.62	13h25	10.28
	0,9	21h25	18.02	15h25	14.08	11h65
	1.0	18.05	19.92	16.68	12.88	13h55
(c) Cote 50						

propositions d'analyse du comportement de l'algorithme pour jauger la contribution de chaque composant. Dans ce cas d'utilisation, puisque nous avons plusieurs nouveaux algorithmes et que le but est de les comparer équitablement, analyser les différents composants de chacun d'entre eux car pourrait impliquer une étude approfondie qui dépasse le cadre de ce travail. Pour un exemple d'analyse en composantes, nous renvoyons à [la section 7.2.5](#).

Conformément à la directive 3, l'utilisation d'un mécanisme de réglage automatique est également recommandée pour les différents algorithmes de la comparaison. En particulier, nous nous concentrons sur les nouveaux algorithmes de proposition (AEO, GSKA et SSA), car les gagnants des concours, EBOwithCMAR et jSO, ont été préalablement optimisés par leurs auteurs pour le concours. Cependant, dans notre cas, la publication originale où l'algorithme le plus prometteur (GSKA et ELSHADE-SPACMA ont été proposés considérât déjà la même référence (CEC'2017), de sorte que la valeur du paramètre utilisée dans cette référence séminale est supposée être la plus adéquate. pour ce benchmark utilisé dans l'étude de cas. En revanche, bien qu'initialement évalué sur un benchmark différent, l'OEA n'a pas de paramètres à optimiser. Son seul paramètre libre (la taille de la population) a été analysé dans plusieurs travaux de ses auteurs, concluant que la variation de la taille de la population n'entraîne pas de changement significatif dans la précision [110] Par conséquent, nous centrons notre discussion sur le deuxième algorithme le plus performant dans les résultats (SSA) : Tuning SSA Pour illustrer ce processus, nous accordons les deux paramètres déterminant le comportement de recherche de l'approche SSA, à savoir la taille de l'essaim et le facteur d'attraction, la valeur inférieure de la distance de glissement Comme il n'y a que deux paramètres, au lieu d'utiliser un outil tel qu'un Comme celles commentées dans [la sous-section 5.4](#), nous avons appliqué une recherche par grille pour les dimensions 10, 30 et 50 (la dimension 100 a été omise pour des raisons d'accessibilité financière). Dans cette grille de recherche, différentes valeurs sont explorées pour chaque paramètre :

- Pour la taille de l'essaim, nous avons testé les valeurs 10, 25, 50 (recommandé par les auteurs), 75 et 100. • Pour le facteur d'attraction, nous avons considéré des valeurs égales à 0,1, 0,3, 0,5 (recommandé par les auteurs), 0,7, 0,9 et 1,0.

À la suite de cette recherche de grille, les combinaisons de paramètres $5 \times 6 = 30$ ont donné lieu à 30 configurations de SSA que nous avons comparées à

Tableau
16 Classement moyen des algorithmes accordés pour chaque dimension.

Algorithme	D10	D30	D50
ELSHADE-SPACMA 2.667		2.133	1.783
EBO avec CMAR	1.933	1.917	2.167
jSO	2.317	2.050	2.117
GSKA	3.883	4.200	4.433
OSP	5,400	5,933 6,067 5,433 (-0,23)	
SSA (réglé)	5,533 (-0,40)	5,200 (+0,13) 6,333 6,233	
OEA	6,267		

- les uns des autres en termes de classement moyen. Ces résultats sont donnés dans [le tableau 15](#), où l'on peut remarquer que :
- Les meilleurs résultats sont obtenus avec un facteur d'attraction de 0,7 et une taille d'essaim égale à 100 pour des valeurs de dimensionnalité de 10 et 50. De plus, cette configuration est la deuxième meilleure pour la dimension 30. Ainsi, ces valeurs peuvent être déclarées comme étant le paramètre recommandé pour l'algorithme.
 - En général, les résultats sont meilleurs avec une taille d'essaim plus grande et avec une valeur moyenne du facteur d'attraction. • La valeur recommandée par les auteurs pour la valeur d'attraction (0,5) est proche de la meilleure obtenue lors du processus de réglage (0,7). De plus, c'est le meilleur pour la dimension 30.
 - La valeur recommandée pour ses auteurs en taille d'essaim, 50, n'est pas adéquate pour ce benchmark.

Nous avons conclu que les résultats nets de SSA s'améliorent via le réglage des paramètres. Cependant, nous devons évaluer si cette amélioration de la performance a un impact sur la comparaison discutée précédemment à la [section 7.1.4](#). En tant que recommandation globale, les paramètres de tous les algorithmes comparés doivent être réglés. Cependant, dans ce cas, les algorithmes les plus compétitifs (ELSHADE-SPACMA, EBOwithCMAR, jSO et GSKA) ont été réglés à l'origine sur cette référence. En conséquence, la seule approche concurrentielle qui nécessite un réglage est la SSA. Le reste des algorithmes n'a pas été réglé compte tenu de ses résultats non compétitifs et des ressources de calcul limitées.

Dans [le tableau 16](#) sont présentés les résultats des algorithmes accordés, montrant pour l'algorithme accordé (SSA) la différence dans le classement moyen en com

paraison avec la version non réglée. La dimension 100 n'a pas été incluse car aucun réglage n'a été effectué dans cette dimension. On peut observer que la version accordée de l'algorithme réalise une amélioration par rapport à son homologue non accordée pour les dimensions 10 et 30, et elle est pire pour la dimension 50. Cependant, l'amélioration n'est pas suffisante pour changer la position relative de l'algorithme dans le classement.

7.1.6. Justifier l'utilité de l'algorithme selon la ligne directrice #4

Suite à la Ligne directrice #4 (Section 6), il y a plusieurs perspectives dont on peut revendiquer l'utilité d'un algorithme :

- **Qualité des résultats** : dans ce cas, la majorité des algorithmes évalués se sont révélés insuffisamment compétitifs par rapport aux méthodes de l'état de l'art. Seul ELSHADE-SPACMA a pu les améliorer, devenant ainsi l'algorithme de pointe actuel dans le benchmark compte tenu de ces résultats. De plus, les comparaisons entre les différents algorithmes révèlent des différences significatives entre eux. Par conséquent, les résultats peuvent être considérés comme intéressants et informatifs pour les chercheurs souhaitant approfondir leur conception. De plus, bien que GSKA ne soit pas compétitif par rapport à jSO et EBOwithCMAR, il est capable de mieux fonctionner lorsque le nombre d'évaluations de fonctions est faible. Dans de nombreux problèmes du monde réel, l'évaluation d'une solution peut être très coûteuse en termes de ressources de calcul (par exemple, lorsque la valeur de fitness est produite par de longues simulations informatiques). Dans ces circonstances, il est indispensable de s'appuyer sur des algorithmes capables d'obtenir de bons résultats en un petit nombre d'évaluations.
- **Nouveauté technique** : les propositions comparées ont non seulement une inspiration biologique très différente, mais elles diffèrent notamment par leur comportement algorithmique. En particulier, ELSHADE-SPACMA, SSA et GSKA proposent des idées intéressantes qui pourraient être envisagées pour de nouveaux algorithmes. ELSHADE-SPACMA combine différents algorithmes précédents (SHADE-ILS et CMA-ES) avec un nouveau mécanisme pour améliorer la diversité dans la population, un opérateur de mutation qui considère non seulement les meilleures solutions mais aussi les pires. Une autre contribution est l'adaptation d'un paramètre utilisé pour augmenter la diversité dans les premiers stades de la recherche. Avec ces deux principaux changements qui le différencient de l'algorithme LSHADE SPACMA précédent, il est en mesure d'améliorer les résultats de l'EBOwithCMAR et du jSO compétitifs. LSHADE-SPACMA n'a pas réussi à le faire. D'autre part, SSA exploite l'idée de classer les différentes solutions pour créer plusieurs catégories (la meilleure actuelle, les meilleures et la normale, qui est le groupe le plus important), et utilise une méthode de mutation différente en tenant compte de la catégorie de chacune. individuel. De plus, les solutions qui ne peuvent pas être améliorées sont périodiquement redémarrées. Ces deux stratégies donnent un algorithme d'optimisation très simple mais efficace. GSKA propose également d'autres concepts intéressants du point de vue technique : il met à jour les variables de chaque individu sous deux critères possibles, parmi lesquels celui sélectionné est mis à jour lors de la recherche pour augmenter l'exploitation lors de l'exécution de l'algorithme. De plus, dans l'exploration, GSKA utilise pour chaque individu les plus similaires en fitness, le meilleur immédiat et le pire immédiat. • **Apport méthodologique** : Il n'y a pas d'apport méthodologique. • Une attention particulière doit être accordée à la simplicité, dans laquelle SSA se démarque dans le benchmark. Cependant, des modifications doivent être apportées pour améliorer ses résultats et éviter son apparence prématurée.

convergence.

7.1.7. Résumé du cas d'utilisation

Pour conclure, ce premier cas d'utilisation suit la plupart des lignes directrices de notre méthodologie proposée. Les principales procédures suivies dans le cas d'utilisation sont mises en évidence dans la Fig. 7. En outre, nous décrivons brièvement maintenant les principales actions entreprises pour chacune des lignes directrices proposées :

- **Ligne directrice n°1** : Nous avons sélectionné deux référentiels à paramètres réels différents, CEC'2017 et COCO. Les deux sont largement acceptés par la com

munauté travaillant dans l'optimisation des paramètres réels. Nous avons également comparé nos algorithmes à des solveurs concurrents qui ont remporté des concours en utilisant les mêmes benchmarks, ainsi qu'une ligne de base de référence pour l'intelligence en essaim (c'est-à-dire PSO). • **Ligne directrice #2** : suivant cette ligne directrice, nous avons montré et validé les résultats selon les bonnes pratiques décrites dans la section 4, y compris les tests statistiques non paramétriques et l'analyse bayésienne. De plus, nous avons fourni des preuves sur la façon dont l'identification du meilleur algorithme peut être fortement biaisée par le critère d'arrêt utilisé (par exemple le nombre maximum d'évaluations de fonctions objectives). • **Ligne directrice n° 3** : en suivant les suggestions de cette ligne directrice, nous avons appliqué un processus de réglage simple et dévoilé que les bonnes valeurs de paramètres peuvent influencer les résultats et les conclusions tirées de la

comparaison précédente. •

Ligne directrice n°4 : dans ce cas d'utilisation, les comparaisons entre les différents algorithmes pourraient être justifiées en fonction des différences pertinentes trouvées entre eux. Un seul des algorithmes comparés (ELSHADE SPACMA) a été capable de surpasser les autres algorithmes concurrents existants. De plus, l'un d'entre eux (GSKA) a montré un comportement supérieur lorsque le nombre d'évaluations de fonctions est faible. Jamais

moins, le fait que la majorité de ces algorithmes modernes ne pouvaient pas améliorer les solveurs concurrents précédents est un fait pertinent qui devrait stimuler des études de comparaison plus justes dans des travaux prospectifs avec des versions nouvelles et/ou améliorées d'algorithmes d'optimisation bio-inspirés.

Enfin, certains de ces algorithmes posent des idées algorithmiques innovantes qui devraient être approfondies en vue de leur utilisation dans la conception de nouvelles méthodes de recherche.

7.2. SHADEILS pour une optimisation globale à grande échelle

Dans cette deuxième étude de cas, nous simulons la situation dans laquelle nous concevons un nouvel algorithme, SHADE-ILS, spécialement conçu pour l'optimisation globale à grande échelle. Dans cette section, nous suivons les lignes directrices pour mener à bien les expériences, les comparaisons avec d'autres algorithmes de référence et l'analyse pour mettre en valeur les avantages de notre proposition méthodologique.

7.2.1. Sélection de l'indice de référence conformément à la

directive n° 1 Tout d'abord, nous devons choisir le bon indice de référence pour l'évaluation expérimentale des performances de notre nouvel algorithme proposé. Suivant les recommandations de la ligne directrice #1 (décrite à la section 3), nous devons :

- **Bien sélectionner le benchmark** : sans biais inattendu, avec le bon niveau de complexité, et pour le type de problème adressé par l'algorithme. • Appliquer l'utilisation d'un référentiel standard qui satisfait aux exigences précédentes

exigences.

La sélection du benchmark ne peut se faire sans considérer l'algorithme proposé, puisqu'il dépend des caractéristiques du problème pour lequel l'algorithme a été implémenté (ou du type de problèmes pour lesquels on veut le tester). Dans notre exemple, nous avons conçu SHADE-ILS [117], un algorithme spécialement conçu pour les problèmes d'optimisation à paramètres réels comportant un nombre élevé de variables. Cette famille de problèmes d'optimisation est collectivement appelée optimisation globale à grande échelle, pour laquelle plusieurs benchmarks ont été proposés [23, 27, 89]. Si l'un d'entre eux permet une comparaison impartiale, nous devons l'utiliser, évitant ainsi la conception de notre propre benchmark. En particulier, notre première option est le benchmark CEC'2013 [23], car il s'agit à la fois du concours le plus récent et le plus populaire à ce jour. De plus, sa popularité donne lieu à de nombreux résultats antérieurs que nous pouvons utiliser à des fins de comparaison. Néanmoins, avant d'aller plus loin, nous devons vérifier si le benchmark sélectionné permet de bonnes comparaisons. À cette fin, les informations et données disponibles sur l'indice de référence doivent répondre à plusieurs exigences :





	Étude de cas 1	Étude de cas 2
 Ligne directrice #1 Repères	Benchmarks CEC'2017 et COCO <input checked="" type="checkbox"/> Deux méthodes de comparaison de pointe <input checked="" type="checkbox"/> Un algorithme classique comme ligne de base <input checked="" type="checkbox"/>	Référence standard adéquate (CEC'2013 LSGO Benchmark) <input checked="" type="checkbox"/> Méthodes de comparaison similaires et à la pointe de la technologie <input checked="" type="checkbox"/>
 Ligne directrice #2 Validation des résultats	Validation statistique non paramétrique <input checked="" type="checkbox"/> Analyse bayésienne <input checked="" type="checkbox"/>	Validation statistique non paramétrique <input checked="" type="checkbox"/> Techniques de visualisation <input checked="" type="checkbox"/>
 Ligne directrice #3 Analyse des composants et réglage des paramètres	Réglage fin des paramètres <input checked="" type="checkbox"/> Implications du réglage des paramètres sur les comparaisons <input checked="" type="checkbox"/>	Énoncé clair des objectifs <input checked="" type="checkbox"/> Des opérateurs prouvant des contributions significatives <input checked="" type="checkbox"/>
 Directive #4 Pourquoi mon algorithme est-il utile ?	Qualité des résultats et convergence des algorithmes <input checked="" type="checkbox"/> Éléments de nouveauté technique identifiés <input checked="" type="checkbox"/>	Compétitivité de SHADE-ILS par rapport aux algorithmes de référence <input checked="" type="checkbox"/> Nouvelles directions de recherche (hybridation avec des méthodes de recherche locales) <input checked="" type="checkbox"/>

Fig. 7. Liste de contrôle des recommandations des lignes directrices suivies par les cas d'utilisation.

done Conditions expérimentales claires : la configuration expérimentale est bien définie et les conditions sont identiques pour tous les algorithmes. done L'implémentation du benchmark est ouvertement disponible : le benchmark CEC'2013 est particulièrement approprié à cet égard, car les implémentations des problèmes composant le benchmark sont non seulement rendues publiques, mais également dans plusieurs langages de programmation : C/C++, Matlab, Java et Python1

done L'optima n'est pas au centre de la recherche de domaine : toutes les fonctions du benchmark choisi sont décalées pour garantir cette fonctionnalité. wontfix Les fonctions sont tournées : bien que cette fonctionnalité ne soit pas présente dans le benchmark choisi, l'importance de cette exigence n'est pas aussi critique que le décalage susmentionné. done Présence d'optima locaux : dans le benchmark, il existe plusieurs fonctions avec différents optima locaux. En fait, ce n'est pas le seul critère pour fournir des fonctions avec différents niveaux de difficulté. En particulier, dans ce benchmark, il existe différents degrés d'interrelation entre les variables, ce qui est logique compte tenu de la grande dimensionnalité des problèmes.

En résumé, l'analyse ci-dessus conclut que le benchmark CEC'2013 pour l'optimisation globale à grande échelle suit la plupart des exigences imposées par notre méthodologie. C'est la raison pour laquelle nous la sélectionnons comme référence pour l'expérimentation.

7.2.2. Sélection de la mesure du rendement conformément à la ligne directrice no 1

Une autre décision importante à prendre est le choix d'une mesure de performance adéquate. À cet égard, nous pouvons mesurer non seulement l'erreur de fitness finale (écart par rapport à l'optimum global connu a priori), mais également l'erreur pour un nombre différent d'évaluations de fitness (appelé niveau de précision). De cette façon, nous pouvons mesurer équitablement l'efficacité des algorithmes. Il y a deux possibilités à cet égard : i) rapporter les performances pour chaque niveau de précision ; et ii) fournir la performance pour le nombre maximum d'évaluations de condition physique prises en compte dans les expériences. Pour présenter les résultats de manière concise, nous n'aborderons que la dernière de ces alternatives (c'est-à-dire les résultats pour le nombre maximum d'évaluations de fitness). Cependant, l'étude doit être effectuée de manière similaire pour chaque niveau de précision.

En ce qui concerne les performances, il existe également plusieurs possibilités : nous pouvons signaler l'erreur de fitness fonction par fonction, ou nous pouvons calculer une mesure agrégée des performances (comme une moyenne). Dans un premier temps, nous optons pour une mesure agrégée, en considérant deux options :

- Classement moyen, qui est calculé en triant les algorithmes de chaque fonction en fonction de son erreur (position inférieure aux meilleures). Ensuite, le classement moyen est calculé de sorte qu'un algorithme avec un

la valeur de classement moyenne est déclarée plus performante, en moyenne, que les autres avec une valeur de classement plus élevée.

- Une mesure particulière proposée dans le benchmark considéré, qui signe pour chaque fonction un score différent à chaque algorithme, en fonction de sa position dans le classement.

La mesure de performance recommandée dans les compétitions avec le benchmark CEC'2013 est la deuxième de ces options. Cependant, nous décrirons d'abord le classement moyen, car il évalue les performances des algorithmes d'une manière plus générale et compréhensible.

7.2.3. Sélection des algorithmes de référence conformément à la directive n° 1

Afin de faire une bonne comparaison, un critère clair est nécessaire pour sélectionner les algorithmes inclus dans la comparaison, visant à prouver équitablement la commodité de l'algorithme en ce qui concerne ses performances compétitives avec d'autres méthodes. En suivant les directives, nous devrions :

- Comparer aux algorithmes de référence : dans ce benchmark DECCG [23] assumera ce rôle.
- Comparer avec des algorithmes similaires : cet aspect est particulièrement pertinent lorsque l'algorithme proposé est une version modifiée d'une approche précédemment publiée. Dans notre cas, SHADE-ILS peut être considéré comme un nouvel algorithme. Cependant, d'autres propositions présentant des concepts similaires ont déjà été proposées dans la littérature, comme IHDELS [118]. Conformément à nos directives, nous avons inclus ces méthodes précédentes pour leur comparaison avec notre proposition.
- Comparer avec des algorithmes concurrents : c'est souvent une décision difficile à prendre, car il est difficile d'examiner l'ensemble de l'état de l'art lié au problème d'optimisation/algorithme/benchmark considéré. Cependant, comme le benchmark est largement utilisé dans les compétitions internationales, nous pouvons utiliser les approches gagnantes de ces compétitions comme algorithmes compétitifs auxquels comparer notre approche proposée. À ce titre, l'un des solveurs dans ce domaine est MOS-CEC2013 [119], qui est depuis des années le meilleur algorithme de ces compétitions. De plus, nous allons également inclure MLSHADE-SPA [120] dans notre comparaison, car il a été signalé qu'il surpassait les résultats de MOS-CEC2013 lors du concours de 2018. De nos jours, il existe d'autres algorithmes concurrents, mais nous nous concentrons sur les algorithmes proposés jusqu'en 2018, année où l'algorithme a été présenté [117].

Dans l'ensemble, nous comparons notre méthode à une version précédente considérée (IHDELS), à des algorithmes concurrents (MOS-CEC2013, MLSHADE SPA) et à un algorithme de référence (DECCG).

7.2.4. Tester et valider les résultats conformément à la ligne directrice n° 2

Après la conception de l'expérimentation, des expériences sont réalisées et les résultats sont validés. Conformément aux recommandations sur la validation statistique dans les lignes directrices 3.2 et 4.1, des tests de normalité ou d'homocédasticité doivent être effectués. Cependant, il a été prouvé que de tels tests

1Code pour le benchmark CEC'2013 Large Scale Global Optimization : <https://www.tflsgo.org/specialsessions/wcci2020.html#new-code> (consulté le 16 avril 2020).

Tableau

17 Classement moyen des algorithmes considérés pour la comparaison statistique.

Algorithme	Classement
SHADE-ILS	1.967
MLSHADE-SPA	2.433
MOS-CEC2013	2.700
IHDELS	3.633
DECCG	4.267

Tableau

18 Validation statistique (SHADE-ILS est l'algorithme de contrôle).

SHADE-ILS versus	Wilcoxon p-value	Wilcoxon p-value
MLSHADE-SPA	1.51e-01	1.51e-01 ≈
MOS-CEC2013	4.79e-02	9.58e-02 ≈
IHDELS	1.53e-03	2.50e-02 √
DECCG	8.36e-03	6.10e-03 √

√ : des différences statistiques existent avec un niveau de signification = 0,05. : valeur de p corrigée avec la procédure de Holm.

ne passent généralement pas pour un repère comme celui choisi [41]. Par conséquent, nous avons opté pour des tests non paramétriques, étant donné qu'il est peu probable que la normalité et l'homocédasticité soient vérifiées pour le benchmark CEC'2013.

Ainsi, la première étape à franchir consiste à calculer le classement moyen, suivi du test d'hypothèse non paramétrique. Afin de comparer les algorithmes, nous avons recours à Tacolab [116]¹⁴, un outil web qui facilite l'application de différentes méthodes de comparaison entre algorithmes.

Le tableau 17 montre le classement moyen des quatre algorithmes comparés par rapport au benchmark CEC'2013 LSGO. Nous rappelons que SHADE-ILS est la nouvelle proposition algorithmique, alors que MLSHADE-SPA est une autre proposition présentée dans le même concours que IHDELS, et MOS CEC2013 et DECCG sont les deux méthodes de pointe considérées comme

algorithmes de référence. Le tableau illustre le classement moyen calculé à partir de la position relative des quatre méthodes lorsqu'elles sont classées pour chacune des fonctions du benchmark. Comme on peut l'observer dans ce tableau, SHADE-ILS présente une performance légèrement meilleure que MLSHADE-SPA et MOS-CEC2013, et un bien meilleur classement que IHDELS et DECCG. En particulier, bien que l'approche précédente (IHDELS) ait donné de moins bons résultats que MOS-CEC2013, SHADE-ILS rend une performance nettement meilleure. Cet aspect est assez important, car il n'est pas courant de concevoir directement un algorithme compétitif à partir de zéro.

Comme indiqué dans la directive n° 2 (section 4), les mesures de performance telles que le classement moyen ne sont pas concluantes, car des écarts de performance peuvent survenir en raison de la nature stochastique des algorithmes comparés. C'est la raison pour laquelle ces résultats doivent être analysés plus en détail pour déterminer si les différences sont significatives. Pour cela, nous utilisons le test de la somme des rangs de Friedman. La valeur de p rapportée par ce test est de 4,87e-03, ce qui est clairement significatif au niveau de confiance = 0,05. Maintenant que les différences susmentionnées ont été évaluées, nous pouvons passer à la comparaison multiple, y compris une correction du taux d'erreur par famille abordée avec la procédure de Holm.

Le tableau 18 présente les résultats de cette analyse. Comme on peut le constater, les différences sont significatives entre SHADE-ILS, DECCG et IHDELS. En ce qui concerne MLSHADE-SPA, il n'y a pas de différences statistiques par rapport à SHADE-ILS. Enfin, dans le cas de MOS-CEC2013, il n'y a pas non plus de différences statistiques après application de la procédure de correction de Holm et utilisation d'un niveau de confiance de 5%, ou augmenter le niveau de confiance jusqu'à 10% permettrait de conclure que les différences sont statistiquement importantes. Cette comparaison doit également prendre en compte différents points de contrôle, par exemple 1 %, 10 % et 100 %, ou tous les 10 % du nombre maximum d'évaluations de la condition physique disponibles. Cette analyse complémentaire refléterait

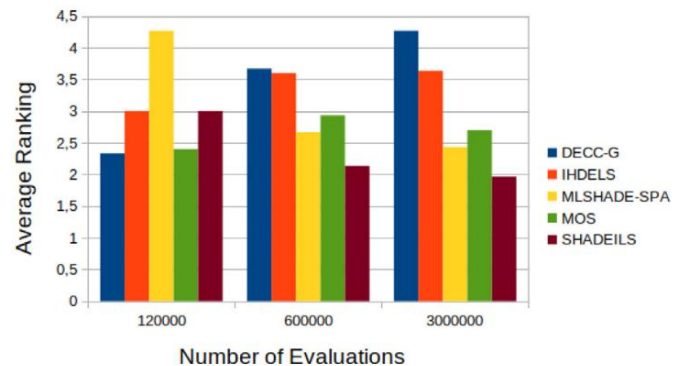


Fig. 8. Classement moyen des algorithmes pour différents nombres d'évaluations de fitness.

non seulement le résultat final des algorithmes, mais aussi leur vitesse de convergence.

De plus, nous avons indiqué dans la section 4.2 qu'une visualisation graphique est utile pour l'analyse. Dans ce cas, nous complétons l'étude résumée dans le tableau 17 avec la figure 8. Dans cette figure, il est plus évident que les différences entre les algorithmes augmentent avec le nombre d'évaluations de fitness : alors que MOS est meilleur que DECCG et IHDELS, tous deux MLSHADE-SPA et SHADE-ILS améliorent leurs résultats depuis 600 000 évaluations, montrant que SHADE-ILS réalise le meilleur classement moyen avec 600 000 et 3 000 000 évaluations. Idéalement, un diagramme de convergence pourrait être plus informatif, mais dans le benchmark CEC'2013, les jalons posés par la concurrence sont très réduits, de sorte qu'un diagramme à barres comme celui illustré s'avère plus utile aux fins de cette analyse.

7.2.5. Analyse et réglage des composants conformément à la directive #3

Comme mentionné précédemment, une comparaison d'une méthode proposée avec uniquement des algorithmes de référence et/ou de pointe n'est généralement pas suffisante. Conformément à la directive n° 3 (section 5), lors de l'analyse de l'algorithme, il est également important de clarifier les objectifs de la conception proposée, puis de montrer des preuves quantitatives des affirmations concernant le comportement de l'algorithme. De cette façon, l'étude peut faire la lumière sur l'influence des différents composants sur les résultats finaux rapportés. Dans notre cas d'utilisation, nous n'expliquons pas les objectifs et les idées principales de l'algorithme. Au lieu de cela, nous remarquons que les principaux changements présentés par SHADE-ILS par rapport à IHDELS sont i) une modification du composant d'évolution différentielle (de SaDE à SHADE); et ii) le mécanisme de redémarrage. Nous renvoyons les lecteurs intéressés à [20] pour plus de détails.

Le tableau 19 montre les résultats obtenus par les différentes composantes de l'algorithme. Ce tableau montre clairement que le comportement de surperformance de la méthode proposée est dû à toutes ses nouvelles contributions, plutôt qu'à un sous-ensemble d'entre elles. De plus, ces changements n'ajoutent pas de complexité au processus de recherche global.

Conformément à la directive 3, l'utilisation d'un mécanisme de réglage automatique est également recommandée pour les différents algorithmes de la comparaison. Cependant, dans notre cas, nous utilisons les résultats rapportés par leurs auteurs dans les contributions où les algorithmes ont été présentés pour la première fois, il est donc attendu que ces résultats ont été obtenus en utilisant les meilleures valeurs de paramètres. En ce qui concerne les valeurs des paramètres de la proposition SHADE-ILS, elles doivent être obtenues par un processus de réglage, idéalement mené par un outil automatique. Dans notre cas, un réglage manuel a été effectué en raison de contraintes de calcul (en particulier, le temps de traitement). Si plus de ressources de calcul étaient disponibles, un processus de réglage complet pourrait être mené en recourant aux outils disponibles tels que ceux commentés dans la sous-section 5.4.

7.2.6. Justifier l'utilité de l'algorithme selon la ligne directrice #4

Conformément à la directive n° 4 (section 6), il existe plusieurs façons de montrer l'utilité d'un algorithme :

¹⁴ Site Web de Tacolab : <https://tacolab.org/>

Tableau
19 Comparaisons entre les différentes composantes de la proposition.

Fonct.	Utilisation de SHADE + nouveau redémarrage	Utilisation de SaDE + nouveau redémarrage	Utilisation de SHADE + ancien redémarrage	IHDELS
1	2.69e-24	1.21e-24	1.76e-28	4.80e-29
2	1.00e+03	1.26e+03	1.40e+03	1.27e+03
3	2.01e+01	2.01e+01	2.01e+01	2.00e+01
4	1.48e+08	1.58e+08	2.99e+08	3.09e+08
5	1.39e+06	3.07e+06	1.76e+06	9.68e+06
6	1.02e+06	1.03e+06	1.03e+06	1.03e+06
7	7.41e+01	8.35e+01	2.44e+02	3.18e+04
8	3.17e+11	3.59e+11	8.55e+11	1.36e+12
9	1.64e+08	2.48e+08	2.09e+08	7.12e+08
dx	9.18e+07	9.19e+07	9.25e+07	9.19e+07
11	5.11e+05	4.76e+05	5.20e+05	9.87e+06
12	6.18e+01	1.10e+02	3.42e+02	5.16e+02
13	1.00e+05	1.34e+05	9.61e+05	4.02e+06
14	5.76e+06	6.14e+06	7.40e+06	1.48e+07
15	6.25e+05	8.69e+05	1.01e+06	3.13e+06
Mieux 12		1	0	2

- Qualité des résultats : dans ce cas, compte tenu des bons résultats dans les comparaisons avec les méthodes de l'état de l'art et de référence, la valeur scientifique de la contribution est évidente. • Nouveauté technique : la combinaison des méthodes de recherche locale et l'hybridation de SHADE sont nouvelles. Cependant, par souci de concision, nous ne développerons pas ici l'originalité de ces ingrédients, car cela nécessiterait une revue exhaustive de l'histoire récente des approches DE pour l'optimisation globale à grande échelle. Nous renvoyons le lecteur à l'analyse faite dans [117] à cet égard.
- Contribution méthodologique : SHADE-ILS améliore une précédente hybridation de DE avec la recherche locale [118] en intégrant, entre autres ajouts algorithmiques, Success-History based Adaptive Differential Evolution (SHADE) en son cœur, qui culmine une série historique de DE adaptative solveurs. Cela ne pose aucun doute sur l'apport scientifique de cette étude, qui peut stimuler de nouvelles directions de recherche vers de nouvelles méthodes de recherche locales hybridées avec SHADE.
- Une attention particulière doit être portée à la simplicité de SHADE-ILS.

Dans cet algorithme, le modèle n'est pas très complexe et le nombre de paramètres est plus simple que dans d'autres propositions (car ses composants nécessitent peu de paramètres). Par ailleurs, les modifications apportées par rapport à IHDELS n'augmentent pas son nombre de paramètres.

7.2.7. Résumé du cas d'utilisation

Pour conclure, le cas d'utilisation décrit dans cette section suit la plupart des directives de notre méthodologie proposée. Comme dans le cas d'utilisation précédent, les principales procédures suivies sont mises en évidence dans la Fig. 7. De plus, nous décrivons brièvement maintenant les principales actions prises pour chacune des lignes directrices proposées :

- Ligne directrice n°1 : nous avons eu recours au référentiel standard CEC'2013, largement accepté par la communauté travaillant sur l'optimisation globale à grande échelle. De plus, nous avons vérifié que le benchmark respecte plusieurs des exigences imposées par les lignes directrices. Enfin, nous avons comparé notre méthode proposée à des techniques de pointe similaires et à une référence de référence sur le terrain. • Ligne directrice #2 : comme cela a été démontré tout au long de la discussion, la validation des résultats a été effectuée selon les bonnes pratiques prescrites à la section 4, y compris les tests d'hypothèses non paramétriques.

En outre, nous avons également montré comment plusieurs résultats peuvent être correctement visualisés pour rendre le résultat des comparaisons plus compréhensible pour le public. • Ligne directrice #3 : nous avons clairement mis en évidence les

objectifs des algorithmes, et nous avons comparé l'influence des différents éléments nouveaux de l'algorithme proposé. Ainsi, nous avons montré que les bons résultats ne sont pas influencés par un seul composant, mais par la synergie entre les différents éléments. Nous avons aussi montré que

la proposition n'est pas inutilement complexe. Enfin, un processus de réglage a également été appliqué. • Guideline #4 : dans ce cas d'utilisation, la proposition de notre méthode est facile à justifier. SHADE-ILS améliore non seulement une hybridation précédente de DE avec recherche locale [118], mais surpasse également MOS-CEC2013 et MLSHADE-SPA (bien que sans signification statistique), qui ont dominé la concurrence au cours des dernières années. Cela ne pose aucun doute sur l'apport scientifique de cette étude, qui peut stimuler de nouvelles directions de recherche vers de nouvelles méthodes de recherche locales hybridées avec SHADE. De plus, le précédent algorithme de pointe, MOS, a été clairement surpris par SHADE-ILS et MLSHADE-SPA, devenant ainsi les algorithmes les plus compétitifs (avec une préférence pour SHADE-ILS, par ses meilleures performances et sa simplicité).

8. Conclusions et perspectives

Dans ce travail, nous avons insisté sur la nécessité de contourner les erreurs et failles courantes observées dans le domaine de l'optimisation bio-inspirée, notamment lorsque de nouveaux algorithmes méta-heuristiques sont proposés et validés expérimentalement sur des benchmarks conçus à cet effet. Plus précisément, nous avons passé en revue et analysé de manière critique les contributions traitant des recommandations expérimentales et des pratiques liées aux métaheuristiques. Suite à notre étude de la littérature, nous avons prescrit un ensemble de recommandations méthodologiques pour préparer une proposition réussie d'algorithmes méta-heuristiques bio-inspirés, de la définition de l'expérimentation à la présentation des résultats. Un certain nombre de techniques utiles (graphiquement résumées dans la Fig. 9) ont été suggérées pour des études prospectives afin de mettre en œuvre notre cadre méthodologique proposé, dans le but d'assurer l'équité, la cohérence et la solidité des études futures sur le sujet. Deux études de cas différentes ont été conçues pour illustrer l'application de notre méthodologie prescrite, en discutant des résultats de l'application de chaque ligne directrice. Bien que les deux études de cas traitent de benchmarks bien connus, nous envisageons que notre méthodologie puisse être un élément central du processus de conception d'algorithmes méta-heuristiques pour des problèmes d'optimisation du monde réel, en suivant les lignes directrices de notre tutoriel récemment publié à ce sujet [121]. Dans ces cas, la validation statistique des résultats ne doit pas être considérée comme l'étape finale de l'analyse : la signification des résultats doit être analysée sous différents angles et en tenant compte d'autres mesures d'intérêt pratique (par exemple, la consommation de mémoire).

Dans un domaine aussi dynamique, avec de nouvelles propositions algorithmiques qui fleurissent vigoureusement, des bases méthodologiques communes sont nécessaires de toute urgence. Les avancées scientifiques dans les années à venir ne seront réalisées que si la communauté parvient à un accord sur la manière dont les algorithmes doivent être testés et comparés les uns aux autres. C'est bien le but de notre travail : rassembler et regrouper les pratiques recommandées autour d'un ensemble unifié de

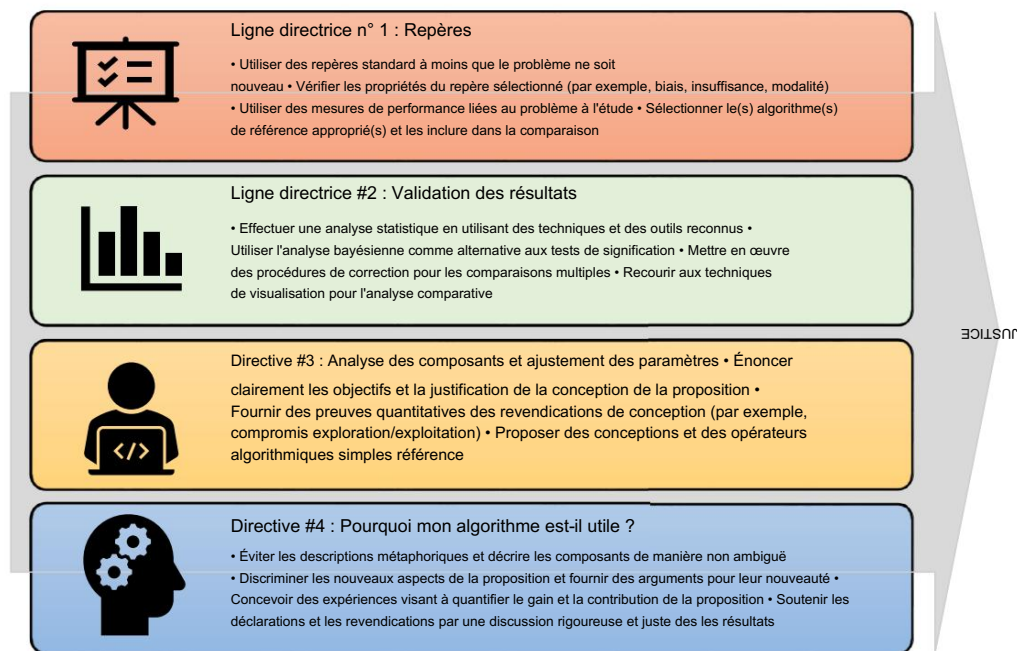


Fig. 9. Lignes directrices composant le cadre méthodologique de comparaison des méta-heuristiques proposées dans ce travail.

directives méthodologiques. Nous espérons sincèrement que le matériel et les prescriptions donnés ici guideront les nouveaux arrivants dans leur arrivée dans cette avenue de recherche passionnante.

Déclaration d'intérêts concurrents

Les auteurs déclarent qu'ils n'ont pas d'intérêts financiers concurrents ou de relations personnelles connus qui auraient pu sembler influencer le travail rapporté dans cet article.

Déclaration de contribution de la paternité du CRediT

Antonio LaTorre : Conceptualisation, Méthodologie, Enquête, Logiciel, Rédaction – ébauche originale, Rédaction – révision et édition.
Daniel Molina : Conceptualisation, Méthodologie, Enquête, Logiciels, Rédaction – brouillon original, Rédaction – révision et édition. Eneko Osaba : Méthodologie, Enquête, Rédaction – projet original. Javier Poyatos : Méthodologie, Recherche, Logiciel, Validation. Javier Del Ser : Conceptualisation, Méthodologie, Enquête, Validation, Supervision, Rédaction – projet original, Rédaction – révision et édition, Acquisition de financement. Francisco Herrera : Conceptualisation, Supervision, Rédaction – révision et édition, Acquisition de financement.

Remerciements

Ce travail a été soutenu par des subventions du ministère espagnol des sciences (TIN2016-8113-R, TIN2017-89517-P et TIN2017-83132-C2-2-R) et de l'Université Politécnica de Madrid (PINV-18-XEOGHQ-19- 4QTEBP). Eneko Osaba et Javier Del Ser tiennent également à remercier le gouvernement basque pour son soutien financier à travers les programmes ELKARTEK et EMAITEK. Javier Del Ser reçoit le soutien financier du Groupe de Recherche Consolidé MATHMODE (IT1294-19) accordé par le Département de l'Éducation du Gouvernement Basque.

Les références

- [1] J. Del Ser, E. Osaba, D. Molina, X.-S. Yang, S. Salcedo-Sanz, D. Camacho, S. Das, PN Suganthan, CAC Coello, F. Herrera, Calcul bio-inspiré : où en sommes-nous et quelle est la suite, *Swarm Evol Comput* 48 (2019) 220–250, doi : 10.1016/j.swevo.2019.04.008.
- [2] D. Molina, J. Poyatos, J. Del Ser, S. García, F. Herrera, Taxonomies complètes de l'optimisation inspirée de la nature et bio : inspiration versus comportement algorithmique, analyse critique et recommandations, *Cognit Comput* 12 (2020) 897–939, doi : 10.1007/s12559-020-09730-8.
- [3] JC Stanley, L'influence de la « conception des expériences » de Fisher sur la recherche pédagogique trente ans plus tard, *Am Educ Res J* 3 (3) (1966) 223–229, doi : 10.3102/00028312003003223 .
- [4] J. Derrac, S. García, D. Molina, F. Herrera, Un didacticiel pratique sur l'utilisation de tests statistiques non paramétriques comme méthodologie de comparaison des algorithmes d'évolution et d'intelligence en essaim, *Swarm Evol Comput* 1 (1) (2011) 3–18, doi : 10.1016/j.swevo.2011.02.002.
- [5] DS Johnson, A theoretician's guide to the experimental analysis of algorithms, dans : MH Goldwasser, DS Johnson, CC McGeech (Eds.), *Data structures, near neighbor searches, and methodology: Fifth and sixième DIMACS implementation challenges*, American Society mathématique, 2002, p. 215–250.
- [6] M. Hellwig, H.-G. Beyer, Algorithmes évolutifs d'analyse comparative pour l'optimisation contrainte à valeur réelle à objectif unique – une revue critique, *Swarm Evol Comput* 44 (2019) 927–944, doi : 10.1016/j.swevo.2018.10.002.
- [7] T. Weise, R. Chiong, K. Tang, Optimisation évolutive : pièges et pièges, *J Comput Sci Technol* 27 (5) (2012) 907–936, doi : 10.1007/s11390-012-1274-4.
- [8] AV Kononova, DW Corne, PD Wilde, V. Shneer, F. Caraffini, Biais structurel dans les algorithmes basés sur la population, *Inf Sci(Ny)* 298 (2015) 468–490.2014.11.035 .
- [9] R. Storn, KV Price, Évolution différentielle – une heuristique simple et efficace pour l'optimisation globale sur des espaces continus, *J. Global Optim.* 11 (4) (1997) 341–359, doi : 10.1023/A : 1008202821328.
- [10] Z. Hu, Q. Su, X. Yang, Z. Xiong, Ne pas garantir la convergence de l'évolution différentielle sur une classe de fonctions multimodales, *Appl Soft Comput* 41 (2016) 479–487, doi : 10.1016/j.asoc.2016.01.001.
- [11] AP Piotrowski, JJ Napiorkowski, Recherche de biais structurels dans l'optimisation des essais de particules et les algorithmes d'évolution différentielle, *Swarm Intell.* 10 (4) (2016) 307–353, doi : 10.1007/s11721-016-0129-y.
- [12] F. Caraffini, AV Kononova, D. Corne, Infaisabilité et biais structurel dans l'évolution différentielle, *Inf Sci (Ny)* 496 (2019) 161–179, doi:10.1016/j.ins.2019.05.019.
- [13] KV Price, Comment la symétrie contraint les optimiseurs évolutifs, dans : 2017 IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 1712–1719. DOI : 10.1109/CEC.2017.7969508
- [14] N. Hansen, A. Ostermeier, Auto-adaptation complètement dérandomisée dans les stratégies d'évolution, *Evol Comput* 9 (2) (2001) 159–195, doi : 10.1162/106365601750190398.
- [15] PN Suganthan, N. Hansen, JJ Liang, K. Deb, Y.-P. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, Technical Report, Nanyang Technological University, 2005. <http://www.ntu.edu.sg/home/EPNSugan/>
- [16] S. Das, PN Suganthan, Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems, Technical Report, Jadavpur University, Inde et Nanyang Technological University, Singapour, 2010.
- [17] JJ Liang, B.-Y. Qu, PN Suganthan, A. Hernández-Díaz, Définitions des problèmes et critères d'évaluation pour la session spéciale et le concours CEC 2013 sur l'optimisation des paramètres réels, Rapport technique, Laboratoire d'intelligence computationnelle,

- Université de Zhengzhou, Chine et Université technologique de Nanyang, Singapour, 2013.
- [18] JJ Liang, B.-Y. Qu, PN Suganthan, Définitions des problèmes et critères d'évaluation pour la session spéciale et le concours CEC 2014 sur l'optimisation numérique à paramètre réel à objectif unique, rapport technique, Laboratoire d'intelligence informatique, Université de Zhengzhou, Chine et Université technologique de Nanyang, Singapour, 2013.
- [19] NH Award, MZ Ali, PN Suganthan, JJ Liang, BY Qu, Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-parameter Numerical Optimization, Technical Report, Nanyang Technological University, Singapore, 2016.
- [20] D. Molina, A. LaTorre, F. Herrera, Un aperçu des algorithmes bio-inspirés et évolutifs pour l'optimisation globale : examen, analyse et leçons apprises sur une décennie de compétitions, Cognit Comput 10 (2018) 517–544, doi : 10.1007/s12559-018-9554-0.
- [21] G. Wu, W. Pedrycz, PN Suganthan, H. Li, Utiliser une stratégie de réduction variable pour accélérer l'optimisation évolutive, Appl Soft Comput 61 (2017) 283–293.
- [22] K. Tang, X. Yao, PN Suganthan, C. MacNish, YP Chen, CM Chen, Z. Yang, Benchmark Functions for the CEC 2008 Special Session and Competition on Large Scale Global Optimization, Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, Chine, 2007.
- [23] X. Li, K. Tang, MN Omidvar, Z. Yang, K. Quin, Benchmark Functions for the CEC 2013 Special Session and Competition on Large Scale Global Optimization, Rapport technique, Evolutionary Computation and Machine Learning Group, 2013.
- [24] R. Bellman, Programmation dynamique, Science 153 (3731) (1966) 34–37.
- [25] F. van den Bergh, AP Engelbrecht, Une approche coopérative de l'optimisation des essais de particules, IEEE Trans. Évol. Calcul. 8 (3) (2004) 225–239, doi : 10.1109/tevc.2004.826069.
- [26] F. Neri, V. Tirronen, Progrès récents dans l'évolution différentielle : une enquête et une analyse expérimentale, Artif Intell Rev 33 (1–2) (2009) 61–106, doi : 10.1007/s10462-009-9137-2.
- [27] K. Tang, X. Li, PN Suganthan, Z. Yang, T. Weise, Benchmark Functions for the CEC 2010 Special Session and Competition on Large Scale Global Optimization, Rapport technique, Nature Inspired Computation and Applications Laboratory, USTC, 2010.
- [28] P. Oliveto, T. Paixão, J. Pérez Heredia, D. Sudholt, B. Trubenová, How to escape local optima in black box optimisation: when non-elitism outperforms elitism, Al gorithmica 80 (5) (2018) 1604–1633, doi : 10.1007/s00453-017-0369-2.
- [29] L. Hernando, A. Mendiburu, J. Lozano, Une évaluation des méthodes d'estimation du nombre d'optimums locaux dans les problèmes d'optimisation combinatoire, Evol Comput 21 (4) (2013) 625–658, doi : 10.1162/EVCO_a_00100.
- [30] AM Sutton, M. Lunacek, LD Whitley, Differential evolution and non-separable ity: Using selective pressure to focus search, in: Actes de la 9e conférence annuelle sur le calcul génétique et évolutif, in: GECCO '07, Association for Computing Machinery, New York, NY, États-Unis, 2007, pp. 1428–1435. DOI : 10.1145/1276958.1277221
- [31] S. Bagheri, W. Konen, T. Bäck, Résoudre les problèmes d'optimisation avec un conditionnement élevé au moyen du blanchiment en ligne, in: GECCO 2019 Companion - Actes de la 2019 Genetic and Evolutionary Computation Conference Companion, 2019, pp. 243–244. DOI : 10.1145/3319619.3322008
- [32] S. Finck, N. Hansen, R. Ros, A. Auger, Real-Parameter Black-Box Optimization Benchmarking 2010 : Presentation of the Noisy Functions, Technical Report, Research Center PPE, 2010.
- [33] J. Rapin, O. Teytaud, Nevergrad - Une plateforme d'optimisation sans gradient, 2018, (<https://GitHub.com/FacebookResearch/Nevergrad>).
- [34] H.-G. Beyer, Algorithmes évolutifs dans des environnements bruyants : problèmes théoriques et lignes directrices pour la pratique, Comput Methods Appl Mech Eng 186 (2–4) (2000) 239–267, doi : 10.1016/s0045-7825(99)00386-2.
- [35] Y. Jin, J. Branke, Optimisation évolutive dans des environnements incertains - une enquête, IEEE Trans. Évol. Calcul. 9 (3) (2005) 303–317, doi : 10.1109/TEVC.2005.846356.
- [36] C. García-Martínez, PD Gutiérrez, D. Molina, M. Lozano, F. Herrera, Depuis le concours CEC 2005 sur l'optimisation des paramètres réels : une décennie de recherche, de progrès et de faiblesse de l'analyse comparative, Soft comput 21 (19) (2017) 5573–5583, doi : 10.1007/s00500-016-2471-9.
- [37] P. Civicioglu, E. Besdok, Une comparaison conceptuelle de la recherche de coucous, de l'optimisation des essais de particules, de l'évolution différentielle et des algorithmes de colonies d'abeilles artificielles, Artif Intell Rev 39 (4) (2011) 315–346, doi : 10.1007 / s10462-011-9276-0.
- [38] V. Osuna-Enciso, E. Cuevas, H. Sossa, Une comparaison d'algorithmes inspirés de la nature pour la segmentation d'images à plusieurs seuils, Expert Syst Appl 40 (4) (2013) 1213–1219, doi:10.1016/j.eswa.2012.08.017.
- [39] J. Demšar, Comparaisons statistiques de classificateurs sur plusieurs ensembles de données, Journal of Recherche sur l'apprentissage automatique 7 (2006) 1–30.
- [40] JM Whitacre, Utilisation de la méthode statistique de détection des valeurs aberrantes dans les algorithmes évolutifs adaptatifs, dans : 2006 Conference on Genetic and Evolutionary Computation (GECCO '06), ACM Press, 2006, pp. 1345–1352. DOI : 10.1145/1143997.1144205 [41] S. García, D. Molina, M. Lozano, F. Herrera, A study on the use of non-parametric tests for analysis the evolutionary algorithms' behavior: a case study on the CEC'2005 special session on real parameter optimisation, Journal of Heuristics 15 (6) (2008) 617–644, doi : 10.1007/s10732-008-9080-4.
- [42] S. García, A. Fernández, J. Luengo, F. Herrera, Tests non paramétriques avancés pour des comparaisons multiples dans la conception d'expériences en intelligence computationnelle et en exploration de données : analyse expérimentale de puissance, Inf Sci (Ny) 180 (10) (2010) 2044–2064, doi : 10.1016/j.ins.2009.12.010.
- [43] CC McGeech, Experimental Analysis of Algorithms, dans : PM Pardalos, HE Romeijn (Eds.), Handbook of Global Optimization : Volume 2, Springer US, Boston, MA, 2002, pp. 489–513. DOI : 10.1007/978-1-4757-5362-2_14
- [44] AE Eiben, M. Jelasity, Une note critique sur la méthodologie de recherche expérimentale en CE, dans : Actes du Congrès 2002 sur le calcul évolutif. CEC'02 (Cat. No.02TH8600), volume 1, 2002, pp. 582–587 vol.1, doi:10.1109/CEC.2002.1006991.
- [45] AH Halim, I. Ismail, S. Das, Évaluation des performances des algorithmes d'optimisation métaheuristique : une revue exhaustive, Artif Intell Rev (2020), doi : 10.1007/s10462-020-09906-6.
- [46] A. Eiben, R. Hinterding, Z. Michalewicz, Parameter control in evolutionary algorithms, IEEE Trans. Evol. des ordinateurs 3 (2) (1999) 124–141, doi : 10.1109/4235.771166.
- [47] J. Grefenstette, Optimisation des paramètres de contrôle pour les algorithmes génétiques, IEEE Trans Syst Man Cybern 16 (1) (1986) 122–128, doi: 10.1109/tsmc.1986.289288.
- [48] IC Trelea, Algorithme d'optimisation de l'essai de particules : analyse de la convergence et sélection des paramètres, Inf Process Lett 85 (6) (2003) 317–325, doi : 10.1016/s0020-0190(02)00447-7.
- [49] AK Qin, VL Huang, PN Suganthan, Algorithme d'évolution différentielle avec adaptation de stratégie pour l'optimisation numérique globale, IEEE Trans. Évol. Calcul. 13 (2) (2009) 398–417, doi : 10.1109/tevc.2008.927706.
- [50] E. Montero, M.-C. Riff, B. Neveu, Guide du débutant sur les méthodes de réglage, Appl Soft Comput 17 (2014) 39–51, doi : 10.1016/j.asoc.2013.12.017.
- [51] P. Balaprakash, M. Birattari, T. Stützle, Stratégies d'amélioration de l'algorithme F-Race : Conception d'échantillonnage et raffinement itératif, vol. 4771 de Hybrid Metaheuristics, Springer Berlin Heidelberg, 2007, pp. 108–122. DOI : 10.1007/978-3-540-75514-2_9 [52] M. López-Ibáñez, J. Dubois-Lacoste, LP Cáceres, M. Birattari, T. Stützle, The irace package: iterated racing for automatic algorithm configuration, Oper. Rés. Par aspect. 3 (2016) 43–58, doi : 10.1016/j.orp.2016.09.002.
- [53] V. Nannen, AE Eiben, Relevance estimation and value calibration of evolutionary algorithm parameters, dans : MM Veloso (Ed.), 20th International Joint Conference on Artificial Intelligence (IJCAI), 2007, pp. 975–980.
- [54] F. Hutter, HH Hoos, K. Leyton-Brown, T. Stützle, Paramils : un cadre de configuration d'algorithme automatique, Journal of Artificial Intelligence Research 36 (2009) 267–306, doi : 10.1613/jair.2861.
- [55] Hansen N, Brockhoff D, Mersmann O, Tusar T, Tusar D, ElHara OA, Sampaio PR, Atamna A, Varelas K, Batu U, Nguyen DM, Matzner F, A. Auger, Comparer les optimiseurs continus : Numbbo/COCO sur Github,
- [56] M. Helbig, AP Engelbrecht, Mesures de performance pour les algorithmes d'optimisation multi-objectifs dynamiques, Inf Sci (Ny) 250 (2013) 61–81, doi : 10.1016/j.ins.2013.06.051.
- [57] S. Mirjalili, A. Lewis, Nouvelles mesures de performance pour des algorithmes d'optimisation multi-objectifs robustes, Swarm Evol Comput 21 (2015) 1–23, doi : 10.1016/j.swevo.2014.10.005.
- [58] X.-S. Yang, Algorithmes Firefly pour l'optimisation multimodale, dans : Stochastic Algorithms : Foundations and Applications, 2009, pp. 169–178.
- [59] S. Saremi, S. Mirjalili, A. Lewis, Algorithme d'optimisation Grasshopper : théorie et application, Adv. Ing. Logiciel 105 (2017) 30–47, doi : 10.1016/j.advengsoft.2017.01.004.
- [60] W. Gong, Z. Cai, Évolution différentielle avec les opérateurs de mutation basés sur le classement, IEEE Trans Cybern 43 (6) (2013) 2066–2081, doi:10.1109/tycy.2013.2239988.
- [61] SS Shapiro, MB Wilk, An analysis of variance test for normality (complete samples), Biometrika 52 (3–4) (1965) 591–611, doi:10.1093/biomet/52.3-4.591.
- [62] NM Razali, Comparaisons de puissance des tests de shapiro-wilk, kolmogorov-smirnov, lilliefors et anderson-darling, Journal of Statistical Modeling and Analytics 2 (1) (2011) 21–33.
- [63] H. Levene, Robust Tests for Equality of Variances, dans : I. Olkin, SG Ghurye, W. Ho edding, WG Madow, HB Mann (Eds.), Contributions to Probability and Statistics : Essays in Honour of Harold Hotelling, Stanford University Press, Stanford, Californie, 1960, pp. 278–292.
- [64] DJ Sheskin, Manuel des procédures statistiques paramétriques et non paramétriques, 4th, Chapman & Hall/CRC, 2007.
- [65] BL Welch, La généralisation du problème des « étudiants » lorsque plusieurs variances de population différentes sont impliquées, Biometrika 34 (1–2) (1947) 28–35, doi : 10.1093/biomet/34.1-2.28.
- [66] F. Wilcoxon, Comparaisons individuelles par méthodes de classement, Biometrics Bulletin 1 (6) (1945) 80–83, doi:10.2307/3001968.
- [67] WW Daniel, Statistiques non paramétriques appliquées, The Duxbury advanced series in statistics and decision sciences, Duxbury Thomson Learning, 1990.
- [68] M. Aikin, H. Gensler, Ajustement pour les tests multiples lors de la communication des résultats de recherche : les méthodes bonferroni vs holm., Am J Public Health 86 (5) (1996) 726–728, doi : 10.2105/AJPH.86.5.726.
- [69] JO Dunn, Comparaisons multiples entre les moyennes, J Am Stat Assoc 56 (293) (1961) 52–64, doi : 10.1080/01621459.1961.10482090.
- [70] S. Holm, Une simple procédure de test multiple à rejet séquentiel, Scand. J. Stat. 6 (1979) 65–70.
- [71] Y. Benjamini, Y. Hochberg, Contrôler le taux de fausses découvertes : une approche pratique et puissante des tests multiples, Journal de la Royal Statistical Society. Série B (méthodologique) 57 (1) (1995) 289–300.
- [72] G. Hommel, Une procédure de test multiple de rejet par étapes basée sur un test bonferroni modifié, Biometrika 75 (2) (1988) 383–386, doi:10.1093/biomet/75.2.383.
- [73] M. Lin, HC Lucas, G. Shmueli, Research commentary - too big to fail: large samples and the p-value problem, Information Systems Research 24 (4) (2013) 906–917, doi:10.1287/isre.2013.0480.
- [74] A. Benavoli, G. Corani, J. Demšar, M. Zaffalon, Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis, Journal of Machine Learning Research 18 (1) (2017) 2653–2688.

- [75] J. Carrasco, S. García, M. del Mar Rueda, F. Herrera, rNPBST : An R package covering non-parametric and bayesian statistics tests, in : *International Conference on Hybrid Artificial Intelligence Systems*, Springer, 2017, p. 281–292.
- [76] A. Benítez-Hidalgo, AJ Nebro, J. García-Nieto, I. Oregi, J. Del Ser, Jmetapy: A Python framework for multi-objective optimization with metaheuristics, *Swarm Evol Comput* 51 (2019) 100598, doi : 10.1016/j.swevo.2019.100598.
- [77] J. Carrasco, S. García, M. del Mar Rueda, S. Das, F. Herrera, Tendances récentes dans l'utilisation de tests statistiques pour comparer les algorithmes de calcul en essaim et évolutifs : directives pratiques et revue critique, *Swarm Evol Comput* 54 (2020) 100665, doi:10.1016/j.swevo.2020.100665.
- [78] A. Herrera-Poyatos, F. Herrera, Algorithme génétique et mémétique avec équilibre de diversité basé sur la diversification gourmande, *CoRR abs/1702.03594* (2017).
- [79] A. LaTorre, S. Muelas, JM Peña, Un algorithme d'évolution différentielle mémétique dynamique basé sur MOS pour l'optimisation continue : un test d'évolutivité, *Soft Computing - A Fusion of Foundations, Methodologies and Applications* 15 (11) (2010) 2187–2199, doi : 10.1007/s00500-010-0646-3.
- [80] M. Črepinšek, SH Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: a survey, *ACM Comput Surv* 45 (3) (2013) 1–33, doi:10.1145/2480741.2480752.
- [81] MG Epitropakis, VP Plagianakos, MN Vrahatis, Équilibrer les capacités d'exploration et d'exploitation de l'algorithme d'évolution différentielle, dans : *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, 2008, pp. 2686–2693. ISSN : 1941-0026, DOI : 10.1109/CEC.2008.4631159
- [82] E.-u. Haq, I. Ahmad, A. Hussain, IM Almanjahie, Une nouvelle approche de sélection pour les algorithmes génétiques pour l'optimisation globale des fonctions continues multimodales, *Comput Intell Neurosci* 2019 (2019), doi : 10.1155/2019/8640218. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6915132/> [83] A. Hussain, YS Muhammad, Compromis entre exploration et exploitation avec un algorithme génétique utilisant un nouvel opérateur de sélection, *Complex & Intelligent Systems* 6 (1) (2020) 1–14, doi : 10.1007/s40747-019-0102-7.
- [84] F. Vafae, G. Turán, PC Nelson, TY Berger-Wolf, Équilibrer l'exploration et l'exploitation dans un algorithme génétique guidé par la diversité adaptative, dans : *2014 IEEE Congress on Evolutionary Computation (CEC)*, 2014, pp. 2570–2577. ISSN : 1941-0026, DOI : 10.1109/CEC.2014.6900257
- [85] A. LaTorre, S. Muelas, JM Peña, Une comparaison complète des optimiseurs globaux à grande échelle, *Inf Sci (Ny)* 316 (2014) 517–549, doi:10.1016/j.ins.2014.09.031.
- [86] AP Piotrowski, JJ Napiorkowski, Certaines métaheuristiques devraient être simplifiées, *Inf Sci (Ny)* 427 (2018) 32–62, doi:10.1016/j.ins.2017.10.039.
- [87] NH Awad, MZ Ali, PN Suganthan, RG Reynolds, Une adaptation de paramètres sinusoïdaux d'ensemble incorporée avec L-SHADE pour résoudre les problèmes de référence CEC 2014, dans : *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 2958–2965. DOI : 10.1109/CEC.2016.7744163 [88] LY Tseng, C. Chen, Recherche de trajectoires multiples pour l'optimisation globale à grande échelle, dans : *2008 IEEE Congress on Evolutionary Computation (CEC)*, IEEE Press, 2008, pp. 3052–3059. DOI : 10.1109/CEC.2008.4631210 [89] M. Lozano, D. Molina, F. Herrera, Editorial : évolutivité des algorithmes évolutionnaires et autres métaheuristiques pour les problèmes d'optimisation continue à grande échelle, *Soft Computing - A Fusion of Foundations, Methodologies and Applications* 15 (2011) 2085–2087, doi:10.1007/s00500-010-0639-2.
- [90] R. Tanabe, AS Fukunaga, Améliorer les performances de recherche de SHADE en utilisant la réduction de la taille de la population d'oreille linéaire, dans : *2014 IEEE Congress on Evolutionary Computation (CEC)*, 2014, pp. 1658–1665. DOI : 10.1109/CEC.2014.6900380
- [91] K. De Jong, Parameter Setting in EAs: A 30 Year Perspective, in: FG Lobo, CF Lima, Z. Michalewicz (Eds.), *Parameter Setting in Evolutionary Algorithms*, Springer, Berlin, Heidelberg, 2007, pp. 1–18. DOI : 10.1007/978-3-540-69432-8_1 [92] G. Eiben, MC Schut, New Ways to Calibrate Evolutionary Algorithms, dans : P. Siarry, Z. Michalewicz (Eds.), *Advances in Metaheuristics for Hard Optimization*, Springer, Berlin, Heidelberg, 2008, pp. 153–177. DOI : 10.1007/978-3-540-72960-0_8 [93] T. Lundstedt, E. Seifert, L. Abramo, B. Thelin, Å. Nyström, J. Pettersen, R. Bergman, Conception expérimentale et optimisation, *Chimométrie et systèmes de laboratoire intelligents* 42 (1–2) (1998) 3–40, doi : 10.1016/s0169-7439(98)00065-3.
- [94] G. Taguchi, Introduction à l'ingénierie de la qualité : Concevoir la qualité dans les produits et processus, *Organisation asiatique de productivité*, 1986.
- [95] S. Wessing, N. Beume, G. Rudolph, B. Naujoks, Parameter Tuning Boosts Performance of Variation Operators in Multiobjective Optimization, dans : R. Schaefer, C. Cotta, J. Kolodziej, G. Rudolph (Eds.), *Résolution de problèmes parallèles à partir de la nature, PPSN XI*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 728–737. DOI : 10.1007/978-3-642-15844-5_73 [96] T. Liao, D. Molina, T. Stützle, Évaluation des performances des optimiseurs continus automatiquement réglés sur différents ensembles de référence, *Appl Soft Comput* 27 (2015) 490–503, doi : 10.1016/j.asoc.2014.11.006.
- [97] S. García, J. Derrac, S. Ramírez-Gallego, F. Herrera, Sur l'analyse statistique de la tendance des paramètres dans un algorithme d'apprentissage automatique, *Progress in Artificial Intelligence* 3 (1) (2014) 51–53, doi : 10.1007/s13748-014-0043-8.
- [98] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Paramètres de contrôle auto-adaptatifs en évolution différentielle : une étude comparative sur les problèmes de référence numérique, *IEEE Trans. Évol. Calcul.* 10 (6) (2006) 646–657, doi : 10.1109/tevc.2006.872133.
- [99] J. Teo, Analyse d'évolutivité de l'évolution différentielle fixe versus auto-adaptative pour une optimisation globale sans contrainte, *Adv Sci Lett* 23 (6) (2017) 5144–5146, doi : 10.1166/asl.2017.7328.
- [100] J. Zhang, AC Sanderson, JADE : Évolution différentielle adaptative avec archive externe optionnelle, *IEEE Trans. Évol. Calcul.* 13 (5) (2009) 945–958, doi : 10.1109/TEVC.2009.2014613.
- [101] R. Tanabe, AS Fukunaga, Evaluating the performance of shade on CEC 2013 benchmark problems, dans : *2013 IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 1952–1959. DOI : 10.1109/CEC.2013.6557798 [102] J. Brest, MS Maucec, B. Boskovic, Optimisation des paramètres réels à objectif unique : Algorithme jso, in : *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 1311–1318. DOI : 10.1109/cec.2017.7969456 [103] T. Bartz-Beielstein, CWG Lasarczyk, M. Preuss, Optimisation des paramètres séquentiels, dans : *2005 IEEE Congress on Evolutionary Computation*, volume 1, 2005, pp. 773–780 Vol.1. DOI : 10.1109/CEC.2005.1554761
- [104] K. Sörensen, Métaheuristiques — la métaphore exposée, *Transactions internationales dans Operational Research* 22 (1) (2015) 3–18, doi:10.1111/itor.12001.
- [105] J. Swan, S. Adriaensen, M. Bishr, EK Burke, JA Clark, P. De Causmaecker, J. Durillo, K. Hammond, E. Hart, CG Johnson, et al., A research agenda for meta standardisation heuristic, dans : *Actes de la XI conférence internationale sur les métaheuristiques*, 2015, pp. 1–3.
- [106] Swan J, Adriaensen S, Brownlee AE, Johnson CG, Kheiri A, Krawiec F, Merelo J, Minku LL, Özcan E, Pappa GL, et al., arXiv preprint arXiv:2011.09821 (2020).
- [107] MA Lones, Mitigating metaphors: a comprehensible guide to recent nature-inspired algorithms, *SN Computer Science* 1 (1) (2020) 49.
- [108] M. Jain, V. Singh, A. Rani, Un nouvel algorithme d'optimisation inspiré de la nature : algorithme de recherche d'écureuils, *Swarm Evol Comput* 44 (2019) 148–175, doi : 10.1016/j.swevo.2018.02.013.
- [109] AW Mohamed, AA Hadi, AK Mohamed, Algorithme basé sur le partage des connaissances pour résoudre les problèmes d'optimisation : un nouvel algorithme inspiré de la nature, *Int. J. Mach. Apprendre. Cybern.* 11 (7) (2020) 1501–1529, doi : 10.1007/s13042-019-01053-x.
- [110] W. Zhao, L. Wang, Z. Zhang, Optimisation basée sur l'écosystème artificiel : un nouvel algorithme méta-heuristique inspiré de la nature, *Neural Computing and Applications* 32 (13) (2020) 9383–9425, doi : 10.1007/s00521-019-04452-x.
- [111] AW Mohamed, AK Mohamed, Algorithme d'évolution différentielle guidée adaptative avec nouvelle mutation pour l'optimisation numérique, *Int. J. Mach. Apprendre. Cybern.* 10 (2) (2019) 253–277, doi : 10.1007/s13042-017-0711-7.
- [112] AW Mohamed, AA Hadi, AM Fattouh, KM Jambi, LSHADE with semi-paramètre adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems, dans : *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 145–152. DOI : 10.1109/CEC.2017.7969307
- [113] AA Hadi, AW Mohamed, KM Jambi, Single-Objective Real-Parameter Optimization: Enhanced LSHADE-SPACMA Algorithm, in : F. Yalaoui, L. Amodeo, E.-G. Talbi (Eds.), *Heuristics for Optimization and Learning*, volume 906, Springer International Publishing, Cham, 2021, pp. 103–121. DOI : 10.1007/978-3-030-58930-1_7 [114] J. Brest, MS Maucec, B. Bošković, Optimisation des paramètres réels à objectif unique : Algorithme jso, dans : *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 1311–1318. DOI : 10.1109/CEC.2017.7969456 [115] A. Kumar, RK Misra, D. Singh, Améliorer la capacité de recherche locale d'un optimiseur de papillon efficace à l'aide d'une phase de retraite adaptée à la matrice de covariance, dans : *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 1835–1842. DOI : 10.1109/CEC.2017.7969524
- [116] D. Molina, A. LaTorre, Boîte à outils pour la comparaison automatique des optimiseurs : comparer les optimiseurs globaux à grande échelle en toute simplicité, in : *2018 IEEE Congress on Evolutionary Computation (CEC)*, 2018, pp. 1–8. DOI : 10.1109/CEC.2018.8477924 [117] D. Molina, A. LaTorre, F. Herrera, SHADE avec recherche locale itérative pour une optimisation globale à grande échelle, dans : *2018 IEEE Congress on Evolutionary Computation (CEC)*, Rio de Janeiro, Brésil, 2018, pp. 1–8. DOI : 10.1109/CEC.2018.8477755 [118] D. Molina, F. Herrera, Hybridation itérative de DE avec recherche locale pour la session spéciale CEC'2015 sur l'optimisation globale à grande échelle, dans : *2015 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2015, pp. 1974–1978. DOI : 10.1109/CEC.2015.7257127 [119] A. LaTorre, S. Muelas, JM Peña, Optimisation globale à grande échelle : résultats expérimentaux avec des algorithmes hybrides basés sur MOS, dans : *2013 IEEE Congress on Evolutionary Computation (CEC)*, IEEE Press, 2013, p. 2742–2749. DOI : 10.1109/CEC.2013.6557901 [120] AA Hadi, AW Mohamed, KM Jambi, Cadre mémétique LSHADE-SPA pour la résolution de problèmes d'optimisation à grande échelle, *Complex & Intelligent Systems* 5 (1) (2019) 25–40, doi : 10.1007/s40747-018-0086-8.
- [121] E. Osaba, E. Villar-Rodriguez, J. Del Ser, AJ Nebro, D. Molina, A. LaTorre, PN Suganthan, C. Coello, F. Herrera, A tutorial on the design, experimental and application of algorithms metaheuristicas aux problèmes d'optimisation du monde réel, *Swarm Evol Comput* (2021) 100888, doi:10.1016/j.swevo.2021.100888.