Titre: Title:	Opportunisme et ordonnancement en optimisation sans dérivées
Auteur: Author:	Loïc Anthony Sarrazin-Mc Cann
Date:	2018
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Sarrazin-Mc Cann, L. A. (2018). Opportunisme et ordonnancement en optimisation sans dérivées [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/3099/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/3099/
Directeurs de recherche: Advisors:	Charles Audet
Programme: Program:	Mathématiques appliquées

UNIVERSITÉ DE MONTRÉAL

OPPORTUNISME ET ORDONNANCEMENT EN OPTIMISATION SANS-DÉRIVÉES

LOÏC ANTHONY SARRAZIN-MC CANN
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(MATHÉMATIQUES APPLIQUÉES)
AVRIL 2018

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

OPPORTUNISME ET ORDONNANCEMENT EN OPTIMISATION SANS-DÉRIVÉES

présenté par : SARRAZIN-MC CANN Loïc Anthony
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de :

M. ADJENGUE Luc Désiré, Ph. D., président
M. AUDET Charles, Ph. D., membre et directeur de recherche
M. ALARIE Stéphane, M. Sc. A., membre

DÉDICACE

À ma mère Nicole.

REMERCIEMENTS

En premier lieu, je voudrais principalement remercier mon directeur Charles Audet de m'avoir donné l'opportunité de faire ce projet avec son équipe, sans qui rien n'aurait été possible. Je voudrais aussi remercier les membres du jury de mon mémoire Luc Adjengue et Stéphane Alarie d'avoir lu et révisé mon travail.

Je voudrais aussi remercier Sébastien Le Digabel et Christophe Tribes de m'avoir introduit à leur domaine d'expertise alors que je n'étais qu'un jeune étudiant de premier cycle en quête d'un projet intégrateur. Ce sont ces rencontres avec vous qui m'ont motivé à aller de l'avant dans ce domaine.

Merci à tous les membres du groupe de recherche en optimisation sans dérivées, soient encore Charles, Sébastien, Christophe, mais aussi Viviane Rochon-Montplaisir pour leurs conseils de toutes natures et leur soutien technique.

Ensuite, je voudrais remercier tous mes collègues de bureau et dorénavant amis, soit Catherine Poissant, Maxime McLaughlin, Kenjy Demeester, Damoon Robatian, Mathieu Besançon, Marie-Ange Dahito, Julien Côté-Massicotte, Émilie Chénier, Mathilde Kelly Bourque, Pierre-Yves Bouchet, Vilmar De Sousa, Michael Dutra, Nadir Amaioua, Stéphane Jacquet, Mariana Rocha, Ludovic Salomon, Vinicius Motta, Christian Bingane et tous les autres pour leur soutien et leurs présences, sans qui ces deux années auraient été bien moins plaisantes. Les parties de tarot et les abus de café vont me manquer.

Je voudrais remercier mes proches de m'avoir supporté, soient premièrement mes amis Vincent "Gogoaw" Dubois, Dave Teoli, David "Snowman" Delorme, Alexandre Veloza, Alexandre Levert, Jean-Nicolas Dang, Éric Gabrielli, Mathieu Goyer et tous les autres qui gâchent systématiquement mes nuits de sommeil. Un merci spécial à mes grands amis Cédrik "Rico Noiseux" Rochon, Cédric Noiseux et Vincent Couturier, sans qui la sarrazinite kirghize m'aurait emporté. Un grand merci à mon père Jean-Marc Mc Cann, à ma soeur Alexandra Sarrazin-Mc Cann et son copain Maxime Piquette, à ma mère Nicole Sarrazin et mon beau-père Michel Cadieux pour leur support et leur confiance en moi. Enfin, je voudrais remercier mille fois mon amoureuse Catherine Boudreau, pour son support, ses encouragements incessants, ses conseils et sa patience.

RÉSUMÉ

Ce mémoire traite de comparaisons de stratégies algorithmiques applicables à certains algorithmes d'optimisation de boîtes noires. Une boîte noire est un problème typiquement lourd en temps de calcul, non différentiable, bruité et potentiellement propriétaire. Afin de résoudre ces problèmes, des techniques n'utilisant pas les informations sur les dérivées des expressions algébriques définissant les problèmes ont été conçues.

Les méthodes ciblées pour y implémenter les stratégies algorithmiques sont celles figurant dans un cadre de travail spécifique. Les méthodes doivent contenir une étape nécessitant une séquence d'évaluations de la fonction objectif à des points figurant dans une liste prédéterminée. Les méthodes de recherche directe correspondent à ces critères ; elles sont des méthodes contenant une série d'évaluations de points, et basent la détermination de la prochaine liste de candidats sur les valeurs obtenues à ces points. Les méthodes d'optimisation par recherche directe identifiées sont la recherche par coordonnées (CS), la recherche par motifs généralisée (GPS), la recherche par ensembles générateurs (GSS), la recherche directe par treillis adaptatifs (MADS) et le filtrage implicite (IMFIL).

La stratégie algorithmique étudiée est la stratégie opportuniste. Elle désigne l'arrêt de l'étape de l'algorithme comprenant une série d'évaluations séquentielles. L'arrêt peut-être fait après un ou plusieurs succès, c'est-à-dire l'évaluation d'un point pour lequel la valeur de la fonction objectif est inférieure à la minimale encourue précédemment dans la résolution. Plusieurs déclinaisons de la stratégie sont identifiées, implémentées et testées sur une batterie de problèmes tests, afin de déterminer numériquement leurs impacts.

L'arrêt d'une série d'évaluations implique que l'ordre des points à évaluer impacte la performance de la stratégie opportuniste. On identifie les stratégies d'ordonnancements suivantes afin de les jumeler à la stratégie opportuniste et d'étudier leurs impacts : lexicographique, aléatoire, utilisant la direction du dernier succès et utilisant les modèles. Toutes ces stratégies sont implémentées et testées sur l'ensemble de problèmes tests.

Les tests numériques sur un ensemble de problèmes incluant une boîte noire révèlent que l'opportunisme le plus rudimentaire, dont le critère d'arrêt est l'obtention d'un simple succès, est bénéfique à divers degrés pour toutes les méthodes de recherche directe identifiées sauf le filtrage implicite. Les stratégies d'ordonnancement utilisant les modèles, aléatoire et utilisant la direction du dernier succès, forment le classement des stratégies bénéfiques. L'ordonnancement lexicographique jumelé à l'opportunisme nuit aux algorithmes.

ABSTRACT

This work considers algorithmic strategies to implement and compare through various blackbox optimization algorithms. A blackbox is a problem to be minimized which is typically computationally heavy, non-differentiable, noisy and may be driven by a private or legacy code. Solving these problems requires optimization techniques that are not using first-order derivative informations of the analytical functions underlying the blackbox.

The methods in which the algorithmic strategies are to be implemented need to fit a specific framework. They must contain a step in which there is a sequence of evaluations of the objective function. Those evaluations are to be made at predetermined list of candidate points. A family of derivative-free methods, the direct search methods, qualifies as methods fitting this framework. Direct search methods choose the next list of candidate points based on the function evaluation values at the current list of candidates. The identified direct search methods for this work include the Coordinate Search (CS), Generalized Pattern Search (GPS), Generative Set Search (GSS), Mesh-Adaptive Direct Search (MADS), and Implicit Filternig (IMFIL).

The main algorithmic strategy studied in this work is the opportunistic strategy, which designates the premature ending of the algorithm's step consisting of serial function evaluations. This termination occurs after one or more successes. A success indicates the finding of a candidate point for which the objective function value is improved from the past best known point. Three variations of the opportunistic strategy are identified.

The termination of the step implies that the order of in the candidate list affects the performance of the strategy. Four main ordering strategies are identified and teamed with the opportunistic strategy in order to determine what impact does ordering have in this context. Those strategies consist of a lexicographic ordering, a random ordering, an ordering based on the direction of the last success and an ordering based on a quadratic surrogate function.

Numerical results are obtained via the resolution of a large problem set including a typical blackbox. The results show that the simplest of the opportunistic strategies, the one that terminates immediately after the identification of a single success, benefits the most to all the methods, beside Implicit Fitlering, for which a standard non-opportunistic use works best. The ordering strategies by order of performance gain is the quadratic surrogate-based, the random and the last success direction-based. Lexicographic ordering mixed with opportunism deteriorates the performance of the methods.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES, ABRÉVIATIONS ET SYMBOLES

CS	Recherche par coordonnées
GPS	Rercherche par motifs généralisée
MADS	Recherche par treillis adaptatifs
GSS	Generating Set Search
IMFIL	Implicit Filtering
BBO	Optimisation de boîte noire
DFO	Optimisation sans dérivées

B	Matrice du modèle quadratique
C^k	Matrice génératrice.
c	Contrainte quantifiable.
D	Ensemble générateur positif.
d	Direction de recherche
d_s	Profil de donnée du solveur s.
e_i	Direction coordonnée.
F^k	Cadre à l'itération k.
f	Fonction objectif.
f_Ω	Fonction objectif sous barrière extrême.
G	Matrice de base.
H^k	Ensemble des directions supplémentaires pour la SEARCH.
h	Quantification de la violation de contraintes
k	Compteur d'itérations.
L^k	Matrice de directions complémentaires à l'itération k.
l	Borne inférieur
M^k	Maillage à l'itération k.
n	Dimension du problème.
P^k	Ensemble des points de sonde à l'itération k.
P	Ensemble de problèmes.
p	Problème.
S^k	Ensemble des points de sonde à l'itération k.
S	Ensemble de solveurs.
s	Solveur.

u	Borne supérieure.
x^k	Centre de sonde à l'itération k.
β	Borne sur la longueur d'une direction.
Γ^k	Matrice génératrice de l'espace.
Δ^k	Longueur du cadre à l'itération k.
δ^k	Longueur du pas à l'itération k.
ϵ_{stop}	Critère d'arrêt de l'algorithme.
$\kappa(D^k)$	Mesure cosinus de l'ensemble D^k.
λ^k	Paramètre de contraction.
ξ	Élément de la base naturelle.
$\rho(\delta^k)$	Fonction de force.
ρ_s	Profil de performance du solveur s.
τ	Paramètre d'ajustement du maillage.
ϕ^k	Paramètre d'expansion.
Ω	Ensemble réalisable de solution.

LISTE DES ANNEXES

CHAPITRE 1 INTRODUCTION

L'optimisation est l'étude de l'obtention d'un minimum ou d'un maximum pour un problème donné. Les problèmes sont définis par leur vecteur de variables x, la fonction objectif $f(x)$ ainsi que l'ensemble réalisable Ω. On cherche alors à résoudre :

$$\min_{x \in \mathbb{R}^n} \{f(x) \ : \ x \ \in \ \Omega\} \tag{1.1}$$

avec $f : \mathbb{R}^n \to \mathbb{R}$ et $\Omega \subseteq \mathbb{R}^n$. Ce problème peut prendre plusieurs formes, et plusieurs disciplines de l'optimisation existent par conséquent. Les problèmes peuvent être tels que x est composé de variables continues, entières ou une combinaison des deux, pour lesquels existent l'optimisation continue, en nombres entiers et mixte. Une grande quantité de spécificités sur les variables, la fonction objectif et les contraintes mènent à une panoplie de sous-disciplines en optimisation.

Les principaux algorithmes d'optimisation exploitent la structure même du problème, telle la connaissance des dérivées du premier et du second ordre pour la résolution. On pense ici à la méthode du simplex pour l'optimisation linéaire [63], ou encore aux méthodes de Newton ou de Quasi-Newton pour l'optimisation non linéaire [65].

1.1 Optimisation de boîtes noires

Dans le cas traité dans ce mémoire, on étudie un certain type de problèmes dit de **boîtes noires**. Les boîtes noires sont caractérisées par leur fonctionnement : elles prennent en entrée un vecteur de variables, et retournent les valeurs de la fonction objectif et des contraintes. Les fonctions sous-jacentes sont inconnues, trop complexes ou trop instables pour en retirer une forme analytique. On peut résumer une boîte noire à un problème complexe, instable, demandant beaucoup de ressources à résoudre, dont les optima locaux sont fréquents et dont les dérivées sont inconnues, difficilement obtenables ou même inexistantes. Les boîtes noires sont typiquement bruitées. Les contraintes peuvent aussi prendre la forme de boîtes noires.

La discipline de l'optimisation qui se concentre sur la résolution de problèmes dont la structure de la fonction objectif et des fonctions contraintes ne peut être exploitée se nomme l'optimisation de boîtes noires (*BBO - Blackbox Optimization*). On distingue de cette discipline l'optimisation sans dérivées (*DFO - Derivative-free optimization*), dans laquelle on étudie la résolution de problème d'optimisation en n'utilisant aucune dérivée de la fonction objectif ou des contraintes [12, 14]. Ainsi, les outils de *DFO* utilisent seulement les valeurs de

la fonction objectif et des contraintes calculées afin de bâtir des modèles qui seront eux-mêmes utilisés dans le processus d'optimisation. Contrairement à la *BBO*, la *DFO* ne sous-entend pas l'inexistence des dérivées de la fonction à optimiser, seulement qu'elles ne sont peut-être pas appropriées comme outil pour la résolution du problème.

Conn, Scheinberg et Vincente [26] regroupent les méthodes de *DFO* en deux familles : les méthodes de recherche directe et les méthodes de régions de confiance. Les méthodes de recherche directe se résument à ce que d'autres nomment les méthodes d'échantillonnage (sampling methods) [48] : l'optimisation est contrôlée par l'évaluation de f dans un ensemble de points, et c'est le résultat de ces évaluations qui détermine le prochain ensemble de points, jusqu'à la convergence. Les méthodes de régions de confiance, quant à elles, utilisent des modèles pour décrire des points candidats, évaluent ces points et mettent à jour les modèles avec les résultats de ces points pour ensuite trouver un autre point candidat et répéter ce processus jusqu'à la convergence.

1.2 Définition de la problématique

Les méthodes de recherche directe diffèrent des méthodes de régions de confiance de par leur structure. Plusieurs définitions existent pour les méthodes de recherche directe. Hooke et Jeeves [43] proposent en 1961 une définition qui éclaircie l'idée derrière les méthodes.

> "On utilise l'expression «recherche directe» pour décrire l'examination séquentielle de solutions candidates impliquant la comparaison de chaque candidat à la meilleure obtenue précédemment, avec une stratégie pour déterminer les prochains candidats [...]."

Depuis l'avènement de problèmes dont les dérivées sont pratiquement inexistantes ou incalculables, les méthodes de recherche directe sont arrivées au cœur des intérêts de plusieurs chercheurs. Cependant, puisque les évaluations sont couteuses, il y a un besoin de développer ces méthodes de façon à conserver leur fondements théoriques de convergence tout en ayant recours à un minimum d'évaluations de la fonction objectif. Par exemple, Audet et al. [13] ont travaillé à la génération d'un espace générateur avec le moins de directions possible pour ainsi réduire le nombre de directions nécessaires à chaque itération. À chaque itération, un algorithme de recherche directe génère une liste d'au moins un point candidat, tandis qu'un algorithme de région de confiance n'en a assurément qu'un seul. Certaines méthodes de recherche directe sont cependant similaires aux méthodes de régions de confiance en ce sens, tel que l'algorithme de Nelder et Mead [64], qui est classé comme une méthode de recherche directe simpliciale par Conn, Scheinberg et Vincente [26].

Puisque chaque évaluation de la fonction objectif est très coûteuse en ressources, on en vient à se demander si l'évaluation de chaque point dans la liste générée à chaque itération est nécessaire au bon déroulement de l'algorithme. Torczon [75] démontre qu'une famille d'algorithmes de recherche directe converge vers un optimum sans avoir à évaluer tous les points dans une liste générée à une itération donnée. Ainsi, la convergence n'est pas assurée par la qualité de la direction de descente utilisée à chaque itération mais par sa propriété de répondre à la définition de décroissance simple $f(x^k + \delta^k d) - f(x^k) < 0$, avec δ^k la longueur du pas et $d \in D^k$ la direction de descente choisie dans l'ensemble des directions propre à l'itération k. On en vient à l'élaboration d'une problématique qui justifie le projet. Pour un algorithme possédant à l'itération k une solution courante x_k et un liste de candidats P^k, on cherche à déterminer si la découverte d'un nouveau meilleur point dans cette liste devrait entrainer un passage prématuré à l'itération suivante de l'algorithme.

1.3 Objectif de la recherche

Le long de ce mémoire, l'interruption prématurée des évaluations d'une liste de candidats sera nommée *stratégie opportuniste* ou *opportunisme*, d'après l'expression introduite par Coope et Price [27] et reprise par Audet et Dennis [8]. La question restant très vague, on devra définir plusieurs aspects afin d'en venir à une quantification de l'impact de l'opportunisme sur le déroulement d'algorithmes de *DFO*.

En premier lieu, la structure de l'algorithme doit admettre qu'au cours d'une itération il existe une liste de points à évaluer séquentiellement avant la détermination d'une autre liste de points. Ainsi, appartenir à la famille des méthodes de recherche directe n'est pas un critère suffisant compte tenu que l'algorithme de Nelder et Mead y figure, malgré qu'il évalue un point à la fois. Cependant, d'autres familles d'algorithmes ouvrent la porte à l'opportunisme par leur structure. Notons ici les méthodes de recherche linéaire basées sur les dérivées simplexes, telles que baptisées par Conn, Scheinberg and Vincente [26], qui ne concordent pas exactement à la définition de méthodes de recherche directe. Un objectif premier de cette recherche est d'identifier les algorithmes pour lesquels l'implémentation de l'opportunisme est possible.

L'utilisation de l'opportunisme soulève une autre problématique au cœur de cette recherche, c'est-à-dire la stratégie utilisée pour ordonner les points candidats dans la liste. Si on permet une interruption de la séquence d'évaluations des points sur la liste lors de l'obtention d'un meilleur point, alors l'ordre dans laquelle apparaît ces points influera sur le comportement de l'algorithme. On nommera la règle définissant l'ordre des points dans la liste la *stratégie d'ordonnancement*. Le second objectif de cette recherche est d'identifier les stratégies d'ordonnancement qui pourraient avoir un impact sur la performance de la stratégie

opportuniste.

Ayant en main un éventail d'algorithmes et de stratégies d'ordonnancement, on pourra procéder à l'objectif principal de la recherche, soit la caractérisation des impacts de l'utilisation de la stratégie opportuniste dans les algorithmes, ainsi que la comparaison des différentes stratégies d'ordonnancement identifiées. On cherche ainsi à déterminer les cas dans lesquels l'opportunisme est une stratégie envisageable, nécessaire, ou contraignante. On cherche aussi à caractériser les différentes stratégies d'ordonnancement en fonction du type de problème à résoudre, des spécificités algorithmiques ou en fonction de la précision demandée à l'algorithme.

1.4 Plan du mémoire

Le mémoire comporte trois thèmes principaux. En premier lieu, au chapitre 2 se trouve une revue de littérature pour déterminer l'état de l'art des méthodes de recherche directe ainsi que les autres outils d'optimisation sans-dérivées nécessaires à la compréhension du lecteur. Le chapitre 3 contient les définitions formelles de la stratégie opportuniste, ses déclinaisons et des stratégies d'ordonnancement ainsi que leur instauration dans les algorithmes identifiés précédemment. Le chapitre 4 porte sur la méthodologie utilisée, les problèmes étudiés et les essais numériques effectués pour caractériser les impacts de la stratégie opportuniste ainsi que des stratégies d'ordonnancement. Enfin, le chapitre 5 résume le sujet, les objectifs et les résultats de ce travail et conclue avec pistes possibles pour la poursuite de ce sujet de recherche.

CHAPITRE 2 REVUE DE LITTÉRATURE : OUTILS D'OPTIMISATION PAR RECHERCHE DIRECTE

Ce chapitre présente un état de l'art sur les outils utilisés en optimisation sans-dérivées par recherche directe. Elle se concentrera sur la description des outils qui sont nécessaires à la définition et à l'application de la stratégie algorithmique étudiée, soit l'opportunisme, décrite au chapitre 3. En premier lieu, une revue des méthodes de recherche directe ciblées pour l'étude est effectuée. Cette étape nécessite une grande attention à l'identification des caractéristiques des méthodes qui garantissent la convergence théorique de la méthode. De cette façon, les modifications apportées aux algorithmes pourront être effectuées sans interférer avec les hypothèses requises pour l'analyse de convergence. Ensuite, on traitera de la gestion des contraintes en fonction de leurs types, de façon à observer subséquemment les interactions entre l'opportunisme et les techniques de gestion de contraintes à disposition. Enfin, on définira les modèles quadratiques utilisés par quelques-unes des implémentations utilisées lors des essais numériques du chapitre 4.

2.1 Recherche par coordonnées

Le premier algorithme à être abordé est aussi le plus intuitif pour l'optimisation sans contraintes, qu'on appellera *Coordinate Search* (CS), ou Recherche par coordonnée, parfois appelée *Compass Search* [51]. On attribue à Fermi et Metropolis [34] cette première méthode de recherche directe. Pour que leur modèle suive bien leur ensemble de données expérimentales sur la diffusion nucléaire, Fermi et Metropolis ont fait varier les paramètres théoriques de déphasage de leur fonction un à la fois avec un pas constant. Lorsque ni l'augmentation ni la diminution de l'un des paramètres n'amélioraient la concordance avec les données expérimentales, la longueur du pas était diminuée de moitié et le processus était recommencé. On continuait ainsi jusqu'à ce que le pas soit considéré comme suffisamment petit. C'est ainsi qu'est née la méthode de la recherche par coordonnées. Plus tard dans la littérature on fera référence à la recherche par coordonnées comme un algorithme de la recherche par motifs. Booker et al. [19] proposent un cadre rigoureux aux algorithmes de recherche par motifs. On y stipule que les algorithmes opèrent en deux étapes, soit la recherche qu'on notera SEARCH et la sonde qu'on notera POLL afin d'être conforme avec la littérature. La SEARCH consiste à explorer le domaine de la fonction sans égard à la détermination d'un optimum. La POLL cherche à minimiser la fonction pour déterminer un optimum. Dans la recherche par coordonnées, la POLL correspond à l'entièreté de la méthode, qui ne contient aucune étape de SEARCH. La

description de l'algorithme CS qui suit est fortement inspirée de celle de Audet et Hare [12].

Algorithme 1 Recherche par coordonnée (CS)

Avec $f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x^0 le point de départ

0. Initialisation des paramètres :

$\delta^0 \in (0, \infty)$ la longueur du pas initial

$\epsilon_{\text{stop}} \in [0, \infty)$ le critère d'arrêt

$k \leftarrow 0$ le compteur d'itérations

1. POLL

if $f(t) < f(x^k)$ pour un $t \in P^k := \{x^k \pm \delta^k \overrightarrow{e}_i : i = 1, 2, \ldots, n\}$ **then**

$x^{k+1} \leftarrow t$ et $\delta^{k+1} \leftarrow \delta^k$

else

$x^{k+1} \leftarrow t$ et $\delta^k \leftarrow \frac{1}{2}\delta^k$

end if

2. Terminaison

if $\delta^k \geq \epsilon_{\text{stop}}$ **then**

$k \leftarrow k + 1$

go to 1.

else

stop

end if

avec $\overrightarrow{e}_i$ les vecteurs unitaires. À l'étape de POLL de l'algorithme, on évalue la fonction $f(x)$ à chaque élément $t \in P^k$ avant de déterminer quel t deviendra le nouveau centre de sonde x^{k+1}. Cependant, si les évaluations sont effectuées en série, on aura une séquence de points à évaluer. La séquence proposée est celle de $2n$ mouvements dans les directions élémentaires suivies de la détermination d'un nouveau centre de sonde si au moins une évaluation est un succès. C'est dans cette séquence que l'opportunisme pourra être introduit, afin de ne pas évaluer le reste de la liste si le test de décroissance simple est positif.

2.2 Recherche par motifs généralisée

Le deuxième algorithme présenté est la recherche par motifs généralisée. La première recherche par motifs en soit est élaboré par Hooke et Jeeves [43]. Ils nomment la recherche par motifs (*Pattern Search* ou PS) la routine de recherche directe visant à minimiser une fonction $f(x)$ avec leur algorithme. Cette routine est composée d'une série de mouvements autour d'un point qui peuvent être divisés en deux types, soit les mouvements exploratoires

(*Exploratory Moves*) ou les mouvements destinés à la détermination d'un minimum, soit les mouvements de motifs (*Pattern Moves*). Les mouvements exploratoires servent à la détermination du motif, c'est-à-dire au comportement de la fonction $f(x^k)$ aux alentours d'un point x^k. Les mouvements de motifs sont ensuite effectués dans la direction que les mouvements exploratoires ont déterminée comme étant celle qui se dirige vers le minimum de la fonction. Hooke et Jeeves introduisent aussi la définition de succès et d'échec. Un succès est un mouvement tel que la valeur de $f(x)$ au point est inférieure à la meilleure valeur connue auparavant. Dans le cas contraire, on dira que c'est un échec.

Lors de l'élaboration de cette technique ils mentionnent que :

> Par souci de simplicité, les mouvements exploratoires sont choisis de façon simple, c'est-à-dire, à chaque mouvement seulement la valeur d'une unique coordonnée est changée.

Pour ensuite affirmer que :

> Suivant un mouvement de motif fructueux, il est raisonnable de conduire une série de mouvements exploratoires et de tenter d'améliorer davantage les résultats.

De ces deux affirmations découlent les principes de bases de PS, soit qu'une succession de mouvements exploratoires dans les directions coordonnées $e_i : i = \pm\{1, 2 \dots n\}$ et d'un mouvement de motifs dans la meilleure direction. Chaque succès de la recherche par motifs entraîne que la prochaine série de mouvements exploratoires sera effectuée autour de ce nouveau meilleur point. Lorsque la succession échoue, la longueur du pas est réduite afin de déterminer un nouveau motif. Pour préciser la méthode, un point initial x^0 doit être déterminé ainsi qu'une longueur de pas initiale δ^0 et un critère ϵ_{stop} pour lequel on jugera que le pas est devenu suffisement petit. De plus, on doit déterminer la méthode utilisée pour procéder à la réduction de la longueur du pas. Dans les mots de Hooke et Jeeves, la POLL serait l'ensemble de mouvements exploratoires accompagné de la mise à jour du centre de sonde, tandis que la SEARCH serait le mouvement de motif restant.

Torczon [75] amène une généralisation du PS de Hooke et Jeeves. Dans cet article, l'auteure approche la méthode d'un autre angle. Le motif propre à l'itération P^k est issu de la multiplication de deux matrices, soient $G \in \mathbb{R}^{n \times n}$ la matrice de base et $C^k \in \mathbb{R}^{n \times p}, p > 2n$, la matrice génératrice. La matrice de base se doit d'être non-singulière et la matrice génératrice se doit d'être composée telle que $C^k = [\Gamma^k \ L^k]$, ou Γ^k est la concaténation d'une matrice de plein rang M et de son opposée. Le rôle de Γ^k est de générer l'espace $\mathbb{R}^n$, tandis que le rôle de la L^k est de complémenter cette dernière avec des directions supplémentaires ne servant pas strictement à générer l'espace, rappelant la SEARCH. Ainsi, GC^k est une matrice qui génère l'espace $\mathbb{R}^n$ et qui se libère du cadre restreignant des directions unitaires proposé par Fermi et

Metropolis et repris par Hooke et Jeeves. Armés de GC^k et d'une longueur de pas spécifique à une itération δ^k, on retrouve $\delta^k G c_i^k$, soit une généralisation de $\delta^k e_i$ présent dans CS, mais pour lequel on a une colonne $c_i^k \in C^k$ perturbée par G remplaçant la direction élémentaire.

Toujours selon Torczon, la définition de mouvements exploratoires est reprise pour être plus générale. Un mouvement exploratoire est en fait un vecteur issu du motif $s^k \in P^k$. L'auteure demande aussi que, si il existe un vecteur colonne $c_i^k \in P^k$ tel que $f(x^k + \delta^k c_i^k) < f(x^k)$, alors les mouvements exploratoires doivent produire un pas s^k tel que $f(x^k + s^k) < f(x^k)$.

L'algorithme s'inscrit alors en cinq étapes. Premièrement, à l'initialisation de l'algorithme, vient le calcul de la fonction $f(x)$ à l'itéré initial x^0. Deuxièmement, le calcul d'une direction s_k issue de la série de mouvements exploratoires. Troisièment, le calcul de $f(x^k + s^k)$. Ensuite vient la mise à jour $x^{k+1} = x^k + s^k$ si l'étape précédente est un succès, sinon $x^{k+1} = x^k$. Enfin, au besoin, on met à jour l'ensemble générateur C^k et la longueur du pas δ^k et on recommence le processus à partir de la deuxième étape jusqu'à ce que δ^k soit jugé suffisamment petit.

Afin de rester dans le cadre de Booker et al. [19], on décrit l'algorithme en se rattachant aux concepts de POLL et de SEARCH. Les directions issues de la matrice supplémentaire $s^k \in GL^k$ introduite par Torczon [75] correspondent à une étape de SEARCH en vertu de sa fonction d'incorporer des directions supplémentaires à la recherche, tandis que les mouvements exploratoires visant à trouver un minimum autour du point de sonde $(f(x^k + s^k), s^k \in D\Gamma^k)$ correspondent à la POLL. Cependant, on généralisera d'avantage $D\Gamma^k$ afin de le remplacer par un ensemble générateur positif [32], noté $D = GZ, Z \in \mathbb{R}^{n \times p}$, qui répond aux exigences exactes énoncées dans [75] sans imposer que la taille de la matrice bornée supérieurement à $p = 2n$, mais bien tel que $n + 1 \leq p \leq 2n$, en concordance avec les propriétés d'une base positive. Cette représentation est fortement inspirée de celle fournie par Audet et Hare dans [12]. On dira de $S^k = GL^k$ qu'il est l'ensemble des directions supplémentaires L^k fournit à l'itération k soumit à une transformation par la matrice inversible $G \in \mathbb{R}^{n \times n}$.

Par ailleurs, il est à noter que les algorithmes de recherche directe énoncés dans [54, 75, 55, 56], qui sont des algorithmes déclinant de la famille d'algorithmes GPS, diffèrent de la formulation donnée dans ce document inspirée de Audet et Hare [12], notamment sur un aspect principal : la mise à jour de la longueur du pas se fait par deux paramètres différents, soient $\delta^{k+1} = \phi^k \delta^k, \phi^k \geq 1$ pour l'expansion du maillage dans le cas ou l'itération k est un succès et $\delta^{k+1} = \lambda^k \delta^k, \lambda^k \in (0, 1)$ dans le cas d'un échec. Dans la formulation présente on utilise un seul paramètre $\tau \in (0, 1)$ et son inverse τ^{-1} pour cette mise à jour. Cependant, cette altération ne contrevient pas aux requis de la démonstration de la convergence de [75] qui assurent que la formulation présente possède les mêmes propriétés. La formulation présente aussi une différence notable avec celle faite par Booker et al. dans [19], c'est-à-dire l'admission

de la mise à jour de la longueur du pas pour une SEARCH fructueuse, alors que Booker et al. incorporent la mise à jour seulement dans un échec de la POLL, à l'image de l'utilisation d'un facteur $\phi^k = 1$.

Algorithme 2 Recherche par motifs généralisée (GPS)

Avec $f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x^0 le point de départ

0. Initialisation des paramètres :

$\delta^0 \in (0, \infty)$	la longueur du pas initial
$D = GZ$	un ensemble générateur positif
$\tau \in (0, 1) \cup \mathbb{Q}$	le paramètre d'ajustement du maillage
$\epsilon_{\text{stop}} \in [0, \infty)$	le critère d'arrêt
$k \leftarrow 0$	le compteur d'itérations

1. SEARCH

if $f(t) < f(x^k)$ pour un $t \in S^k$ **then**
$x^{k+1} \leftarrow t$ et $\delta^{k+1} \leftarrow \tau^{-1}\delta^k$
go to 3
else
go to 2.
end if

2. POLL

détermination d'un ensemble générateur positif $\mathbb{D}^k \subseteq \mathbb{D}$
if $f(t) < f(x^k)$ pour un $t \in P^k := \{x^k + \delta^k d : d \in D^k\}$ **then**
$x^{k+1} \leftarrow t$ et $\delta^{k+1} \leftarrow \tau^{-1}\delta^k$
else
$x^{k+1} \leftarrow t$ et $\delta^k \leftarrow \tau\delta^k$
end if

3. Terminaison

if $\delta^k \geq \epsilon_{\text{stop}}$ **then**
$k \leftarrow k + 1$
go to 1.
else
stop
end if

L'opportunisme pourra être implémenté dans l'étape de POLL selon la même logique que dans CS, c'est-à-dire à même la liste des points $x^k + \delta^k d : d \in D^k$. Pour ce qui est de la SEARCH, l'implémentation de la stratégie est faisable mais sa performance n'en vient qu'à

dépendre de la pertinence de la méthode pour générer S^k.

2.3 Mesh Adaptive Direct Search

Le troisième algorithme présenté est dans la même lignée que CS et GPS. Il s'agit de la recherche directe par maillage adaptatif (*Mesh Adaptive Direct Search* ou MADS), formulé par Audet et Dennis [9]. Contrairement aux algorithmes précédents, MADS est un algorithme basé sur un maillage, c'est-à-dire que chaque évaluation de la fonction objectif se fera à un point sur une discrétisation de l'espace des variables. On dénotera

$$M^k := \{x^k + \delta^k Dy : y \in \mathbb{N}^p\} \subset \mathbb{R}^n$$

le maillage à l'itération k sur lequel les itérés seront définis, générés par l'ensemble générateur positif $D = GZ$ et y les entiers naturels. Il s'agit ici d'un maillage de cardinalité infini. Dans la formulation de Audet et Hate reprise ici, les algorithmes CS et GPS sont aussi considérés comme étant basé sur un maillage, quoique leurs premières références [43, 75] n'en indique pas autant.

La grande distinction de MADS ne réside alors pas dans son utilisation d'un maillage comme structure, mais bien dans l'incorporation d'un cadre à son étape de sonde. On définit le cadre tel que $F^k := \{x \in M^k : ||x - x^k||_\infty \leq \Delta^k b\}$ à l'itération k, soit l'ensemble des points sur le maillage M^k pour lesquels la distance de Chebyshev à l'itéré courant est inférieure à une valeur $\Delta^k b$. Cette valeur est déterminée par $b = \max\{||d'||_\infty : d' \in \mathbb{D}\}$, soit la plus grande composante vectorielle présente dans toutes les colonnes de la base positive $\mathbb{D}$ issue de l'ensemble générateur D, multipliée par un paramètre introduit dans MADS, soit le paramètre de longueur du cadre Δ^k.

La figure 2.1 montre une itération hypothétique de MADS pour laquelle $\Delta^k = \frac{1}{2}, \delta^k = \frac{1}{8}$ avec l'ensemble générateur $D = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix}$.

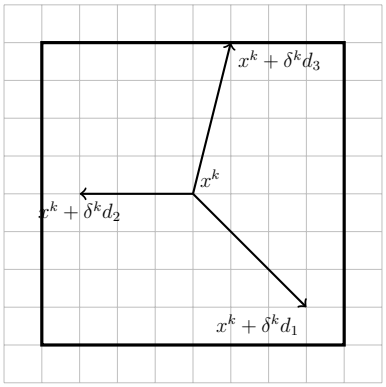


Figure 2.1 Ensemble générateur, maillage (gris) et cadre (noir)

Dans GPS, quoique l'algorithme décrit n'inclue pas de maillage formellement, les évaluations sont limitées aux directions de D mises à l'échelle avec la longueur du pas. Dans MADS, on pourra choisir à l'itération k n'importe quelle direction d^k tant que l'ensemble des directions D^k_Δ forme un ensemble générateur positif qui soit compris dans le cadre F^k. La mise à jour de la longueur du cadre se fait avec le paramètre d'ajustement du maillage qui, dans CS et GPS, servait pour la mise à jour de δ^k. Dans MADS, la mise à jour de δ^k suggérée est $\delta^k = \min(\Delta^k, (\Delta^k)^2)$. Ainsi, on s'assure de respecter $\delta^k \leq \Delta^k$ et on augmente exponentiellement le nombre de directions possibles si $\Delta^k \geq 0$. Afin de rester dans le cadre de Booker et al. [19], on décrit l'algorithme en se rattachant aux concepts de POLL et SEARCH avec un étape SEARCH limitée à un ensemble S^k de points. La description de l'algorithme est fortement inspirée de celle de Audet et Hare, issue de [12].

Algorithme 3 Recherche par treillis adaptatif (MADS)

Avec $f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x_0 le point de départ

0. Initialisation des paramètres :

$\Delta^0 \in (0, \infty)$	la longueur du cadre initial
$D = GZ$	un matrice génératrice positive
$\tau \in (0, 1) \cup \mathbb{Q}$	le paramètre d'ajustement du maillage
$\epsilon_{\text{stop}} \in [0, \infty)$	le critère d'arrêt
$k \leftarrow 0$	le compteur d'itérations

1. Mise à jour des paramètres

   $\delta^k \leftarrow \min(\Delta^k, (\Delta^k)^2)$

2. SEARCH

   if $f(t) < f(x^k)$ pour un $t \in S^k$ **then**

   $x^{k+1} \leftarrow t$ et $\Delta^{k+1} \leftarrow \tau^{-1}\Delta^k$

   go to 4

   else

   go to 3

   end if

3. POLL

   avec le cadre F^k de demi-coté Δ^k

   détermination d'un ensemble générateur positif $\mathbb{D}_\Delta^k \subset F^k$

   if $f(t) < f(x^k)$ pour un $t \in P^k := \{x^k + \delta^k d : d \in \mathbb{D}_\Delta^k\}$ **then**

   $x^{k+1} \leftarrow t$ et $\Delta^{k+1} \leftarrow \tau^{-1}\Delta^k$

   else

   $x^{k+1} \leftarrow t$ et $\Delta^k \leftarrow \tau\Delta^k$

   end if

4. Terminaison

   if $\delta^k \geq \epsilon_{\text{stop}}$ **then**

   $k \leftarrow k + 1$

   go to 1.

   else

   stop

   end if

L'implémentation de la stratégie opportuniste sera faite à l'étape de POLL selon la même logique que pour CS et GPS.

2.4 Generating Set Search

Le quatrième algorithme est la recherche par ensemble générateur. Il est introduit par Kolda, Lewis et Torczon dans [51] sous le nom de *Generating Set Search* qu'on abrègera GSS. Il s'agit d'un algorithme très similaire à GPS, mais qui incorpore certains aspects laissés de côté lors de la formulation de GPS et certains aspects nouveaux propres à GSS. À l'instar de GPS, un ensemble de directions à chaque itération est dédié à la SEARCH, soit H^k et un ensemble dédié à la POLL, soit D^k.

La différence principale entre GSS et GPS est le critère d'acceptation d'un itéré comme étant un succès. Pour les algorithmes précédents, chaque itération entrainant une simple diminution $f(x^k + \delta^k d) < f(x^k)$ était considérée comme un succès. Dans GSS, on permet que le critère d'acceptation soit resserré avec l'addition du terme $\rho(\delta^k) > 0$ au test de diminution, soit que $f(x^k + \delta^k d) < f(x^k) + \rho(\delta^k)$, où $\rho(\delta)$ sera appelée la fonction de force et doit satisfaire à une des deux définitions suivantes. Soit $\rho(\delta)$ est continue, $\rho(\delta) = o(\delta)$ lorsque $\delta \downarrow 0$ et que $\rho(\delta_1) < \rho(\delta_2)$ si $\delta_1 < \delta_2$, ou soit $\rho(\delta) \equiv 0$. Le premier cas impose une diminution suffisante de la fonction, rappelant le critère d'Armijo [5, 41]. Cet outil permet d'assurer la convergence globale théorique de l'algorithme en utilisant les résultats de convergence de Lucidi et Sciandrone [57] avec seulement une fonction objectif $f(x)$ bornée inférieurement. Le deuxième cas correspond à une condition de diminution simple. Sous cette condition, les résultats de convergence globale sont identiques à ceux des algorithmes basés sur treillis. Les conditions de convergence sont resserrées, de façon à ce que GSS converge dans le cas où les paramètres fournis restreignent les points candidats à un maillage, et qu'aucune expansion de ce maillage n'est admise [51, 27]. Il est à noter que CS et GPS sont des algorithmes de la famille de GSS, c'est-à-dire qu'une paramétrisation spécifique de GSS permet d'obtenir GPS ou CS.

Les auteurs spécifient deux autres grandes différences en comparaison avec CS. Premièrement, on y fait valoir l'utilisation d'un ensemble générateur positif comme ensemble de directions de sonde, un aspect déjà incorporé dans notre définition de GPS. L'autre différence se situe dans la souplesse des paramètres de contraction τ^k et des paramètres d'expansion ϕ^k.

Plusieurs paramètres supplémentaires sont nécessaires tels que la mesure du cosinus, le plus petit angle entre deux directions de l'ensemble générateur $\kappa(G^k)$ qui empêche le choix de mauvaises directions, à l'instar de la mesure d'angle [41, 67] en recherche linéaire. On y fait mention aussi de la longueur des vecteurs de l'ensemble générateur étant bornée telle que $\beta_{\min} < \|d\| < \beta_{\max}, \beta_{\max} \geq \beta_{\min} > 0$.

Algorithme 4 Recherche par ensemble générateur (GSS)

Avec $f : \mathbb{R}^n \rightarrowtail \mathbb{R}$ la fonction objectif et x^0 le point de départ

0. Initialisation des paramètres :

$\delta^0 \in (0, \infty)$	la longueur du pas initial
$\lambda_{\max} \in (0, 1) \cup \mathbb{R}$	le paramètre de contraction maximale de la longueur du pas
$\lambda^0 \in (0, 1) \cup \mathbb{R}, \phi^0$	les paramètres de contraction et d'expansion du pas
$\rho : [0, +\infty] \to \mathbb{R}$	une fonction continue telle que $\nabla(\rho(\delta)) < 0$ lorsque $\delta \to 0$ et $\frac{\rho(\delta)}{\delta} \to 0$ quand $\delta \downarrow 0$
$\beta_{\max} \geq \beta_{\min} > 0$	les bornes sur la longueur des vecteurs de D^k
$\epsilon_{\text{stop}} \in [0, \infty)$	le critère d'arrêt
$\kappa_{\min} \geq 0$	La mesure cosinus minimale d'un ensemble
$k \leftarrow 0$	le compteur d'itérations

1. SEARCH

détermination de $H^k = \{s \in \mathbb{R}^n,\ \beta_{\min} \leq \|s\|\}$

if $f(t) < f(x^k) - \rho(\delta^k)$ pour un $t \in S^k = \{x^k + \delta^k s, s \in H^k\}$ **then**

$x^{k+1} \leftarrow t$ et $\delta^{k+1} \leftarrow \phi^k \delta^k$

go to 3

else

go to 2.

end if

2. POLL

détermination de $D^k = \{s \in \mathbb{R}^n,\ \beta_{\min} \leq \|s\| \leq \beta_{\max},\ \kappa(H^k \cup D^k) \geq \kappa_{\min}\}$

un ensemble générateur

if $f(t) < f(x^k) - \rho(\delta^k)$ pour un $t \in P^k := \{x^k + \delta^k d : d \in D^k\}$ **then**

$x^{k+1} \leftarrow t$ et $\delta^{k+1} \leftarrow \phi^k \delta^k$

else

$x^{k+1} \leftarrow t$ et $\delta^k \leftarrow \tau^k \delta^k$

end if

3. Terminaison

if $\delta^k \geq \epsilon_{\text{stop}}$ **then**

$k \leftarrow k + 1$

go to 1.

else

stop

end if

2.5 Implicit Filtering

Le dernier algorithme est celui du filtrage implicite (*Implicit Filtering* ou IMFIL). Cette méthode, développée par Kelley [47, 48], se veut un algorithme d'optimisation hybride, sans contraintes explicites, de recherche linéaire se basant sur le gradient du stencil qui incorpore des particularités de CS. Un stencil V consiste en un ensemble de points répartis selon un motif autour d'un point central, par exemple x^k. Si le motif consiste en l'ensemble des directions coordonnées, le stencil prendra la forme suivante :

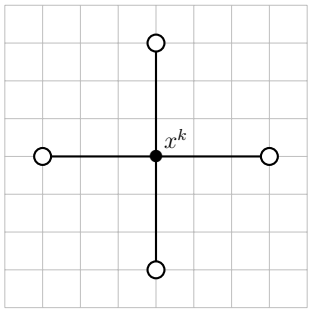


Figure 2.2 Stencil V autour du point x^k avec $n = 2$

On définira le gradient simplex de IMFIL comme l'approximation du gradient de la fonction objectif f en calculant la valeur de la fonction à chaque point d'un stencil et en posant :

$$\nabla_s f(x^k) = \frac{1}{h^k}\delta(f, x^k, V, h)V^\dagger$$

avec $V \in \mathbb{R}^{n\times 2n}$ une matrice représentant les directions du stencil,$V^\dagger$ son inverse de Moore-Penrose [36], $\delta(F, x, V, h^k)$ le vecteur colonne ayant $f(x+hv_j)-f(x)$ à son j^e élément et h^k le pas de l'itération. L'utilisation du pseudoinverse est justifiée par la nécessité de calculer des différences finies d'un seul côté si un point du stencil n'est pas réalisable. L'idée fondamentale de IMFIL est de calculer le gradient stencil en un itéré courant x^k, avec des différences simples ou centrées, et d'utiliser le gradient simplex $\nabla_s f(x^k) \in \mathbb{R}^n$ comme direction pour effectuer une recherche linéaire à l'image d'une descente du gradient [66].

L'algorithme appartient à la famille de méthodes sans-dérivées puisqu'il n'utilise pas le calcul de dérivées analytiques de la fonction. Cependant, contrairement aux étapes de POLL des algorithmes présentés précédemment, l'algorithme approxime la dérivée à l'aide de différences finies. Pour déterminer les valeurs de $f(x+hv_i)$ servant au calcul du gradient simplex, l'auteur introduit une nomenclature proche de la notre, soit le stencil poll, qu'on généralisera sonde du stencil. Cette sonde est constituée de n à $2n$ évaluations de la boîte noire. Le motif suggéré

par la méthode est une combinaison des vecteurs unitaires dans $\mathbb{R}^n$. Dans le cas où la sonde se limite à seulement n évaluations, le centre de sonde courant se trouve à la jonctions de n bornes ; les n autres points générés sont hors de Ω. Dans le cas opposé aucune contrainte de borne n'est active, tous les points générés peuvent être réalisables, pour un total de $2n$ points. C'est cette sonde du stencil qui sera analogue aux POLL des algorithmes présentés antérieurement.

Soit $\tau > 0$ un paramètre de tolérance pour la terminaison sur la norme du gradient fournie par l'utilisateur. Suivant une sonde du stencil fructueuse, si la norme de l'approximation du gradient du stencil n'est pas plus petite que τh^k, l'algorithme effectue une recherche linéaire dans la direction opposée au gradient du stencil $d = -\nabla_s f(x^k)$ avec un nombre de pas maximal spécifié par l'utilisateur. Alternativement, la direction peut-être calculée avec la résolution d'un système d'équations linéaires $Hd = \nabla_s f(x^k) f(x^k)$, où $H \in \mathbb{R}^{n \times n}$ est une matrice Hessienne issue d'un modèle mis à jour selon une méthode de Quasi-Newton (par exemple Broyden-Fletcher-Goldfarb-Shanno (BFGS) [20, 35]),$\nabla_s f(x^k) \in \mathbb{R}^n$ le gradient du stencil et $f(x^k)$ la fonction objectif évaluée à l'itéré courant. La recherche linéaire qui suit peut s'écrire de la façon suivante qu'on abrègera BLS pour *Backtracking Line Search*.

$$\min_m \{m \ : \ f(x^k + \beta^m d) < f(x^k), m \in \{0, \text{maxitarm}\} \cup \mathbb{N}\}. \tag{2.1}$$

où $\beta < 1$ agit comme un facteur diminuant pour déterminer la longueur du pas nécessaire, $d \in \mathbb{R}^n$ la direction de descente, m un entier servant de puissance à β et maxitarm est le nombre maximal de pas de recherche linéaire imposé par l'utilisateur. L'idée ici est de trouver un m minimal pour lequel la fonction évaluée $f(x^k + \beta^m) < f(x^k)$.

Dans le cas où la sonde échoue, la longueur du pas est diminuée de moitié et aucune autre recherche n'a lieu. L'algorithme est ainsi recommencé jusqu'à ce que la longueur du pas soit diminuée en deçà d'un seuil prescrit. La description de l'algorithme simplifié sort du cadre de travail précédemment imposé [19]. Par contre, il est possible de faire un parallèle entre celui-ci et IMFIL. La sonde du stencil peut être vue comme l'étape de POLL des méthodes de recherche directe, sur laquelle repose l'analyse de convergence. La sonde du stencil est suivie d'une recherche linéaire, une heuristique qui a pour but d'accélérer la convergence de l'algorithme. Un parallèle peut être fait entre la recherche linéaire de IMFIL et une étape de SEARCH des méthodes de recherche directe. Cependant, la recherche linéaire est détaillée et fixée, contrairement aux étapes de SEARCH possibles pour les autres méthodes. Pour conclure l'analogie, on évoque qu'une itération de IMFIL peut être interprétée comme une étape de POLL suivie d'une étape de SEARCH, qui celle-ci est conditionnelle au succès de la POLL, à savoir la mise en marche d'une recherche linéaire seulement lorsque la sonde du stencil est

fructueuse.

Algorithme 5 Filtrage implicite (IMFIL)

Avec $f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x^0 le point de départ

0. Initialisation des paramètres :

$h^0 \in (0, \infty)$	la longueur du pas initial
$V = \{\pm e_1, \pm e_2, \ldots, \pm e_k\}$	un stencil
budget	le nombre d'évaluations maximal
maxitarm	nombre maximal d'itérations de recherche linéaire
ϵ_{stop}	le critère d'arrêt sur le pas
$k \leftarrow 0$	le compteur d'itérations de l'algorithme
$\tau > 0$	la tolérance admise sur le gradient

1. Boucle principale :

while $f_{\text{count}} <$ budget & $h^k > \epsilon_{\text{stop}}$ **do**
  Effectuer la sonde du gradient
  Poser $P^k := \{x^k + h^k v_i : v_i \in V\}$ et choisir $x^{n+1} \in \arg\min\{f(x) : x \in P^k \cup \{x^k\}\}$
  Descente linéaire
  Mise à jour de H si nécessaire
  évaluer $-\nabla_s f(x^k)$
  if $\left\|\nabla_s f(x^k)\right\| \geq \tau h^k$ & $x^{k+1} \neq x^k$ **then**
    $d \leftarrow -\nabla_s f(x^k)$ ou $d \leftarrow H^{-1}\nabla_s f(x^k) f(x^k)$
    if BLS est réalisable **then**
      $x^{k+1} \leftarrow x^k + \beta^m d$
    end if
  else
    $h^{k+1} \leftarrow h^k/2$
  end if
end while

2.6 Gestion des contraintes en DFO

Le problème 1.1 demande que la variable de décision x soit comprise dans l'ensemble Ω. On peut représenter Ω de la façon suivante afin de représenter le cas ou les seules contraintes existantes sont des bornes :

$$X = \Omega = \{x \ \in \ \mathbb{R}^n \ : \ l_i \leq x_i \leq u_i\} \tag{2.2}$$

où $l \in \mathbb{R}^* \cup \{-\infty\}$ représente le vecteur des bornes inférieures et $u \in \mathbb{R}^* \cup \{+\infty\}$ le vecteur des bornes supérieures. Sous cette même forme, on peut définir un problème comme étant non borné, si toutes les valeurs de l et de u prennent la valeur infinie, ce qui équivaut à $X = \mathbb{R}^n$.

Si la variable x, en plus d'être limitée par des bornes, est contrainte par un ensemble d'inégalité, tel que :

$$\begin{aligned} \min_{x \in X \subset \mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & c(x) \leq 0 \end{aligned}$$

On peut représenter son ensemble Ω réalisable tel que

$$\Omega = \{x \ \in \ X \ \subset \mathbb{R}^n \ : \ c(x) \leq 0\}$$

où $c : \mathbb{R}^n \mapsto \mathbb{R}^m$ avec m le nombre de contraintes et où X est défini à l'équation 2.2. Une approche pour traiter ce problème se nomme **la barrière extrême** [9]. On définit une fonction de barrière suivante

$$f_\Omega(x) = \begin{cases} f(x) & \text{si } x \in \Omega \\ \infty & \text{sinon} \end{cases}$$

On peut alors redéfinir la fonction à optimiser comme étant f_Ω en ne se souciant plus des contraintes. Cette approche ne prend cependant pas en compte la nature des contraintes. Dans cet ordre d'esprit, Le Digabel et Wild [53] proposent une taxonomie des contraintes rencontrées en optimisation de boîtes noires. Dépendemment de la forme de la boîte noire, il est possible qu'une solution $f(x)$ existe, et ce même si l'évaluation de la contrainte est violée $c(x) < 0$. Selon la terminologie de Le Digabel et Wild, il s'agit de contraintes de type QR*K, soit l'ensemble des contraintes connues, quantifiables et relaxables. On peut alors quantifier la violation de ces contraintes ainsi [12] :

$$h(x) = \begin{cases} \sum_{j \in J} (\max(c_j(x), 0))^2 & \text{si } x \in X \\ \infty & \text{sinon} \end{cases}$$

avec J l'ensemble des indices des contraintes et X l'ensemble des points. Ces contraintes seraient alors relaxables. Audet et Dennis [10] proposent une façon de gérer la relaxation avec la **barrière progressive**. Afin de déterminer une hiérarchie des points selon leur valeur de $h(x)$ et $f(x)$, on définit des notions de dominance entre deux points. On dit que le point

x réalisable domine le point y réalisable si l'expression suivante est respectée

$$f(x) < f(y) \implies x \prec_f y \text{ si } x, y \in \Omega.$$

Cependant, si x est non-réalisable, il domine le point y non-réalisable si

$$f(x) \leq f(y), h(x) \leq h(y) \implies x \prec_h y \text{ si } x, y \in \Omega/X$$

avec au moins une inégalité stricte.

L'algorithme de la barrière progressive admet deux solutions courantes, soient x^{feas} la solution réalisable dont la valeur de la fonction objectif est minimale, et x^{inf}, la solution irréalisable non-dominée dont la valeur de la fonction objectif est minimale et où $h(x^{\text{feas}})$ est inférieur à $h^k_{\max}$, un paramètre limite pour h spécifique à l'itération k.

Dans la Figure 2.3, on peut observer quelques points candidats. Les points pleins sont les points non dominés et les points vides sont dominés ou exclus par la barrière $h_{\max}$. Ceux présents dans la zone grise sont dominés par des points irréalisables pour lesquels $f(x)$ ou $h(x)$ sont inférieurs, ou encore leur valeur de leur fonction de violation $h(x)$ est supérieure à $h_{\max}$. L'interaction entre l'utilisation des concepts de barrière et les différents algorithmes d'optimisation sans dérivées sera vue lors de la description des algorithmes. La portion de

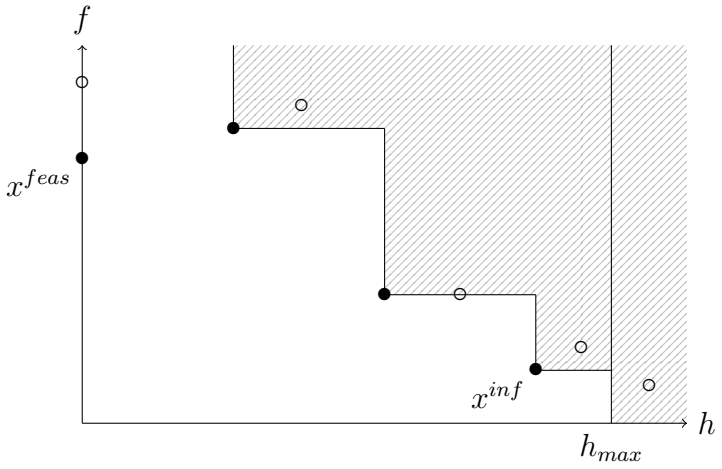


Figure 2.3 Barrière progressive, dominance et solutions courantes multiples

l'algorithme de la barrière progressive analogue à la sonde se déroule de la façon suivante. On utilise la nomenclature propre à CS à des fins de simplification :

— Si il y a un point $t \in P^k$ qui domine la solution réalisable ou la solution irréalisable, alors on met à jour la solution nouvellement dominée. On mets à jour δ^k selon un cas de succès et on met à jour la barrière $h_{\max}^{k+1} \leftarrow h(t)$ si t domine x^{inf}. Il s'agit d'une itération *dominante*.

— Si aucun nouveau point dominant n'est trouvé, mais qu'on est en présence d'un point $t \in P^k$ pour lequel $0 < h(t) < h(x^{inf})$, alors on met à jour la barrière $h_{\max}^{k+1} \leftarrow h(t)$. Il s'agit d'une itération *améliorante*.

— Sinon, on met à jour δ^k selon un cas d'échec. Il s'agit d'un échec.

2.7 Modèles quadratiques

Conn et Le Digabel [23] font mention de l'utilisation de modèles quadratiques en optimisation sans dérivées. Ces modèles peuvent servir de fonction substitut qui donne une approximation de la fonction et pour laquelle le coût de calcul est réduit. Pour obtenir un modèle quadratique, on considère la base naturelle de l'espace des polynômes de degré deux et moins.

$$\xi(x) = (\xi_0(x), \xi_1(x), ..., \xi_q(x))^T = \left(1, x_1, x_2, ..., x_n, \frac{x_1^2}{2}, \frac{x_2^2}{2}, ..., \frac{x_n^2}{2}, x_1x_2, x_1x_3, ..., x_{n-1}, x_n\right)^T$$

Cette base possède $q+1 = (n+1)(n+2)/2$ éléments. Le modèle $\tilde{f}$ de la fonction f est tel que $\tilde{f}(x) = \alpha^T \xi(x)$, $\alpha \in \mathbb{R}^{q+1}$. Pour obtenir ce modèle, un ensemble de points $Y = \{y^0, y^1, ..., y^p\}$ ayant $p+1$ éléments est nécessaire. On cherche alors à minimiser les différences entre les valeurs de la boîte noire évaluées aux points de Y et celles du modèle. Au long de l'exécution présente, les algorithmes procèderont à l'évaluation de la boîte noire à différents points. Les valeurs de la fonction objectif évaluée à ceux-ci sont enregistrées dans une cache, parmi laquelle pourront être choisis les $p = q$ points nécessaires à l'élaboration du modèle. Les points devront satisfaire la propriété qui valide le modèle :

$$B(\xi, Y)\alpha = f(Y)$$
$$f(Y) = (f(y^0, f(y^1), ..., f(y^p))^T$$
où
$$B(\xi, Y) = \begin{bmatrix} \xi_0(y^0) & \xi_1(y^0) & \dots & \xi_q(y^0) \\ \xi_0(y^1) & \xi_1(y^1) & \dots & \xi_q(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \xi_0(y^p) & \xi_1(y^p) & \dots & \xi_q(y^p) \end{bmatrix}.$$

Ce système peut être résolu seulement si $p = q$ et si la matrice est de plein rang. Dans le cas où $p \geq q$, on tentera de résoudre le problème de minimisation suivant :

$$\min_{\alpha \in \mathbb{R}} \quad \|B(\xi, Y)\alpha - f(Y)\|^2 \quad .$$

Dans le cas où $p < q$, on minimisera le même problème, mais en régularisant le problème à l'aide d'une interpolation dans le sens de la norme de Frobenius minimale [62, 30]. La norme de Frobenius pour une matrice A est définie par :

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{i,j}|}.$$

Il est possible de réarranger $\tilde{f}(x)$ de façon à l'illustrer comme une fonction quadratique en utilisant la notation précédente, mais en divisant le modèle en ses expressions linéaires et quadratiques :

$$\tilde{f}(x) = \alpha_L^T \xi_L(x) + \alpha_Q^T \xi_Q(x).$$

L'indice L dénote les termes linéaires et d'ordre 0 de $\xi(x)$, au compte de $n + 1$, soient les n variables et le terme de degré 0, $\xi_0(x) = 1$. L'indice Q dénote les termes quadratiques au compte de $(n+1)(n+2)/2 - (n+1) = \frac{n(n+1)}{2}$. Ainsi, on peut réécrire la quadratique en trois termes, soit le terme constant, le terme des composantes linéaires et le terme des composantes quadratiques :

$$\tilde{f}(x) = c + g^T x + \frac{1}{2} x^T H x.$$

Avec $g \in \mathbb{R}^n$ et la matrice $H \in \mathbb{R}^{n \times n}$ la matrice Hessienne symétrique du modèle. Avec un problème sous déterminé où $p < q$, on choisira le modèle tel que :

$$\begin{aligned} \min_{H \in \mathbb{R}^{n,n}} \quad & \|H\|_H^2 \\ \text{sujet à} \quad & c + g^T y^i + \frac{1}{2}(y^i)^T H(y^i) = f(y^i) \qquad i = 1, \ldots, p. \end{aligned}$$

On entend ici réduire autant que possible l'influence des termes quadratiques pour diminuer l'amplitude du modèle entre les points de Y utilisés. Par exemple, pour un problème de dimension n avec $p \leq (n + 1)$, on résoudra seulement la portion linéaire de $\alpha^T \xi(y^i) = f(y^i)$ pour ainsi laisser $h_{i,j} = 0, i, j = 1 \ldots n$ et donc $\alpha_Q = 0$.

CHAPITRE 3 OPPORTUNISME ET ORDONNANCEMENT

Dans le contexte d'optimisation de boîte noire, chaque évaluation de la fonction objectif requiert un temps d'attente non négligeable par l'utilisation de ressources informatiques ; il est indispensable de veiller à ce que l'algorithme choisi soit conçu avec des spécificités réduisant le nombre d'appels à la boîte noire. Dans cette optique, les étapes de POLL des algorithmes présentés précédemment sont revues afin d'y incorporer une stratégie opportuniste. Dans cette section, l'opportunisme est formellement défini, ainsi que les stratégies pouvant s'y coller. Enfin, on élabore comment cette stratégie peut être incorporée aux algorithmes identifiés aux sections 2.1 à 2.5

3.1 Stratégie opportuniste

Pour les algorithmes présentés à la section précédente, les analyses de convergence reposent sur une sous-suite d'échecs à l'étape de POLL, tandis que les étapes de SEARCH sont d'avantage accessoires et simplement destinées à accélérer la convergence en pratique. Afin d'adapter les méthodes de recherche directe aux problèmes pouvant être traités comme des boîtes noires, il est primordial d'étudier chaque aspect pouvant entraîner une réduction d'évaluation de la fonction objectif dans la POLL sans interférer avec ses propriétés. Ainsi, l'outil fourni est soutenu par une base théorique forte prouvant l'obtention d'une solution satisfaisant des conditions d'optimalité [75, 27, 9, 48, 51] et est muni d'un processus soigneusement conçu pour éviter les évaluations couteuses. Les quelques définitions suivantes seront nécessaires à l'introduction de la stratégie principale qui reste à être formellement introduite.

Définition 3.1.1 (Ensembles de recherche et de sonde)**.** *L'ensemble de recherche, désigné par S^k, est l'ensemble de points candidats pour l'étape de SEARCH. L'ensemble de sonde, désigné P^k, est l'ensemble des points candidats pour l'étape de POLL à l'itération k d'un algorithme de recherche directe.*

Par exemple, pour CS, $S^k := \{\}$ et $P^k := \{x^k \pm \delta^k e_i \text{ avec } i \in \{1, 2, \ldots, n\}\}$.

Définition 3.1.2 (Diminutions simple et suffisante)**.** *Pour un algorithme de recherche directe, on dit qu'un candidat de P^k entraîne une diminution simple si*

$$\exists\ \tau\ \in P^k \text{ tel que } f(\tau) < f(x^k). \tag{3.1}$$

Analoguement, on dit qu'un candidat dans P^k entraîne une diminution suffisante si

$$\exists\ \tau\ \in P^k \text{ tel que } f(\tau) < f(x^k) + \rho \tag{3.2}$$

où $\rho \geq 0$.

Ces définitions peuvent s'étendre à un autre ensemble que P^k, notamment S^k pour une SEARCH. La présence d'un candidat dans P^k entraînant une diminution simple peut être un critère de succès pour une étape de POLL. Ce n'est pas toujours le cas, tel que présenté précédemment dans GSS, où le critère de succès d'une POLL requiert une diminution suffisante. Outre les diminutions simples et suffisantes, on introduit un autre critère caractérisant un candidat.

Définition 3.1.3 (Meilleure diminution)**.** *Pour une étape k de POLL d'un algorithme de recherche directe, on dit qu'un candidat τ de P^k entraîne la meilleure diminution de la fonction objective si*

$$f(\tau) < f(x^k) \text{ et } f(\tau) \leq f(p)\ \forall\ p \in P^k.$$

Le cas traité ici est l'obtention d'un optimum en utilisant un algorithme dans sa forme séquentielle. En pratique, il s'agit d'évaluer les points de façon séquentielle ou parallèle. Cependant, dans le cas d'optimisation de boîte noire, il est nécessaire de choisir si l'utilisation des ressources parallèles sont destinées à l'évaluation de la boîte noire, ou si elles sont dédiées à une version parallèle de l'algorithme [44, 11]. Il est intéressant de paralléliser un algorithme pour un ensemble de tâches indépendantes, par exemple dans l'optimisation de paramètres algorithmiques [7]. Dans le cas où la boite noire est lourde en calculs et nécessite de grandes ressources en parallèle pour une évaluation unique, il est envisageable d'utiliser un algorithme dans sa version en séquentielle. L'utilisation d'un algorithme de recherche directe séquentiel signifie qu'au moment initial d'une étape de POLL, $2n$ évaluations en série de la boîte noire sont possiblement à venir. Dans les algorithmes présentés, pour considérer la POLL courante comme un succès, on n'exige pas la détermination d'un candidat entraînant la meilleure diminution au sens de la définition 3.1.3 ; elle exige la détermination d'un candidat apportant une diminution simple. Ainsi, pour le premier candidat satisfaisant l'équation (3.1), l'algorithme pourrait passer à l'étape suivante en considérant la présente comme un succès. Alternativement, on pourrait aussi continuer à évaluer la fonction objectif aux autres points de l'ensemble de sonde, de façon à identifier le candidat entraînant la meilleure diminution. Ces décisions algorithmiques sont au cœur de l'étude présente.

Définition 3.1.4 (Stratégie opportuniste simple)**.** *La stratégie opportuniste désigne l'arrêt prématuré de l'étape SEARCH ou POLL courante d'un algorithme de recherche directe dès*

la première obtention d'un point τ satisfaisant le critère de succès de l'algorithme, soit la diminution simple ou la diminution suffisante.

En opposition à la stratégie opportuniste, on définit l'idée consistant à évaluer tous les points de l'ensemble P^k ou S^k.

Définition 3.1.5 (Sonde complète et recherche complète)**.** *La sonde complète désigne l'évaluation de la fonction objectif à tous les points candidats de P^k de l'étape de POLL sans égard à l'identification encourue d'un candidat entrainant une diminution simple ou suffisante.*

La recherche complète désigne l'évaluation de la fonction objectif à tous les points candidats de S^k de l'étape de SEARCH sans égard à l'identification encourue d'un candidat entrainant une diminution simple ou suffisante.

L'opportunisme pour les étapes de SEARCH, quoique figurant comme paramètre pour certaines implémentations [15], est laissée de côté pour la suite du présent travail, compte tenu de l'extrême versatilité de l'étape de SEARCH. Il en va de même pour la descente linéaire encourue dans IMFIL puisqu'elle n'y est pas applicable. Ces définitions concordent avec les mentions de l'opportunisme dans la littérature. Coope et Price [27] sont les premiers à utiliser le terme d'opportunisme. De prime abord, ils identifient deux cadres de travail pour des algorithmes d'optimisation sans-contraintes dont les points sont limités à des maillages, soit les cadres A et B. Les deux cadres peuvent être résumés ainsi.

Algorithme 6 Cadre algorithmique opportuniste (A) de Coope et Price

$f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x^0 le point de départ.

1. Déterminer un pas δ^k et une base positive ordonnée $D^k := d_1^k, d_2^k, \ldots, d_l^k$.
   Fixer $i \leftarrow 1$.
2. Déterminer une direction de descente $d_i^k \in D^k$ qui emmène une diminution simple de la fonction. Si une telle direction existe, mettre à jour la solution courante x^k. Sinon, essayer avec la prochaine direction dans la base positive ordonnée en incrémentant $i \leftarrow i + 1$. Si aucun i ne satisfait la condition, passer à l'étape 3.
3. Déterminer un ensemble de points finis avec une procédure arbitraire (SEARCH). Si cette procédure produit une diminution simple de la fonction objective, mettre à jour la solution. Retour à 1.

Algorithme 7 Cadre algorithmique non-opportuniste (B) de Coope et Price

$f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x^0 Le point de départ

1. Déterminer un pas δ^k et une base positive ordonnée D^k.
   Fixer $i \leftarrow 1$.
2. Déterminer la direction menant à la meilleure descente d_i^k dans D^k. Faire une recherche linéaire, car cette direction mène à un succès. Mettre x^k à jour et recommencer. Si aucune direction de descente n'est trouvée, un minimum sur le maillage est atteint.
3. Déterminer un ensemble de points finis avec une procédure arbitraire (SEARCH). Si cette procédure produit une diminution simple, mettre à jour la solution. Retour à 1.

Les auteurs stipulent que le cadre algorithmique B est une adaptation du A et enchaînent avec la démonstration que la procédure répétée de A mène ultimement à la détermination d'un point stationnaire de la fonction f. Ils remarquent que l'algorithme B est beaucoup plus restreint que l'algorithme A. Par contre, c'est la routine primitive de A qui assure la convergence, malgré que le candidat entraînant la meilleure diminution n'est pas obligatoirement identifié. C'est ici qu'ils mentionnent l'opportunisme associé à ce concept algorithmique, la première instance dans la littérature autant que nous sachions.

> «Pour le cadre algorithmique A, la convergence peut être démontrée seulement pour une sous-séquence de minimum locaux du maillage. Ceci est dû au fait que la recherche effectuée à l'étape 2 est opportuniste ; le premier membre rencontré dans D qui donne une descente (diminution simple) mène à un nouvel itéré.»

Le concept de l'opportunisme, parfois désigné autrement dans la littérature, impacte l'analyse de convergence. Dans [75], Torczon introduit les mouvements exploratoires, sur lesquels l'analyse de convergence de GPS, et par extension CS, reposent. Deux résultats sont identifiés. En premier lieu, on statue que la convergence globale est assurée pour une méthode de GPS sur une fonction objectif f continue différentiable et sur un ensemble avoisinant un ensemble compact. La conclusion étant que sous ces conditions, une série infinie d'évaluations mène à un point où la norme du gradient est nulle.

$$\liminf_{k\to\infty} \left\|\nabla f(x^k)\right\| = 0. \tag{3.3}$$

Ce résultat peut être renforcé par des hypothèses plus strictes sur la norme des colonnes de la matrice génératrice Z, sur le paramètre d'ajustement du maillage τ selon la notation de [75] et, plus important encore, les hypothèses de mouvements exploratoires. Initialement,

ces hypothèses se limitaient à

— La direction d est choisie dans $D = GZ$ et la longueur est contrôlée par δ^k.

— Si un des points existants dans P^k entraîne une diminution simple, alors un mouvement exploratoire produira un pas qui entrainera une diminution simple.

Le résultat de convergence s'obtient en conservant la première hypothèse et en remplaçant la deuxième par la suivante :

— Si un des points existants dans P^k entraîne une diminution simple, alors un mouvement exploratoire produira un pas qui entrainera la meilleure descente.

Le résultat de convergence renforcé assure que :

$$\lim_{k\to\infty} \left\|\nabla f(x^k)\right\| = 0. \tag{3.4}$$

L'hypothèse sur les mouvements exploratoires est uniquement applicable si tous les points de P^k sont évalués. Il en découle que la convergence est influencée par l'opportunisme. L'utilisation de la stratégie opportuniste contredit l'hypothèse nécessaire pour la convergence forte, ainsi une implémentation de GPS opportuniste assure seulement une convergence dans le sens de la limite inférieure. Toujours dans [75], l'auteure adapte CS au cadre énoncé pour une méthode de GPS, entraînant ainsi le résultat de convergence applicable à CS. Une situation semblable est répétée dans [51] pour GSS, soit que la convergence plus forte de l'équation (3.3) est atteignable avec la prémisse que x^{k+1} soit le candidat procurant la meilleure descente. Les algorithmes présentés dans la section 3 ne vont pas dans les détails de leurs implémentations et la présence de l'opportunisme y est laissée floue. Conn, Scheinberg et Vincente [26] évoquent explicitement la présence de l'opportunisme dans les cadres algorithmiques de méthodes de recherche directe directionnelles qu'ils utilisent.

3.2 Ordonnancement

L'utilisation de l'opportunisme entraîne que l'ordre des points dans l'ensemble de sonde P^k prend soudainement un rôle important. Dans l'optique de réduire le nombre d'appels à la boîte noire, on s'intéresse ici à la possibilité de réordonner les points de P^k de façon à évaluer les candidats possiblement intéressants en premier lieu. Dans cet ordre d'esprit, Custodio et Vincente [31] déterminent des indicateurs de descente, c'est-à-dire une direction avec un potentiel de descente intéressant identifié à l'aide des informations obtenues à présent sur le problème.

Définition 3.2.1 (Ordonnancement)**.** *L'ordonnancement est la permutation des points de la*

liste de points, soit des colonnes de P^k, les ensembles de sonde et de recherche. Une stratégie guidant un ordonnancement est désignée comme stratégie d'ordonnancement.

Dans leur étude, Custodio et Vincente utilisent une estimation du gradient de la fonction f, appelé *gradient simplexe* $\nabla_s f(x)$. Cette approximation est calculée à l'aide des évaluations précédentes, compte tenu de certaines contraintes sur la géométrie de l'ensemble de points choisis [24]. Ils utilisent la distance angulaire minimale entre $-\nabla_s f(x)$ et d_i comme stratégie d'ordonnancement. Les résultats numériques montrent que, sur une version opportuniste de CS, cet ordre réfléchi des points dans P^k est bénéfique lorsque comparé à un ordre statique, surtout si combiné à d'autres stratagèmes identifiés par les auteurs. Un premier stratagème consiste en l'utilisation des informations obtenues sur chaque point, en opposition à utiliser seulement les points garantissant un succès. Un deuxième, proposée dans [44], porte sur la limitation de l'expansion de la longueur du pas, soit de limiter une série de p succès consécutifs issus de la même direction. Ces deux stratagèmes se retrouvent à être conceptuellement très près des idées derrière deux des stratégies d'ordonnancement présentées à la sous-section 3.2. Dans [29], Custodio, Dennis et Vincente [29] réutilisent la stratégie dans le contexte d'optimisation non lisse et observent toujours une amélioration en comparant avec un ordre arbitraire restant inchangé au cours de l'optimisation. Abramson, Audet et Dennis [2] montrent quant à eux qu'il est possible d'élaguer l'ensemble P^k de façon à n'avoir qu'une seule évaluation par étape de POLL fructueuse en utilisant les informations sur les signes des dérivés. Lorsque Audet et Dennis introduisent MADS [9], ils spécifient que les ordres des directions dans leurs ensembles P^k sont aléatoirement déterminées. Une autre mention de l'opportunisme est présente dans [70], où on y étudie l'impact de celle-ci jumelé à une stratégie d'ordonnancement induite par l'utilisateur dans le contexte d'optimisation de boîtes grises. L'idée de ces travaux est de limiter le nombre d'évaluations nécessaires dans une étape de POLL courante. Cependant, ces mêmes évaluations peuvent être utiles à la répartition d'un modèle ou encore à l'identification du comportement en situation de barrière progressive. Ainsi, il est à noter que l'objectif de réduire le nombre d'itérations par sonde peut être en contradiction avec l'objectif de la résolution complète, qui peut faire intervenir d'autres facteurs, notamment diverses étapes de SEARCH.

Dans le cadre de ce travail, on tente de déterminer si le choix de la stratégie d'ordonnancement a un impact observable sur la performance de la stratégie opportuniste. On procède alors avec la définition du concept de la précédence entre deux points, qui sera au coeur des définitions des différentes stratégies d'ordonnancement.

Définition 3.2.2 (Précédence)**.** *On dit que $\tau_1 \in P^k$ précède $\tau_2 \in P^k$, dénoté $\tau_1 \prec \tau_2$, lorsque τ_1 apparait avant τ_2 dans l'ordre des évaluations à effectuer dans P^k, l'ensemble de sonde*

ordonné.

La relation de précédence est transitive :

$$\forall\ \tau_1, \tau_2, \tau_3,\ (\tau_1 \prec \tau_2 \wedge \tau_2 \prec \tau_3) \implies \tau_1 \prec \tau_3.$$

Les sous-sections suivantes détaillent les différentes stratégies d'ordonnancement qui seront comparées dans la suite du travail. Chaque candidat à l'étape de sonde τ_i est défini par un itéré courant x^k, une longueur de pas δ^k,une direction $d_i \in D^k$ pour $i = 1, \ldots, |D^k|$. Certaines stratégies d'ordonnancement utilisent des caractéristiques des points τ_i alors que d'autres utilisent des caractéristiques sur leurs directions correspondantes $d_i \in D^k$.

3.2.1 Ordonnancement déterministe

Une stratégie d'ordonnancement déterministe est une stratégie qui impose à P^k un ordre de point indépendant des facteurs de la résolution (fonction objectif, contraintes, historique des points évalués, etc.) ou d'un facteur externe. L'intérêt de comparer une telle stratégie est de pouvoir observer les impacts d'une application stratégie opportuniste qui n'utilise aucune information sur le problème. La stratégie d'ordonnancement déterministe choisie est la stratégie *lexicographique* [28].

Définition 3.2.3 (Ordonnancement lexicographique)**.** *Soient $\tau_i, \tau_j \in P^k$ deux points distincts de l'ensemble de sonde avec leurs directions correspondantes $d_i = (a_1, a_2, \ldots, a_n)$ et $d_j = (b_1, b_2, \ldots, b_n) \in \mathbb{R}^n$. On définit la précédence lexicographique $\prec_l$ de façon à ce que $\tau_i \prec_l \tau_j$ si et seulement si il existe un indice $\ell \in \{1, 2, \ldots, n\}$ tel que*

$$a_m = b_m \text{ pour } m \in \{1, 2 \ldots \ell - 1\} \text{ et } a_\ell < b_\ell.$$

Une étape de sonde opportuniste ordonnancée de façon lexicographique dépend des étapes de sonde précédentes. Les directions présentent des plus petits nombres dans leurs premières composantes sont avantagées par cet ordonnancement, alors que l'amélioration possible de $f(x)$ suivant ces directions peut diminuer au courant de la résolution. Tout dépendamment de la nature de $f(x)$, l'amélioration possible dans les directions priorisées peut aussi être restreinte ou inexistante.

3.2.2 Ordonnancement aléatoire

Une stratégie d'ordonnancement aléatoire est une stratégie qui ordonne les points de P^k de façon aléatoire. L'intérêt de comparer une telle stratégie est toujours de pouvoir observer les

impacts de l'application d'une stratégie d'ordonnancement qui n'utilise aucune information sur le problème. La stratégie aléatoire implique l'indépendance complète du déroulement de la sonde quant aux évaluations qui l'ont précédée. La grande différence pratique avec la stratégie lexicographique est la fréquence avec laquelle certaines directions sont explorées. Aucune direction n'est artificiellement avantagée par la nature de ses composantes vectorielles. Cette stratégie n'est évidemment pas déterministe.

Définition 3.2.4 (Ordonnancement aléatoire)**.** *Soient $\tau_i, \tau_j \in P^k$ deux points distincts de l'ensemble de sonde, et r_i, r_j deux nombres aléatoires tels que $Pr[r_i > r_j] = 0.5$ et $Pr[r_i = r_j] = 0$. On définit la précédence aléatoire $\prec_a$ de façon à ce que $\tau_i \prec_a \tau_j$ si et seulement si*

$$r_i > r_j.$$

Dans [69], l'auteur exprime la différence avec la stratégie lexicographique et justifie sa présence en option dans son implémentation [39].

> «Quand les évaluations sont effectuées de façon asynchrone, l'ordonnancement aléatoire réduit la tendance des directions de recherche initiales à être priorisées.»

3.2.3 Ordonnancement en fonction de la direction du dernier succès

Les stratégies énoncées auparavant n'utilisent pas les informations sur le problème ni les évaluations préalablement effectuées. Une façon peu couteuse de combler ce manque est d'utiliser seulement les informations qui découlent du dernier succès identifié à l'étape de POLL ou de SEARCH fructueuse précédente. Pour ce faire, on tiendra compte de la direction correspondante au dernier candidat ayant été un succès. Il est ainsi possible d'ordonner les points de l'ensemble de sonde P^k en fonction de l'angle formé par leurs directions correspondantes et la dernière direction de succès, tel que

Définition 3.2.5 (Ordonnancement en fonction de la direction du dernier succès)**.** *Soient $\tau_i, \tau_j \in P^k$ deux points distincts de l'ensemble de sonde, $d_i, d_j \in D^k$ leurs directions correspondantes, t^{ds}, le point ayant entrainé le succès de la dernière sonde fructueuse et d^{ds} sa direction correspondante. On définit la précédence en fonction de la direction du dernier succès $\prec_d$ telle que $\tau_i \prec_d \tau_j$ si et seulement si*

$$\frac{d^{ds} \cdot d_i}{\|d^{ds}\| \, \|d_i\|} > \frac{d^{ds} \cdot d_j}{\|d^{ds}\| \, \|d_j\|}.$$

Cette stratégie est inspirée de la sonde dynamique proposée par Audet et Dennis [9], une stratégie consistant en un pas supplémentaire dans la dernière direction de succès, une stratégie

qui entre cependant dans la définition de SEARCH. Le produit scalaire entre d^{ds} et d_i, d_j permet d'ordonnancer en fonction de la valeur du cosinus de l'angle formé par les deux directions. S'il n'y a aucune direction de dernier succès, l'ordonnancement est lexicographique.

3.2.4 Ordonnancement selon un modèle

On cherche à guider l'ordonnancement de façon à ce que le point le plus prometteur soit considéré en premier. La stratégie de la section 3.2.3 utilise uniquement les informations d'une seule itération de l'algorithme. Afin d'utiliser le plus d'information possible, on doit avoir recours à plus d'outils. On peut avoir recours à des modèles dynamiques, tels que ceux décrits à la section 2.7, à des modèles statiques de la boîte noire à optimiser. Ces modèles fournissent des approximations de $f(x)$ en une fraction du coût computationnel de la boîte noire. On ordonne alors les points en fonction de leur approximation avec le modèle choisi, de façon à placer les points les plus prometteurs au début de l'ordre d'évaluation.

Définition 3.2.6 (Ordonnancement selon un modèle)**.** *Soient $\tau_i, \tau_j \in P^k$ deux points de l'ensemble de sonde et $\tilde{f}(x)$, une fonction substitut de $f(x)$. On définit la précédence en fonction d'un modèle $\prec_m$ tel que $\tau_i \prec_m \tau_j$ si et seulement si*

$$\tilde{f}(\tau_i) < \tilde{f}(\tau_j).$$

Évidemment, la performance de la stratégie d'ordonnancement dépendra largement du caractère du modèle. La méthode développée par Custodio et Vincente [31] peut être perçue comme appartenant à cette catégorie. Malgré le fait que ces auteurs n'élaborent pas une approximation de la fonction, ils le font avec son gradient.

Aucune stratégie précédemment définie n'intervient avec la gestion des contraintes QR*K selon la nomenclature proposée par [53]. En situation de barrière progressive, la condition de précédence $\prec_m$ ne suffit pas pour déterminer la stratégie d'ordonnancement à adopter lorsqu'on est en présence d'un modèle pour chaque contrainte relaxable. Audet et Hare [12] proposent l'ordonnancement suivant, utilisant la notion de dominance propre à la barrière progressive résumée à la section 2.6.

Définition 3.2.7 (Ordonnancement selon un modèle)**.** *Soient $\tau_i, \tau_j \in P^k$ deux points de l'ensemble de sonde, $\tilde{f}(x)$, une fonction substitut de $f(x)$ et $\tilde{h}(x)$, une fonction substitut de $h(x)$, la fonction de violation de contrainte. On définit la précédence en fonction d'un modèle $\prec_m$ telle que $\tau_i \prec_m \tau_j$ si et seulement si*

i. $\tilde{h}(\tau_i) < \tilde{h}(\tau_j)$ quand les deux sont dominants

ii. τ_i *est dominant et* τ_j *ne l'est pas*

iii. $\tilde{h}(\tau_i) < \tilde{h}(\tau_j)$ *quand les deux sont non-dominants*

iv. $\tilde{f}(\tau_i) < \tilde{f}(\tau_j)$ *quand les deux sont non-dominants et que* $\tilde{h}(\tau_i) = \tilde{h}(\tau_j)$.

Les points $\tau \in P^k$ perçus par les modèles comme étant dominants quant à x^{feas}, c'est-à-dire que $\tilde{f}(\tau) < f(x^{inf})$ et que $\tilde{h}(\tau) = 0$, sont priorisés. L'ordonnancement à même ce sous-ensemble de points est fait selon la même précédence qu'à la définition 3.2.6. Les points $\tau \in P^k$ pour lesquels $\tilde{f}(\tau) < f(x^{inf})$ et $\tilde{h}(\tau) \neq 0$, soient les points perçus comme dominant la solution irréalisable courante, viennent ensuite. Ce sous-ensemble de points est lui-même ordonné selon la précédence de la définition 3.2.6, en utilisant $\tilde{h}(x)$ afin de les ordonner selon la plus petite violation des contraintes possible. Les points restants sont ordonnés dans leur sous-ensemble selon la même précédence qu'à la définition 3.2.6, mais avec en remplaçant $\tilde{f}(x)$ par $\tilde{h}(x)$.

Le choix d'avantager les points dominants et qui violent le moins possible les contraintes est arbitraire. En pratique, il existe des problèmes pour lesquels un utilisateur aurait avantage à ordonnancer selon un autre critère. Par exemple, en priorisant les points qui violent le plus les contraintes, on pourrait obtenir une meilleure géométrie des points dans l'espace contraint et utiliser ces informations pour améliorer les modèles des contraintes $\tilde{h}_i$.

L'ordonnancement à l'aide de modèles comporte en effet un aspect paradoxal. En ordonnançant ainsi, on détermine possiblement un point qui amènera un succès, ce qui rend la POLL susceptible de contenir qu'une seule évaluation. Quoiqu'intéressant de prime abord, ce nombre limité de points évalués ne bénéficie pas au modèle lui-même. Conn et Le Digabel [23] insistent sur la bonne répartition des points pour la précision des modèles en optimisation sans dérivées, constat repris par Conn, Scheinberg et Vincente [25, 24]. D'autant plus que, le nombre de points environnants respectant les conditions géométriques pour une bonne répartition sera limité dans le cas de plusieurs étapes de POLL fructueuses d'une seule évaluation, produisant une série de centres de sonde alignés. Ces aspects sont sujets à intervenir dans la performance de la stratégie, surtout en l'absence de SEARCH pouvant inhiber ces lacunes.

3.2.5 Ordonnancement omniscient

Suivant les résultats favorables de Custodio et Vincente [31] envers un ordonnancement tirant avantage des informations sur le problème, on en vient à imaginer un ordonnancement idéal dans ce sens. Il s'agit d'utiliser une stratégie qui dictera un ordre de points pour lequel le premier sera toujours le meilleur candidat présent dans la liste. Chaque étape de sonde de l'algorithme opportuniste possédant au moins un point entraînant un succès pourrait ainsi

être réduite à une seule évaluation. On désignera cette stratégie *l'ordonnancement omniscient*. Cette stratégie est introduite uniquement à des fins de comparaison. Son implémentation en pratique est impossible à coût raisonnable.

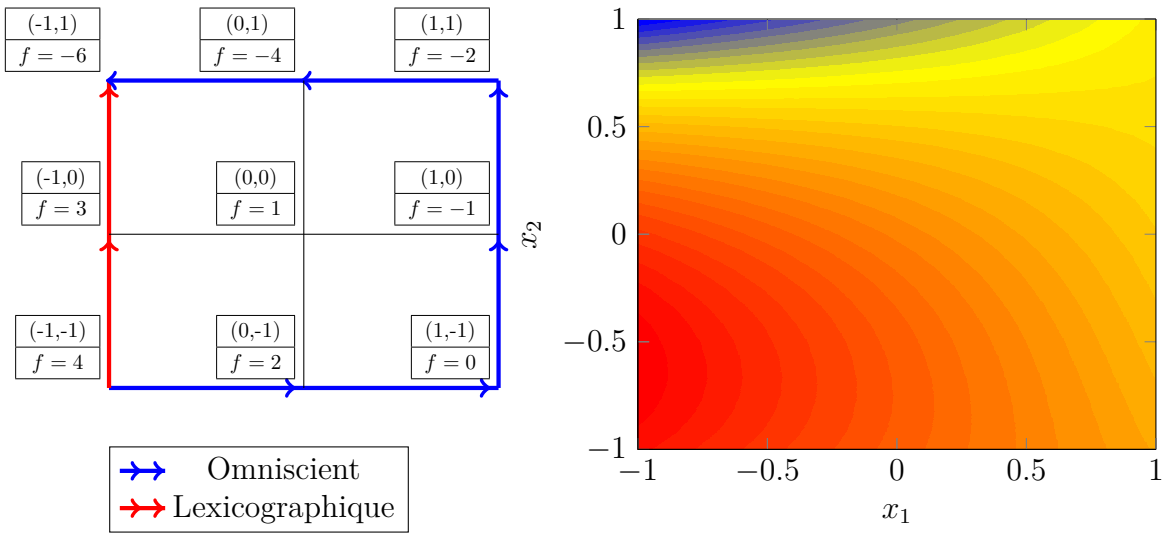
Cet ordonnancement est idéal pour des problèmes d'optimisation sans contraintes relaxables, convexes et lisses. Par contre, cet idéal ne se transmet pas dans toutes les disciplines de l'optimisation. En pratique, on peut s'attendre à ce que l'ordonnancement omniscient démontre une très bonne performance. Cependant, il est facile d'imaginer un problème pour lequel la fonction objectif est une boîte noire possédant plusieurs discontinuités et optimums locaux. L'exemple suivant illustre le point pour un problème lisse :

$$\min_x \{x_2(x_2-1)(1-x_1) + (x_2^2-1)(2x_1-1) + x_2(x_2+1)(x_1-2) \ : \ x \ \in [-1,1]\}.$$

La solution optimale se trouve à $x^* = (-1, 1)$ avec $f(x^*) = -6$. Dans un premier cas, la résolution est effectuée avec l'algorithme CS opportuniste avec ordonnancement lexicographique à partir du point de départ $x^0 = (-1, -1)$ et le pas initial $\delta^0 = 1$. En priorisant les points ayant -1 comme première composante, l'algorithme se dirigera vers $x^1 = (-1, 0)$ plutôt que $(0, -1)$. La deuxième itération sera dans la même direction, et l'algorithme aura atteint la solution x^* en seulement 3 évaluations.

Dans un deuxième cas, la résolution est effectuée avec l'algorithme CS opportuniste avec ordonnancement omniscient. Sachant à l'avance que $f(0, -1) < f(-1, 0)$, l'algorithme se dirigera vers $x^1 = (0, -1)$. En faisant ainsi, l'algorithme s'est éloigné de la solution et il nécessitera 5 autres étapes de sonde pour atteindre x^*, totalisant 7 évaluations. La figure 3.1 illustre les détails des résolutions ainsi qu'un aperçu du caractère de la fonction.

L'ordonnancement omniscient est impossible sans la connaissance au préalable de la valeur de la fonction objectif à chaque point de P^k. La simulation du comportement de la stratégie peut être obtenue avec l'utilisation d'un ordonnancement selon un modèle, avec la fonction objectif elle-même comme modèle $\tilde{f}(x) := f(x)$. En terme d'évaluations de la fonction objectif, l'algorithme opportuniste avec ordonnancement omniscient aura le même coût que l'algorithme non opportuniste. Cependant, la performance théorique de la stratégie omnisciente sera observable en considérant seulement $\frac{1}{|P^k|}$ pour chaque k correspondant à une étape de POLL fructueuse. Il est ainsi assuré que la stratégie omnisciente sera au moins aussi performante que la sonde complète. Cette performance serait en mesure de fournir une amélioration pour les autres stratégies. C'est d'autant plus vrai pour la stratégie d'ordonnancement à l'aide de modèles, dont les idées de base sont d'ordonnancer selon la meilleure diminution possible prédite par un modèle.



(a) Chemins pris par les algorithmes (b) Courbes de niveau de $f(x)$

Figure 3.1 Exemple de mauvaise performance de la stratégie omnisciente

3.2.6 Ordonnancement inverse-omniscient

Si l'opportunisme est susceptible d'influencer positivement la performance d'algorithmes d'optimisation, alors il a aussi le pouvoir de l'influencer négativement. Afin de pouvoir observer l'ampleur de l'impact inverse possible, il est possible d'identifier une stratégie d'ordonnancement qui fera exactement l'opposé de l'ordonnancement omniscient. On désignera cette stratégie d'ordonnancement comme *l'ordonnancement inverse-omniscient*. Pour les itérations de POLL consistant en un succès, les points sont ordonnés de façon à forcer l'évaluation des points n'entrainant pas de succès en premier, pour ensuite évaluer le point entrainant l'amélioration la plus modeste possible.

Encore une fois, il s'agit d'une stratégie avec aucune application pratique. Son unique utilité est de pouvoir simuler, selon notre intuition, l'obtention du pire ordre possible suivie d'une interruption prématurée de l'étape de POLL. En pratique, il est possible d'obtenir ce comportement en utilisant l'ordonnancement selon un modèle, utilisant comme modèle une fonction retournant l'opposé de la fonction objectif $\tilde{f}(x) := -f(x)$. Il est difficile d'estimer le coût en terme d'évaluations de la fonction objectif, étant donné que le nombre d'itérations nécessaire à la convergence risque d'être plus élevé.

Il est important de noter que la stratégie inverse-omnisciente n'est pas universellement la pire possible pour tout problème donné, au même titre que la stratégie omnisciente n'est pas la meilleure. Il s'agit de la pire dans l'optique où on veut minimiser le nombre d'évaluations de

chaque étape de POLL, sans considérer les impacts possibles sur les étapes de SEARCH d'un algorithme.

3.3 Mise en place de l'opportunisme

En forçant l'évaluation séquentielle de la fonction sur les candidats de P^k et S^k, l'opportunisme demandent que les cadres algorithmiques soient plus restrictifs et limités à des formes séquentielles. L'application de la stratégie est donc limitée à certains algorithmes. Les algorithmes de méthodes de régions de confiance [26, 22] en optimisation sans dérivées possèdent un seul candidat par itération de l'algorithme. D'autres ne peuvent être vus de façon non opportuniste sans les dénaturer. Prenons, par exemple, les transformations géométriques générant les candidats à chaque étape d'une itération de Nelder Mead [64]. Chacune de ces transformations pourrait être effectuée préalablement de façon à identifier l'ensemble des candidats possiblement rencontrés lors du déroulement d'une itération de l'algorithme, créant ainsi une liste de points à ordonnancer. Cependant, ceci mettrait en péril le bon fonctionnement de la méthode en détruisant les propriétés qui en font une méthode attrayante. Pour l'algorithme de Nelder Mead, si on considère une liste de points, on aura un ordonnancement sans aucune flexibilité.

La stratégie opportuniste est incompatible avec certains algorithmes, ce qui justifie la tâche d'identifier préalablement un ensemble d'algorithmes compatibles. Le critère de compatibilité principal consiste en l'existence d'une étape où une liste de points qui doit être évaluée, soit les étapes de SEARCH et de POLL ainsi que dans leurs équivalents dans IMFIL. L'opportunisme ne doit pas modifier les hypothèses émises lors des analyses de convergence de ces algorithmes. Il est vérifié à la section 3.1 que c'est le cas pour les algorithmes étudiés dans le cadre de ce travail. La section qui suit détaille les modifications nécessaires à la mise en place de l'opportunisme dans les étapes de POLL pour les algorithmes de recherche directe identifiés préalablement.

La stratégie opportuniste réfère à l'arrêt prématuré des évaluations dans la sonde. Jusqu'ici, seulement la stratégie opportuniste la plus intuitive a été évoquée, soit celle de l'arrêt à l'obtention d'un premier succès. La définition de succès varie selon les algorithmes, et diverses stratégies opportunistes peuvent être élaborées sur mesure en fonction de l'étape de POLL de la méthode à l'étude. Dans cette section, on définit les différentes versions de l'opportunisme applicables aux étapes de POLL de CS, GPS, GSS et MADS et dont la performance est étudiée.

Définition 3.3.1 (Stratégie opportuniste au $p^{\text{ème}}$ succès)**.** *La stratégie opportuniste au $p^{ème}$ succès désigne l'arrêt prématuré de l'étape de SEARCH ou de POLL courante d'un algorithme*

de recherche directe après l'obtention de p points satisfaisants le critère de succès de l'algorithme, soit la diminution simple ou la diminution suffisante.

La stratégie opportuniste au $p^{\text{ème}}$ succès consiste à attendre l'évaluation de p points de sonde qui satisfont le critère de succès avant de procéder à la prochaine itération de l'algorithme avec le meilleur candidat évalué comme nouvel itéré courant x^k . Cette variation est destinée à modérer l'utilisation de la stratégie opportuniste simple et à la rendre plus robuste. Par exemple, un point de sonde entrainant une diminution minime, par exemple le fruit d'un bruit stochastique présent dans la boîte noire, pourrait être considéré à tort comme un nouveau succès par un algorithme opportuniste.

L'opportunisme au $p^{\text{ème}}$ succès propose de décaler l'arrêt de la sonde. Ce décalage dépend du nombre de points entrainant un succès identifiés en cours de celle-ci. Il est possible d'utiliser d'autres mécaniques pour prolonger la sonde tout en étant opportuniste.

Définition 3.3.2 (Stratégie opportuniste avec un minimum de q évaluations)**.** *La stratégie opportuniste minimum q évaluations désigne l'arrête prématuré de l'étape de* SEARCH *ou de* POLL *courante d'un algorithme de recherche directe après q évaluations si un point τ satisfaisant le critère de succès de l'algorithme est évalué au cours des q premières évaluations. Dans le cas contraire, l'arrêt prématuré se fait à la première obtention d'un point τ satisfaisant le critère de succès de l'algorithme*

La stratégie opportuniste avec un minimum de q évaluations est une autre déclinaison de l'opportunisme. Elle permet un autre type de contrôle sur le décalage de l'interruption de la sonde qui peut être utile selon la méthode utilisée. Par exemple, une étape de SEARCH dont le fonctionnement dépends de l'évaluation d'un ensemble de points bien dispersés pourrait bénéficier d'une sonde avec un minimum d'évaluations, sans nécessité que plus d'un des points entraine un succès.

Les différentes stratégies opportunistes sont applicables aux étapes de POLL des algorithmes de recherche suivant l'algorithme 8.

Algorithme 8 POLL opportuniste

Avec $f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x^0 le point de départ

1. POLL

Ordonner P^k avec la stratégie d'ordonnancement choisie.

for $t \in P^k$ **do**

if Critère d'arrêt opportuniste de la sonde **then**

$x^{k+1} \in \arg\min\{f(t) : t \in P_e^k\}$ avec P_e^k les points de l'ensemble de sonde qui ont été évalués.

Mise à jour de δ^k et Δ^k selon un succès de l'étape

Fin de la sonde, poursuivre à la terminaison.

end if

$x^{k+1} \leftarrow x^k$

Mise à jour de δ^k et Δ^k selon un échec de l'étape

end for

Fin de la sonde, poursuivre à la terminaison.

Ce cadre algorithmique n'est pas applicable à la cinquième méthode présentée, soit IMFIL. Dans son œuvre élaborant les principes de sa méthode [48], Kelley commente la relation de IMFIL à l'opportunisme.

> «La version opportuniste de CS est de plusieurs façons plus naturelle que la version non opportuniste (pensez à un animal cherchant de la nourriture). Toutefois, IMFIL est une méthode conceptuellement non opportuniste. La raison de ce choix est que IMFIL utilise un échantillonnage pour approximer un gradient et donc une direction de recherche et pour construire un modèle de f.»

L'utilisation de l'opportunisme dans IMFIL nécessite une modification de la méthode. Premièrement, la notion de succès pour IMFIL est différente de celle établie pour les autres algorithmes identifiés. On dira que la sonde du stencil comporte un succès si il existe un point t dans $P^k := \{x^k + h^k v_i : v_i \in V\}$ pour lequel $f(t) < f(x^k)$. Pour respecter la définition 3.2.1 de l'opportunisme sans compromis, on devrait terminer la sonde du stencil au premier succès. On désignera cette méthode comme IMFIL-op pour IMFIL avec opportunisme pur. IMFIL enchaîne l'étape de sonde du stencil avec une descente du gradient. C'est cette dernière qui confère un intérêt à la méthode, d'autant plus que la qualité de sa contribution est dépendante du nombre de points calculés dans la sonde du stencil. L'interruption prématurée de la sonde du stencil empêchera l'obtention de l'information nécessaire pour la détermination d'une direction de descente cohérente, puisque celle-ci est déterminée à l'aide de différences finies obtenues avec les valeurs de la fonction objectif aux points de P^k. Pre-

nons pour exemple l'obtention d'un succès à une première évaluation dans la sonde. En ayant échantillonné seulement dans un sous-espace de dimension 1, d'un espace de dimension n, le gradient du stencil aura $n-1$ composantes nulles. On anticipe alors que cette stratégie nuira au bon fonctionnement pratique de la méthode.

Algorithme 9 Sonde du stencil IMFIL-op

Avec $f : \mathbb{R}^n \to \mathbb{R}$ la fonction objectif et x^0 le point de départ
Poser $P^k := \{x^k + h^k v_i : v_i \in V\}$ et ordonner selon la stratégie d'ordonnancement choisie.
for $t \in P^k$ **do**
  if $f(t) < f(x^{k+1})$ **then**
    $x^{k+1} \leftarrow t$
    Fin de la sonde, poursuivre à la recherche linéaire
  end if
end for
Fin de la sonde, poursuivre à la recherche linéaire

Pour éviter les cas avec des directions comportant plusieurs composantes nulles, on propose une autre version de l'algorithme incorporant la stratégie opportuniste. Rappelons que IMFIL est un algorithme d'optimisation sous contraintes de bornes. Il utilise le motif $V = \{\pm e_i : i = 1, \ldots, n\}$ formé par les directions élémentaires, duquel découle un gradient du stencil $\nabla_s f(x^k) = \frac{1}{h^k}\delta(f, x^k, V, h)V^\dagger$. En considérant le cas le plus restrictif en terme de points réalisables, x^k se trouve à la frontière de n contraintes actives. Les n points du motif, maintenant superposés à x^k, figurant dans Ω seront nécessaires pour évaluer n différences finies avant, qui fourniront une approximation du gradient dans chaque dimension de n. Ces approximations sont moins précises tel que le stipule le théorème de Taylor, soit $O(h)$ plutôt que $O(h^2)$ dans le cas de IMFIL sans opportunisme, où chaque approximation est calculée avec une différence centrée.

L'idée de la deuxième approche est de favoriser l'utilisation de différences avant dans le calcul de $\nabla_s f(x^k)$, à l'instar du cas précédemment décrit. Le fonctionnement de l'algorithme serait le suivant. En premier lieu, les points $x^k + h^k e_j$ et $x^k - h^k e_j$ sont ordonnés deux à deux avec la stratégie d'ordonnancement choisie. Dans le cas où une contrainte est active pour la dimension j tel que $x^k + h^k e_j \notin \Omega$ ou $x^k - h^k e_j \notin \Omega$, aucun ordonnancement n'est nécessaire pour cette paire et c'est le point réalisable qui est considéré comme le premier. Ensuite, on évalue f pour les n premiers points des W_j définis à l'étape précédente. Si un succès est encouru, alors la sonde du stencil s'arrête et on procède à la recherche linéaire. Si aucun point n'a entrainé de diminution simple, alors on réordonne les points restants, soit les deuxièmes

points dans les paires, avec la même stratégie choisie et on poursuit la sonde du stencil de façon opportuniste.

La méthode proposée implique au minimum une approximation du gradient dans chaque dimension et incorpore la notion d'opportunisme. Pour chaque étape de sonde, n évaluations peuvent être sauvées. Cependant, pour chaque étape de sonde fructueuse dont la recherche linéaire subséquente est non-fructueuse, les évaluations supplémentaires auront été en vain. On désignera cette méthode comme IMFIL-od pour IMFIL avec opportunisme dimensionnel. Le cadre algorithmique présenté est généralisé pour le cas où tous les points du motif V sont réalisables. Les ensembles W_i y représentent les ensembles de deux points opposés dans V.

Algorithme 10 Sonde du stencil IMFIL-od

Avec $f : \mathbb{R}^n \mapsto \mathbb{R}$ la fonction objectif et x^0 le point de départ
Soit la paire de points $x^k + h^k v_i$ et $x^k - h^k v_i$. Ordonner selon la stratégie d'ordonnancement choisie pour tout i. Poser $W_{i,1}$ le point apparaissant en premier dans l'ensemble et $W_{i,2}$ le point apparaissant en deuxième.
Poser $P_1^k := \{W_{i,1} \ : \ i = 1, 2, \ldots, n\}$
$x^{k+1} \in \arg\min\{f(x) : x \in P_1^k \cup \{x^k\}\}$
if $x^{k+1} \neq x^k$ **then**
  Fin de la sonde, poursuivre à la recherche linéaire
end if
Poser $P_2^k := \{W_{i,2} \ : \ i = 1, 2, \ldots, n\}$ et ordonner P_2^k selon la stratégie d'ordonnancement choisie
for $t \in P_2^k$ **do**
  if $f(t) < f(x^{k+1})$ **then**
    $x^{k+1} \leftarrow t$
    Fin de la sonde, poursuivre à la recherche linéaire
  end if
end for
Fin de la sonde, poursuivre à la recherche linéaire

CHAPITRE 4 RÉSULTATS NUMÉRIQUES

Ce chapitre contient les détails portant sur la comparaison des stratégies opportunistes et des stratégies d'ordonnancement. Dans un premier lieu, on y décrit les outils destinés à la comparaison utilisés. Les implémentations et les problèmes tests sont expliqués en détails. Enfin, on y fait la comparaison des stratégies et on en retire les tendances principales.

4.1 Outils de comparaison

Les premières recherches visant à comparer les performances de solveurs sur un problème unique ou une série de problèmes utilisaient des métriques facilement quantifiables telles que le nombre d'appels à la fonction [18] ainsi que le temps de résolution en secondes [60]. Cependant, ces indicateurs sont moins appropriés en situation de résolution d'un grand nombre de problèmes. Puisque ces métriques sont spécifiques à un problème, le banc d'essai qui incorpore chacune d'entre elles prend des proportions énormes. Dans cet ordre d'idée, certains outils ont été développés pour analyser une série de problèmes tests, avec des dimensions et des spécificités différentes, sans qu'il en devienne laborieux d'en tirer des conclusions.

4.1.1 Tests et graphes de convergence

Pour comparer la performance d'un ensemble d'algorithmes sur un seul problème, on aura recours au graphe de convergence. On y trace la courbe de la meilleure valeur de la fonction objectif obtenue en fonction du nombre d'appels à la fonction objectif.

En optimisation sans dérivées, contrairement à l'optimisation classique, rien n'assure qu'un solveur arrivera à l'optimum global, ou encore qu'un optimum soit atteint dans le budget alloué. De plus, en situation de résolution de boîte noire, il est impossible de connaître l'optimum théorique du problème, et ce même si celui-ci existe. On aimerait tout de même quantifier la performance d'un solveur. À cette fin, il est entendu dans la littérature de faire appel à un test de convergence qui mesure la capacité à l'algorithme à s'approcher de la meilleure solution obtenue par l'ensemble des algorithmes à l'étude, soit

$$f(x^0) - f(x) \geq (1 - \tau)(f(x^0) - f^*) \tag{4.1}$$

issu de [59] avec x^0 le point initial, x le point courant, f la fonction objectif, $\tau \geq 0$ un seuil de tolérance et f^* la meilleure solution courante trouvée par un des solveurs comparés sur ce

problème. Le test de convergence peut être utilisé pour savoir si un solveur atteint au moins une solution proche de la meilleure en utilisant un τ grossier, par exemple $\tau = 0.1$, ou encore pour savoir si le solveur est capable d'obtenir f^* en utilisant $\tau = 0$. Il est possible que le test ne soit jamais réussi si on observe la résolution avec un budget d'évaluation fixe, ce qui sera nécessaire pour des boîtes noires bruitées, pour lesquelles la convergence de l'algorithme n'est pas assurée.

En situation de barrière progressive, le test de convergence doit être adapté pour tenir compte des cas où $x \notin \Omega$. Afin de comparer seulement des points réalisables, on utilisera le test issu de [17]

$$\bar{f}_{\mathsf{fea}} - f(x_e) \geq (1 - \tau)(\bar{f}_{\mathsf{fea}} - f^*)$$

où $\bar{f}_{\mathsf{fea}}$ y représente la valeur moyenne des premiers points réalisables obtenus dans les résolutions des différentes instances du même problème et $f(x_e)$ la fonction objectif obtenue à un point réalisable. Si x_e n'est pas un point réalisable, le test est échoué.

4.1.2 Profils de performance

Un outil établi pour la comparaison de solveurs d'optimisation classique est le profil de performance [33]. La performance est donnée par une mesure $t_{p,s}$, qui peut être par exemple le nombre d'évaluations nécessaire pour l'obtention du minimum global, pour une paire de p et s, un problème et un solveur. En optimisation sans dérivées, on choisira le nombre d'itérations nécessaire à la satisfaction du test de convergence énoncé précédemment. Afin de comparer les solveurs sur chaque problème, on défini le ratio de comparaison suivant

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}}$$

On peut alors définir le profil de performance d'un solveur sur un ensemble de problèmes $\mathcal{P}$ ainsi

$$\rho_s(\alpha) = \frac{1}{|\mathcal{P}|}|\{p \in \mathcal{P} : r_{p,s} \leq \alpha\}|$$

On peut ainsi connaître la proportion de problèmes que chaque solveur a su résoudre en se rapprochant à un facteur de $(1-\tau)$ de la meilleure solution trouvée par l'ensemble des solveurs, en fonction des ratios de comparaison. Le test de convergence varie selon la nature des problèmes à résoudre. On peut utiliser un profil de performance pour des outils d'optimisation avec dérivés avec un $t_{p,s}$ qui agit sur les informations de premier ordre de la fonction, tel que le nombre d'évaluations pour atteindre x tel que $f'(x) = 0$. Ainsi, $\rho_s(\tau)$ revient à être la probabilité que le ratio de performance $r_{p,s}$ du solveur s soit à un facteur τ du meilleur ratio

possible. La lacune principale des profils de performance consiste en l'incapacité du ratio utilisé de prendre en compte la dimension du problème.

Il est important de noter que les profils de performance ont une faille telle qu'élaboré dans [37]. Les auteurs concluent que l'utilisation de f^* dans le test de convergence implique que la simple observation des courbes sur un graphe n'est pas suffisante pour classes les algorithmes. Puisque la comparaison se fait avec le nombre d'itérations minimal observé, il se peut que, lorsqu'on compare entre eux deux solveurs qui n'ont pas performé en tant que meilleur, l'ordre apparent ne soit pas celui qu'on observerait si on éliminait du graphe les données du meilleur. On se doit alors d'utiliser un autre outil pour mesurer les performances des solveurs sur un ensemble de données.

4.1.3 Profils de données

Moré et Wild proposent [62] une définition d'un profil de donnée,

$$d_s(\alpha) = \frac{1}{|\mathcal{P}|}|\{p \; \epsilon \; P : \frac{t_{p,s}}{n_p + 1} \leq \alpha\}|$$

avec $t_{p,s}$ le nombre d'évaluations pour atteindre la convergence nécessité par le solveur s sur le problème p. Le facteur $n_p + 1$ est le nombre d'évaluations nécessaire pour le calcul de l'estimation finie d'un gradient pour un problème de taille n_p. Il est alors possible de mesurer la performance relative des solveurs comme une fonction du budget d'évaluations, sans la comparaison que le ratio de performance imposait. Ainsi, pour un α, soit une quantité de $n+1$ évaluations de la fonction objectif, on peut savoir la proportion de problèmes que chaque solveur a su résoudre en se rapprochant à un facteur de $(1-\tau)$ de la meilleure solution trouvée par l'ensemble des solveurs. L'avantage des profils de données sur les profils de performance est qu'il est possible de quantifier la performance du meilleur algorithme, étant donné que celui-ci n'est plus comparé à lui-même. La faille évoquée par Gould et Scott [37] n'est pas applicable aux profils de données.

4.2 Problèmes tests

La comparaison des stratégies se fait sur plusieurs ensembles de problèmes tests. En premier lieu, un ensemble de problèmes non linéaires sans contraintes et sans bornes est décrit afin de caractériser le comportement des algorithmes opportunistes sur un large ensemble de problèmes. En deuxième lieu, un échantillon de problèmes avec des contraintes relaxables et quantifiables est décrit afin d'observer la performance des différentes versions de la stratégie

opportuniste en situation de barrière progressive. Enfin, on décrit en détail des cas de problèmes d'ingénierie connus comme pouvant être résolus avec des outils d'optimisation sans dérivées dans le but de déterminer si la stratégie opportuniste a le potentiel d'influencer la résolution d'une boîte noire d'expression non analytique.

4.2.1 Ensemble de problèmes Moré-Wild

L'échantillon de problèmes analytiques caractéristiques $\mathcal{P}$ utilisé est celui proposé par Moré et Wild [62]. Cet ensemble de problèmes est fréquemment réutilisé dans la communauté [76, 30, 77, 23, 4, 71, 16], ce qui justifie son utilisation comme base de problèmes analytiques pour la comparaison d'algorithmes ou de stratégies algorithmiques. Il s'agit d'une banque de problèmes sans contraintes et sans bornes.

L'ensemble de problèmes est décliné en 22 fonctions non linéaires de moindres carrés tirées de la collection CUTEr [38]. Ces 22 fonctions sont remaniées pour donner un ensemble de 53 problèmes. L'indice k_p pour un problème $p \in \mathcal{P}$ fait référence à la fonction de base issue de CUTEr utilisé pour ce problème.

Chaque problème possède trois autres paramètres en plus de l'indice k_p, soient n_p pour la dimension du problème, m_p le nombre de composantes du problème et le paramètre binaire s_p, indiquant si une homothétie de facteur 10 est activée ou non sur le point de départ x_0. L'ensemble $\mathcal{P}$ contient 53 problèmes pour lesquels existe un vecteur (k_p, n_p, m_p, s_p) unique. Aucune fonction sous-jacente n'est surreprésentée puisqu'au plus six problèmes possèdent le même k_p. Les bornes sur les paramètres sont

$$1 \leq k_p \leq 22, \quad 2 \leq n_p \leq 12, \quad 2 \leq m_p \leq 65, \quad s_p \in \{0, 1\}, \quad p = 1, \ldots, 53$$

avec $n_p \leq m_p$.

La structure par morceaux des problèmes de $\mathcal{P}$ permet la fabrication de fonctions objectif ayant différentes propriétés. On entend ainsi permettre la comparaison d'algorithmes ou de stratégies algorithmiques sur différentes classes de problèmes dont les caractéristiques peuvent être représentées dans des problèmes de boîtes noires. Les classes utilisées dans [62] qui sont réutilisées ici sont : les problèmes lisses, les problèmes lisses définis par partie, les problèmes à bruit stochastique et les problèmes à bruit déterministe. **Les problèmes lisses** sont formés tels que :

$$f(x) := \sum_{i=1}^{m} \left(f_i(x)\right)^2.$$

Les problèmes non différentiables sont obtenus à l'aide d'une transformation apportée à l'expression des problèmes lisses telle que :

$$f(x) := \sum_{i=1}^{m} |f_i(x)|.$$

Ainsi, l'expression du gradient $\nabla f_i(x)$ est inexistante lorsque $f_i(x) = 0$. Afin d'assurer que chaque déclinaison de problème possède un minimum global, pour les fonctions issues d'un ensemble de problèmes sous-jacents $k_p \in \{8, 9, 13, 16, 17, 18\}$, la fonction est redéfinie dans l'orthant positif tel que

$$f(x) := \sum_{i=1}^{m} |f_i(\max(x, 0))|.$$

Les problèmes à bruit déterministe sont obtenus à l'aide d'une transformation apportée à l'expression des problèmes lisses. On multiplie la fonction $f(x)$ par le terme $(1 + \epsilon_f \theta(x))$ qui induit une oscillation

$$f(x) := (1 + \epsilon_f \theta(x)) \sum_{k=1}^{m} f_i(x)^2.$$

Le terme $\theta(x)$ est issu d'une composition d'une fonction trigonométrique $\theta_0(x)$ avec un polynôme de Chebyshev de degré 3 tel que $T_3(\alpha) = \alpha(4\alpha^2 - 3)$

$$\theta_0(x) = 0.9 \sin(100||x||_1) \cos(100||x||_\infty) + 0.1 \cos(||x||_2)$$
$$\theta(x) = T_3(\theta_0(x)) = T_3(0.9 \sin(100||x||_1) \cos(100||x||_\infty) + 0.1 \cos(||x||_2)).$$

Cette composition élimine la périodicité de θ_0. La formulation est aussi composée du terme ϵ_f, le niveau du bruit relatif, qui est fixé à $\epsilon_f = 10^{-3}$.

Les problèmes à bruit stochastique sont obtenus avec une transformation apportée à chaque composante de la sommation

$$f(x) := \sum_{k=1}^{m} ((1 + \epsilon_f u_i) f_i(x))^2.$$

Le scalaire u_i est issu du vecteur $u = (u_1, u_2, \ldots, u_m)$ dont chaque valeur $-0.5 < u_i < 0.5$ est déterminée aléatoirement. La formulation est aussi composée du terme de bruit relatif fixé à $\epsilon_f = 10^{-3}$.

4.2.2 Problèmes analytiques contraints

Les problèmes Moré-Wild n'ont pas de contraintes ni de bornes. L'impact observé de l'opportunisme et de l'ordonnancement sur ces problèmes n'est pas nécessairement transférable en optimisation sous contraintes, compte tenu des différences que comporte la sonde lorsqu'on se trouve en situation de barrière active. Un ensemble de problèmes contraints issus de [23, 17] est mis en place pour former un banc d'essai. Il s'agit de problèmes analytiques recueillis dans la littérature comportant des contraintes relaxables de type QR*K. Le nombre n de variables varie de 2 à 20 et le nombre m de contraintes de 1 à 15.

Tableau 4.1 – Description de l'ensemble de problèmes contraints

#	Nom	Source	n	m	#	Nom	Source	n	m
1	CHENWANG_F2	[21]	8	6	10	MAD6	[42]	5	7
2	CHENWANG_F3	[21]	10	8	11	MEZMONTES	[61]	2	2
3	CRESCENT	[10]	10	2	12	OPTENG_RBF	[50]	3	4
4	DISK	[10]	10	1	13	PENTAGON	[58]	6	15
5	G2_10	[11]	10	2	14	PIGACHE	[68]	4	11
6	G2_20	[11]	20	2	15	SNAKE	[42]	2	2
7	HS19	[42]	2	2	16	SPRING	[10]	3	4
8	HS83	[58]	5	6	17	TAOWANG_F2	[74]	7	4
9	HS114	[42]	9	6	18	ZHAOWANG_F5	[78]	13	9

4.2.3 STYRENE

Les méthodes de recherche directe sont appliquées à la résolution d'un problème issue de l'ingénierie de procédé proposé par Audet et al. [6]. Il s'agit d'une simulation de la production de styrène, un composé organique utilisé dans la fabrication de plastique, particulièrement le polystyrène. Cette simulation comprend quatre étapes principales.

i. La préparation des réactifs par mise sous pression et par chauffage précède la réaction chimique. Cette étape est influencée par la pression à la sortie de la pompe x_1, en atm, et la température à la sortie du réchauffeur x_2, en K, qui constituent des variables du problème. Il est à noter que les entrées dans le système proviennent d'une source externe et d'un retour de matière recyclée provenant de la sortie du procédé chimique.

ii. La réaction chimique prenant place dans le réacteur, faisant intervenir x_3, en mètres, la longueur du réacteur.

iii. La récupération du styrène par la distillation des produits de la réaction. Les produits sont initialement refroidis à une certaine température x_4, en K. Les produits sont

dégazés et les gaz indésirables sont ensuite brulés et éjectés par la cheminée, dont la fraction d'air excédentaire x_5 pour la combustion figure comme variable d'optimisation. Les produits refroidis et dégazés vont ensuite au séparateur à styrène, dans lequel a lieu la première distillation. Le séparateur nécessite deux composantes, soit la lourde et la légère. La fraction x_6 de composante légère utilisée dans le séparateur constitue une variable du problème. À la sortie du séparateur, les résidus (ethylbenzène) peuvent être réutilisés comme réactifs. La fraction des résidus x_7 qui retourne à la première étape est une variable nécessaire pour la simulation.

iv. La portion non recyclée des résidus est ensuite distillée à son tour pour en retirer le benzène des éléments réactifs restants, lesquels sont aussi retournés à la première étape. La fraction x_8 de composante légère utilisée dans le séparateur à benzène constitue une variable du problème.

Avec $x = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$ le vecteur de variables, le simulateur peut estimer la valeur nette estimée du projet de production de styrène, en agrégeant des facteurs comme les coûts d'opérations, le profit issu des ventes du styrène produit, les investissements nécessaires pour maintenir la production et la dépréciation du projet sur un nombre donné d'années.

Selon la terminologie de [53], le problème comporte 11 contraintes de type K et un nombre indéfini de type H. Les contraintes cachées sont des contraintes de type NUSH (Non-quantifiable, non-relaxable, encourue durant la simulation et cachée). Les contraintes connues sont explicitées au tableau 4.2 avec leur taxonomie respective. Une caractéristique intéressante de

Tableau 4.2 – Contraintes de STYRENE groupées par type

#	Type	Description	Taxonomie
1	Simulateur	Succès de la simulation	NUSK
2	Procédé	Structure du séparateur à styrène acceptable	NUSK
3		Structure du séparateur à benzène acceptable	NUSK
4		Combustion des gaz possible et réglementaire	NUSK
5		Borne minimale sur la pureté du styrène produit	QRSK
6		Borne minimale sur la pureté du benzène produit	QRSK
7		Borne minimale sur la conversion des résidus en styrène	QRSK
8	Économique	Borne maximale sur le temps requis pour atteindre la rentabilité	QRSK
9		Borne minimale sur les flux de trésorerie actualisés	QRSK
10		Borne maximale sur l'investissement total	QRSK
11		Borne maximale sur le coût annuel	QRSK

STYRENE est que la simulation échoue sur une proportion inconnue des évaluations.

4.3 Méthodologie

Cette section décrit la méthodologie employée pour obtenir les algorithmes et leurs implémentations opportunistes.

4.3.1 Logiciel NOMAD

L'implémentation des l'algorithme CS, GPS et MADS utilisée est NOMAD 3.8.1 [52] (Nonlinear Optimization with the Mesh Adaptive Direct search), disponible sur son site officiel [1]. Initialement une implémentation de GPS pour la recherche non linéaire, NOMAD est aujourd'hui un des solveurs principaux dans la littérature pour l'optimisation de boîtes noires [26, 72] à l'aide de recherche directe. L'architecture de NOMAD est telle que chaque point $t \in P^k$ ou $t \in S^k$ doit figurer sur le maillage $M^k := \{x^k + \delta^k Dy : y \in \mathbb{N}^p\}$. Chaque point déterminé avec sa direction correspondante $t = x^k + \delta^k d : d \in D^k$ qui ne correspond pas à un point sur le maillage sera remplacé par un point sur le maillage $\tilde{t} \in M^k$ correspondant au point existant sur le maillage le plus près. Un paramétrage spécifique est nécessaire à NOMAD afin que le logiciel soit en mesure d'appliquer CS, GPS et MADS selon leurs définitions des sections 2.1, 2.2 et 2.3. Les descriptions des différents paramètres sont issues de [15].

Afin de ne pas obtenir le comportement d'un maillage granulaire, la valeur `XMESH`, qui correspond au maillage anisotrope décrit dans [16] est affectée au paramètre `MESH_TYPE`.

CS

La valeur `GPS 2N STATIC` est donnée au paramètre `DIRECTION_TYPE`, qui contrôle l'algorithme et la mécanique gérant la génération des directions. En ayant `GPS` comme méthode de génération de directions, on a qu'un seul paramètre pour les directions, soit δ^k, la longueur du pas. En ayant $2n$ directions définies de manière statique (en opposition à aléatoire), le logiciel génère toujours les mêmes directions, qui sont par défaut les directions coordonnées. Pour l'option `GPS` dans NOMAD, le maillage est obligatoirement isotrope, ce qui correspond à la définition de CS et qui annule l'anisotropie induite par l'option `XMESH`.

Puisque l'ajustement du maillage dans CS ne se fait qu'en cas de sonde fructueuse, on affecte au paramètre `MESH_COARSENING_COEFFICIENT` la valeur `0`.

L'implémentation de l'opportunisme dans la recherche par coordonnées permet d'observer l'impact de la stratégie sur la plus archaïque et la plus grossière des méthodes de recherche directe. Afin de désactiver les stratégies de recherche globale présente par défaut dans NOMAD, certains paramètres se doivent d'être désactivés (Tableau 4.3).

Tableau 4.3 – Paramètres de SEARCH désactivés pour CS dans NOMAD

Nom	Description	Valeur
MODEL_SEARCH	SEARCH basée sur des modèles quadratiques [23]	NO
SPECULATIVE_SEARCH	*Dynamic search* de [9]	NO
SNAP_TO_BOUND	Accolement des points aux bornes	NO

GPS

L'algorithme GPS est obtenu a l'aide d'un paramétrage similaire à celui de CS. Afin de correspondre à la définition de l'auteure [75] sans imposer de variations arbitraires, on utilise les directions coordonnées en affectant `GPS 2N STATIC` à `DIRECTION_TYPE`, qui annule encore une fois l'anisotropie du maillage.

La différence se situe au niveau du `MESH_REFINING_COEFFICIENT` prenant la valeur `1`, qui correspond à la puissance affectant le paramètre d'ajustement de maillage en cas de succès τ^{-1} de l'Algorithme 2. En laissant la valeur par défaut, on a que l'exposant -1 affecte le $\tau = 2$, contrairement à l'exposant 0 dans le cas de CS.

Étant donné la volonté, en premier lieu, d'isoler les étapes de POLL des algorithmes, les mêmes paramètres figurant dans le tableau 4.3 sont désactivés pour GPS.

MADS

L'implémetation de MADS est OrthoMADS [3] avec $n+1$ directions obtenue avec l'affection de `ORTHO N+1 NEG`. Il s'agit d'une instance de MADS déterministe produisant n direction orthogonales. La génération des directions $\mathbb{D}^k$ telle qu'élaborée dans [16] est effectuée en quatre étapes.

i. La séquence aléatoire produit un vecteur $u^t \in [0,1]^n$ correspondant à une direction aléatoire.

ii. La direction u^t subit une transformation de Householder [45]. Cette transformation génère une base orthogonale positive de $\mathbb{R}^n$ composée de vecteurs d'entiers, qu'on dénote H.

iii. L'ensemble des directions H sont mises à l'échelle à l'aide du vecteur Δ^k correspondant au paramètre de la longueur du cadre dans chaque direction. Les directions sont ensuite arrondies pour être accolées au maillage.

On choisit le nombre $n+1$ de directions afin d'avoir le minimum de directions dans l'ensemble générateur positif, ce qui accélère la convergence. On choisit `NEG` pour indiquer au logiciel de fabriquer la $(n+1)$-ième direction en prenant la direction opposée à la somme des directions

issues de la transformation de Householder $d_{n+1} = -\sum_{d \in H_{t,l}} d$. L'ensemble $D_\Delta^k = [H_{t,l}\ d_{n+1}]$ est ainsi une base positive et donc un ensemble générateur positif, condition nécessaire pour MADS tel qu'évoqué à la section 2.3. On choisit cette méthode à celle de la complétion par modèle quadratique proposée par Audet et al. [13] par souci d'interférence avec les paramètres nécessaires à l'obtention des stratégies d'ordonnancement.

Les valeurs des paramètres du maillage et celles des paramètres figurant dans le tableau 4.3 sont identiques à celles de GPS.

MADS par défaut de NOMAD

Afin d'observer l'impact de l'ordonnancement sur la résolution complète des problèmes d'optimisation identifiés, on résout l'ensemble des problèmes décrits à la section 4.2 avec une instance de MADS en communicant à NOMAD seulement les valeurs de paramètres indispensables. Ceci implique l'activation des étapes de SEARCH désactivées pour les autres algorithmes, soient celles du Tableau 4.3, l'utilisation du maillage granulaire `GMESH` [15] et la génération de direction `DIRECTION_TYPE ORTHO N+1 QUAD`. Cette méthode de génération de directions indique la complétion de la base positive issue de Householder avec une direction d_{n+1} qui permet l'obtention d'un ensemble générateur positif tout en minimisant l'approximation de f au point candidat $x^k + \delta^k d_{n+1}$. Les résultats obtenus pour cet algorithme ne sont pas à l'abri d'interférences entre les paramètres nécessaires pour l'obtention des différentes stratégies d'ordonnancement et les paramètres de génération de directions. Ils sont aussi grandement influencés par les heuristiques présentes dans les étapes de SEARCH. Dans ces mesures, on utilisera ces résultats afin de visualiser l'apport possible de l'opportunisme et de l'ordonnancement sur une résolution réaliste, plutôt que sur l'étape de sonde seulement.

Opportunisme et stratégies d'ordonnancement

NOMAD permet à l'utilisateur de déterminer si la POLL ou la SEARCH se doit d'être opportuniste. Le paramètre nécessaire à la sonde opportuniste est `OPPORTUNISTIC_EVAL`, pour lequel la valeur `YES` implique l'opportunisme simple tel que décrit dans la définition 3.1.4 et la valeur `NO` implique la sonde complète de la définition 3.1.5. L'obtention des autres déclinaisons de la stratégie opportuniste se fait à l'aide des paramètres `OPPORTUNISTIC_MIN_NB_SUCCES` suivi de son argument pour la stratégie opportuniste au $p^{\text{ème}}$ succès de la définition 3.3.1 et du paramètre `OPPORTUNISTIC_MIN_EVAL` suivi de sa valeur pour l'obtention de la stratégie opportuniste avec un minimum de q évaluations de la définition 3.3.2.

Les stratégies d'ordonnancement lexicographique, en fonction de la direction du dernier succès

et en fonction d'un modèle sont implémentées dans NOMAD et accessibles à l'utilisateur avec la bonne combinaison de paramètres. Le Tableau 4.4 décrit les combinaisons nécessaires à l'obtention des stratégies. Les stratégies restantes sont obtenues à l'aide de l'activation du

Tableau 4.4 – Paramètres pour l'obtention de certaines stratégies d'ordonnancement

Paramètre	OPPORTUNISTIC_EVAL	DISABLE	MODEL_EVAL_SORT
Sonde complète	NO	N/A	N/A
Lexicographique	YES	EVAL_SORT	NO
Dernier succès	YES	N/A	NO
Avec modèles	YES	N/A	YES

paramètre SGTE_EVAL_SORT. La stratégie omnisciente est obtenue en attribuant au paramètre SGTE_EXE l'exécutable destiné à servir de boîte noire. Dans le même ordre d'esprit, la stratégie inverse-omnisciente est obtenue en fournissant à ce même paramètre le nom d'un exécutable retournant $\tilde{f} := -f(x)$ en argument. Enfin, on lui donne un exécutable retournant un nombre aléatoire en argument pour obtenir la stratégie d'ordonnancement aléatoire.

4.3.2 Recherche par ensembles générateurs

La comparaison de certaines stratégies dans GSS est possible grace à son implémentation présente dans le logiciel d'optimisation HOPSPACK 2.0 [69, 40], héritier de APPSPACK [39]. Le logiciel est avant tout destiné à l'interfaçage de différents algorithmes d'optimisation de tout genre en parallèle, mais contient une implémentation de GSS présente par défaut. La gestion des contraintes est possible seulement si les contraintes sont formulables dans leurs formes analytiques, ce qui ne convient pas à la définition du domaine d'optimisation sans dérivées étudié dans cet ouvrage.

L'implémentation de GSS de HOPSPACK est opportuniste simple, sans option pour désactiver la stratégie. Cependant, l'implémentation offre de changer la stratégie d'ordonnancement à l'aide du paramètre `Use Random Order`, pour lequel l'argument `False` entraine un ordonnancement déterministe non précisée, tandis que `True` entraine la stratégie d'ordonnancement aléatoire.

4.3.3 Filtrage implicite

L'algorithme IMFIL est indissociable de son implémentation en MATLAB, dénommée imfil 1.0 [48, 49]. Il s'agit d'une implémentation de l'algorithme défini à la section 2.5 munie d'un mécanisme de gestion d'échec de la fonction objectif à retourner une valeur, ce qui en fait un solveur applicable aux boîtes noires. Ces contraintes cachées sont traitées avec un procédé

similaire à la barrière extrême. L'implémentation nécessite que des bornes soient fournies pour chaque problème.

Tel que mentionné à la Section 3.3, l'auteur et architecte de l'implémentation C.T. Kelley reconnaît la pertinence de l'opportunisme, mais ne la recommande pas et ne l'incorpore pas dans son implémentation. L'implémentation de la stratégie est effectuée en modifiant les portions `imfil_poll_stencil` du code pour y implémenter les stratégies opportunistes et d'ordonnancement voulues. L'idée principale de l'implémentation de l'opportunisme dans IMFIL est qu'on y simule que les points non évalués suivant l'arrêt prématuré de la sonde ont échoué à retourner une valeur. L'implémentation procède ensuite au BLS avec seulement une portion des valeurs aux points évalués. Les points artificiellement communiqués comme des échecs sont réinitialisés avant la prochaine itération.

4.4 Comparaison des stratégies d'ordonnancement sur les problèmes Moré-Wild

Les résolutions sont effectuées avec un budget d'évaluations de $1000(n+1)$. Les profils de données présents dans le corps du travail sont tracés avec les seuils de tolérance $\tau = 10^{-3}$ et $\tau = 10^{-7}$ pour avoir un aperçu des résolutions à basse et haute précision. Chaque combinaison de problème, de stratégie opportuniste et de stratégie d'ordonnancement est résolue 10 fois avec des germes aléatoires différents. Les algorithmes issus de NOMAD qui peuvent bénéficier de la gestion des contraintes par barrière progressive sont testés sur chaque banque de problème. Pour la stratégie opportuniste de la définition 3.3.1, on utilise $p = 2$, afin d'observer l'arrêt de la sonde après deux succès. Pour la stratégie opportuniste de la définition 3.3.2, on déterminera que $q = \lceil \frac{n}{2} \rceil$, afin de permettre à un algorithme d'explorer au moins un orthan.

En raison de l'incapacité de leurs implémentations à gérer les contraintes de type boîte noire, GSS et IMFIL sont testées seulement avec l'ensemble de problèmes non contraints de Moré-Wild. Les profils de données incluent les 212 instances possibles des problèmes Moré-Wild. Les graphes présentés ici ne sont qu'un sous-ensemble de ceux tracés. L'annexe A en contient davantage.

4.4.1 CS

Dans le cas des tests avec cette famille d'algorithme, l'utilisation de la stratégie omnisciente requiert le traçage de profils de données pour bien pouvoir comparer les stratégies. Ce type de métrique est choisi plutôt que les profils de performance, car la nature de leur abscisse est relative au nombre d'itérations avant d'atteindre le minimum, permettant ainsi de juger de la performance d'une stratégie d'ordonnancement indépendamment de ses concurrents. Ainsi,

les profils de données permettent de mieux distinguer la courbe dominante sans qu'elle soit jointe aux axes, comme il serait le cas pour un profil de performance.

La recherche par coordonnées offre une opportunité sans pareil d'observer l'impact de l'ordonnancement. Puisqu'il s'agit essentiellement d'une étape de sonde rudimentaire, elle sera très sensible à l'opportunisme et à la stratégie d'ordonnancement qui le guidera. On s'attend ici à des courbes très distinctes puisque l'influence de l'opportunisme n'est pas noyée par d'autres spécificités algorithmiques.

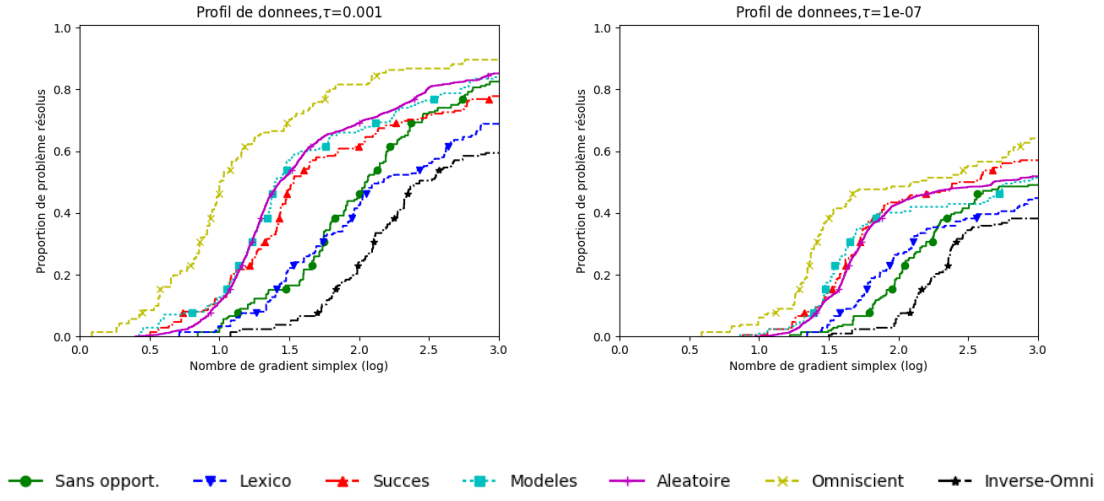


Figure 4.1 Comparaison sur tous les problèmes Moré-Wild avec CS, avec opportunisme simple.

Premièrement, dans la figure 4.1 on observe que la stratégie d'ordonnancement omnisciente semble bien remplir son objectif d'agir en tant que borne supérieure lorsque jumelée à l'opportunisme simple. Sa performance supérieure est indéniable, pouvant aller jusqu'à 50% de plus de problèmes résolus à précision $\tau = 10^{-3}$ que son plus proche adversaire, pour une abscisse donnée.

Il en va de même pour la stratégie inverse-omnisciente. Sa performance telle qu'observée indique que l'utilisation de la stratégie opportuniste peut être nuisible à la résolution de l'algorithme.

Les stratégies d'ordonnancement lexicographique, aléatoire, en direction du dernier succès et utilisant les modèles quadratiques sont des stratégies dites réalisables, contrairement aux stratégies de comparaison. Les stratégies réalisables les plus performantes sont les stratégies aléatoires et guidées par les modèles. Les profils de données montrent qu'elles sont de performances comparables. On remarque que, pour cet algorithme, seule la stratégie d'or-

donnancement aléatoire contient des résolutions qui diffèrent selon le germe aléatoire, ce qui confère au profil un aspect beaucoup plus lisse. La courbe représentant l'ordonnancement guidé par modèle, contrairement à la courbe aléatoire, pourrait être améliorée avec des modèles plus représentatifs de la fonction qu'un modèle quadratique, tel qu'élaboré pour NOMAD dans [73].

Malgré que la stratégie en fonction du dernier succès soit plus raffinée que la stratégie aléatoire, elle ne tirera avantage de la forme du problème que si celui-ci n'est pas composé de plusieurs minimums locaux et de points de selle. On pourrait conclure que l'ensemble de problèmes utilisé ici possède une proportion de problèmes dont la structure n'est pas idéale pour un ordonnancement basé uniquement sur le succès précédent. La stratégie fonctionne relativement bien si comparée à la méthode non opportuniste et présente une option déterministe peu raffinée.

La stratégie lexicographique n'est pas bonne en terme de performance. Elle pousse l'algorithme à épuiser ses sources de directions de descente une par une. Au fil des évaluations, si le point initial est un maximum, les directions de descente déjà épuisées seront évaluées en début de sonde, et celles prometteuses seront toujours à la fin de la liste, ce qui aura pour effet de décaler les succès de plus en plus au cours du déroulement de l'algorithme. On voit dans la stratégie lexicographique un exemple de mauvaise utilisation de la stratégie opportuniste.

La stratégie non-opportuniste nécessite n fois plus d'évaluation que la stratégie omnisciente pour chaque étape fructueuse de sonde et converge assurément vers le même point si la fonction n'incorpore pas d'aspects aléatoires. On peut observer quantitativement l'effet des évaluations supplémentaires nécessitées par la méthode sans opportunisme. Celle-ci n'est certainement pas gage de résolution optimale, puisque les stratégies d'ordonnancement aléatoires et guidées par modèles y sont en tout point supérieures. Cependant, on remarque que sa courbe est davantage linéaire, ce qui indique que pour un grand budget d'évaluations, elle consiste en un choix sûr.

Pour $p = 2$, la figure 4.2 montre que tous les profils semblent avoir le même comportement. Par ailleurs, le profil de la stratégie omnisciente est indiscernable. Ceci est dû au fait que la mécanique derrière le paramètre `OPPORTUNISTIC_MIN_NB_SUCCES` impose que le deuxième succès trouvé soit par rapport au premier. Ainsi, la sonde est arrêtée seulement si $f(\tau_2) < f(\tau_1) < f(x^*)$, avec $\tau_1 \in P^k$ le premier succès obtenu dans la séquence d'évaluations de la sonde et τ_2 le deuxième succès. C'est pourquoi chaque courbe est très similaire à la courbe de la sonde complète. Ceci explique aussi pourquoi il est impossible d'observer la courbe de l'ordonnancement omniscient, étant donné que celle-ci est jointe à la courbe de l'algorithme sans opportunisme. En effet, si la première évaluation est le meilleur succès possible, alors il

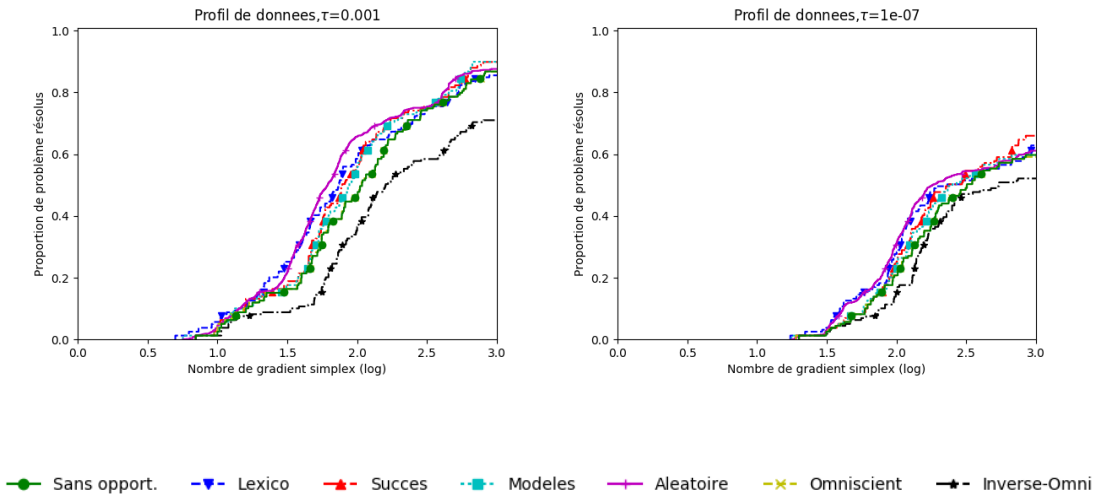


Figure 4.2 Comparaison sur tous les problèmes Moré-Wild avec CS, avec opportunisme au $p = 2^{\text{ème}}$ succès.

sera impossible d'en trouver un deuxième satisfaisant le critère d'opportunisme du paramètre.

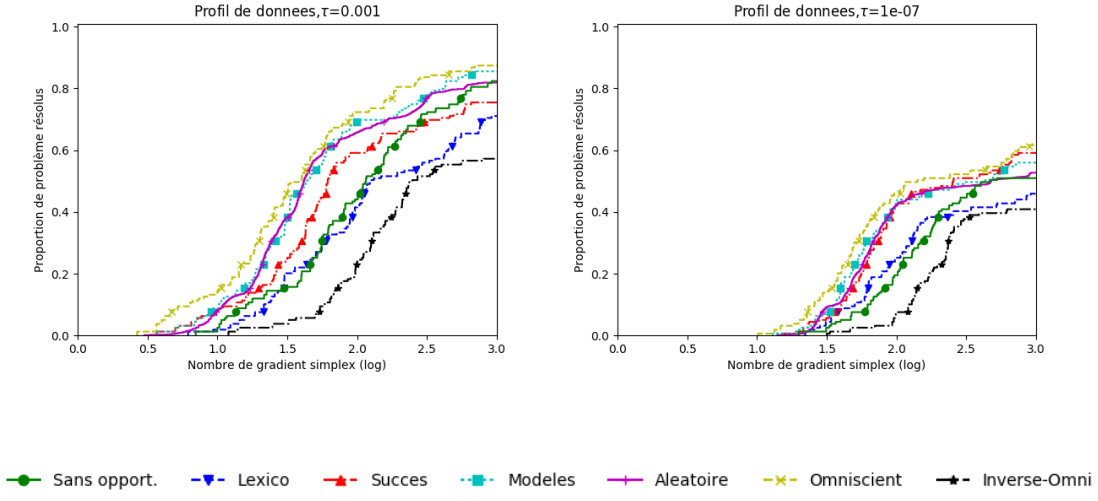


Figure 4.3 Comparaison sur tous les problèmes Moré-Wild avec CS, avec minimum $q = \lceil \frac{n}{2} \rceil$ évaluations.

Dans la figure 4.3, on observe le même ordre apparent entre les stratégies d'ordonnancement, ainsi que les mêmes tendances propres à chaque stratégie que celles observables à la figure 4.3. Toutefois, on remarque aussi que les courbes sont rapprochées les unes avec les autres. La dominance de la stratégie omnisciente n'est pas aussi évidente que dans celle observable dans

la figure 4.1. L'opportunisme avec un minimum de $q = \lceil \frac{n}{2} \rceil$ ne semble pas avoir le potentiel d'accélérer autant la résolution que l'opportunisme simple. La stratégie d'ordonnancement avec modèles semble bénéficier des évaluations supplémentaires dans la mesure où elle semble être un peu supérieure à la stratégie aléatoire, contrairement à ce qui était observé à la figure 4.1.

Enfin, on compare l'impact des différentes déclinaisons de la stratégie opportuniste sur les problèmes analytiques de Moré-Wild. La stratégie d'ordonnancement utilisée est l'ordonnancement guidé par modèles quadratiques.

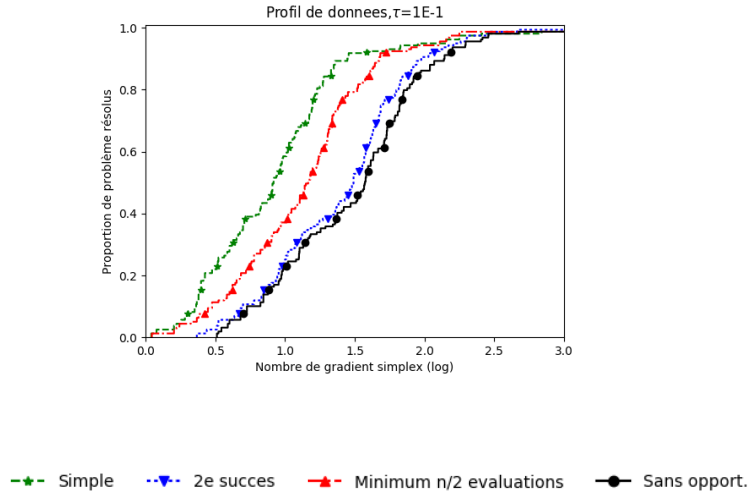


Figure 4.4 Comparaison des stratégies opportunistes sur tous les problèmes Moré-Wild avec CS

La figure 4.4 montre que la stratégie opportuniste simple présente un gain de performance net sur la sonde complète. La stratégie opportuniste avec au minimum $\lceil \frac{n}{2} \rceil$ présente un gain réduit, mais tout de même significatif, ce qui laisse entrevoir que l'utilisation de l'opportunisme pour des évaluations en paquet pourrait être bénéfique à un usager doté de ressources en parallèle. On remarque que la stratégie opportuniste avec $p = 2^{\text{ème}}$ succès présente un gain limité en performance.

En conclusion, on observe que la stratégie d'ordonnancement impacte fortement la performance d'une étape de sonde opportuniste. Les ordonnancements aléatoires et guidés par modèles sont ceux qui entraînent le plus grand gain possible. L'ordonnancement par la direction du dernier succès est aussi bénéfique, mais dans une moindre mesure. La stratégie lexicographique nuit à la résolution de problèmes. On en conclu aussi qu'en utilisant le profil de données de l'algorithme sans opportunisme, on observe que l'opportunisme simple tel qu'exis-

tant dans NOMAD offre une meilleure possibilité d'accélérer la résolution de problèmes. La stratégie opportuniste au 2ème succès, tel qu'implémentée dans NOMAD, ne propose pas une alternative intéressante à l'opportunisme simple. La stratégie opportuniste avec un minimum de $\lceil \frac{n}{2} \rceil$ évaluations non plus, étant donné que les tests démontrent qu'elle limite les gains en performances issus de l'utilisation de l'opportunisme.

4.4.2 GPS

La recherche par motifs généralisée est une version améliorée de la recherche par coordonnées. On s'attend à voir les mêmes impacts des stratégies opportunistes et la même hiérarchie entre les stratégies, mais dans une mesure moins prononcée. À la figure 4.5, on remarque en effet

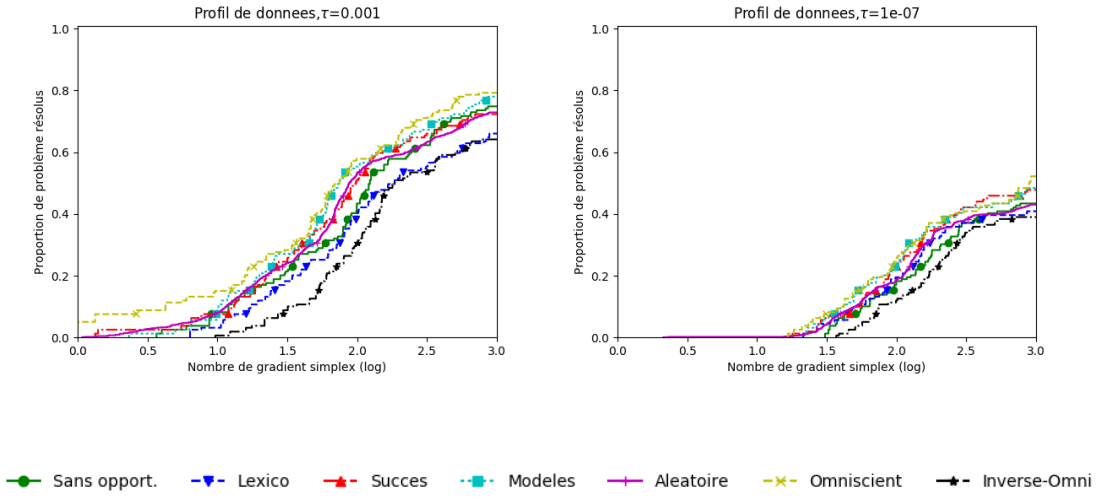


Figure 4.5 Comparaison sur tous les problèmes Moré-Wild avec GPS, avec opportunisme simple.

que les courbes sont rapprochées les unes des autres. L'expansion du maillage permet d'éviter plusieurs étapes de sondes fructueuses, ce qui a pour effet de limiter l'impact possible de l'opportunisme. On remarque la même hiérarchie des stratégies observée avec l'opportunisme simple est présente, mais de façon moins prononcée.

On observe toujours que la stratégie omnisciente surplombe les autres, mais avec GPS elle est talonnée par la stratégie d'ordonnancement guidée par modèles. Il est moins évident avec GPS qu'avec CS de définir un classement des stratégies par performance, mais on peut affirmer que la stratégie lexicographique semble toujours nuire aux résolutions, mais dans une moindre mesure. Les stratégies de la sonde complète, l'opportunisme avec ordonnancement aléatoire

et avec ordonnancement selon la direction du dernier succès semblent avoir des performances très comparables. On observe principalement que l'écart entre la courbe de l'opportunisme omniscient et l'opportunisme inverse-omniscient n'est pas aussi importante qu'avec CS. On reconnaît ainsi que l'opportunisme a un moins grand potentiel d'influence sur GPS que sur CS.

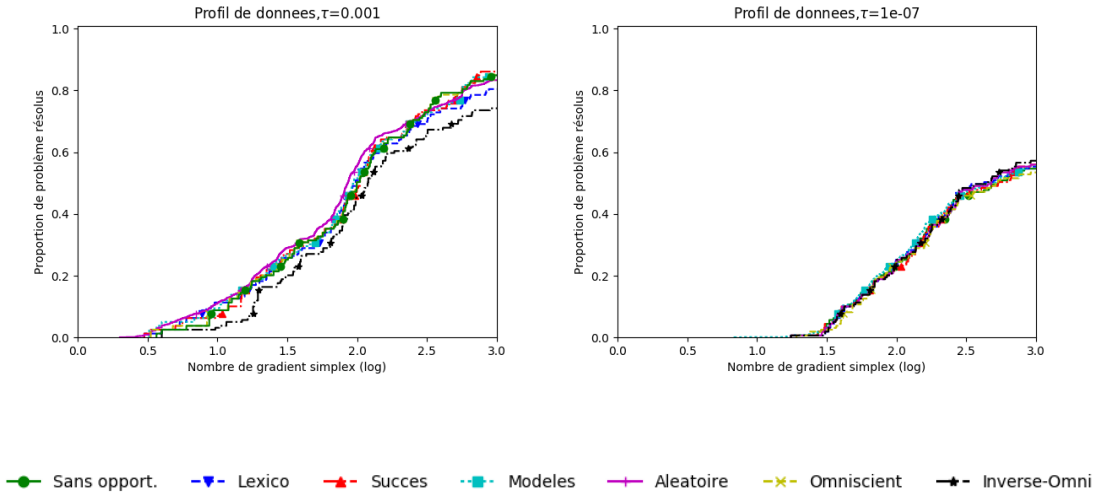


Figure 4.6 Comparaison sur tous les problèmes Moré-Wild avec GPS, avec opportunisme au $p = 2^{\text{ème}}$ succès.

La figure 4.6 révèle que la tendance observée pour GPS avec l'opportunisme simple se transfère dans celle observée pour l'opportunisme au deuxième succès. On voit que les profils sont difficilement dissociables les uns des autres, ce qui concrétise l'idée que l'opportunisme au deuxième succès tel qu'implémenté dans NOMAD porte l'algorithme à se comporter de façon non opportuniste.

La figure 4.7 présente des résultats similaires à la figure 4.5, mais en plus condensés. Il s'agit d'observations similaires que celles qui avaient été faites pour CS. Il semble y avoir toutefois une bonne performance de la stratégie d'ordonnancement avec modèles, qui semble bénéficier des évaluations supplémentaires lui permettant de raffiner les modèles nécessaires à l'ordonnancement.

Enfin, on compare l'impact des différentes déclinaisons de la stratégie opportuniste sur les problèmes analytiques de Moré-Wild. La stratégie d'ordonnancement utilisée est l'ordonnancement guidé par modèles quadratiques.

La figure 4.8 montre que la stratégie opportuniste simple ne présente plus un gain de performance aussi significatif qu'avec CS. La stratégie opportuniste avec au minimum $\lceil \frac{n}{2} \rceil$ présente

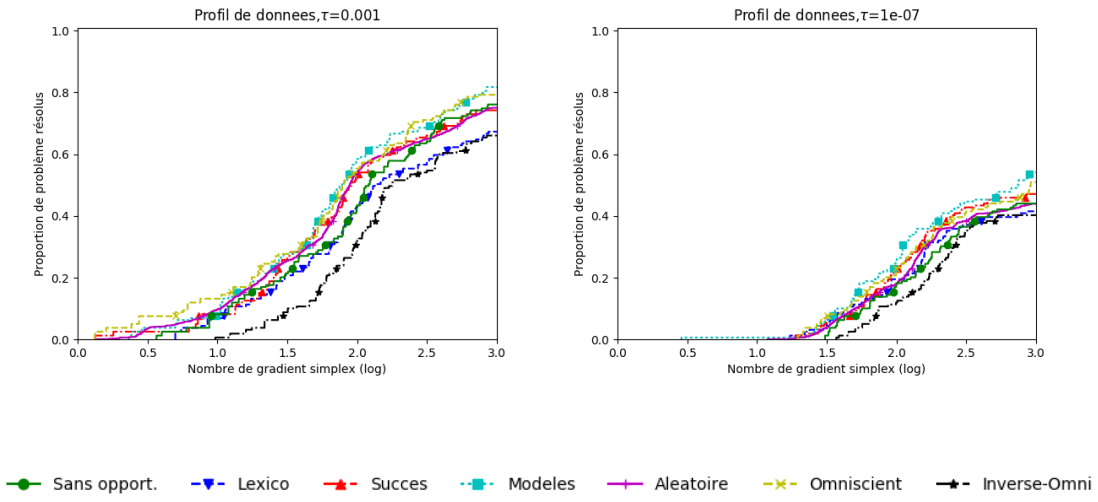


Figure 4.7 Comparaison sur tous les problèmes Moré-Wild avec GPS, avec minimum $q = \lceil \frac{n}{2} \rceil$ évaluations.

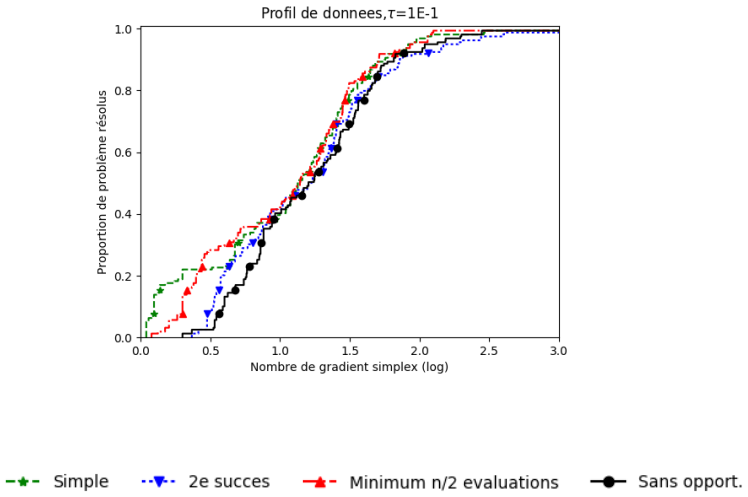


Figure 4.8 Comparaison des stratégies opportunistes sur tous les problèmes Moré-Wild avec GPS

un gain similaire à l'opportunisme simple, ce qui peut être expliqué par le moins grand nombre de succès marginaux et donc de mise à jour de $\delta^k \leftarrow \tau^{-1}$. On remarque que la stratégie opportuniste avec $p = 2$ succès présente encore une fois un gain limité en performance. L'importance de l'opportunisme est diluée avec GPS en vertu de la plus grande proportion des évaluations qui sont destinées à la convergence. En effet, si la longueur du pas est augmentée,

plusieurs succès intermédiaires sont évités. Toutefois, chaque augmentation de la longueur du pas entraîne qu'une diminution sera nécessaire à la convergence, diminution qui nécessitent que $2n$ évaluations soient faites.

En conclusion, l'opportunisme a un effet moins important sur GPS que sur CS. La stratégie d'ordonnancement préférable est la stratégie d'ordonnancement par modèle, quoique la stratégie aléatoire semble très performante pour son niveau de complexité. L'ordonnancement au $p = 2^{\text{ème}}$ succès semble rapprocher l'algorithme à sa version non opportuniste, et ce peu importe la stratégie d'ordonnancement employée.

4.4.3 MADS

MADS est un algorithme qui génère un plus grand nombre de directions que GPS, en plus de pouvoir nécessiter moins d'évaluations pour ses étapes infructueuses. Puisque la convergence est plus rapide, les étapes de sondes qui sont des succès auront plus d'impact dans la vitesse de résolution. On constate à la figure 4.9 que l'impact de la stratégie opportuniste est plus im-

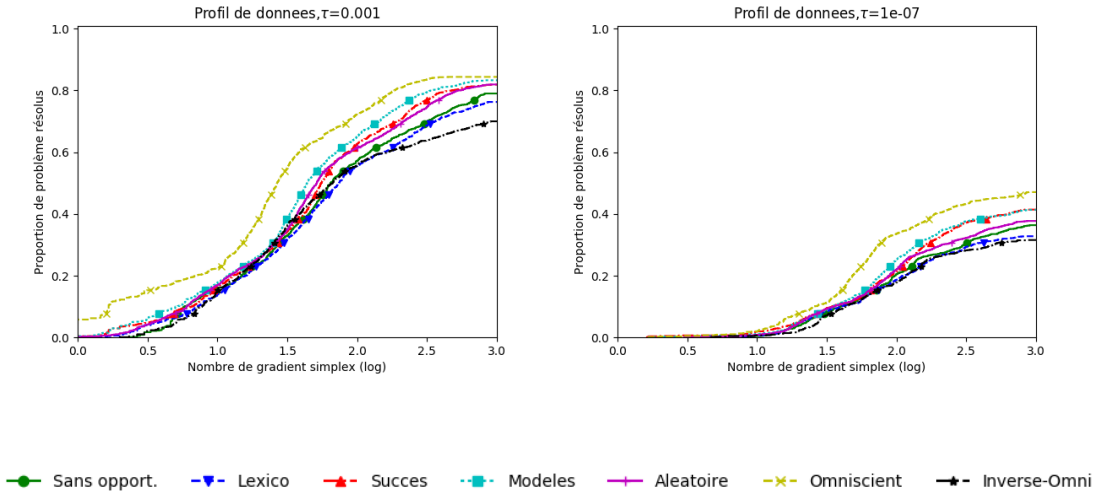


Figure 4.9 Comparaison sur tous les problèmes Moré-Wild avec MADS, avec opportunisme simple.

portant sur MADS que sur GPS. La stratégie omnisciente indique qu'un gain en performance est possible avec MADS beaucoup plus qu'avec GPS, et ce même à haute précision.

Outre le gain possible, la figure montre que la stratégie d'ordonnancement avec modèles est la stratégie réalisable la plus fiable pour cet ensemble de problèmes. Les ordonnancements aléatoires et en fonction de la direction du dernier succès offrent une meilleure performance

que la méthode sans opportunisme. Celle-ci est comparable à la stratégie lexicographique. Toutes les stratégies réalisables offrent une performance similaire pour un budget de $10(n+1)$ à $100(n+1)$ évaluations, ce qui nous indique que, pour des budgets d'évaluations limités, l'importance de la stratégie d'ordonnancement, ou même de l'utilisation de l'opportunisme, peut être mise en doute.

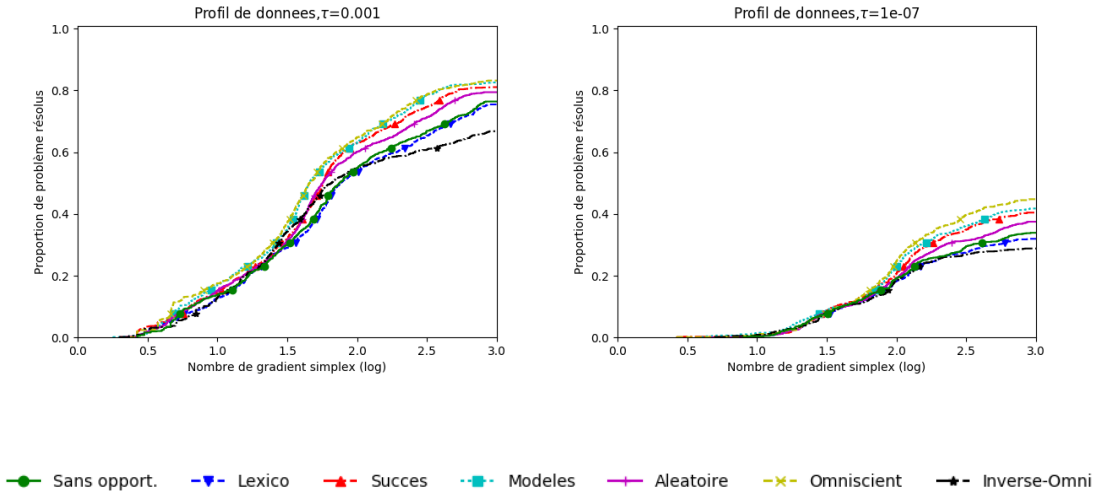


Figure 4.10 Comparaison sur tous les problèmes Moré-Wild avec MADS, avec opportunisme au $p = 2^{\text{ème}}$ succès.

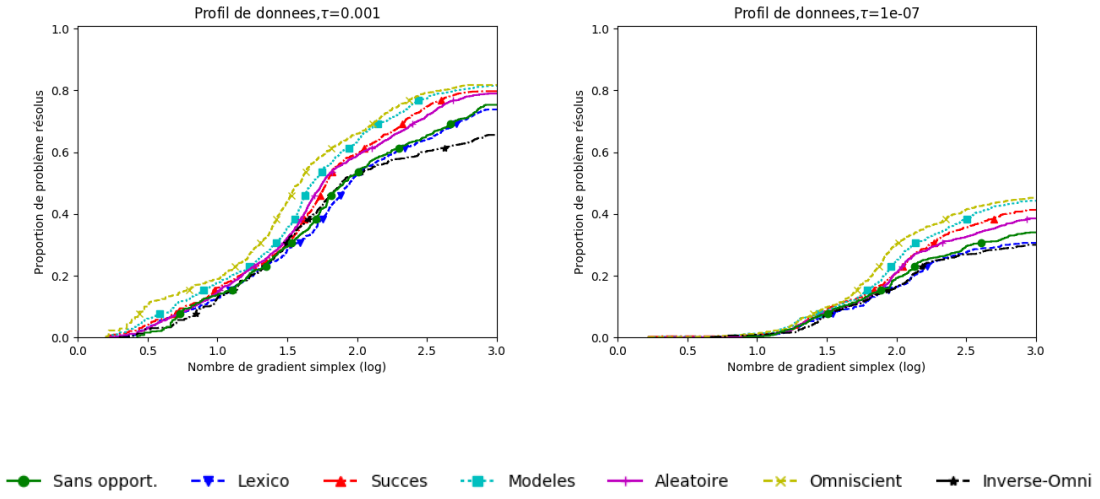


Figure 4.11 Comparaison sur tous les problèmes Moré-Wild avec MADS, avec minimum $q = \lceil \frac{n}{2} \rceil$ évaluations.

Les figures 4.10 et 4.11 montrent que l'opportunisme avec au minimum $p = 2$ succès et au minimum $q = \lceil \frac{n}{2} \rceil$ évaluations ne sont pas très différentes de l'opportunisme simple, outre la performance diminuée de la stratégie omnisciente. L'apport de cette stratégie opportuniste est limité, mais encore une fois, on peut y percevoir que la stratégie utilisant les modèles pour l'ordonnancement bénéficie des évaluations supplémentaires. On peut aussi en comprendre que, pour MADS avec opportunisme simple et une stratégie d'ordonnancement réalisable, les étapes de sondes comportent un nombre d'évaluations comparable à $\lceil \frac{n}{2} \rceil$, sans quoi l'allure des courbes ne serait pas aussi similaire.

Enfin, on compare l'impact des différentes déclinaisons de la stratégie opportuniste sur les problèmes analytiques de Moré-Wild. La stratégie d'ordonnancement utilisée est l'ordonnancement guidé par modèles quadratiques.

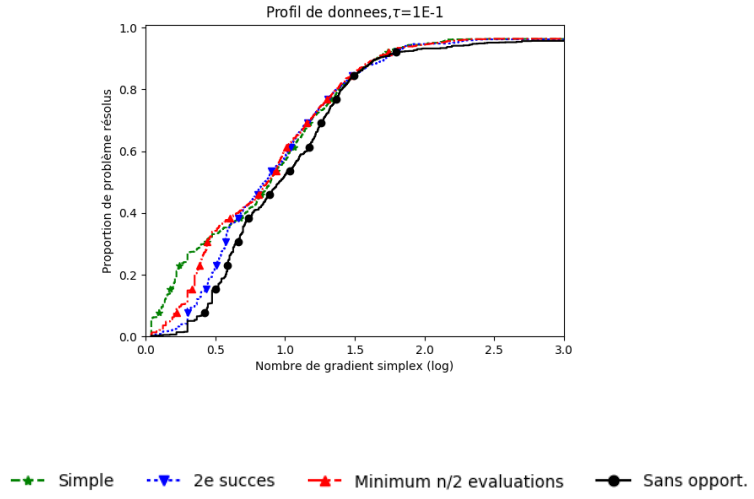


Figure 4.12 Comparaison des stratégies opportunistes sur tous les problèmes Moré-Wild avec MADS

La figure 4.12 montre que la stratégie opportuniste simple présente un gain de performance supérieur à bas budget d'évaluations. Cependant, cette conclusion n'est pas absolue puisque cet écart pourrait être le fruit d'une portion de problèmes bénéficiant de l'opportunisme simple par leur aspect lisse et différentiable. La stratégie opportuniste avec au minimum $\lceil \frac{n}{2} \rceil$ présente un gain semblable à l'opportunisme simple. On remarque que la stratégie opportuniste avec $p = 2$ succès présente un gain limité en performance à bas budget. L'impact de l'opportunisme est moins important que CS et plus important a bas budget qu'à haut budget si comparé à GPS.

En conclusion, MADS a sensiblement autant à gagner que GPS à utiliser une stratégie op-

portuniste simple et une stratégie d'ordonnancement telle que l'ordonnancement à l'aide de modèles quadratiques. Cette stratégie domine les stratégies aléatoires et en fonction de la direction du dernier succès, qui sont elles aussi des stratégies bénéfiques. La stratégie d'ordonnancement lexicographique n'est pas recommandée, étant donné que la méthode sans opportunisme semble avoir la même performance pour un comportement plus stable.

Une mise en garde est nécessaire au moment d'observer les courbes comparant les différentes stratégies sur les différents algorithmes. Les allures des courbes d'opportunisme simple et sans opportunismes pour les algorithmes CS et MADS des figures 4.4 et 4.12 pourraient porter à croire que CS opportuniste domine largement n'importe quelle déclinaison de MADS. Cependant, lors qu'on trace les profils comparant CS et MADS avec opportunisme tel qu'à la figure 4.13, cette illusion disparaît et laisse place à la simple comparaison de méthodes. Les profils de données utilisent $t_{p,s}$ comme métrique, qui corresponds au nombre d'évaluations nécessaires pour satisfaire le test de convergence (4.1). Or, ce test est effectué en admettant f^*, soit la meilleure solution obtenue par tous les solveurs, comme solution de convergence. Ainsi, la qualité des solutions f^* utilisées est uniquement relative aux algorithmes comparés dans le profil. Afin de bien caractériser l'impact de la stratégie algorithmique en soi, nous jugeons préférable de ne pas comparer les algorithmes entre eux mais de seulement faire les comparaisons entre les différentes variantes opportunistes des algorithmes. Le lecteur est avisé de ne pas se laisser tenter de comparer la performance des différents algorithmes.

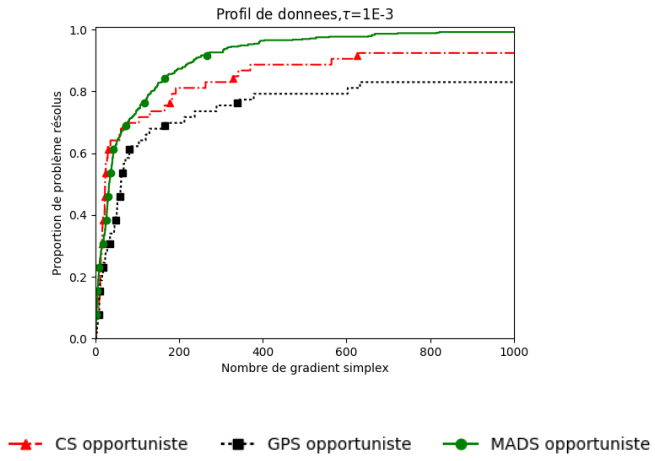


Figure 4.13 Comparaison de CS, GPS et MADS opportuniste

4.4.4 MADS par défaut

Afin d'observer l'impact de l'opportunisme et le gain possible en performance, les problèmes de l'ensemble Moré-Wild sont résolus à l'aide de NOMAD avec ses paramètres par défaut. La stratégie lexicographique n'est pas représentée, compte tenu d'un conflit existant au sein de l'implémentation avec l'utilisation de modèles quadratiques en tant que SEARCH.

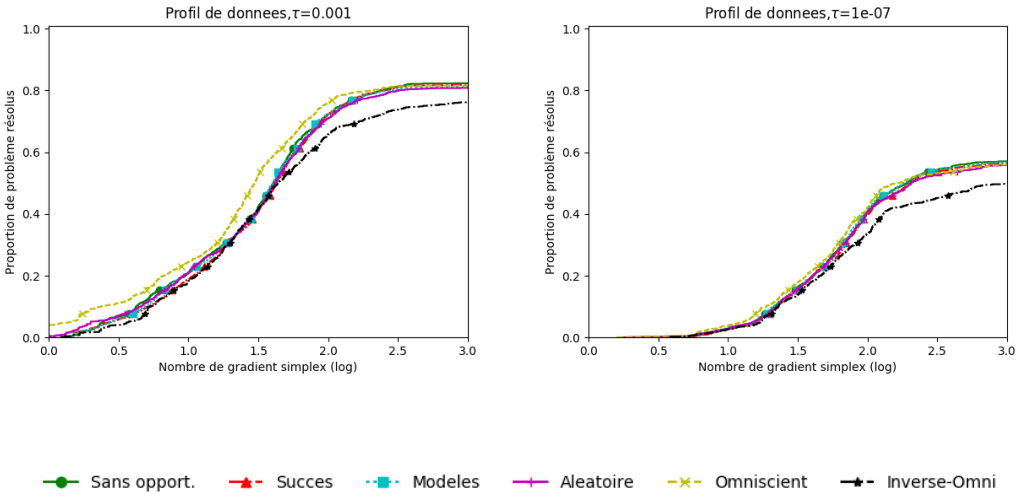


Figure 4.14 Comparaison sur tous les problèmes Moré-Wild avec MADS par défaut de NOMAD, avec opportunisme simple.

Dans la figure 4.14, on observe que le gain possible avec un ordonnancement parfait est réduit lorsque NOMAD est utilisé au meilleur de ses capacités. Seules les courbes des stratégies d'ordonnancement irréalisables se distinguent des autres. En observant la courbe omnisciente, on comprend que le gain possible est limité à un maximum de 15% de problèmes pour un budget donné. Cependant, il est important rappeler que l'ensemble de problèmes Moré-Wild est un ensemble de problèmes analytiques sans contraintes peu représentatif des problèmes encourus en optimisation de boîtes noires. Les figures pour NOMAD par défaut avec les deux autres types d'opportunisme sont omises, compte tenu de leur similarité avec la figure 4.14.

Enfin, on compare l'impact des différentes déclinaisons de la stratégie opportuniste sur les problèmes analytiques de Moré-Wild. La stratégie d'ordonnancement utilisée est l'ordonnancement guidé par modèles quadratiques.

La figure 4.15 montre qu'aucune stratégie opportuniste ne semble avoir un impact considérable sur MADS avec étapes de recherches heuristiques.

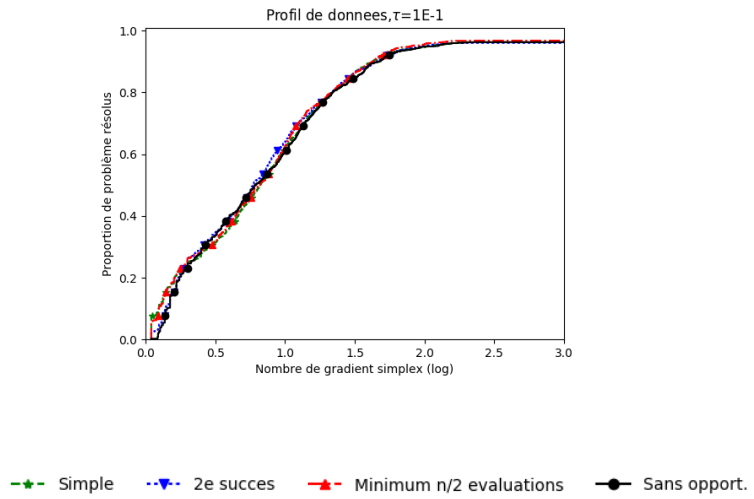


Figure 4.15 Comparaison des stratégies opportunistes sur tous les problèmes Moré-Wild avec MADS par défaut de NOMAD

4.4.5 GSS

Les problèmes Moré-Wild sont résolus avec HOPSPACK, qui est par défaut opportuniste, avec les deux stratégies d'ordonnancement déjà implémentées à même le logiciel. Une mise à niveau a du être fournie à HOPSPACK, inspirée de [16], consistant en l'obtention de

$$\Delta_j^0 = \begin{cases} \frac{|x_j^0|}{10} & \text{si } |x_j^0| \neq 0 \\ 1 & \text{si } |x_j^0| = 0 \end{cases}$$

où Δ_j^0 est la $j^{\text{ème}}$ composante du pas initial, et x_j^0 la $j^{\text{ème}}$ composante du point de départ x^0. Seul l'opportunisme simple est implémenté dans HOPSPACK. La figure 4.16 nous permet d'observer la comparaison de deux stratégies d'ordonnancement présente dans une autre implémentation que NOMAD. On observe que la stratégie aléatoire est nettement supérieure à la stratégie déterministe de HOPSPACK. On observe ici directement l'impact qu'un choix d'ordonnancement déterministe peut avoir sur la résolution. La mise en valeur arbitraire de certaines directions tend à nuire à l'algorithme. Il est à noter que le paramètre par défaut de HOPSPACK quant à l'ordonnancement est l'ordonnancement déterministe. On en conclut que la supériorité de l'ordonnancement aléatoire quant à l'ordonnancement déterministe se transmet dans diverses implémentations, et que l'ordonnancement aléatoire se distingue comme un choix judicieux nécessitant peu d'efforts.

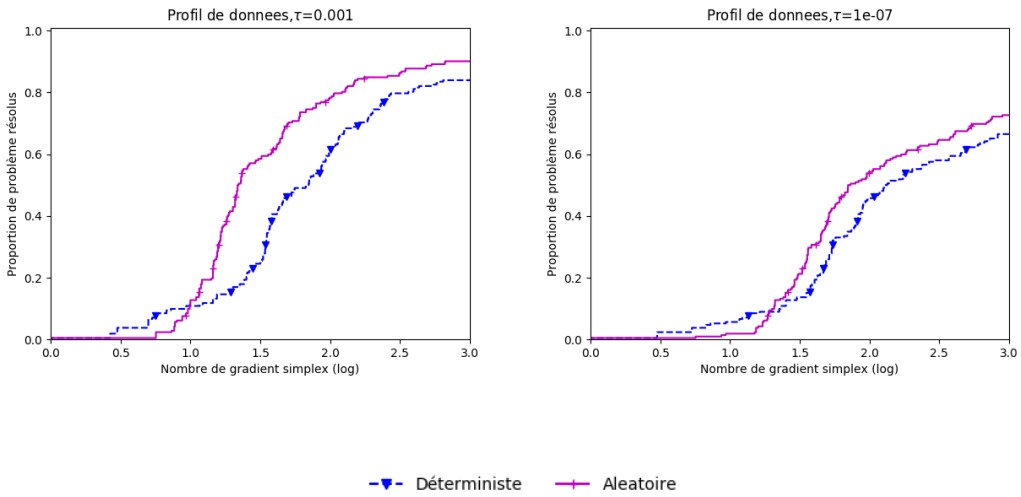


Figure 4.16 Comparaison sur tous les problèmes Moré-Wild avec GSS avec opportunisme simple.

4.4.6 IMFIL

L'implémentation utilisée de IMFIL nécessite que l'utilisateur fournisse des bornes, duquel en découle une mise à l'échelle pour la résolution. Les problèmes Moré-Wild étant non bornés, des bornes arbitraires ont dû être déterminées. Pour ce faire, on utilise

$$u_i = 10^{1+\lfloor \log(\max(|x_i^0|,|x_i^*|)) \rfloor}$$
$$l_i = -u_i$$

avec u le vecteur des bornes supérieures, l le vecteur des bornes inférieures, x^0 le point initial fourni avec le problème et x^* la solution trouvée par l'algorithme MADS par défaut de NOMAD sur le problème donné. Ainsi, on fournit à IMFIL un domaine dont les limites sont une puissance de 10, centré en 0, symétrique et qui contient assurément le point de départ et la solution. À la figure 4.17, on observe que IMFIL n'est pas en mesure de bénéficier pleinement de la stratégie opportuniste op de l'algorithme 9, définie à la section 3.3. On observe en premier lieu que le profil correspondant à la stratégie d'ordonnancement omnisciente offre un meilleur rendement que la méthode sans opportunisme pour un certain budget, et ce à précision $\tau = 1E-3$ seulement. Le profil de l'ordonnancement négatif-omniscient se comporte de façon très similaire à la méthode sans opportunisme. On explique ceci par la nature de IMFIL. Puisque la sonde opportuniste inverse-omnisciente permet l'évaluation d'un grand nombre de points, le BLS est effectué dans une direction comprenant plusieurs composantes

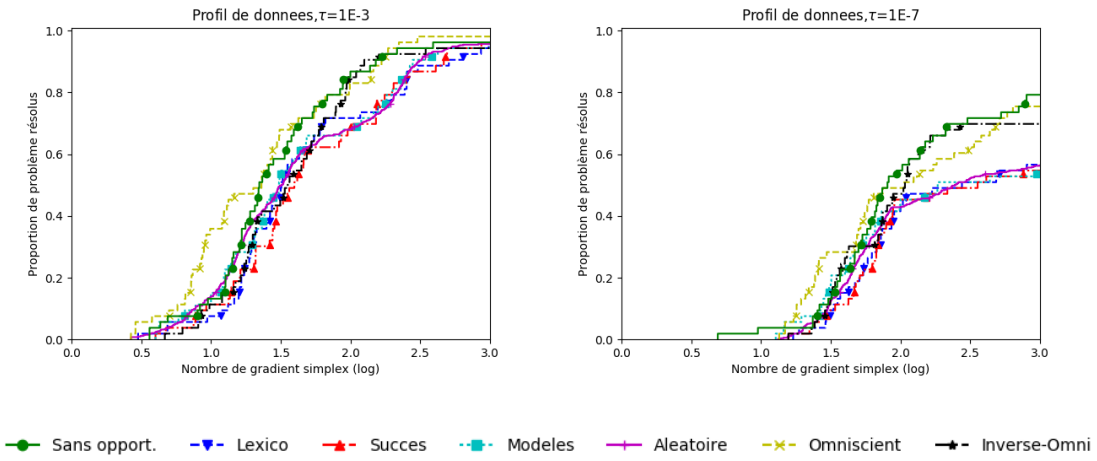


Figure 4.17 Comparaison sur tous les problèmes Moré-Wild avec IMFIL avec opportunisme op

non nulles.

Bien que la stratégie d'ordonnancement omnisciente semble procurer un gain en performance à basse précision, on observe que les stratégies réalisables nuisent aux résolutions. Aucune stratégie n'est en mesure de se démarquer comme étant un choix intéressant pour l'algorithme. À haute précision $\tau = 1E-7$, on remarque que la version opportuniste de IMFIL avec une stratégie d'ordonnancement réalisable n'est pas en mesure d'obtenir les solutions précises pour 50% des problèmes seulement, comparativement à près de 80% avec la méthode sans opporutnisme, ce qui témoigne que les meilleurs résultats f^* proviennent en majorité de celles-ci. La figure 4.18 témoigne de l'incompatibilité entre la stratégie opportuniste op et IMFIL. La méthode sans opportunisme résout la meilleure proportion de problèmes, peu importe le budget alloué. Les stratégies omniscientes et inverse-omnisciente ne sont pas discernables des autres stratégies. À haute précision, on remarque qu'aucune stratégie d'ordonnancement ne permet d'obtenir un rendement comparable à la méthode sans opportunisme.

Enfin, on compare l'impact des différentes déclinaisons de la stratégie opportuniste sur les problèmes analytiques de Moré-Wild. La stratégie d'ordonnancement utilisée est l'ordonnancement guidé par modèles quadratiques.

Les courbes des stratégies opportunistes utilisées dans IMFIL montrent qu'aucune version opportuniste de IMFIL n'équivaut à sa version originale. En conclusion, l'opportunisme implanté dans IMFIL n'est pas bénéfique à la méthode, tel que prédit par son auteur, puisqu'aucune combinaison de stratégie opportuniste et de stratégie d'ordonnancement réaliste n'est en

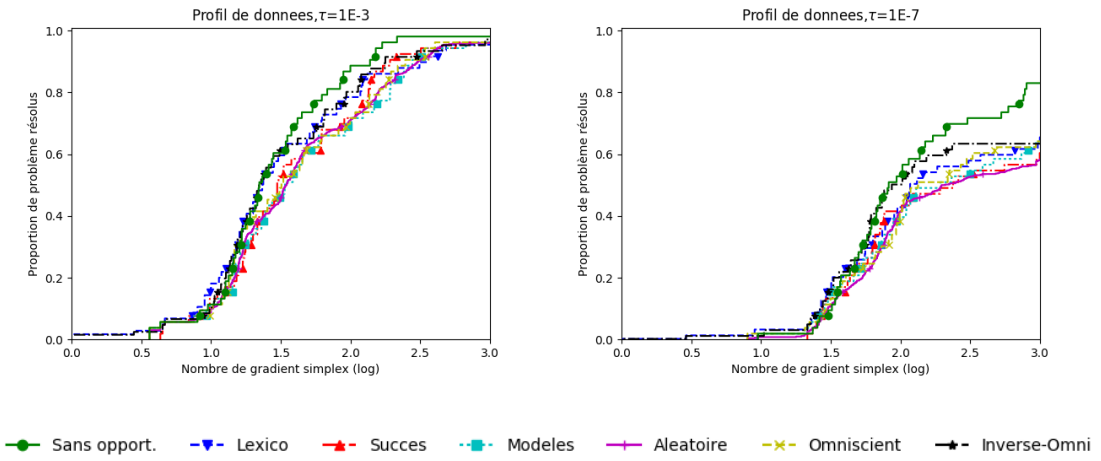


Figure 4.18 Comparaison sur tous les problèmes Moré-Wild avec IMFIL avec opportunisme od

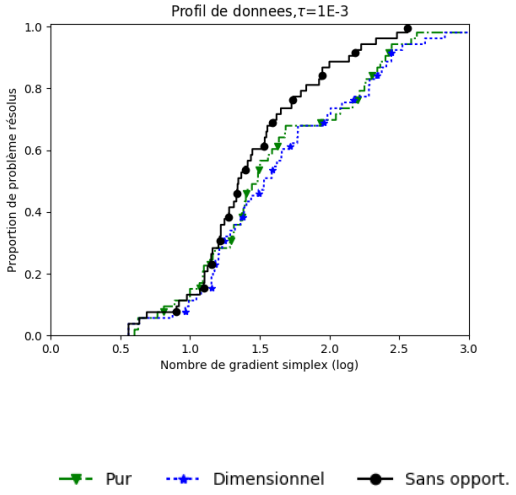


Figure 4.19 Comparaison des stratégies opportunistes sur tous les problèmes Moré-Wild avec IMFIL

mesure d'accélérer les résolutions.

4.5 Comparaison des stratégies d'ordonnancement sur les problèmes contraints

Pour l'ensemble de problèmes contraints, les profils de données sont tracés avec les seuils de tolérance $\tau = 10^{-1}$ et $\tau = 10^{-3}$, puisque les solutions trouvées par les algorithmes n'ont pas

la même reproductibilité que les problèmes Moré-Wild. Les paramètres p et q des stratégies opportunistes des définitions 3.3.1 et 3.3.2 sont respectivement $p = 2$ et $q = \lceil \frac{n}{2} \rceil$, à l'instar des comparaisons de la section 4.4. Les profils de données pour ces stratégies opportunistes sont fournis à l'annexe A. Seuls les algorithmes implémentés dans NOMAD, soit CS, GPS et MADS, pouvant gérer les contraintes à l'aide de la barrière progressive, sont testés sur ces problèmes. Chaque point initial utilisé est un point irréalisable.

4.5.1 CS et GPS

Les algorithmes CS et GPS sont jumelés dans cette section en vertu des résultats similaires que les profils de données permettent d'observer.

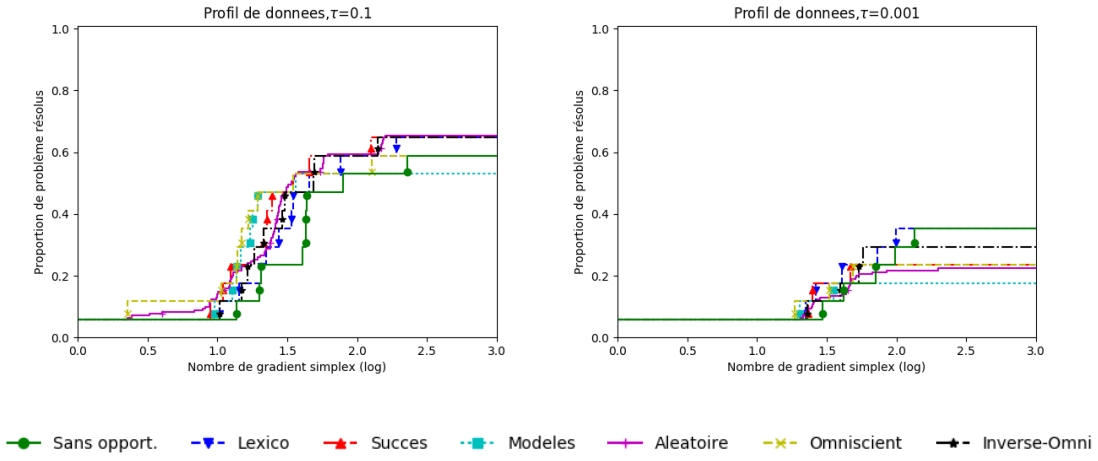


Figure 4.20 Comparaison sur tous les problèmes contraints avec CS avec opportunisme simple.

Premièrement, les figure 4.20 et 4.21 montrent que l'ensemble de problèmes utilisés est trop petit pour obtenir des courbes lisses. La tendance générale est qu'aucune stratégie d'ordonnancement jumelée à l'opportunisme simple pour CS et GPS n'est en mesure de se démarquer comme celle pouvant potentiellement apporter la meilleure amélioration. Cependant, en distinguant le profil correspondant à la méthode sans opportunisme à précision $\tau = 1E-1$, on remarque que celui-ci se trouve sous les courbes opportunistes, et ce pour tout budget d'évaluation. Ces observations sont valables pour les courbes des autres stratégies opportunistes, figurant sur les figures A.1, A.2, A.3 et A.4 de l'annexe A.

On en conclut que l'opportunisme simple est une pratique qui impacte positivement la résolution de problèmes contraints, sans qu'aucune stratégie d'ordonnancement ne soit favorisée.

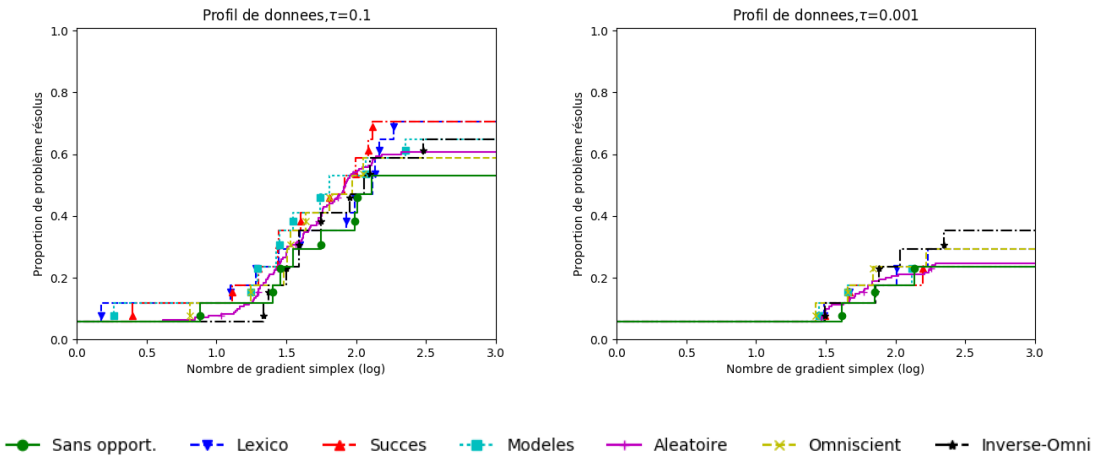


Figure 4.21 Comparaison sur tous les problèmes contraints avec GPS avec opportunisme simple.

4.5.2 MADS

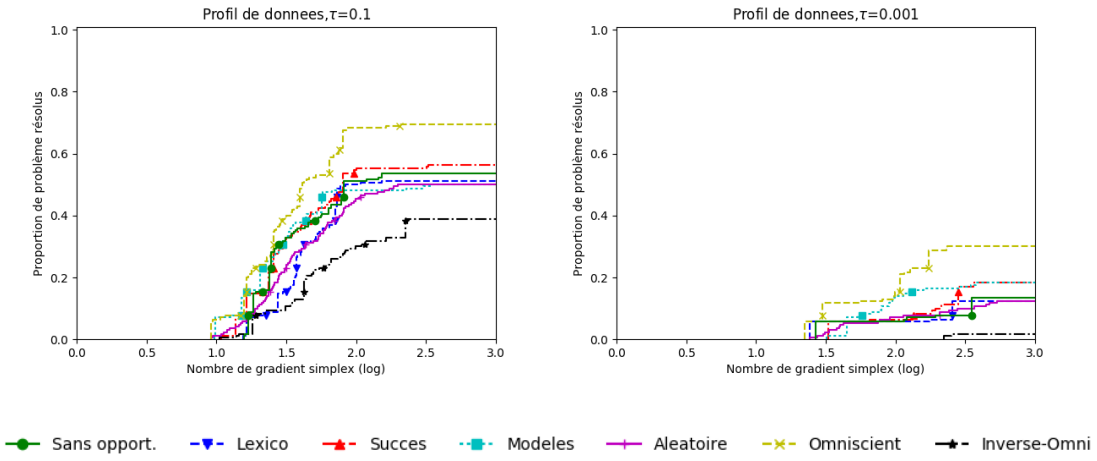


Figure 4.22 Comparaison sur tous les problèmes contraints avec MADS avec opportunisme simple.

La figure 4.22 permet d'observer une plus grande hiérarchie des stratégies d'ordonnancement qu'à la section 4.5.1. Les courbes sont plus lisses en vertu de la génération aléatoire de directions possible avec MADS, contrairement à CS et GPS, pour lesquels les résolutions sont parfaitement reproductibles indépendamment du germe aléatoire.

L'ordonnancement omniscient jumelé à l'opportunisme simple semble offrir une nette amé-

lioration de l'algorithme MADS en situation de barrière progressive. La performance de l'ordonnancement inverse-omniscient, quant à lui, démontre que l'intuition sur une mauvaise stratégie d'ordonnancement est toujours valide pour les problèmes contraints.

Les stratégies d'ordonnancement réalisables sont cependant de performance comparable avec la méthode sans opportunisme. La stratégie aléatoire et la stratégie lexicographique semblent nuire à l'algorithme. Les autres stratégies opportunistes ne semblent pas avoir le même effet sur la sonde, tel que les figures A.5 et A.6 en témoignent. On en conclu qu'un gain en performance serait envisageable avec l'implémentation d'un ordonnancement guidé par un outil plus élaboré que les modèles quadratiques, mais que dans l'état actuel, ceux-ci permettent à l'opportunisme de ne pas nuire à la sonde de MADS.

4.5.3 MADS par défaut

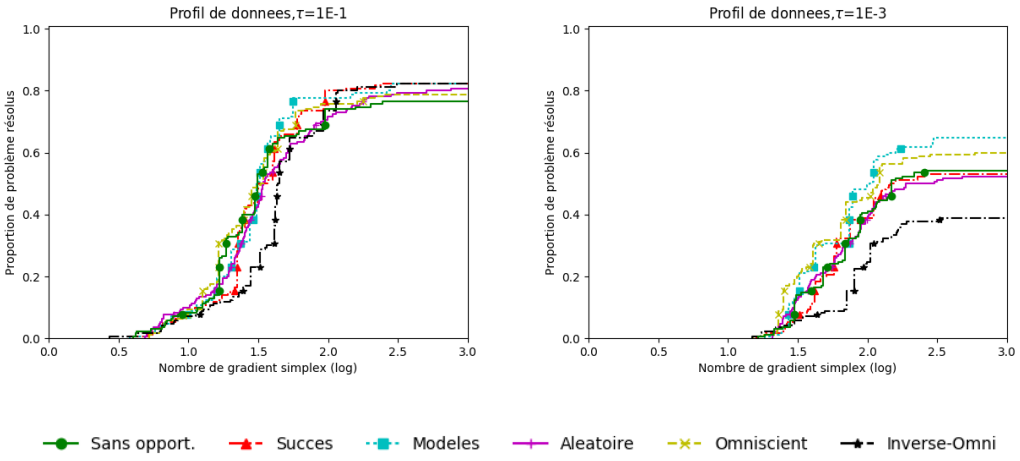


Figure 4.23 Comparaison sur tous les problèmes contraints avec MADS par défaut de NOMAD avec opportunisme simple.

La figure 4.23 montre que l'opportunisme a peu d'effet si on utilise MADS par défaut de NOMAD. Les spécificités de OrthoMADS ainsi que l'influence d'une étape de SEARCH noient l'impact de l'opportunisme sur cet ensemble de problèmes. Cependant, on remarque que le profil correspondant à l'ordonnancement inverse-omniscient nuit quelque peu les résolutions de problèmes, notamment au niveau de la proportion de problèmes résolus avec une précision de $\tau = 1E - 3$. C'est avec cette précision qu'on observe que l'ordonnancement guidé par modèles semble bénéficier à l'algorithme.

4.6 Comparaison des stratégies d'ordonnancement sur STYRENE

Les stratégies d'ordonnancement sont comparées à l'aide de graphes de convergence, dans lesquels les courbes correspondantes aux stratégies omniscientes et inverse-omnisciente sont présentes. Avec celles-ci figure la courbe correspondante à une stratégie comparée, de façon à observer la performance de cette stratégie relativement à la performance des stratégies de comparaison. Le point de départ des résolutions est irréalisable et déterminé suite à un échantillonnage du domaine de la boîte noire. Puisque CS et GPS ne contiennent pas d'aspect aléatoire, chaque résolution d'un problème avec un point de départ x^0 est identique. Les algorithmes étudiés sont limités à MADS et à MADS par défaut de NOMAD.

4.6.1 MADS

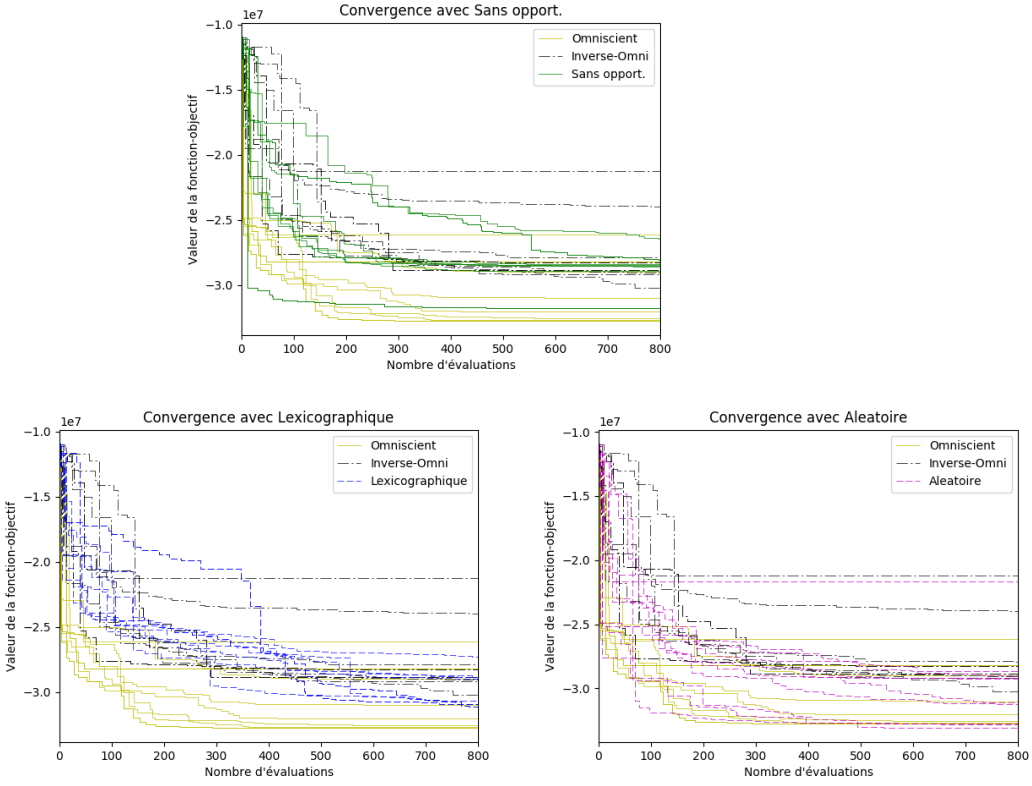


Figure 4.24 Comparaison sur STYRENE avec MADS et opportunisme simple, sonde complète, stratégie lexicographique et stratégie aléatoire.

Premièrement, le premier graphe de la convergence de figure 4.24 présente les courbes des stratégies de comparaison ainsi que la sonde complète. On observe que la convergence est significativement plus rapide avec la stratégie omnisciente qu'avec la stratégie inverse-omnisciente. De

plus, on observe que la solution vers laquelle la stratégie omnisciente converge est beaucoup plus satisfaisante que celle obtenue par la stratégie inverse-omnisciente. La sonde complète a un comportement qui se rapproche davantage à celui de la stratégie inverse-omnisciente. Il s'agit d'une indication de la performance de la barrière progressive jumelée à la stratégie omnisciente, pour laquelle la mise à jour de x^{inf} est effectuée en admettant le point ayant la plus petite violation de contrainte. Inversement, l'algorithme de la barrière progressive de la sonde complète met à jour $x^{\text{inf}} \leftarrow t$, avec $t \in \arg\min(T)$ et $h(t) < h_{\max}$, où T est l'ensemble des points non-dominés. Il en résulte que le premier point réalisable n'est pas le même.

La vitesse de résolution de la stratégie lexicographique s'apparente à celle de la stratégie inverse-omnisciente. Cependant, on remarque une certaine proportion de courbes convergeant vers une solution intermédiaire entre la solution de la sonde complète et celle de la stratégie omnisciente.

La stratégie aléatoire semble apporter une amélioration de performance en comparaison avec la sonde complète. La vitesse de résolution est similaire, mais la qualité de la solution obtenue est plus souvent semblable à celle obtenue avec la stratégie omnisciente.

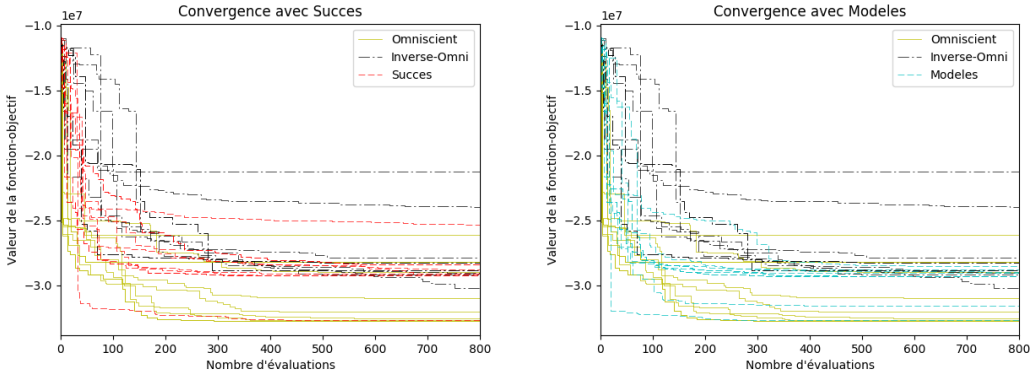


Figure 4.25 Comparaison sur STYRENE avec MADS et opportunisme simple, stratégie avec direction du dernier succès et stratégie guidée par modèles.

La stratégie d'ordonnancement avec la direction du dernier succès ne permet pas à l'algorithme de converger vers la solution optimale déterminée avec la stratégie omnisciente, tel qu'observé sur la figure 4.25. Il en est de même pour la stratégie avec modèles. La fonction objectif et les contraintes de la boîte noire STYRENE sont difficilement modélisables à l'aide de fonctions quadratiques. Une fonction substitut plus représentative pourrait permettre à la solution de se rapprocher de la courbe omnisciente. Cependant, la vitesse de résolution pour la stratégie avec modèles est plus grande qu'avec la stratégie de la direction du dernier succès.

Avec l'étape de sonde seule de MADS, l'influence de la stratégie d'ordonnancement est importante pour la qualité de la solution obtenue et la vitesse de résolution. La stratégie lexicographique ralenti la résolution, mais son aspect quasi aléatoire peut entraîner l'algorithme vers une meilleure solution. La stratégie aléatoire présente une vitesse de résolution significativement plus rapide que la stratégie lexicographique, mais semblable à la sonde complète. Cependant, son aspect aléatoire peut permettre à l'algorithme de converger vers une meilleure solution. La stratégie de la direction de succès mène à la même solution que la sonde complète avec un léger gain en vitesse de résolution. La stratégie utilisant les modèles semble celle qui amène le meilleur gain en vitesse de résolution. Par contre, la solution vers laquelle 80% des résolutions converge n'est pas une amélioration quant à la sonde complète. Puisque le point de départ x^0 est irréalisable, on en conclu que la détermination du x^{inf} à chaque itération a un effet important sur la qualité de la solution obtenue, et qu'aucune stratégie réaliste utilisée ici ne permet d'ordonnancer les points de l'ensemble P^k de façon à mettre de l'avant le meilleur candidat pour x^{inf}.

4.6.2 MADS par défaut

Il est à noter que la stratégie lexicographique n'est pas utilisée pour les comparaisons avec étape de SEARCH. Dans la figure 4.26, on remarque que la sonde complète converge vers

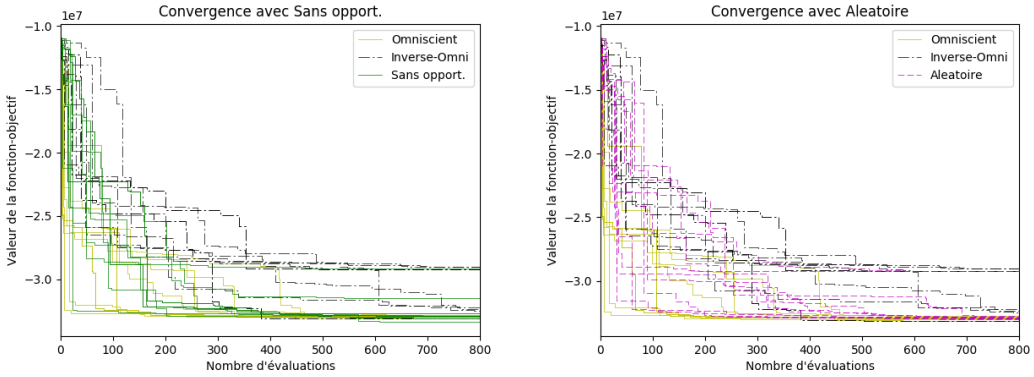


Figure 4.26 Comparaison sur STYRENE avec MADS par défaut et opportunisme simple, sonde complète et stratégie aléatoire.

la même solution que les stratégies de comparaison. La vitesse de résolution de la sonde complète est moins satisfaisante que celle de l'ordonnancement omniscient. La différence est beaucoup moins grande que celle observée avec MADS. Les courbes de la stratégie aléatoire montrent que parfois la vitesse est supérieure à la sonde complète, parfois inférieure. Elles convergent en majorité vers la même solution.

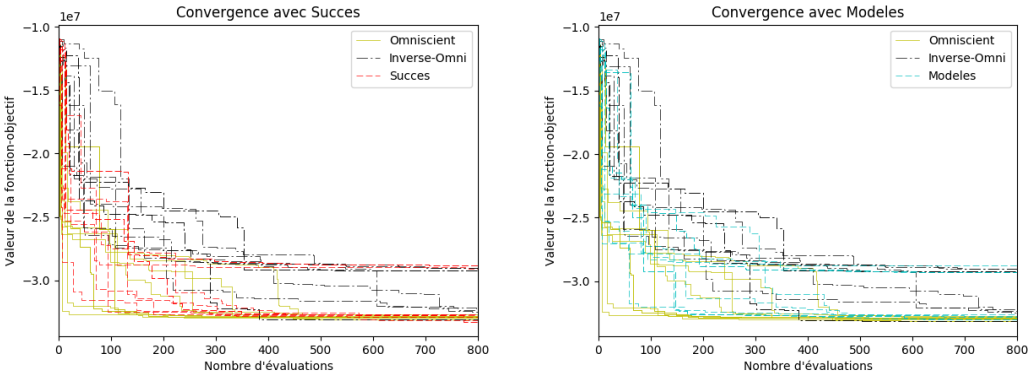


Figure 4.27 Comparaison sur STYRENE avec MADS par défaut et opportunisme simple, stratégie avec direction du dernier succès et stratégie guidée par modèles.

La stratégie avec la direction du dernier succès présente une performance intéressante. Jumelée avec l'étape de SEARCH de MADS par défaut, on observe que la stratégie a une performance similaire à celle de l'ordonnancement par modèles. L'ordonnancement guidé par modèles n'est pas aussi performant en présence de SEARCH. Une mauvaise géométrie de points pour l'interpolation issue de la détermination de points seulement par les modèles peut être à l'origine de cette baisse d'impact.

Les impacts des stratégies d'ordonnancement sur STYRENE sont moins prononcés qu'avec MADS sans SEARCH de la section 4.6.1. Les solutions trouvées sont plus robustes avec la SEARCH et l'ordonnancement influence en général seulement la vitesse de résolution. La stratégie d'ordonnancement utilisant la direction du dernier succès offre le meilleur rendement, mais l'impact de l'opportunisme jumelé à la SEARCH reste minime.

CHAPITRE 5 CONCLUSION

5.1 Synthèse des travaux

Ce travail comporte une comparaison de stratégies algorithmiques implémentables dans certains algorithmes d'optimisation sans dérivées. Des algorithmes d'optimisation sans dérivées identifiés pour la comparaison de ces stratégies, quatre sont des algorithmes de recherche directe directionnels, soient la recherche par coordonnée, la recherche par motifs généralisée, la recherche par ensemble générateur et la recherche directe par treillis adaptatif. Un algorithme de recherche directe basée sur les dérivées du stencil, soit l'algorithme de filtrage implicite.

La principale stratégie algorithmique étudiée, la stratégie opportuniste, consiste en l'arrêt prématuré d'une étape de l'algorithme dans laquelle il y a une succession d'évaluations de la fonction objectif à effectuer. La stratégie est implémentée dans les étapes de sonde des algorithmes identifiés préalablement. L'arrêt de la sonde est effectué après l'évaluation d'un point dont la valeur de la fonction objectif à ce point est inférieure à la valeur minimale obtenue précédemment dans la résolution. Le critère d'arrêt peut être complexifié en y incorporant un nombre minimal de succès ou un nombre minimal d'évaluations.

L'arrêt prématuré implique que l'ordre d'évaluation séquentielle des points importe sur la performance. Plusieurs stratégies d'ordonnancement sont identifiées afin d'être comparées. Les stratégies sont l'ordonnancement lexicographique, l'ordonnancement aléatoire, l'ordonnancement en fonction de la direction du dernier succès et l'ordonnancement en fonction d'un modèle. À ces stratégies s'ajoute les stratégies de comparaison, soient les stratégies omniscientes et inverse-omniscientes.

L'ensemble de ces stratégies algorithmiques est testé sur un ensemble de problèmes analytiques, un ensemble de problèmes contraints et la boîte noire STYRENE. Les tests numériques révèlent que, des différentes stratégies opportunistes, l'opportunisme après un seul succès est celle qui améliore le plus les méthodes. On y observe que l'impact de la stratégie opportuniste décroit avec le raffinement de l'étape de sonde de l'algorithme utilisé pour les méthodes directes directionnelles. Son impact sur la méthode de filtrage implicite est en apparence négatif. On y observe aussi que l'ordonnancement par modèles est la stratégie d'ordonnancement qui offre le meilleur rendement, suivi de la stratégie aléatoire et la stratégie de la direction du dernier succès. En général, on observe que l'opportunisme avec ordonnancement lexicographique nuit à la résolution.

Les stratégies de comparaison permettent d'affirmer que les stratégies utilisées dans cette

étude ne sont pas optimales et que l'opportunisme jumelé à une mauvaise stratégie d'ordonnancement peut nuire à la résolution de problèmes. Les résolutions de problèmes contraints montrent que l'interaction entre la stratégie opportuniste et l'algorithme de la barrière progressive rend le classement des performances moins évident. Ce travail mène à la conclusion que l'opportunisme, jumelé à une stratégie d'ordonnancement adéquate, peut être bénéfique aux méthodes de recherche directe directionnelle, mais qu'il s'agit d'un outil à utiliser avec soin.

5.2 Limitations de la solution proposée

Une boîte noire est un problème souvent long, bruité, non différentiable et instable. Les problèmes analytiques proposés par Moré et Wild [62] ne sont pas représentatifs des problèmes pour lesquels les algorithmes de résolution de boîte noire existent. Leurs aspects non différentiables sont limités et les amplitudes des bruits introduits sont faibles.

L'ensemble de problèmes contraints avec points de départ non réalisables n'est pas suffisamment volumineux pour obtenir une tendance nette pour la performance de chaque stratégie d'ordonnancement identifiée sur cette catégorie de problème. L'ensemble de boîtes noires est limité à une seule, et les tendances sur ces problèmes complexes ne peuvent être validées ou infirmées sur d'autres boîtes noires.

5.3 Améliorations futures

Afin d'élargir les connaissances sur l'impact de la stratégie opportuniste en soi, il est envisageable d'identifier davantage de méthodes d'optimisation dans lesquelles elle pourrait être implémentée. Dans [46], les auteurs introduisent l'algorithme DiRect. Cet algorithme semble à première vue être un candidat qui possède les caractéristiques nécessaires pour l'implémentation de la stratégie. D'autres méthodes basées sur la recherche par coordonnées, notamment PSwarm [76], figurent dans les algorithmes susceptibles d'être compatible avec la stratégie opportuniste.

Dans l'esprit de GSS, un succès est accepté seulement s'il entraîne une diminution suffisante de la fonction objectif. Ce critère de diminution suffisante pourrait être appliqué seulement comme critère d'opportunisme dans des algorithmes dont l'analyse de convergence ne nécessite pas la diminution suffisante. De cette façon, les succès marginaux seraient ignorés.

Les stratégies de comparaison montrent que le gain de performance de l'opportunisme pourrait être supérieur avec une stratégie d'ordonnancement plus efficace. Une utilisation future

des modèles plus sophistiqués pour guider l'ordonnancement, tels que ceux proposés dans [73], serait à envisager. Davantage de stratégies d'ordonnancement pourraient être identifiées. Dans [26], les auteurs mentionnent une stratégie d'ordonnancement déterministe comme la stratégie lexicographique, qui consisterait en l'ordonnancement des points sans remise à zéro de l'ordre des points à chaque nouvelle sonde. De cette façon, aucune direction ne serait priorisée et la probabilité de choisir deux nouveaux centres de sonde issus de la même direction serait nulle, ce que la stratégie aléatoire ne garantit pas. Les stratégies présentes dans les travaux de Custodio et al. [29] pourraient être testées sur une série d'algorithmes.

Dans ce travail, l'ordonnancement en situation de barrière progressive est fixé à une dominance élaborée à la section 3.2.4. Différentes relations d'ordre entre les points pourraient être étudiées pour déterminer les impacts de l'ordre d'évaluation de ces points en concert avec l'utilisation de la stratégie opportuniste.

RÉFÉRENCES

[1] M.A. Abramson, C. Audet, G. Couture, J.E. Dennis, Jr., S. Le Digabel, et C. Tribes. The NOMAD project. Software available at `https://www.gerad.ca/nomad`, 2015.

[2] M.A. Abramson, C. Audet, et J.E. Dennis, Jr. Generalized pattern searches with derivative information. *Mathematical Programming*, Series B, 100(1) :3–25, 2004.

[3] M.A. Abramson, C. Audet, J.E. Dennis, Jr., et S. Le Digabel. OrthoMADS : A Deterministic MADS Instance with Orthogonal Directions. *SIAM Journal on Optimization*, 20(2) :948–966, 2009.

[4] L. Adjengue, C. Audet, et I. Ben Yahia. A variance-based method to rank input variables of the Mesh Adaptive Direct Search algorithm. *Optimization Letters*, 8(5) :1599–1610, 2014.

[5] L. Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16(1) :1–3, 1966.

[6] C. Audet, V. Béchard, et S. Le Digabel. Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search. *Journal of Global Optimization*, 41(2) :299–318, 2008.

[7] C. Audet, C.-K. Dang, et D. Orban. Efficient use of parallelism in algorithmic parameter optimization applications. *Optimization Letters*, 7(3) :421–433, 2013.

[8] C. Audet et J.E. Dennis, Jr. A pattern search filter method for nonlinear programming without derivatives. *SIAM Journal on Optimization*, 14(4) :980–1010, 2004.

[9] C. Audet et J.E. Dennis, Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on Optimization*, 17(1) :188–217, 2006.

[10] C. Audet et J.E. Dennis, Jr. A Progressive Barrier for Derivative-Free Nonlinear Programming. *SIAM Journal on Optimization*, 20(1) :445–472, 2009.

[11] C. Audet, J.E. Dennis, Jr., et S. Le Digabel. Parallel space decomposition of the mesh adaptive direct search algorithm. *SIAM Journal on Optimization*, 19(3) :1150–1170, 2008.

[12] C. Audet et W. Hare. *Derivative-Free and Blackbox Optimization*. Springer Series in Operations Research and Financial Engineering. Springer International Publishing, 2017.

[13] C. Audet, A. Ianni, S. Le Digabel, et C. Tribes. Reducing the Number of Function Evaluations in Mesh Adaptive Direct Search Algorithms. *SIAM Journal on Optimization*, 24(2) :621–642, 2014.

[14] C. Audet et M. Kokkolaras. Blackbox and derivative-free optimization : theory, algorithms and applications. *Optimization and Engineering*, 17(1) :1–2, 2016.

[15] C. Audet, S. Le Digabel, et C. Tribes. NOMAD user guide. Rapport technique G-2009-37, Les cahiers du GERAD, 2009.

[16] C. Audet, S. Le Digabel, et C. Tribes. Dynamic scaling in the mesh adaptive direct search algorithm for blackbox optimization. *Optimization and Engineering*, 17(2) :333–358, 2016.

[17] C. Audet et C. Tribes. Mesh-based nelder-mead algorithm for inequality constrained optimization. Rapport technique Les Cahiers du GERAD G-2017-90, 2017.

[18] I. Bongartz, A.R. Conn, N. Gould, et Ph.L. Toint. CUTE : constrained and unconstrained testing environment. *ACM Transactions on Mathematical Software*, 21(1) :123–160, 1995.

[19] A.J. Booker, J.E. Dennis, Jr., P.D. Frank, D.B. Serafini, V. Torczon, et M.W. Trosset. A Rigorous Framework for Optimization of Expensive Functions by Surrogates. *Structural and Multidisciplinary Optimization*, 17(1) :1–13, 1999.

[20] C.G. Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of Computation*, 19(92) :577–593, 1965.

[21] X. Chen et N. Wang. Optimization of short-time gasoline blending scheduling problem with a DNA based hybrid genetic algorithm. *Chemical Engineering and Processing : Process Intensification*, 49(10) :1076–1083, 2010.

[22] A.R. Conn, N.I.M. Gould, et Ph.L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. SIAM, 2000.

[23] A.R. Conn et S. Le Digabel. Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. *Optimization Methods and Software*, 28(1) :139–158, 2013.

[24] A.R. Conn, K. Scheinberg, et L.N. Vicente. Geometry of interpolation sets in derivative free optimization. *Mathematical Programming*, 111(1–2) :141–172, 2008.

[25] A.R. Conn, K. Scheinberg, et L.N. Vicente. Geometry of sample sets in derivative free optimization : Polynomial regression and underdetermined interpolation. *IMA Journal of Numerical Analysis*, 28(4) :721–749, 2008.

[26] A.R. Conn, K. Scheinberg, et L.N. Vicente. *Introduction to Derivative-Free Optimization*. MOS-SIAM Series on Optimization. SIAM, Philadelphia, 2009.

[27] I.D. Coope et C.J. Price. On the convergence of grid-based methods for unconstrained optimization. *SIAM Journal on Optimization*, 11(4) :859–869, 2001.

[28] T.H. Cormen, C.E. Leiserson, R.L. Rivest, et C. Stein. *Introduction to Algorithms, Second Edition*, pages 873–876. MIT Press and McGraw-Hill, 2001.

[29] A.L. Custódio, J.E. Dennis, Jr., et L.N. Vicente. Using simplex gradients of nonsmooth functions in direct search methods. *IMA Journal of Numerical Analysis*, 28(4) :770–784, 2008.

[30] A.L. Custódio, H. Rocha, et L.N. Vicente. Incorporating minimum Frobenius norm models in direct search. *Computational Optimization and Applications*, 46(2) :265–278, 2010.

[31] A.L. Custódio et L.N. Vicente. Using Sampling and Simplex Derivatives in Pattern Search Methods. *SIAM Journal on Optimization*, 18(2) :537–555, 2007.

[32] C. Davis. Theory of positive linear dependence. *American Journal of Mathematics*, 76 :733–746, 1954.

[33] E.D. Dolan et J.J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2) :201–213, 2002.

[34] E. Fermi et N. Metropolis. Numerical solution of a minimum problem. Los Alamos Unclassified Report LA–1492, Los Alamos National Laboratory, Los Alamos, USA, 1952.

[35] R. Fletcher. Function minimization without evaluating derivatives – a review. *The Computer Journal*, 8(1) :33–41, 1965.

[36] G.H. Golub et C.F. Van Loan. *Matrix Computations*. The John Hopkins University Press, Baltimore and London, third edition, 1996.

[37] N. Gould et J. Scott. A note on performance profiles for benchmarking software. *ACM Transactions on Mathematical Software*, 43(1) :15 :1–15 :5, May 2016.

[38] N.I.M. Gould, D. Orban, et Ph.L. Toint. CUTEr (and SifDec) : A constrained and unconstrained testing environment, revisited. *ACM Transactions on Mathematical Software*, 29(4) :373–394, 2003.

[39] G.A. Gray et T.G. Kolda. Algorithm 856 : APPSPACK 4.0 : Asynchronous parallel pattern search for derivative-free optimization. *ACM Transactions on Mathematical Software*, 32(3) :485–507, 2006.

[40] J.D. Griffin, T.G. Kolda, et R.M. Lewis. Asynchronous parallel generating set search for linearly-constrained optimization. *SIAM Journal on Scientific Computing*, 30(4) :1892–1924, 2008.

[41] I. Griva, S.G. Nash, et A. Sofer. *Linear and Nonlinear Optimization*. Society for Industrial and Applied Mathematics, 2009.

[42] W. Hock et K. Schittkowski. *Test Examples for Nonlinear Programming Codes*, volume 187 of *Lecture Notes in Economics and Mathematical Systems*. Springer, Berlin, Germany, 1981.

[43] R. Hooke et T.A. Jeeves. “Direct Search" Solution of Numerical and Statistical Problems. *Journal of the Association for Computing Machinery*, 8(2) :212–229, 1961.

[44] P.D. Hough, T.G. Kolda, et V. Torczon. Asynchronous parallel pattern search for nonlinear optimization. *SIAM Journal on Scientific Computing*, 23(1) :134–156, 2001.

[45] A.S. Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the Association for Computing Machinery*, 5(4) :339–342, 1958.

[46] D.R. Jones, C.D. Perttunen, et B.E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Application*, 79(1) :157–181, 1993.

[47] C.T. Kelley. *Iterative Methods for Optimization*. Number 18 in Frontiers in Applied Mathematics. SIAM, Philadelphia, 1999.

[48] C.T. Kelley. *Implicit Filtering*. Society for Industrial and Applied Mathematics, Philadephia, PA, 2011.

[49] C.T. Kelley. Users’ guide for imfil version 1.0. Rapport technique, May 2011.

[50] S. Kitayama, M. Arakawa, et K. Yamazaki. Sequential approximate optimization using radial basis function network for engineering optimization. *Optimization and Engineering*, 12(4) :535–557, 2011.

[51] T.G. Kolda, R.M. Lewis, et V. Torczon. Optimization by direct search : New perspectives on some classical and modern methods. *SIAM Review*, 45(3) :385–482, 2003.

[52] S. Le Digabel. Algorithm 909 : NOMAD : Nonlinear Optimization with the MADS algorithm. *ACM Transactions on Mathematical Software*, 37(4) :44 :1–44 :15, 2011.

[53] S. Le Digabel et S.M. Wild. A Taxonomy of Constraints in Simulation-Based Optimization. Rapport technique G-2015-57, Les cahiers du GERAD, 2015.

[54] R.M. Lewis et V. Torczon. Rank ordering and positive bases in pattern search algorithms. Rapport technique 96–71, Institute for Computer Applications in Science and Enginee-

ring, Mail Stop 132C, NASA Langley Research Center, Hampton, Virginia 23681–2199, 1996.

[55] R.M. Lewis et V. Torczon. Pattern search algorithms for bound constrained minimization. *SIAM Journal on Optimization*, 9(4) :1082–1099, 1999.

[56] R.M. Lewis et V. Torczon. Pattern search methods for linearly constrained minimization. *SIAM Journal on Optimization*, 10(3) :917–941, 2000.

[57] S. Lucidi et M. Sciandrone. On the Global Convergence of Derivative-Free Methods for Unconstrained Optimization. *SIAM Journal on Optimization*, 13(1) :97–116, 2002.

[58] L. Lukšan et J. Vlček. Test problems for nonsmooth unconstrained and linearly constrained optimization. Rapport technique V-798, ICS AS CR, 2000.

[59] M. Marazzi et J. Nocedal. Wedge trust region methods for derivative free optimization. *Mathematical Programming*, 91(2) :289–305, 2002.

[60] H. D. Mittelmann. Benchmarking interior point lp/qp solvers. *Optimization Methods and Software*, 11 :655–670, 1999.

[61] M.J. Montes, A. Abánades, J.M. Martínez-Val, et M. Valdés. Solar multiple optimization for a solar-only thermal power plant, using oil as heat transfer fluid in the parabolic trough collectors. *Solar Energy*, 83(12) :2165–2176, 2009.

[62] J.J. Moré et S.M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM Journal on Optimization*, 20(1) :172–191, 2009.

[63] S.G. Nash et A. Sofer. *Linear and Nonlinear Programming*. McGraw–Hill, New York, 1996.

[64] J.A. Nelder et R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4) :308–313, 1965.

[65] J. Nocedal et S.J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, New York, 1999.

[66] J. Nocedal et S.J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, Berlin, second edition, 2006.

[67] J.M. Ortega et W.C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970. Reissued in 2000 by SIAM Publications, Philadelphia, as Vol. 30 in the series Classics in Applied Mathematics.

[68] F. Pigache, F. Messine, et B. Nogarède. Optimal design of piezoelectric transformers : a rational approach based on an analytical model and a deterministic global optimization. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, 54 :1293–1302, 2007.

[69] T. D. Plantenga. Hopspack 2.0 user manual. Rapport technique SAND2009-6265, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, October 2009.

[70] C. Poissant. Exploitation d'une structure monotone en recherche directe pour l'optimisation de boîtes grises. Mémoire de maitrise, École Polytechnique de Montréal, 2018.

[71] E. Polak et M. Wetter. Generalized pattern search algorithms with adaptive precision function evaluations. Department of Electrical Engineering, University of California at Berkeley, Berkeley, CA 94720 and Simulation Research Group, Building Technologies Department, Environmental Energy Technologies Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720., 2001.

[72] L.M. Rios et N.V. Sahinidis. Derivative-free optimization : a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3) :1247–1293, 2013.

[73] B. Talgorn, C. Audet, S. Le Digabel, et M. Kokkolaras. Locally weighted regression models for surrogate-assisted design optimization. *Optimization and Engineering*, 19(1) :213–238, 2018.

[74] J. Tao et N. Wang. Dna double helix based hybrid ga for the gasoline blending recipe optimization problem. *Chemical Engineering & Technology*, 31(3) :440–451, 2008.

[75] V. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7(1) :1–25, 1997.

[76] A.I.F. Vaz et L.N. Vicente. A particle swarm pattern search method for bound constrained global optimization. *Journal of Global Optimization*, 39(2) :197–219, 2007.

[77] H. Zhang, A. R. Conn, et K. Scheinberg. A derivative-free algorithm for least-squares minimization. *SIAM Journal on Optimization*, 20(6) :3555–3576, 2010.

[78] J. Zhao et N. Wang. A bio-inspired algorithm based on membrane computing and its application to gasoline blending scheduling. *Computers & Chemical Engineering*, 35(2) :272–283, 2011.

ANNEXE A PROFILS SUPPLÉMENTAIRES POUR LA COMPARAISON DES STRATÉGIES D'ORDONNANCEMENT

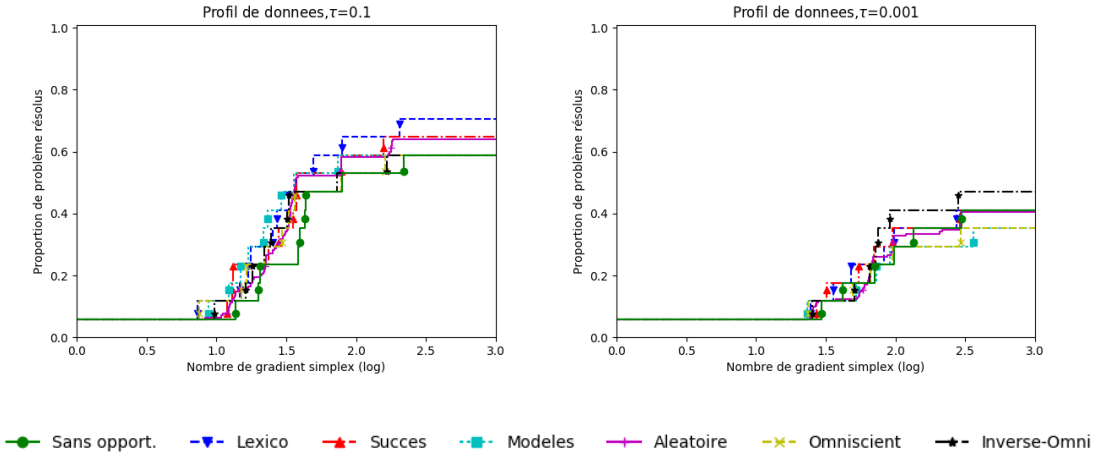


Figure A.1 Comparaison sur tous les problèmes contraints avec CS, avec opportunisme au $p = 2^{\text{ème}}$ succès.

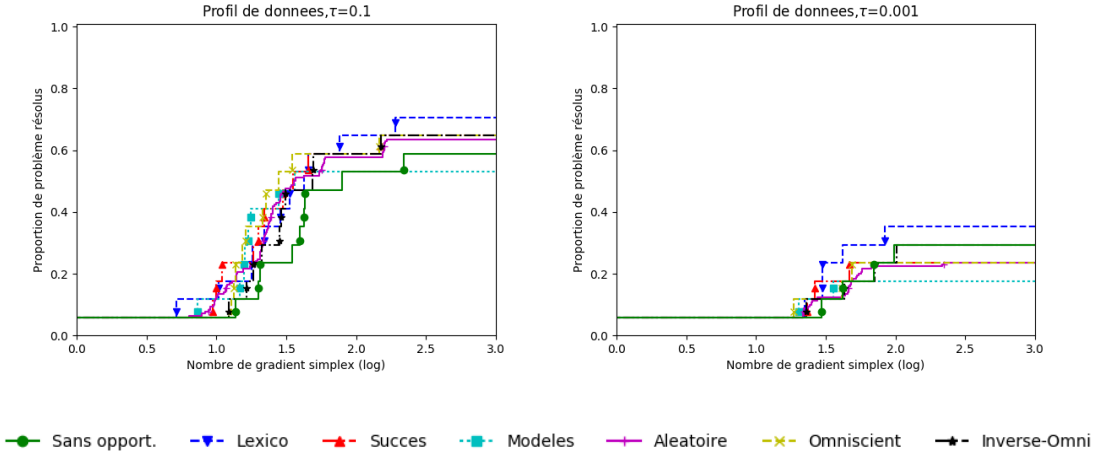


Figure A.2 Comparaison sur tous les problèmes contraints avec CS, avec minimum $q = \lceil \frac{n}{2} \rceil$ évaluations

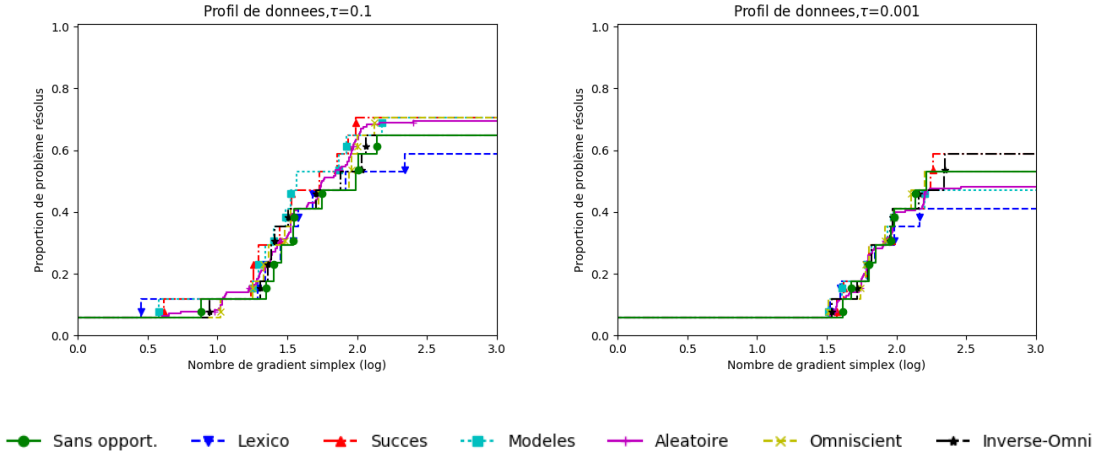


Figure A.3 Comparaison sur tous les problèmes contraints avec GPS, avec opportunisme au $p = 2^{\text{ème}}$ succès.

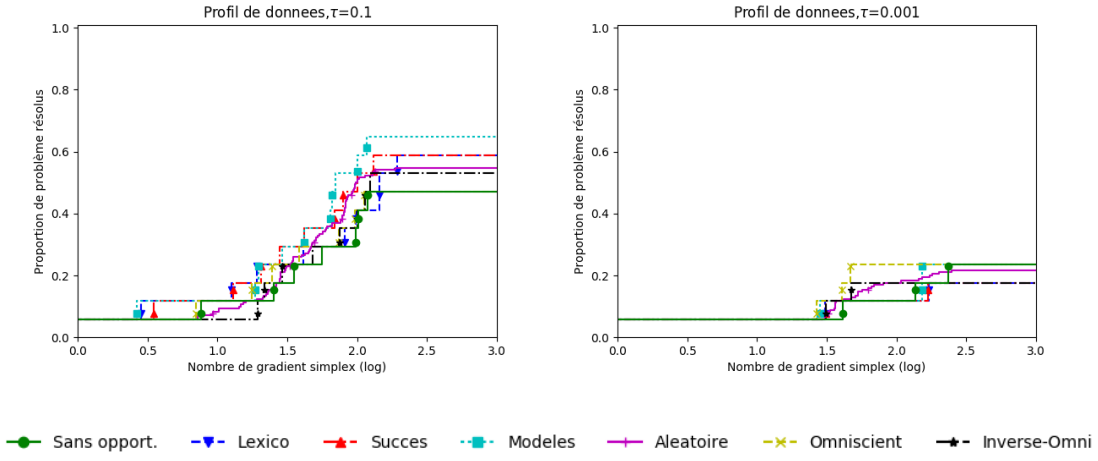


Figure A.4 Comparaison sur tous les problèmes contraints avec GPS, avec minimum $q = \lceil \frac{n}{2} \rceil$ évaluations

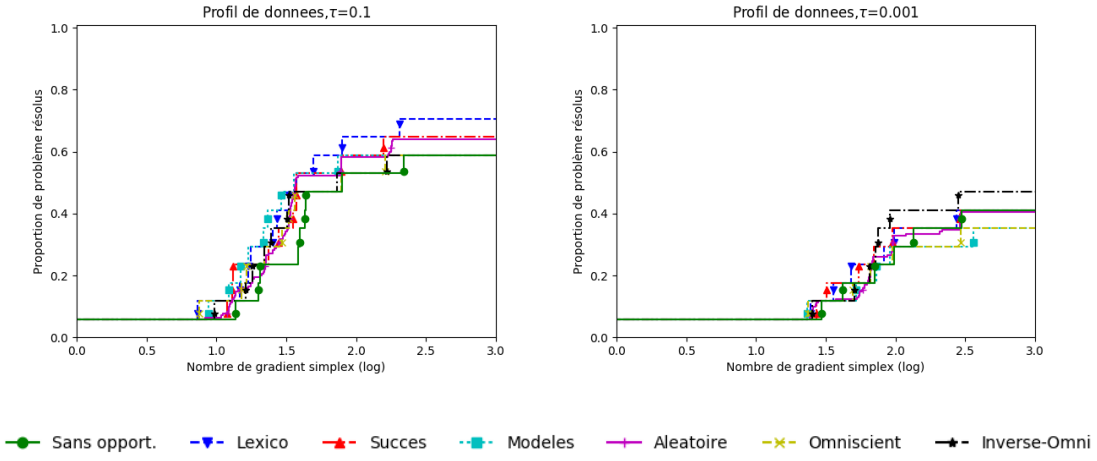


Figure A.5 Comparaison sur tous les problèmes contraints avec MADS, avec opportunisme au $p = 2^{\text{ème}}$ succès.

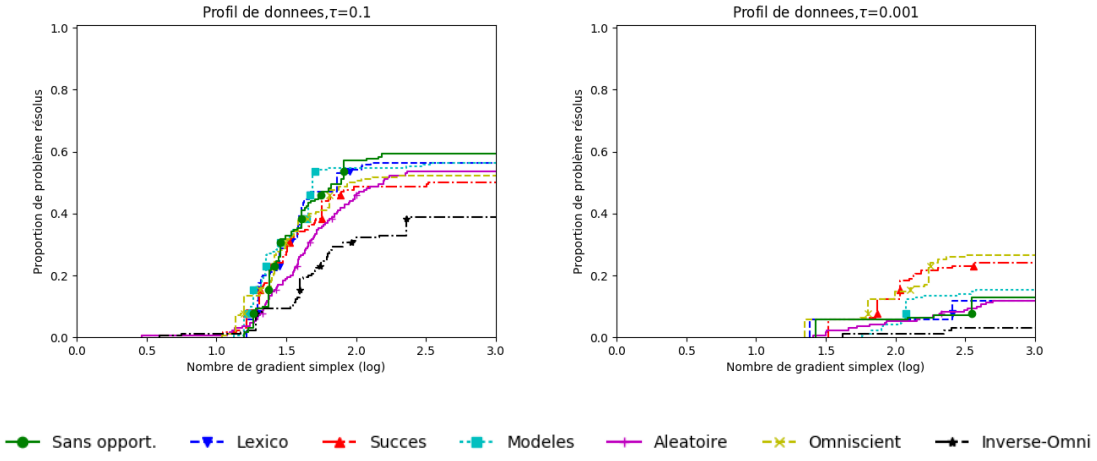


Figure A.6 Comparaison sur tous les problèmes contraints avec MADS, avec minimum $q = \lceil \frac{n}{2} \rceil$ évaluations

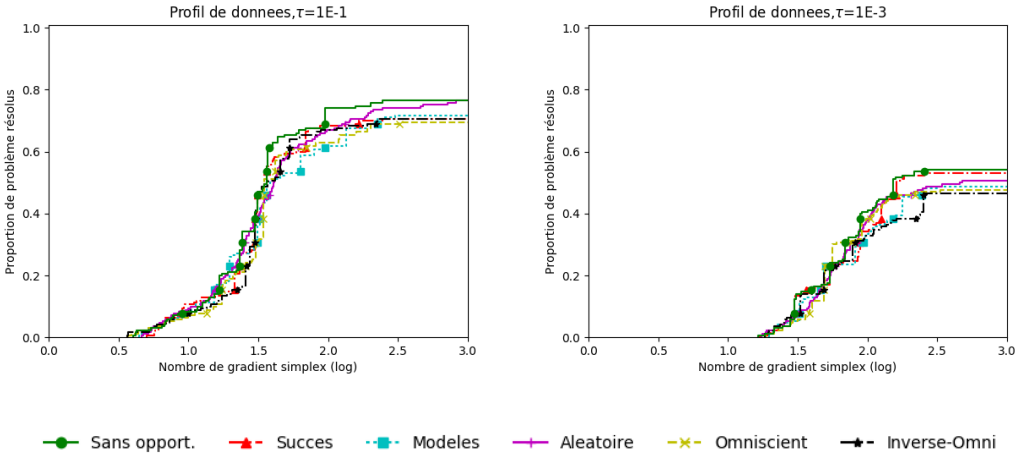


Figure A.7 Comparaison sur tous les problèmes contraints avec MADS par défaut de NOMAD avec opportunisme au $p = 2^{\text{ème}}$ succès.

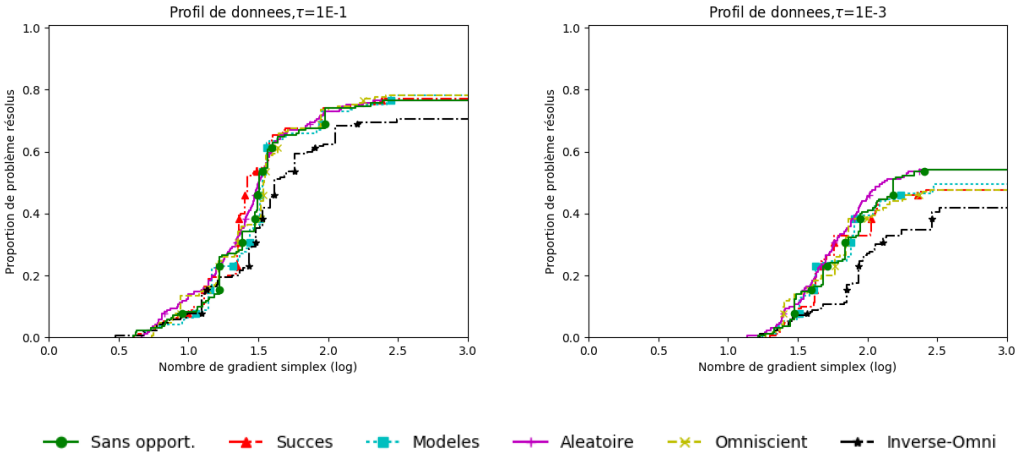


Figure A.8 Comparaison sur tous les problèmes contraints avec MADS par défaut de NOMAD avec minimum $q = \lceil \frac{n}{2} \rceil$ évaluations