République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique
Université Dr. Yahia Farès de Médéa



Faculté des Sciences
Département Des Mathématiques

MÉMOIRE

présenté en vue de l'obtention du

DIPLÔME DE MASTER EN MATHÉMATIQUES

Spécialité : Recherche Opérationnelle

par

AZIRI Souhila
MANSOURI Naziha

Thème

L'OPTIMISATION MULTI-OBJECTIF PAR LA MÉTHODE P.A.S.A

---

devant le jury composé de :

Mr.	M. SAHRAOUI	Maître assistant	UYF Médéa	Président.
Mr.	R. LEMDANI	Maître assistant	UYF Médéa	Examinateur.
Mr.	K. TABLENNEHAS	Maître assistant	UYF Médéa	Promoteur.
Mr.	K. SEBAA	Professeur	UYF Médéa	Co-Promoteur.

Année Universitaire 2017/2018

Remerciements

Nous tenons à remercie Dieu tout puissant clément et miséricordieux de nous avoir donné la force d'accomplir ce travail.

Nous tenons avant tout, à exprimer notre profond gratitude à notre promoteur monsieur TABLENNEHAS Kamel, Maître assistant à l'université Dr. Yahia Farès de Médéa, qui a assumé la direction de ce travail.

sa patience, sa disponibilité, ses conseils et son aide constant qu'il nos apporté tout au long de ce travail.

Nous voudrons sincèrement exprimer nos plus vifs remerciements à notre co-promoteur monsieur SEBAA Karim , Professeur à l'université Dr. Yahia Farès de Médéa s'est toujours montrée à l'écoute et très disponible tout au long de la réalisation de ce travail, ainsi, pour son aide et le temps qu'il a bien voulu nos consacrer.

Nous remerciements tous les membres de jury qui nous ont fait l'honneur d'examiner notre travail, monsieur LEMDANI.R, Maître assistant à l'université Dr. Yahia Farès de Médéa

et monsieur SAHRAOUI.M, Maître assistant à l'université Dr. Yahia Farès de Médéa

Nous adressons notre vifs remerciements à tous les enseignants de spécialité de mathématique de l'université de Dr.Yahia Farès de Médéa. qui, par leur enseignement, leur encouragement et leur aide, ont contribué à notre formation durant toutes mes études à l'université de Dr.Yahia Farès de Médéa.

Enfin, nous adressons nos plus sincère remerciements, nos parent pour leur encouragement tout au long de nos études, et tous nos proches et amis(es) qui nous ont toujours soutenus et encouragé au cours de la réalisation de ce mémoire.

Dédicaces

Nous tenons à remercie le Dieu pour nous tout puissant clément
et miséricordieux de nous avoir soigné et aidé.
Je dédie ce travail à mes chers parents,
Ma mère pour ses encouragement
et ses prières tout au long de mes études
Mon père pour tous ce qu'il avait fait pour avoir ce résultat.
Je le dédie à mes frères Hasan, Omar et Imad et mes sœurs Saida,Fatima et Fayza et surtout le petit fils al katkot Walid
Et je les remercie pour leurs encouragements et leurs aides ainsi que toute ma grande famille.
A tous les enseignants et a tous mes amis au long de mes études, surtout ma chère ahlam et et bien sur la plus proche naziha.
A mes collègues de la promotion 2017-2018 de recherche opérationnelle et. A tous ceux qui aiment souhila et ceux qui souhila aime.

SOUHILA

Dédicaces

Je remercie le Dieu pour m'avoir donné la force d'accomplir ce travail pour aller plus loin. Je dédie ce travail à mes parents **REDOUANE** et ma mère **RABIA** pour ses encouragements et ses prières tout au long de mes études, mon père pour tous ce qu'il avait fait pour avoir ce résultat.
Je le dédie à mes frères faisal ,hamza et ma soeurs marwa, et je les remercie pour leurs encouragements et leurs aides ainsi que toute ma grande famille.
Je le dédie à l'homme qui m'a choisi pour me donner son nom à jamais, mon mari **smain**.
A mes collègues et amies "ahlem,wissam,mariam,remaisa"et surtout souhila.
A tous mes amis sans citer les noms.
A mes collègues de la promotion 2017/2018 .
A tous ceux qui aiment Naziha et ceux qui Naziha aime.

NAZIHA

Résumé

Résoudre un problème d'optimisation consiste à trouver la ou les meilleures solutions vérifiant un ensemble des contraintes et objectifs définis par l'utilisateur. Il est nécessaire que le problème introduise un critère de comparaisons. Ainsi, la meilleure solution appelée aussi solution optimale, est la solution ayant obtenu la meilleure évaluation au regard un critère défini.

Malheureusement, dans la plupart des problèmes du monde réel, il ne s'agit pas d'optimiser seulement un seul critère mais plutôt d'optimiser simultanément plusieurs critères et qui sont généralement conflictuels. L'optimisation multi-objectif consiste donc à optimiser simultanément plusieurs fonctions. La notion de solution optimale unique dans l'optimisation uni-objectif disparait pour les problèmes d'optimisation multi-objectif au profit de la notion d'ensemble de solutions Pareto optimales. L'utilisation des méta-heuristiques pour résoudre des problèmes multi-objectifs a fait l'objet d'un intérêt de plus en plus croissant. A présent, les méta-heuristiques constituent un des champs de recherche les plus actifs dans l'optimisation multi-objectifs. La plupart des travaux existants concernent l'optimisation bi-objectif. Le cas multi-objectif reste difficile à résoudre, non seulement à cause de la complexité de ces problèmes, mais aussi à cause du nombre élevé des solutions Pareto optimales à un problème d'optimisation multi-objectif.
Nous proposons, dans ce mémoire, de traiter les problèmes d'optimisation multi-objectif par la méthode P.A.S.A (pareto archived simulated annealing). Nous donnerons aussi plus en détail des définitions et principaux concepts de la méthode de recuit simulé et comment la méthode P.A.S.A utilise ce dernier pour résoudre un problème d'optimisation multi-objectif selon des étapes de ce méthode.

ملخص

ينطوي حل مشكلة التحسين على إيجاد أفضل حل أو الحلول التي تلبي مجموعة من القيود والأهداف المحددة من قبل المستخدم. من الضروري أن تقدم المشكلة معيار المقارنات. وبالتالي ، فإن الحل الأفضل ، والذي يسمى أيضًا الحل الأمثل ، هو الحل الذي حصل على أفضل تقييم مقابل معيار محدد.

لسوء الحظ ، في معظم مشاكل العالم الواقعي ، لا يتعلق الأمر بتحسين معيار واحد فحسب ، بل بالأحرى تحسين العديد من المعايير بشكل متزامن والتي تكون متعارضة بشكل عام. لذا ، فإن التحسين متعدد الأهداف يتكون في الوقت نفسه من تحسين وظائف متعددة. مفهوم الحل الأمثل فريد من نوعه في تحسين وحيد موضوعي يختفي المشكل الأمثل متعدد الأهداف لصالح المفهوم العام للباريتو الأمثل الحلول. لقد كان استخدام الاستدلال الفوقي لحل مشكلات متعددة الأهداف موضع اهتمام متزايد. الآن ، الاستدلال الفوقي هي واحدة من المجالات الأكثر نشاطا للبحوث في التحسين متعدد الأهداف. معظم العمل الحالي يتعلق بالتحسين ثنائي الهدف. تظل القضية متعددة الأهداف صعبة الحل ، ليس فقط بسبب تعقيد هذه المشاكل ، ولكن أيضا بسبب العدد الكبير من حلول باريتو التي تعتبر مثالية لمشكلة تحسين متعددة الأهداف. في هذه الأطروحة ، نقترح التعامل مع مشكلات تحسين متعددة الأهداف باستخدام طريقة التلدين المحاكية ذات الباريتو (عا.صا). سنقدم أيضا تعريفات أكثر تفصيلا والمفاهيم الرئيسية لطريقة محاكاة الصلب وكيف يستخدم أسلوب عا.صا من أجل حل مشكلة الأمثل متعدد الأهداف في مراحل هذه الطريقة.

Abstract

Solving an optimization problem involves finding the best solution or solutions that satisfy a set of user-defined constraints and objectives. It is necessary that the problem introduces a criterion of comparisons. Thus, the best solution, also called optimal solution, is the solution that has obtained the best evaluation against a defined criterion.
Unfortunately, in most real-world problems, it is not a question of optimizing only one criterion but rather of simultaneously optimizing several criteria and which are generally conflicting. Multi-objective optimization therefore consists in simultaneously optimizing several functions. The notion of optimal solution unique in the uni-objective optimization disappears for the problems of multi-objective optimization in favor of the notion of set of optimal solutions Pareto. The use of meta-heuristics to solve multi-objective problems has been of increasing interest. Now, meta-heuristics are one of the most active fields of research in multi-objective optimization. Most of the existing work concerns bi-objective optimization. The multi-objective case remains difficult to solve, not only because of the complexity of these problems, but also because of the high number of Pareto solutions that are optimal for a multi-objective optimization problem.
In this thesis, we propose to deal with multi-objective optimization problems using the pareto archived simulated annealing (P.A.S.A) method. We will also give more details on the definitions and main concepts of the simulated annealing method and how the P.A.S.A method uses this method to solve a multi-objective optimization problem according to steps of this method.

Table des matières

Résumé **6**

Abstract **8**

Introduction générale **13**

1 OPTIMISATION MONO-OBJECTIF **15**
1.1 Définition d'un problème d'optimisation : . . . 15
1.1.1 Terminologie de la solution . . . 16
1.2 Définitions et notations : . . . 16
1.2.1 Minimum /maximum local : . . . 16
1.2.2 Minimum /maximum global : . . . 17
1.2.3 Notion de convexité : . . . 17
1.2.4 Gradient et Matrice Hessien : . . . 17
1.3 Classification générale des méthodes d'optimisation mono-objectif . . . 18
1.4 Optimisation Combinatoire . . . 18
1.5 Quelque problème de l'optimisation combinatoire : . . . 19
1.5.1 Problème du sac à dos : . . . 19
1.5.2 Problème voyageur de commerce : . . . 19
1.5.3 La Complexité des problème et efficacité des algorithmes : . . . 19
1.5.4 Définition d'un algorithme et sa convergence : . . . 20
1.6 Problème de l'optimisation non linaire : . . . 21
1.6.1 Cas sans contrainte : . . . 21
1.6.2 Les condition d'optimalités . . . 21
1.6.3 Les condition nécessaire d'optimalités : . . . 21
1.6.4 Les méthodes de résolution de ces problèmes : . . . 22
1.6.5 Cas avec contrainte : . . . 23
1.6.6 Plan tangent à une surface : . . . 23
1.6.7 Condition nécessaire d'optimalité du première ordre : . . . 23
1.6.8 Condition nécessaire d'optimalité du deuxième ordre : . . . 23
1.6.9 La nature des points candidat : . . . 24
1.6.10 Problème d'optimisation non linéaire avec contrainte mixte : . . . 24
1.6.11 Définitions et Notations : . . . 24
1.6.12 Les conditions d'optimalités : . . . 24
1.6.13 Condition nécessaire d'optimalité deuxième ordre : . . . 25
1.6.14 Condition suffisant d'optimalité deuxième ordre : . . . 25
1.7 Les Méthodes de résolution des problèmes avec contraintes mixte : . . . 25
1.7.1 Définition : . . . 25
1.7.2 Remarque : . . . 25
1.7.3 Méthode du gradient projeté : . . . 25

1.8 Problème d'optimisation linéaire (ou programmation linéaire) : . . . 26
1.8.1 Définition : . . . 26
1.8.2 Formulation de modèle mathématique : . . . 26
1.8.3 La forme matricielle d'un PL . . . 27
1.8.4 La forme canonique d'un pb linéaire : . . . 27
1.8.5 La forme standard d'un Pb linéaire : . . . 27
1.9 Méthodes de résolution : . . . 27
1.9.1 La résolution graphique : . . . 27
1.9.2 Les étapes de résolution graphique d'un PL : . . . 28
1.9.3 Méthode de simplexe . . . 28
1.9.4 La dualité : . . . 30

2 L'OPTIMISATION MULTI-OBJECTIF 31
2.1 Définition d'un Problème d'optimisation multiobjectif : . . . 31
2.1.1 Vocabulaire et définitions : . . . 32
2.2 La multiplicité des solution : . . . 32
2.3 Notion de dominance : . . . 33
2.3.1 La relation de dominance : . . . 33
2.3.2 Efficacité : . . . 33
2.3.3 Optimalité au sens de Pareto : . . . 33
2.4 La surface de compromis (front de Pareto) : . . . 35
2.4.1 Définition : . . . 35
2.4.2 La convexité : . . . 37
2.4.3 La représentation de la surface de compromis : . . . 38
2.5 Les Conditions d'optimalités d'un problème d'optimisation multi-objectif : . . . 39
2.5.1 Les Conditions nécessaires d'optimalités : . . . 39
2.5.2 Conditions suffisantes d'optimalité : . . . 40
2.6 Les différents méthodes de résolution des problèmes d'optimisation multiobjectif : . . . 40
2.6.1 Les méthodes à préférence a priori : . . . 40
2.6.2 Les méthodes à préférence progressive ou interactive : . . . 41
2.6.3 Les méthodes à préférence a posteriori : . . . 41
2.6.4 les méthodes scalaires : . . . 41
2.6.5 les méthodes interactives : . . . 41
2.6.6 les méthodes floues : . . . 42
2.6.7 les méthodes exploitant une méta-heuristique : . . . 42
2.6.8 les méthodes d'aide à la décision : . . . 42
2.7 Classification des Méthodes d'optimisation multiobjectif : . . . 42

3 LES MÉTHODE EXPLOITANT UNE MÉTA-HEURISTIQUE 44
3.1 Heuristique / Méta-heuristique d'optimisation : . . . 44
3.1.1 Définition de la heuristique . . . 44
3.1.2 Définition d'une méta-heuristique . . . 45
3.2 Classification des méthodes : . . . 45
3.2.1 Les méta-heuristiques à solution unique . . . 46
3.2.2 Les méta-heuristiques à population de solutions : . . . 47
3.3 Le recuit simulé : . . . 49
3.3.1 Les origines : . . . 49
3.3.2 Définition : . . . 50
3.3.3 Algorithme : . . . 50
3.4 Domaines d'applications : . . . 51
3.5 Avantages et Inconvénients : . . . 52

3.5.1 Avantages : . . . 52
3.5.2 Inconvénient : . . . 52
3.6 La méthode P.A.S.A(Pareto Archived Simulated Annealing) . . . 52
3.6.1 Principe : . . . 52
3.6.2 Présentation de la méthode : . . . 52
3.6.3 Discussion : . . . 53
3.6.4 Algorithme La méthode P.A.S.A : . . . 53
3.7 L'équilibre souhaité entre l'intensification et la diversification . . . 54
3.8 Procédure de Crowding : . . . 58

conclusion **59**

Table des figures

1.1 Classification générale des méthodes d'optimisation mono-objectif 18

2.1 L'optimalité locale au sens de Pareto. 34
2.2 Le théorème du contact. 34
2.3 Les niveaux de préférence dans la relation de dominance. 35
2.4 Le front de Pareto. 36
2.5 Formes les plus courantes de la surface de compromis dans le cas de deux objectifs. 36
2.6 Front pareto. 37
2.7 Exemples d'ensemble convexe et d'ensemble non convexe. 38
2.8 La représentation de la surface de compromis. 38
2.9 Classification des Méthodes d'optimisation multi-objectif 43

3.1 Fonctionnement d'un algorithme génétique. 49
3.2 Comparaison entre le recuit simulé et une heuristique classique 51
3.3 Illustration de la convergence et la diversité en multi-objectif 54
3.4 Distance de Crowding (les points noirs sont des Solutions appartenant au même front) 58

Introduction générale

L'optimisation est une branche des mathématique cherchant à modéliser, à analyser et à résoudre analytiquement ou numériquement les problèmes qui consistent à minimiser ou maximiser une fonction sur un ensemble. L'optimisation joue un rôle important en recherche opérationnelle, dans les mathématiques appliquées en analyse et en analyse numérique, en statistique pour l'estimation du maximum de vraisemblance d'une distribution, pour la recherche de stratégies dans la cadre de la théorie des jeux, ou encore en théorie du contrôle de la commande.

L'optimisation est au cœur de tout problème lié à la prise de décision, Le but ultime de toutes ces décisions est soit de minimiser l'effort requis et/ou maximiser le bénéfice désiré.
Les problèmes d'optimisation sont utilisés pour modéliser de nombreux problèmes appliqués dans la conception des systèmes de l'industrie, le traitement d'images, les problèmes rencontrés dans les réseaux industriels, la conception d'emplois du temps, . . . etc.
Il existe deux types des problèmes d'optimisation, l'optimisation mono-objectif, qui se base sur la minimisation (ou la maximisation) d'une seule fonction objectif où le but est de trouver la meilleure solution appelée solution optimale qu'est facilement définie suivant des méthodes de résolution d'optimisation mono-objectif. Et l'optimisation multi-objectif qui optimise simultanément plusieurs fonctions objectives qui sont souvent contradictoires, Où le résultat d'un problème d'optimisation multicritère n'est pas unique, mais généralement un assortiment de solutions, qui se distinguent par différents compromis réalisés entre les objectifs. Cet assortiment est connu par l'ensemble de solutions Pareto optimales
Donc le but de l'optimisation multicritère, est d'obtenir les solutions de Pareto, par conséquent, à connaître l'ensemble des compromis possibles entre les objectifs.

Depuis quelques années, l'approche Pareto définie initialement dans des travaux en économie au 19 ème siècle a été utilisée dans les sciences pour l'ingénieur. Cette approche a l'avantage de traiter les problèmes multi-objectifs sans transformation, sans favoriser un objectif par rapport à un autre. Dans ce cas la solution optimale ou de bonne qualité n'est plus une solution unique (cas mono-objectif) mais un ensemble de solutions compromis entre les différents objectifs à optimiser, ce qui donne une aide précieuse aux décideurs.
La qualité des résultat et des prédictions dépend de la pertinence du modèle, l'efficacité de l'algorithme et des moyens pour le traitement numérique.
La plupart des problèmes d'optimisation appartiennent à la classe des problèmes NP-difficile : classe où il n'existe pas d'algorithme qui fournit la solution optimale en temps polynomial en fonction de la taille du problème.
La plupart des travaux réalisés dans ce domaine étaient dédiés à l'optimisation d'un seul objectif, or la plupart des applications réelles intègrent plusieurs objectifs souvent contradictoires à optimiser simultanément. Pendant longtemps, les approches de résolution multiobjectif consistaient principalement à les transformer en problèmes mono-objectifs.

Tout comme pour l'optimisation mono-objectif, La résolution d'un problème d'optimisation multicritère, a conduit les chercheurs à proposer deux classes de méthodes de résolution pour traiter les problèmes

multi-objectifs : les méthodes exactes dédiées à résoudre optimalement les petites instances (les méthodes exactes pour la résolution à des problèmes de petites tailles) et les méthodes approchées : les heuristiques et en particulier les méta-heuristiques (génériques) permettant d'approximer les meilleures solutions sur les plus grandes instances (les méthodes approchées pour la résolution à des problèmes de grande taille). Les méta-heuristiques sont des algorithmes génériques, souvent inspirés de la nature, conçues pour résoudre des problèmes d'optimisation complexes. Parmi les méta-heuristiques les plus récentes, nous retenons celle basée sur la recuit simulé qui est utilisé par la méthode de P.A.S.A. qui nous avons étudier dans ce mémoire.

Organisation du mémoire :

Cette mémoire est organisée en 3 chapitres :
Dans le premier chapitre , nous présentons les principes de base de l'optimisation mono-objectif(continu et discret(combinatoire)) , Nous présentons l'optimisation mono-objectifs tout en introduisant des concepts fondamentaux des problèmes d'optimisation mono-objectif, et une classification général de différents méthodes de résolution utilisées pour résoudre les problèmes d'optimisation mono-objectif linéaire et non linéaire dans le cas avec et sans contraintes.

Le deuxième chapitre, nous introduisons dans un premier temps quelques concepts et ensembles des définitions et généralités de base très importantes liées au thème de l'optimisation combinatoire multi-objectifs tels que la dominance, la surface de compromis,l'optimalité. Nous décrivant aussi le problème d'optimisation multi-objectifs et les différents méthodes de résolution des problèmes d'optimisation multiobjectif et les différentes approches existantes pour leur résolution

Le troisième chapitre, Ces dernières années, la plupart des méthodes exploitant une méta-heuristiques présentées dans la littérature pour résoudre des problèmes d'optimisation combinatoire multi-objectif, Parmi les méta-heuristiques les plus récentes, nous nous intéressons à étudies et données quelque test sur la méthode P.A.S.A(Pareto Archived Simulated Annealing) qui utilise le recuit simulé.
Finalement, nous clôturons ce mémoire par une conclusion générale,

Chapitre 1

OPTIMISATION MONO-OBJECTIF

Dans ce chapitre, nous présentons les principes de base de l'optimisation mono-objectif (l'optimisation avec des variables dont le domaine est continu et l'optimisation combinatoire dont le domaine est discret). Nous donnerons aussi des définitions et principaux concepts d'un problème optimisation Mono-objectif et une classification général des méthodes d'optimisation mono-objectif. Puis on présentera quelques une de ces méthodes pour la résolution d'un problème d'optimisation mono- objectif .

1.1 Définition d'un problème d'optimisation :

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

$$P : \begin{cases} \min / max(f(x)) \\ x \in \Omega \end{cases} \tag{1.1}$$

avec
$f : \mathbb{R}^n \longrightarrow \mathbb{R}$: une fonction objectif ou bien la fonction optimiser sur l'ensemble Ω

1. si $\Omega = \mathbb{R}^n$ alors le problème est dit problème d' optimisation sans contrainte
2. si $\Omega = \{x \in \mathbb{R}^n / g_i(x) \leqslant 0 \forall\, i \in \{1, ..., m\}, h_j(x) = 0 \forall\, j \in \{1, ..., m'\}\}$
   et $g_i, h_j : \mathbb{R}^n \longrightarrow \mathbb{R}$ des fonction de contraintes, alors le problème est dit problème d'optimisation avec contraintes mixtes . Ou bien :

$$P : \begin{cases} \min / max(f(x)) \\ h(x) = 0 \\ g(x) \leqslant 0 \\ x \in \Omega \end{cases} \tag{1.2}$$

Le problème est dit problème d'optimisation avec contraintes égalité :
Si $\Omega = \{x \in \mathbb{R}^n, h(x) = 0\}$
Si $\Omega = \{x \in \mathbb{R}^n, g(x) \leqslant 0\}$ alors :
P est dit pb d'optimisation avec contraintes d'inégalités.
L'ensemble des points $\{x \in \Re^n / g_i(x) \leqslant 0 \forall\, i \in \{1, ..., m\}, h_j(x) = 0 \forall\, j \in \{1, ..., m'\}\}$: est appelé ensemble de candidats, s'il est non vide

On appelle solution du problème d'optimisation tout minimum/maximum global de la fonction f restreinte à cet ensemble de candidat,noté x^*.
f, g, h sont linéaires ,alors (P) est dit problème d'optimisation linéaire
Si f, g, h sont convexe ,alors (P) est dit problème d'optimisation convexe
Si le problème

$$P : \begin{cases} \min / max f(x) = \frac{1}{2} X^t A X - C^t X \\ AX \leqslant 0 \end{cases} \tag{1.3}$$

Est appelé problème d'optimisation quadratique.
Chaque programme mathématique contient trois genres d'éléments :

1. Les variables :
   • Ensemble des variables qui régissent la situation à modéliser ou bien
   •Elles représentent les décisions ou les choix qui doivent être prises dans le modèle.
   • Variables réelles, entières, binaires
2. Préciser la fonction objectif :
   • Fonction mathématique composée des variables de décision qui représente le modèle physique modélisé
   • Une définition mathématique du but du modèle
3. les contraintes du problème :
   • Ensemble des paramètres qui limitent le modèle réalisable
   • équations ou inéquations composées des variables de décision
   • Les limitations imposé sur le décision dans la situations modélisée.

1.1.1 Terminologie de la solution

Solution réalisable

•Solution où toutes les contraintes du modèle sont satisfaites

Zone de solution

• Ensemble de toutes les solutions réalisables

Solution optimale

• Solution réalisable où la fonction objectif atteint la meilleure valeur, maximum ou minimum
• Plusieurs solutions optimales possibles .

1.2 Définitions et notations :

1.2.1 Minimum /maximum local :

Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$

1. Minimum local : f possède un minimum local (resp min local strict) ou point x^* s'il existe $\delta > 0$,si $\forall x \in v_\delta(x^*)$ tel que :
   $f(x^*) \leqslant f(x)$ (respect :$f(x^*) < f(x), \forall x \in v_\delta(x^*)$).
   $v_\delta(x^*) = \{x \in \mathbb{R}^n : \| x - x^* \| \leqslant \delta\}$

2. Maximum local : f possède un maximum local (respect max local strict) ou point x^* si $\forall x \in v_\delta(x^*)$ ona :
   $f(x^*) \geq f(x)$ (respect :$f(x^*) > f(x), \forall x \in v_\delta(x^*)$).

1.2.2 Minimum /maximum global :

.

1. Minimum global : f possède un minimum global (resp min global strict) ou point $x^* \in v_\delta(x^*)$ tel que :
   $f(x^*) \leqslant f(x)$ (respect :$f(x^*) < f(x), \forall x \in \mathbb{R}^n$).
2. Maximum global : f possède un maximum global (resp max global strict) ou point $x^* \in v_\delta(x^*)$ tel que :
   $f(x^*) \geq f(x)$ (respect :$f(x^*) > f(x), \forall x \in \mathbb{R}^n$).

Remarque :

On dit que x^* est un point d'extremum global (resp local)de f sur$\mathbb{R}^n$ si x^* est un point max global (resp local) de f sur $\mathbb{R}^n$

1.2.3 Notion de convexité :

soit D un ensemble convexe :

Définition :

On dit que un ensemble $D \in \mathbb{R}^n$ est convexe ssi pour tout $(x_1, x_2) \in D^2$ et$\alpha \in [0,1]$ $\alpha x_1 + (1-\alpha)x_2 \in D$
C'est-à-dire,le segment reliant x_1 et x_2 doit être contenu entièrement dans D

Définition d'un fonction convexe :

La fonction $f : D \longrightarrow \mathbb{R}^n$ est dit convexe ssi $\forall (x_1, x_2) \in D^2, \alpha \in [0,1]$
$f(\alpha x_1 + (1-\alpha)x_2) \leqslant \alpha f(x_1 + (1-\alpha)f(x_2)$
Et on dit que f est strictement convexe si $\forall (x_1, x_2) \in D^2, \alpha \in [0,1]$
$f(\alpha x_1 + (1-\alpha)x_2) < \alpha f(x_1 + (1-\alpha)f(x_2)$.

Remarque :

En changement le sens de l'inégalité de la définition d'une fonction convexe on obtient le définition d'une fonction concave.

1.2.4 Gradient et Matrice Hessien :

Définition :

1. Si f est dérivable au point x, on définit le gradient de f au point x noté par $\nabla f(x)$
   $\nabla f(x) = (\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, ..., \frac{\partial f}{\partial x_n})^t$
   Ou $x = (x_1, x_2, ..., x_n)$.
2. Si f est deux fois dérivable alors la matrice hessien de f au point x noté par $\nabla^2 f(x)$ ou bien $H_f(x)$
   c-à-d : $H_f(x) = \frac{\partial^2 f}{\partial x_i \partial x_j}$ $\quad i = 1, ..., n$ et $j = 1, ..., n$
3. Si f est fonction de classe C^2(admet des dérivées partielles d'ordre 2 continues),le hessien de f est une matrice symétrique.
   Si f est de classe C^2,en peut écrire la formule de Taylor d'ordre 2 :

$f(x) = f(x^*) + (x - x^*)\nabla f(x^*) + \frac{1}{2}(x - x^*) + \|x - x^*\|^2 \varepsilon(x)$
Ou ε est une fonction qui tend vers 0 quand x tend vers x^*.

1.3 Classification générale des méthodes d'optimisation mono-objectif

La résolution d'un problème d'optimisation est réalisée à l'aide des méthodes d'optimisation dont la classification est illustrée dans la figure 1.1.

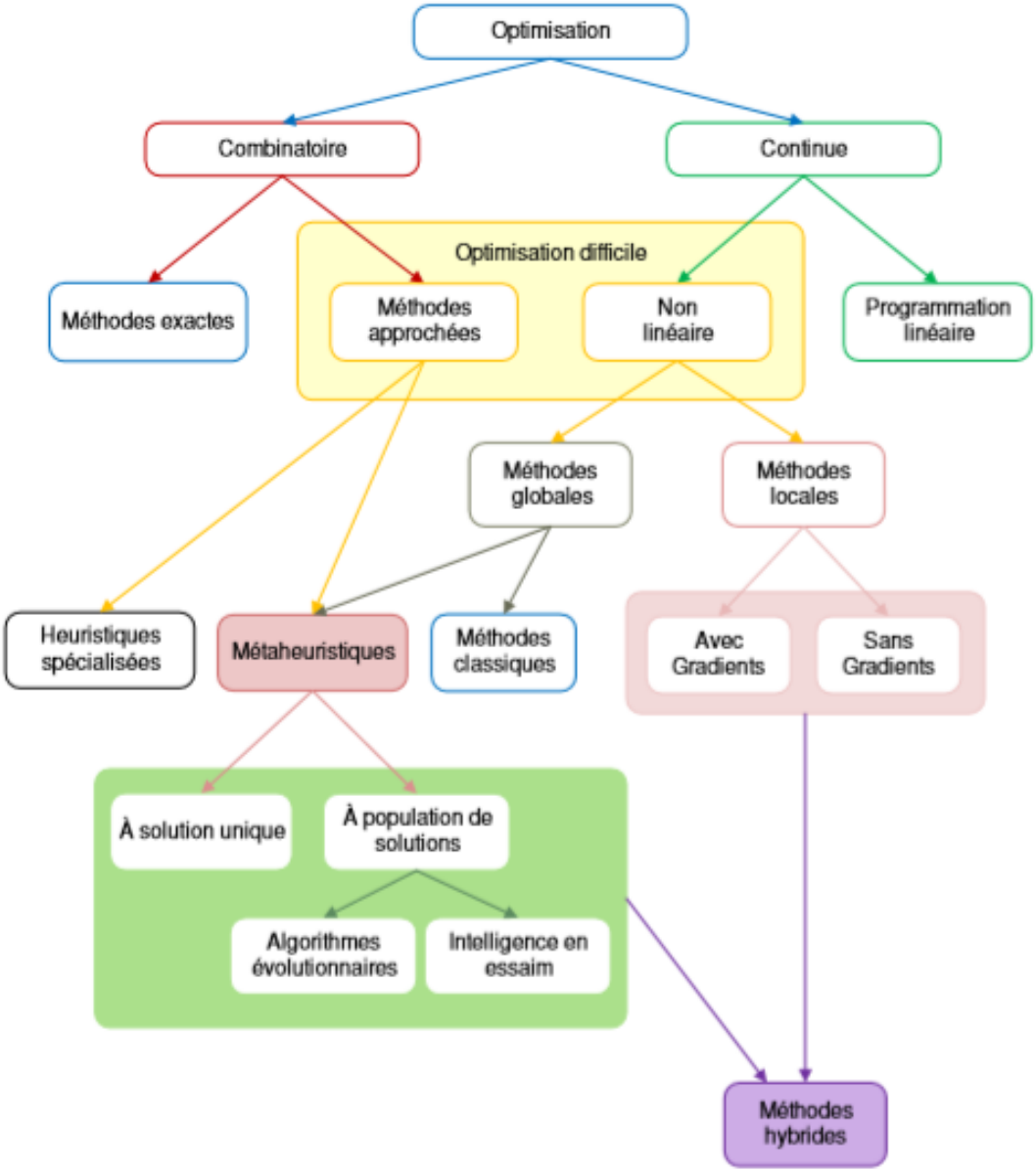


FIGURE 1.1 – Classification générale des méthodes d'optimisation mono-objectif

1.4 Optimisation Combinatoire

L'optimisation combinatoire occupe une place très importante en recherche opérationnelle, en mathématiques discrètes et en informatique.

Son importance se justifie d'une part par la grande difficulté des problèmes d'optimisation et d'autre part par de nombreuses applications pratiques pouvant être formulées sous la forme d'un problème d'optimisation combinatoire. Bien que les problèmes d'optimisation combinatoire soient souvent faciles à définir, ils sont généralement difficiles à résoudre.
En effet, la plupart de ces problèmes appartiennent à la classe des problèmes NP-difficiles et ne possèdent donc pas à ce jour de solution algorithmique efficace valable pour toutes les données.
L'optimisation combinatoire est minimiser (ou maximiser) une fonction souvent appelée fonction coût, d'une ou plusieurs variables soumises à des contraintes. Le sujet de l'optimisation combinatoire dans un domaine discret.
Il faut trouver parmi toutes les possibilités, souvent en nombre fini, la possibilité optimale. Ceci parait facile mais devient infaisable dès que la taille du problème est suffisamment grande.

1.5 Quelque problème de l'optimisation combinatoire :

1.5.1 Problème du sac à dos :

On considère n objet possédant chacun un bénéfice et un poids (c_i, a_i), $i = 1, ..., n$.
On cherche à ranger ces objet dans un sac-à-dos de capacité b les objet mis dans le sac doivent maximiser la valeur totale du bénéfice sans dépasser la capacité du sac .

$$PAC : \begin{cases} (max) Z = \Sigma_{i=1}^{n} c_i x_i & \\ \Sigma_{i=1}^{n} a_i x_i \leq b & i \in \{1....., n\} \\ x_i \in \{0, 1\} & \end{cases} \tag{1.4}$$

1.5.2 Problème voyageur de commerce :

Un représentant doit visiter n ville une et une seule fois, et revenir à sa ville de départ, en minimiser le coût total du trajet.
Soit un graphe complet $G = (V, E)$ muni d'une application $c : E \longrightarrow \mathbb{R}$.
Le problème est de trouver le plus court circuit Hamiltonien dans le graphe G .

$$PAC : \begin{cases} & \\ (min) Z = \Sigma_{i=1}^{n} \Sigma_{j=1}^{n} c_{ij} x_{ij} & \\ \Sigma_{j=1}^{n} x_{ij} = 1 & i \in \{1....., n\} \\ \Sigma_{i=1}^{n} x_{ij} = 1 & j \in \{1....., n\} \\ \Sigma_{(i,j) \in s^2} x_{ij} \leq |S| - 1 & S \subset \{1, ..., n\} : 2 \leq |S| \leq n - 1 \\ x_{ij} \in \{0, 1\} & (i, j) \in \{1, ..., n\}^2 \end{cases} \tag{1.5}$$

1.5.3 La Complexité des problème et efficacité des algorithmes :

La théorie de la complexité s'attache à classifier les problèmes selon leur difficulté.

Un problème de la classe P :

La classe des problème qui peuvent être résolus en temps polynomial sur une machine déterministe ces problème sont souvent considérée comme ceux pour lesquels il existe un algorithme efficace.
Un problème est dit polynomial s'il existe un algorithme de complexité polynomiale permettant de répondre à la question posée dans ce problème, quelque soit la donnée de celui-ci.
La classe P est l'ensemble de tous les problèmes de reconnaissances polynomiaux.

Un problème de la classe NP :

La classe des problème qui peuvent être résolus on temps polynomial sur une machine non déterministe
Un problème de reconnaissance est dans la classe NP si, pour toute instance de ce problème, on peut vérifier, en un temps polynomial par rapport à la taille de l'instance, qu'une solution proposée ou devinée permet d'affirmer que la présence est « oui » pour cette instance.

Problème facile et difficiles

Un problème est facile s'il appartient à la classe des Problème pour lesquels il existe un algorithme efficace pour les résoudre.
Un problème de décision est difficile s'il appartient à la classe NP-complet, pour lesquels il n'existe pas à l'heure actuelle un algorithme efficace pour les résoudre, cependant, il existe un moyen pour vérifier, en temps polynomial, problème véracité d'une solution donnée.
Un problème d'optimisation appartient à la classe NP-difficile si son problème décisionnel appartient à la classe NP-Complet.

1.5.4 Définition d'un algorithme et sa convergence :

Définition d'un algorithme :

Un algorithme est défini par une application A de $\mathbb{R}$ permettant la génération d'une suite d'élément de $\mathbb{R}^n$ par formule

$$\begin{cases} x_0 \in \mathbb{R}^n \\ x_{k+1} = A(x_k) \end{cases} \tag{1.6}$$

x_0 donnée , k=0 (étape initiale)
$x_{k+1} = A(x_k)$, k=k+1 (itération k)

Convergence d'un algorithme :

Théorème :

On dit qu'un algorithme converge si la suite $(x_k)_{k\in\mathbb{N}}$ engendré par l'algorithme, converge vers une limite x .

Vitesse de convergence d'un algorithme :

Soit $(x_k)_{k\in\mathbb{N}}$ une suite de limite x défini par le donnée d'un algorithme A, on dit que la convergence est :
Linéaire :
Si l'erreur $e_k = \|x_k - x\|$ décroit linéairement :
$\exists \varepsilon \in [0,1]; \exists x_0 : \forall k \geq k_0$
$e_{k+1} \leqslant \varepsilon . e_k$
super linéaire :
Si l'erreur e_k décrit tel que :$e_{k+1} \leqslant \alpha . e_k$ avec α_k est le terme d'une suite positive converge vers 0 .
Si α est le terme d'un suite géométrique on dit alors que le convergence est géométrique.
l'ordre P :
Si l'erreur e_k décroit de manière suivant :
$\exists \varepsilon \geq 0; \exists k_0 : \forall k \geq k_0 : e_{k+1} \leqslant \varepsilon . (e_k)^2$.
Si P=2 :la convergence est dit quadratique .

Convergence local :

la convergence local existe si elle n'a lien que pour dz points de départ x_0 dans un voisinage de x^*.

Remarque :

Dans le cas contraire (non local) le convergence est dit globale.

1.6 Problème de l'optimisation non linaire :

Le problème de l'optimisation non linéaire ,consiste à chercher le minimum (resp maximum) d'une fonction f dans un ensemble Ω. Il s'agit de déterminer un point $x^* \in \Omega$ tel que :
$f(x^*) \leqslant f(x)$ (resp $f(x^*) \geq f(x)$) pour tout $x \in \Omega$ .on dit, alors que x^* est la solution optimal du problème de minimisation(resp maximisation) sur Ω.
Où f est la fonction objective, qui est une fonction continue non linéaire. Et Ω est l'ensemble des solution réalisables (l'ensemble définissant ce problème qui est non linéaire) qui est déjà définie mais sont aussi différentiables autant de fois que nécessaire pour l'établissement des outils théorique, comme les conditions d'optimalité ou pour la bonne marche des algorithmes de résolution qui y sont introduit et analyser .
x : un vecteur de composantes $(x_1, x_2, ..., x_n)$ qui sont les inconnues du problème .

1.6.1 Cas sans contrainte :

Les problèmes d'optimisation sans contraintes consiste à chercher le point (où les points) au que une fonction donnée est minimale (où maximale)
- Ces problèmes d'optimisation peuvent être résolu en appliquant les condition d'optimalités où bien en utilisant les méthodes numériques
L'objectif de l'étude de ce problème est de donne les principaux résultats mathématiques concernant les condition d'optimalités.

1.6.2 Les condition d'optimalités

Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$.
Et

$$P : \begin{cases} \min / max(f(x)) \\ x \in \mathbb{R}^n \end{cases} \tag{1.7}$$

1.6.3 Les condition nécessaire d'optimalités :

Théorème :

Soit x^* un point on f atteint son minimum /maximum, si f est continue dérivable en ce point alors :
$\nabla f(x^*) = 0_{\mathbb{R}^n}$.

Les condition suffisantes d'optimalités :

Les condition suffisant peuvent être du 1^{ere} ordre ou de 2^{eme} ordre.

Théorème 01 :

(condition nécessaire du 1^{ere} ordre)
Lorsque f est convexe si $f(x)$ est de classe C^1 on tout point intérieur d'un ensemble convexe D,alors :
$\forall x, y \in D : f(y) - f(x) \geq \nabla f(x)(y - x)$.

Théorème 02 :

(condition suffisant de 1 ère ordre)
Soit $f : D \longrightarrow \mathbb{R}$ une fonction convexe sur le domaine convexe D tel que :
$D \subseteq \mathbb{R}^n$ alors,
tout minimum local de f sur D est un minimum globale de f sur D .

Théorème 03 :

(condition suffisant du 2 ème ordre)
Lorsque f n'est pas convexe et $f \in C^2$
- Si x^* un point ou la fonction f est deux fois continue et dérivable (2 différentiable)
- Si $\nabla f(x^*) = 0$ et $H_f(x^*)$ est défini positive, alors x^* est un minimum local strict de f sur D
- Si $\nabla f(x^*) = 0$ et$H_f(x^*)$ est défini négative, alors x^*est un maximum local strict de f sur D .

Remarque :

la condition suffisant du 2 ème ordre nous permet du savoir la nature de point candidat ou les point critiques (max où min).

1.6.4 Les méthodes de résolution de ces problèmes :

Les méthodes qui nous allons présenter sont les méthodes itératives d'optimisation sans contraintes C'est à dire à partir d'une donnée initial x_0 on construit un suite $(x_k)_{k\in\mathbb{N}}$ et étudier sa convergence .

Les Méthodes de descente :

Soit (P) le problème d'optimisation sans contrainte suivant :

$$P : \begin{cases} \min / max(f(x)) \\ x \in \mathbb{R}^n \end{cases} \tag{1.8}$$

Principe :

Le principe de tout ces méthodes est la descente c'est à dire on commence d'un point $x_0 \in \mathbb{R}^n$.
Et en suit on construit $(x_k)_{k\in\mathbb{N}}$ avec $x_{k+1} = x_k + l_k d_k$.
Avec d_k :le direction de déplacement de x_k vers x_{k+1}.
l_k : est le pas de déplacement en vérifiant :
- La condition suivante(de descente).
Définition : la valeur d_k direction réalisable choisie à partir de x_k est dit descente si et seulement si
$\nabla f(x_k)^t . d_k < 0$.

Algorithme de la méthode à direction de descente :

à l'itération k,on dispose du points $x_k \in \mathbb{R}^n$.
— test d'arrêt : $\nabla f(x_k) = 0$, alors $x^* = x_k$.
— choix de la direction de descente $d_k \in \mathbb{R}^n$.
— déterminer le pas l_k le long de d_k de manière de faire décroit f suffisante.

1.6.5 Cas avec contrainte :

On cherche à résoudre le problème

$$(P):\begin{cases}\min f(x)\\ g(x)\in\Omega\end{cases} \tag{1.9}$$

$f:\mathbb{R}^n\longrightarrow\mathbb{R}$
$g:\mathbb{R}^n\longrightarrow\mathbb{R}^m$
Ω :ensemble convexe du $\mathbb{R}^n$.
$g(x)$:les contraintes
$f(x)$:les fonction objective
problème est appelé problème d'optimisation avec contrainte

L'étude du problèmes avec contraintes égalités :

$$Pce:\begin{cases}\min f(x)\\ x\in\Omega:\{x\in\mathbb{R}^n/h(x)=0\}\\ h=\left(h1,h2,.....,hm\right)'\end{cases} \tag{1.10}$$

Définition d'un point régulier :

soit $S:\{x\in\mathbb{R}^n\diagup h(x)=0\}$ une surface de $\mathbb{R}^n$, on dit que x^* est un point régulier si et seulement si les vecteurs gradient de h au points x^* sont linéairement indépendants c-à-d $\nabla h_1(x^*),\nabla h_2(x^*),.....,\nabla h_n(x^*)$ sont linéairement indépendants

1.6.6 Plan tangent à une surface :

$S=\{x\in\mathbb{R}^n\diagup h(x)=0\}$
On consiste toutes les courbes différentiables paramétrique à la surface passant par un point x_0 de S, $x(t)=t\in[-a,a]$ avec $x(0)=x_0$ le plan tangent à la surface S au point x^* est l'ensemble des dérivée des courbes paramétrique différentiable au point $t=0$

Théorème :

soit $S:\{x\in\mathbb{R}^n\diagup h(x)=0\}$
x^* est point régulier de S.
T est un tangent à la surface Sau point x^* alors
$T=\{y\in\mathbb{R}^n\diagup\nabla h(x^*)'y=0\}$

1.6.7 Condition nécessaire d'optimalité du première ordre :

soit $S=\{x\in\mathbb{R}^n\diagup h(x)=0\}$ et soit x^* un point régulier de S si x^* est un point minimum local de f sur S alors $\exists\lambda\in\mathbb{R}^m$ tel que
$\nabla f(x^*)+\lambda\nabla h(x^*)=0$

1.6.8 Condition nécessaire d'optimalité du deuxième ordre :

soit $S:\{x\in\mathbb{R}^n\diagup h(x)=0\}$ et soit x^* un point régulier de S
si x^* est un minimum local de f sur S alors $\exists\lambda\in\mathbb{R}^m:\nabla f(x^*)+\lambda\nabla h(x^*)=0$ et la matrice
$L(x^*)=H_f(x^*)+\lambda H_h(x^*)$ est S.D.P sur le plant tangent T à la surface au point x^*
$H_f(x^*)=\nabla^2 f(x^*)$
$H_h(x^*)=\nabla^2 h(x^*)$

1.6.9 La nature des points candidat :

Condition suffisant d'optimalité :

Théorème/
Soit $S : \{x \in \mathbb{R}^n \diagup h(x) = 0\}$
Si x^* est un point régulier de S et $\exists \lambda \in \mathbb{R}^m : \nabla f(x^*) + \lambda \nabla h(x^*) = 0$ telle que
$L(x^*) = H_f(x^*) + \lambda H_h(x^*)$ est D.P sur le plan tangente T de f sur s alors x^* est minimum local strict sur S.

1.6.10 Problème d'optimisation non linéaire avec contrainte mixte :

Soit le problème

$$(P) : \begin{cases} \min f(x) \\ x \in \Omega = \{x \in \mathbb{R}^n : h(x) = 0, g(x) \leq 0\} \end{cases} \tag{1.11}$$

avec $f(x)$ fonction objective.
$h(x) : (h_1(x), ..., h_n(x))'$.
$g(x) : (g_1(x), ..., g_m(x))'$.
$X = (x_1, ..., x_n)$.

1.6.11 Définitions et Notations :

1/une contrainte $g_i(x) \leq 0$ est dit active au point x^* si $g(x^*) = 0$
2/une contrainte $g_i(x) \leq 0$ est dit non active au point x^* si $g(x^*) < 0$
3/un point x^* est dit régulier si et seulement si $(\nabla h_1(x^*), ..., \nabla h_n(x^*), \nabla g'_j(x^*))$ sont linéairement indépendants.
$j \in J$ telle que :
J= l'ensemble des contraintes active.
J'=l'ensemble des contraintes non active.

1.6.12 Les conditions d'optimalités :

soit

$$p : \begin{cases} minf(x) \\ x \in \Omega = \{\Phi \in \mathbb{R}^n : h(x) = 0, g(x) \leq 0\} \end{cases} \tag{1.12}$$

Les Conditions nécessaires d'optimalités :

Théorème :

Soit x^* est un minimum local de f sur Ω alors :

$$\exists \lambda \in \mathbb{R}^n, \exists \mu \in \mathbb{R}^p : \begin{cases} \nabla f(x^*) + \lambda \nabla h(x^*) + \mu \nabla g(x^*) = 0 \\ \mu g(x^*) = 0 \\ \mu \geqslant 0 \end{cases} \tag{1.13}$$

1.6.13 Condition nécessaire d'optimalité deuxième ordre :

Théorème :

soit x^* est un minimum local de f sur S alors :

$$\exists \lambda \in \mathbb{R}^n, \exists \mu \in \mathbb{R}^p : \begin{cases} \nabla f(x^*) + \lambda \nabla h(x^*) + \mu \nabla g(x^*) = 0 \\ \mu g(x^*) = 0 \\ \mu \geqslant 0 \\ L(x^*) = H_f(x^*) + \lambda H_h(x^*) + \mu H_g(x^*) \end{cases} \tag{1.14}$$

Est semi définit positive sur le plan tangente à la surface $\{x \in \mathbb{R}^n / h(x) = 0 \text{ et } g_j(x) = 0\}$ au point x^*, $j \in J$

1.6.14 Condition suffisant d'optimalité deuxième ordre :

Théorème :

soit x^* est un point régulier si $\exists \lambda \in \mathbb{R}^n, \exists \mu \in \mathbb{R}^p$:

$$\begin{cases} \nabla f(x^*) + \lambda \nabla h(x^*) + \mu \nabla g(x^*) = 0 \\ \mu g(x^*) = 0 \\ \mu \geqslant 0 \\ L(x^*) = H_f(x^*) + \lambda H_h(x^*) + \mu H_g(x^*) \end{cases} \tag{1.15}$$

Est définit positive sur M tel que $\{y \in \mathbb{R}^n / \nabla h(x^*) y = 0\}$
Et $\{\nabla g_j(x^*) y = 0_{j \in J}\}$ au point x^*.
$j \in J'$, $J' = \{j \in J / \mu_i > 0\} \Longrightarrow x^*$ est un minimum local strict f de S.

1.7 Les Méthodes de résolution des problèmes avec contraintes mixte :

Soit

$$(P) \begin{cases} minf(x) \\ X \in \Omega : \{x \in \mathbb{R}^n / h(x) = 0, g(x) \leq 0\} \end{cases} \tag{1.16}$$

1.7.1 Définition :

Le vecteur d_k au point x_k est dit réalisable s'il existe $\overline{\alpha} \geq 0$, tel que $\forall \alpha, 0 < \alpha < \overline{\alpha}$ on a $(x_k + \alpha d_k) \in \Omega$

1.7.2 Remarque :

Les méthode qu'on va voir sont bases sur la direction réalisable de descente c-à-d : on part d'un pt x_0,on construit une suit $\{x_k\}$ qui converge vers la solution où $x_{k+1} = x_k + \alpha_k d_k$ et $0 < \alpha < \overline{\alpha}$ et le pas de direction réalisable de descente

1.7.3 Méthode du gradient projeté :

La méthode du gradient projeté consiste à résoudre un pb d'optimisation de type

$$\begin{cases} minf(x) \\ X \in \Omega : \{x \in \mathbb{R}^n / h(x) = 0, g(x) \leq 0\} \end{cases} \tag{1.17}$$

et à prendre comme direction de descente réalisable la projection du gradient négatif au point x_k à chaque itération sur le plan tangent a la surface Ω au pt x_k avec
$M_k : \{y \in \mathbb{R}^n / \left(\nabla h(x_k); \nabla g_I(x_k)\right) Y = 0\}$
I :l'ensemble des indices des contraintes inégalités actives au pt x_k

Algorithme :

Étape (0) :trouver un point $x_0 \in \Omega$, mettre $k = 0$
Étape (1) :soit A_q matrice dont les lignes sont les gradients des contraints active au point x_k on pose $g_k = \nabla f(x_k)$
calculer $d_k = [-I + A'_q(A_q A'_q)^{-1} A_q] g'_k$
aller en (2)
Étape (2) : :si $d_k \neq 0$ alors d_k est une direction réalisable de descente
calculer $x_{k+1} = x_k + \alpha d_k$ avec $\alpha_k = argminf(x_k + \alpha d_k)$ et $0 < \alpha < \overline{\alpha}$
poser $k = k + 1$ et aller en (1)
si $d_k = 0$,aller en (3)
Étape (3) :calculer $B'_k = -(A_q A'_q)^{-1} A_q] g'_k$
si la partie de B_k correspondant au contraintes active et positive,alors x_{k+1} vérifie les C.K.T terminer
si non ,il existe au moins une composant de B_k qui est strictement négative.
soit A_k :matrice obtenu à partir de A_k en supprimant une ligne correspondante à la contrainte active pour la quelle le composante B_k est négative
mettre $A_k = \overline{A}_q$et aller en (1)

1.8 Problème d'optimisation linéaire (ou programmation linéaire) :

1.8.1 Définition :

Un programme linéaire est un problème d'optimisation qui consiste à optimiser un fonction linéaire sous des contraintes linéaires (un ensemble d'équation et /ou d'inégalité linéaire).

1.8.2 Formulation de modèle mathématique :

est l'étape la plus délicate de la résolution d'un problème , de la forme d'un PL ,elle nécessite un effort de conception aboutir à la détermination des trois éléments suivants :

Les variables de décision :

(les inconnus du problème)pour les quelles ont doit décides du niveau à atteindre ,tq le niveau d'activité dans l'entreprise .

La fonction objectif :

qui décrit la relation linéaire représentant l'objectif de l'entreprise à l'aide des variables de décision.

Les contraintes :

qui décrivent les relation linéaire entre les variables de décision .
(P) un programme linéaire :

$$pL : \begin{cases} optimiser f(x) = C^t X \\ g(x) \leqslant 0.....(1) \\ h(x) = 0.....(2) \end{cases} \tag{1.18}$$

Avec f,g,h sont des fonction linéaires

1.8.3 La forme matricielle d'un PL

$$P:\begin{cases} optimiser(f(x)) = C^t X \\ AX \geq B \\ A^{'}X \leqslant B^{'} \\ A^{"}X = B^{"} \\ X \in \mathbb{R}^n \\ C \in \mathbb{R}^n \end{cases} \tag{1.19}$$

1.8.4 La forme canonique d'un pb linéaire :

soit (P) :

$$P:\begin{cases} max(Z) = C^t X \\ AX \leqslant B \\ X \geq 0 \end{cases} \tag{1.20}$$

$$P:\begin{cases} min(Z) = C^t X \\ AX \geq B \\ X \geq 0 \end{cases} \tag{1.21}$$

où :
n : nombre des variables
m : nombre des contraintes
$X \in \mathbb{R}^n, C \in \mathbb{R}^n, B \in \mathbb{R}^m$
$A = (X_{i,j})$
$i = 1,..,n$
$j = 1,...,m$

1.8.5 La forme standard d'un Pb linéaire :

$$P:\begin{cases} max(Z) = C^t X \\ AX = B \\ X \geq 0 \end{cases} \tag{1.22}$$

$$P:\begin{cases} min(Z) = C^t X \\ AX = B \\ X \geq 0 \end{cases} \tag{1.23}$$

1.9 Méthodes de résolution :

1.9.1 La résolution graphique :

Définition :

1/ On appelle polytope l'intersection d'un nombre fini des demi-espaces fermés de $\mathbb{R}^n$.
un polytope est un ensemble convexe fermé
La domaine des solution réalisables est un polytope convexe borné (polyèdre)ou non borné
le domaine des solutions réalisables de tout PL à n variables est une partie convexe de $\mathbb{R}^n$.

Effet : il s'agit d'un sous ensemble de $\mathbb{R}^n$ s'exprime comme l'intersection d'un nombre fini de demi-espace fermé de $\mathbb{R}^n$.
2/ On appelle polyèdre un polytope borné .
3/ Étant donné un polyèdre D ,un vecteur $x \in D$ est un point extrême (ou sommet) de D s'il n'existe pas deux vecteurs $y, z \neq x \in D$ et $\alpha \in [0,1]$ tq $x = \alpha y + (1-\alpha)z$

1.9.2 Les étapes de résolution graphique d'un PL :

La résolution d'un PL par la méthode graphique on fait selon une démarche commune ,celle-ci peut être résumé en quatre étapes :

1. Schématiser le domaine réalisable dans un repère orthonormé.
2. Dans le même repère représenter l'hyperplan $(D0) : C^t X = 0$
   Noter que tout hyperplan (D_k) de l'équation $C^t X = k, k \in \mathbb{R}$ parallèle à (D_0).
3. Si le PL considérer est de type maximisation (resp minimisation)indique par une flèche le sens de déplacement parallèle de (D_k) qui fait augment (resp diminue)k .
4. Déterminer la droite D_{kmax} la plus éloignée dans le sens de la flèche de D_0 qui coupe le domaine réalisable D
   Tout point de $D_{kmax} \cap D$ est une solution optimale .
   Si une telle droite D_{kmax} n'existe pas alors on peut dire que l'optimum est infini .

Remarque :

Si $n \geq 3$ alors la méthode graphique n'est pas efficace .

1.9.3 Méthode de simplexe

Nous présentons une méthode de résolution d'un programme linéaire qui a été développé par Dantzig (États-unis) en 1947. Cette méthode appelé algorithme de s implexe,elle consiste à effectuer une exploration dirigée de l'ensemble des solutions de bases réalisable. c'est l'ensemble des points extrême défini par le système des contraintes du problème de programme linéaire.

1. Forme standard d'un programme linéaire par rapport à une base

Soit (P) un P.L sous forme standard :

$$(P) : \begin{cases} \max Z = c^T x \\ Ax \leq b \\ x \geq 0 \end{cases} \tag{1.24}$$

où $x \in \mathbb{R}^n$ et $c \in \mathbb{R}^n$,$b \in \mathbb{R}^m$.

On dispose d'une base B et d'une solution de base réalisable x,avec $A = (B, H)$, $x = (x_B, x_H)^T$ et $c^T = (c_B^T, c_H^T)$.
où
B : une matrice carré inversible d'ordre m.
H : matrice d'ordre $m * (n-m)$.
x_B :variable de base.
x_H :variable hors base.
La fonction objectif s'écrit : $\max Z = c^T x = c_B^T x_B + c_H^T x_H$
Le système $Ax = b$ s'écrit comme suit :$Bx_B + Hx_H = b \Rightarrow x_B = B^{-1}b - B^{-1}HX_H$.
posons $\pi = c_B^T B^{-1}$ appelé vecteur multiplicateur relatif à la base B.Alors $c_H^T - \pi H$ est appelé le cout réduit

relatif à la base B.
Le programme (P) s'écrit sous la forme standard par rapport à la base B :

$$(P): \begin{cases} \max Z = \pi b + (c_H^T - \pi H) x_H \\ x_B + B^{-1} H x_H = B^{-1} b \\ x_B, x_H \geq 0 \end{cases} \tag{1.25}$$

2. Algorithme de simplexe

L'idée de base de ce algorithme consiste à démarrer d'un point extrême (solution de base réalisable) x_0 de l'ensemble des solution réalisable D en suite à se déplacer vers un autre point extrême x_1 de D dont la valeur de la fonction objectif $Z(x_0)$ est meilleur que $Z(x_0)$ ainsi se suite jusqu'à ce qui on arrive à une solution de base réalisable optimale ou bien jusqu'à on se rendre compte que le(P) n'est pas borné inférieurement.

L'algorithme suppose qu'on a une solution de base réalisable initiale,on note par Il'ensemble des indices qui correspondant à l'ensemble de bases réalisable et par Jl'ensemble des indices qui correspondant à l'ensemble des solution hors bases.

Étapes de l'algorithme

1. Étape 0 : Mettre le PL sous forme standard par rapport à une solution de base réalisable I.
2. Étape 1 : Déterminer le coefficient des variable hors base de la fonction objectif,
   et on pose $C_s = \max_{j \in J}(c_j)$
   — Si $C_s \leq 0$,stop la solution de base correspondant à I est optimale.
   — Si $C_s > 0$, aller en étape 2.
3. Étape 2 : Examiner le vecteur A_s (vecteur colonne) .
   — Si $\{i : A_s^i > 0\} = \varnothing$ stop le PL n'a pas de solution bornée.
   — Si $\{i : A_s^i > 0\} \neq \varnothing$ alors déterminer K

   $$K = \{k : \frac{b_k}{A_s^k} = \min\{\frac{b_i}{A_s^i} : A_s^i > 0\}\}$$

   et aller en étape 3.
4. Étape 3 : Choisir $r \in K$ et mettre le PL sous forme standard par rapport à la nouvelle base
   $I = (I - \{r\}) \cup \{s\}$ et aller en 1.

3. Méthode de deux phases

L'application de la méthode du simplexe à la résolution des PL suppose la connaissance préalable d'une solution de base réalisable de départ.
Lorsque celle ci ne peut s'obtenir de manière immédiate, on peut utilise des méthodes qui permettent de construire de manière systématique cette solution. La méthode de deux phases repasse sur l'introduction des variable artificielles.

Phase 1 : Consiste annuler toutes les variables artificielles en utilisent une fonction objectif artificielle.
Si l'on ne peut pas annules toutes les variables alors problème n'est pas réalisable.

Phase 2 : Consiste à remplacer la fonction objectif artificielle de la phase 1 par la vrais fonction objectif à maximiser, on utiliser alors la solution de base réalisable obtenue à la fin de la phase 1.

1.9.4 La dualité :

Définition :

soit :

$$P : \begin{cases} max(Z) = C^t X \\ AX \geq B \\ X \geq 0 \\ X \in \mathbb{R}^n, C \in \mathbb{R}^n, B \in \mathbb{R}^m, A \in \mathbb{R}^{n \times m} \end{cases} \tag{1.26}$$

(P) est un pb primale avec n variables et m contraintes
On appelle le duale de ce programme linéaire le programme linéaire suivant :

$$D = Dual(P) : \begin{cases} min(W) = Y^t B \\ Y^t A \geq C \\ Y \geq 0 \\ avec Y \in \mathbb{R}^m \end{cases} \tag{1.27}$$

Théorème 01 :

Le problème dual du problème dual est le problème primal.

Théorème 02 :

Soit $\bar{X}$ solution réalisable de (P) et $\bar{Y}$ solution réalisable de(D) tel que : $C^t.\bar{X} = \bar{y}^t.B$.
alors $\bar{X}$ et $\bar{Y}$ sont des solutions optimales de (P) et (D) respectivement.

Chapitre 2

L'OPTIMISATION MULTI-OBJECTIF

Dans ce chapitre, nous introduisons dans un premier temps quelques concepts et ensembles des définitions et généralités de base très importantes liées au thème de l'optimisation combinatoire multi-objectifs qui seront très utiles pour la suite de ce travail.Utiles pour comprendre la suite du ce mémoire.
Ensuite, nous ferons un tour rapide des méthodes de résolution du domaine,

2.1 Définition d'un Problème d'optimisation multiobjectif :

Un problème d'optimisation multi-objectif a pour but de optimiser simultanément plusieurs critères définis dans le même espace. Par exemple, de nombreuses conceptions de produits ou systèmes cherchent à minimiser le coût de leur production tout en maximisant leur performance. Ces deux critères d'optimisation sont contradictoires et la solution du problème est un compromis entre les deux objectifs
Cependant lorsque l'on modélise un problème, on cherche souvent à satisfaire plusieurs objectifs.
Par exemple, on veut un système performant et on veut aussi que ce système consomme peu. Dans ce cas, on parle de problème d'optimisation multi-objectif (ou problème d'optimisation multicritère). Celui-ci s'écrit de la manière suivante :

$$P : \begin{cases} optimiser\, f(x) \\ h(x) = 0 \\ g(x) \leqslant 0 \\ où \\ x \in \mathbb{R}^n, f(x) \in \mathbb{R}^k, g(x) \in \mathbb{R}^m, h(x) \in \mathbb{R}^p \end{cases} \tag{2.1}$$

Plus précisément un problème d'optimisation multi-objectif peut être défini comme un problème où le but est d'optimiser k fonctions objectif simultanément (Maximiser k fonctions, Minimiser k fonctions, Maximiser et Minimise k fonctions).
La présence de multiples objectifs est en conflit naturel dans de nombreux problèmes et rend le problème d'optimisation intéressant à résoudre, car aucune solution ne peut être qualifiée comme une solution optimale aux objectifs contradictoires multiples, dans un problème d'optimisation multicritère, on rencontre souvent des objectifs contradictoires. Deux objectifs sont contradictoires lorsque la diminution d'un objectif entraîne une augmentation de l'autre objectif. Le problème d'optimisation multi-Objectif résultant recourt à un certain nombre de solutions compromis optimal .
Le but que l'on se fixe dans la résolution d'un problème d'optimisation multiobjectif est de optimiser "au mieux" les différents objectifs.

2.1.1 Vocabulaire et définitions :

Tout d'abord nous définissons les notions communes à n'importe quelle méthode d'optimisation multicritère :

Fonction objective :

C'est le nom donné à la fonction f (on l'appelle encore fonction de coût ou critère d'optimisation). C'est la fonction que l'algorithme d'optimisation va devoir "optimiser" (trouver un optimum)

L'ensemble de contraintes :

Définit des conditions sur l'espace d'état que les variables doivent satisfaire. Ces contraintes sont souvent des contraintes d'inégalité ou d'égalité et permettent en général de limiter l'espace de recherche.

Vecteur de décision :

Une action (ou un vecteur de décisions) sera notée $x = (x_1, x_2, \ldots, x_n)$ avec x_i les variables du problème (variables de décision) et n le nombre de variables, C'est en faisant varier ce vecteur que l'on recherche un optimum de la fonction f .

Espace de recherche :

Représentant l'ensemble des valeurs pouvant être prises par les variables.

Espace réalisable :

Représentant l'ensemble des valeurs des variables satisfaisant les contraintes.

2.2 La multiplicité des solution :

Contrairement à l'optimisation mono-objectif,la solution d'un problème multi-objectif n'est pas unique, en raison de la contradiction et d'une éventuelle incompatibilité des fonctions objectifs.
De ce fait il est impossible de trouver une seule solution qui serait optimale pour tous les objectifs simultanément, mais un ensemble de solutions non dominées, connues comme l'ensemble des solutions de meilleur compromis. Compromis signifie que l'on ne peut pas augmenter le niveau de satisfaction pour un objectif sans réduire cela pour un autre objectif. Pour des problèmes de programmation multi-objectifs le concept de solutions non-dominées est utilisé. Une solution de compromis est choisie parmi l'ensemble des solutions non-dominées
Autrement dit

Lorsque l'on cherche à obtenir une solution optimale à un problème d'optimisation multiobjectif donné, on s'attend souvent à trouver une solution et une seule. En fait, La plupart du temps, on trouve une multitude de solutions, du fait que certains des objectifs sont contradictoires.

Donc, quand on résoudra un problème d'optimisation multiobjectif, on obtiendra une grande quantité de solutions. Ces solutions, comme on peut s'en douter, ne seront pas optimales, au sens où elles ne optimisera pas tous les objectifs du problème. Un concept intéressant, qui nous permettra de définir les solutions obtenues, est le compromis. En effet, les solutions que l'on obtient lorsque l'on a résolu le problème sont des solutions de compromis. Elles optimisent un certain nombre d'objectifs tout en dégradant les performances sur d'autres objectifs.

2.3 Notion de dominance :

En raison de la nature conflictuelle des objectifs, Lorsque nous avons résolu notre problème d'optimisation multiobjectif, il n'y a généralement pas une solution réalisable qui optimise simultanément tous les objectifs alors nous avons obtenu une multitude de solutions. Seul un nombre restreint de ces solutions va nous intéresser. Pour qu'une solution soit intéressante, il faut qu'il existe une relation de dominance et l'efficacité au sens de Pareto entre la solution considérée et les autres solutions, dans le sens suivant :

2.3.1 La relation de dominance :

On dit que le vecteur x_1 domine le vecteur x_2 si :
– x_1 est au moins aussi bon que x_2 dans tous les objectifs, et,
– x_1 est strictement meilleur que x_2 dans au moins un objectif.

2.3.2 Efficacité :

La notion d'une solution efficace a été introduite par Pareto en 1896, par conséquent, la connotation d'un ensemble efficace est aussi connue sous le nom Pareto-optimale .

Définition :

Une solution $x \in D$ est une solution efficace pour un problème d'optimisation multiobjectif s'il n'existe pas de $x_0 \in D$ tel que $f(x_0)$ domine $f(x)$.
Un point est efficace si son image par f est un vecteur critère non dominé.

Remarque :

A partir d'un point efficace, il est impossible d'augmenter la valeur d'un des critères sans diminuer la valeur d'au moins un autre critère.

2.3.3 Optimalité au sens de Pareto :

Les solutions qui dominent les autres mais ne se dominent pas entre elles sont appelées solutions optimales au sens de Pareto (ou solutions non dominées). Ensemble des solutions optimales au sens Pareto sont situées sur la limite de l'espace de critère F
Nous pouvons maintenant définir le concept d'optimalité locale et globale au sens de Pareto :

Optimalité local au sens de pareto :

Un vecteur $x \in \mathbb{R}^n$ est optimal localement au sens de Pareto s'il existe un réel $\delta > 0$ tel qu'il n'y ait pas de vecteur x' qui domine le vecteur x avec $x' \in \mathbb{R}^n \cap B(x,\delta)$ où $B(x,\delta)$ représente une boule de centre x et de rayon δ. Un vecteur x est donc optimal localement au sens de Pareto s'il est optimal au sens de Pareto sur une restriction de l'ensemble $\mathbb{R}^n$,Cette définition est illustrée par la figure 2.1 .

Optimalité global au sens de pareto :

Un vecteur x est optimal globalement au sens de Pareto (ou optimal au sens de Pareto) s'il n'existe pas de vecteur x' tel que x' domine le vecteur x.

La différence entre cette définition et celle de l'optimalité locale tient dans le fait que l'on ne considère plus une restriction de l'ensemble $\mathbb{R}^n$.

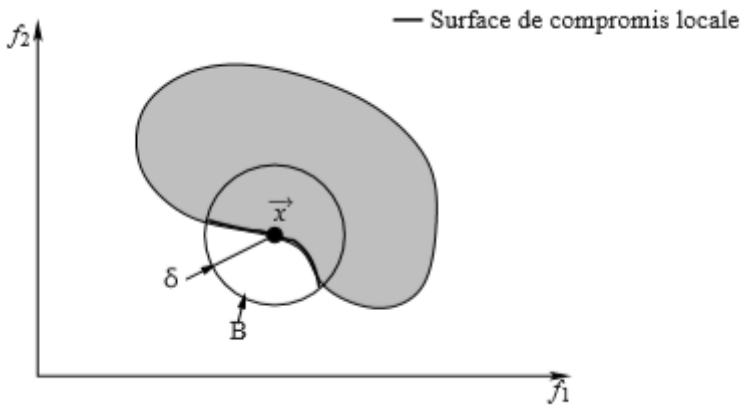


FIGURE 2.1 – L'optimalité locale au sens de Pareto.

Une version "graphique" de la précédente définition (optimalité global au sens de pareto) utilise le théorème du contact.

Théorème du contact :

Un vecteur x est optimal au sens de Pareto pour un problème d'optimisation multiobjectif donné si :
$(C^- + x) \cap F = \{x\}$
où F désigne l'espace des solutions réalisables.
L'utilisation de ce théorème est illustrée par la figure 2.2.

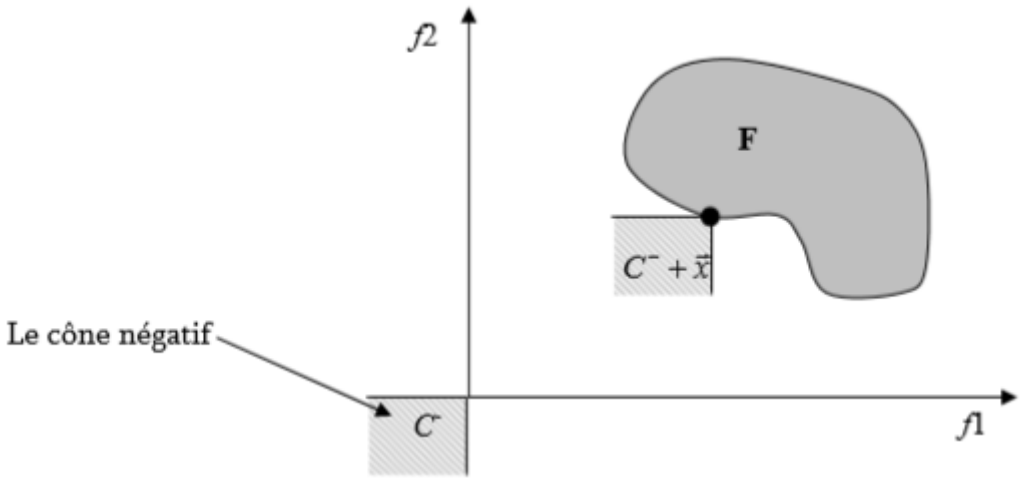


FIGURE 2.2 – Le théorème du contact.

Définition d'un C*ô*ne négative :

Un cône négatif est défini dans $\mathbb{R}^k$ de la manière suivante : $C^- = \{x \mid f(x) \in \mathbb{R}^k et f(x) \leq 0\}$.

Les niveaux de préférence dans la relation de dominance :

Lorsque l'on applique la définition de la dominance, on peut définir quatre régions aux quelles on peut attribuer des niveaux de préférence. Ces régions sont représentées à la figure 2.3. Cette figure reprend le découpage défini par le cône négatif que l'on a introduit précédemment et l'étend à tout l'espace.
Par exemple, si ce graphique est centré sur une solution A et que l'on compare cette solution avec une solution B, on aura les possibilités suivantes :
• si la solution B se trouve dans le quadrant 1, alors la solution A est préférée à la solution B
• si la solution B se trouve dans le quadrant 3, alors la solution A est dominée par la solution B
• si la solution B se trouve dans l'un des quadrants 2 ou 4, alors, on ne peut pas se prononcer sur la préférence de A par rapport à B ou de B par rapport à A.

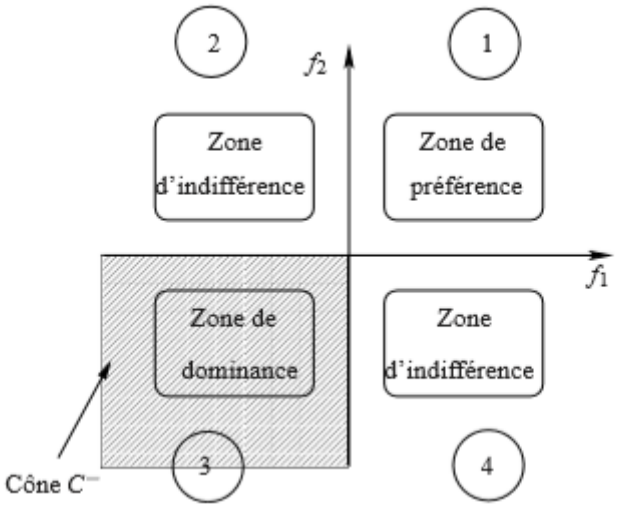


FIGURE 2.3 – Les niveaux de préférence dans la relation de dominance.

2.4 La surface de compromis (front de Pareto) :

2.4.1 Définition :

Soit $x = [x_1, ...x_n]^t$ un vecteur de décision avec :
$x \in X$ (L'espace réalisable) x est dit Pareto optimal, s'il n'existe pas une solution y domine x.
Une solution Pareto optimal appelée aussi : solution efficace ou non dominée solution.
Autrement
Le front (frontière) de Pareto est l'ensemble des solutions Pareto optimales qui sont composées des points, ne sont dominés par aucun autre
Le front de Pareto appelé aussi surface de compromis ou l'ensemble des solutions efficaces .
La figure 2.4 , représente le Front du Pareto pour un problème de minimisation et maximisation de deux fonctions objective
Une propriété est remarquable : en fonction du type de problème que l'on cherche à résoudre, on obtient une forme de surface de compromis.
Les formes les plus courantes de surfaces de compromis sont réunies à la figure 2.6.

Un exemple de surface de compromis (front Pareto) en dimension 2 est montré à a la figure 2.6. Dans cet exemple, le problème considéré est un problème de minimisation avec deux critères. Deux points particuliers apparaissent clairement :

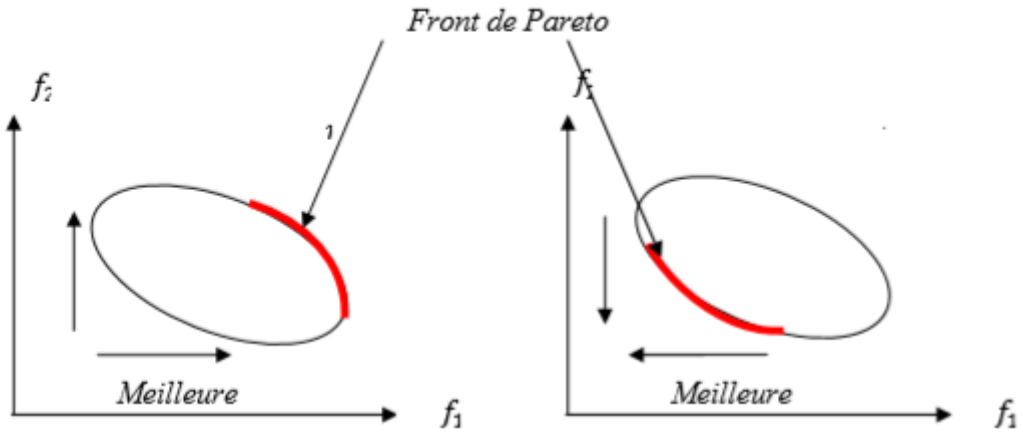


FIGURE 2.4 – Le front de Pareto.

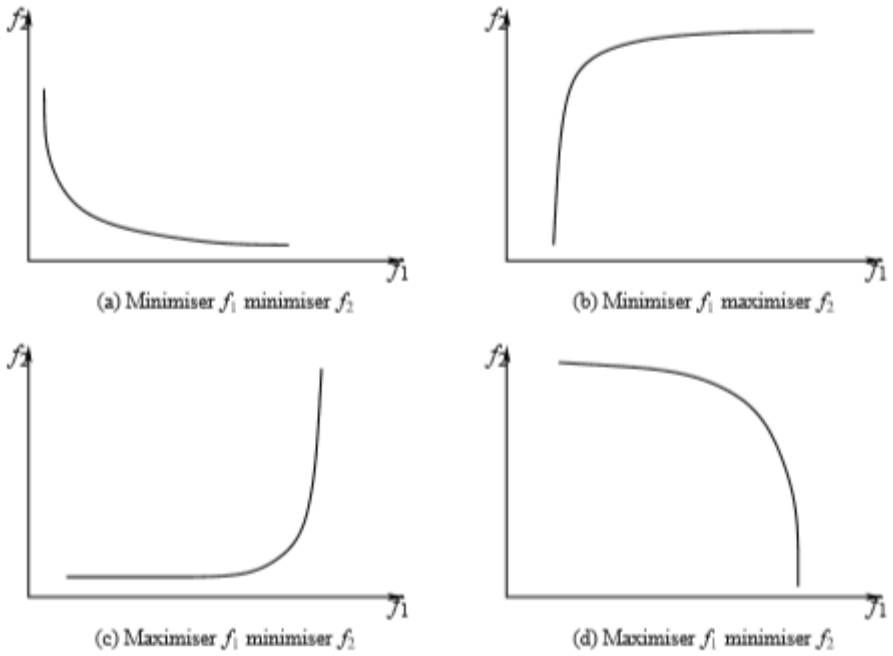


FIGURE 2.5 – Formes les plus courantes de la surface de compromis dans le cas de deux objectifs.

Point idéal :

Les coordonnées du point idéal correspondent aux meilleurs valeurs de chaque objectif des points du front Pareto.
Les coordonnées de ce point correspondent aussi aux valeurs obtenues en optimisant chaque fonction objectif séparément.

Point nadir :

Les coordonnées de ce point correspondent aux pires valeurs obtenues par chaque fonction objectif lorsque l'on restreint l'espace des solutions à la surface de compromis.
Le point idéal est utilisé dans beaucoup de méthodes d'optimisation comme point de référence. Le point nadir, lui, sert à restreindre l'espace de recherche ; il est utilisé dans certaines méthodes d'optimisation interactives.
Les coordonnées de ces deux points sont maintenant définies formellement.

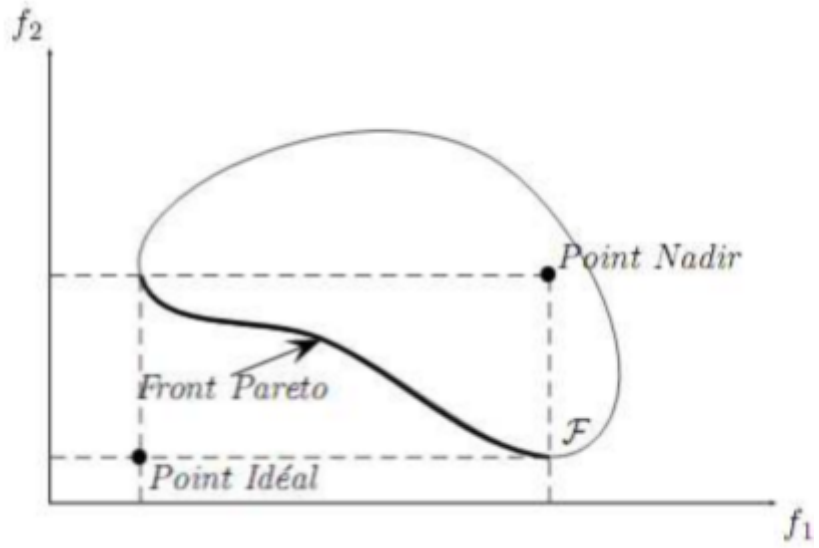


FIGURE 2.6 – Front pareto.

2.4.2 La convexité :

Comme nous le verrons plus loin, certaines méthodes d'optimisation multiobjectif nécessitent de respecter certaines hypothèses.
Le plus souvent, la méthode réclame de travailler sur un espace S des valeurs de $f(x)$ qui soit convexe.

Définition :

Un ensemble S est convexe si, étant donnés deux points distincts quelconques de cet ensemble, le segment qui relie ces deux points est contenu dans l'ensemble S.
Un exemple d'ensemble convexe et un exemple d'ensemble non convexe sont représentés à la figure 2.7.
La convexité est le premier indicateur de la difficulté du problème.
En effet, certaines méthodes sont dans l'incapacité de résoudre des problèmes non convexes de manière optimale.
Mais il existe d'autres indicateurs tout aussi importants, notamment la continuité, la multi-modalité, la nature des variables de décision (entières ou réelles), . . .

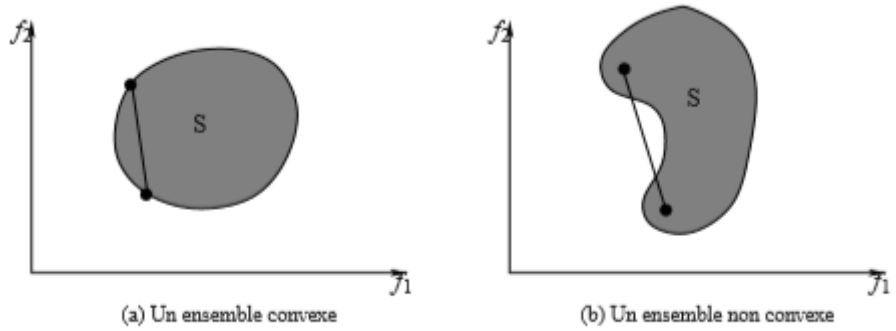


FIGURE 2.7 – Exemples d'ensemble convexe et d'ensemble non convexe.

2.4.3 La représentation de la surface de compromis :

Toutes les représentations de la surface de compromis, pour un même problème, ne sont pas équivalentes.
En effet, la représentation idéale de la surface de compromis devra être constituée de points solution de notre problème répartis de manière uniforme sur la surface de compromis (voir figure 2.8).
Dans le premier cas, les points représentant la surface de compromis ne sont pas répartis de manière uniforme. L'utilisateur n'aura alors pas en sa possession un ensemble de solutions très utile. En effet, s'il décide que la solution qu'il avait choisie ne lui convient pas, le choix d'une autre solution risque de faire varier brusquement tous ses objectifs, et cette nouvelle solution ne lui conviendra pas non plus. Il est alors probable que la solution offrant le "meilleure " compromis se trouve dans une zone qui ne soit pas représentée par des points solution.
La détermination d'une bonne représentation de la surface de compromis sera un critère de choix d'une méthode d'optimisation multiobjectif.

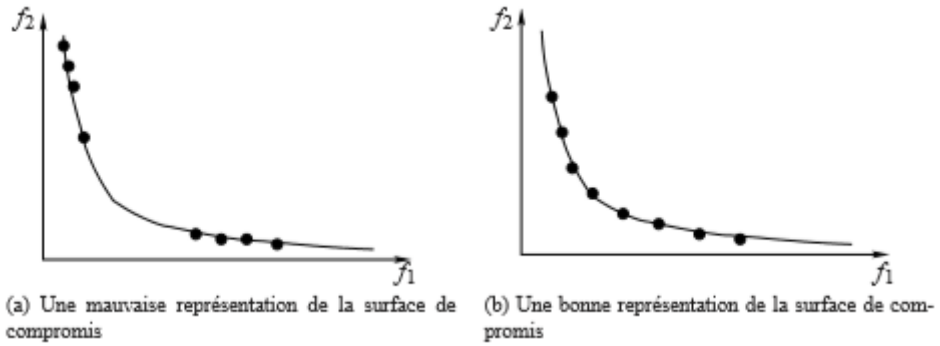


FIGURE 2.8 – La représentation de la surface de compromis.

2.5 Les Conditions d'optimalités d'un problème d'optimisation multi-objectif :

Résoudre les problèmes d'optimisation multi-objectif de la forme :

$$P:\begin{cases} optimiser f(x) \\ h(x)=0 \\ g(x)\leqslant 0 \\ où \\ x\in\mathbb{R}^n, f(x)\in\mathbb{R}^k, g(x)\in\mathbb{R}^m, h(x)\in\mathbb{R}^p \end{cases} \tag{2.2}$$

consiste à rechercher les points qui ne sont pas dominés. Ces solutions seront les solutions optimales au sens de Pareto (l'ensemble des meilleurs compromis).
Ces problèmes d'optimisation peuvent être résolu en appliquant les condition d'optimalités mais les problèmes de grande dimension sont très difficiles à résoudre alors on ne peut pas appliquant les condition d'optimalités, alors en utilisant les méthodes de résolution des problèmes d'optimisation multi-objectif.

2.5.1 Les Conditions nécessaires d'optimalités :

Comme en optimisation mono-objectif, un point solution d'un problème d'optimisation multi-objectif sous contraintes vérifie des conditions du premier ordre (c'est-à-dire portant sur la dérivée de la fonction co*û*t et des contraintes) qui donnent lieu à un système d'équations vérifié par le point solution et des paramètres associés aux contraintes, les coefficients de Lagrange.

Théorème :

Soient $f:\mathbb{R}^n\longrightarrow\mathbb{R}^k, g:\mathbb{R}^n\longrightarrow\mathbb{R}^m$ et $h:\mathbb{R}^n\longrightarrow\mathbb{R}^p$ des fonctions de classe C^1. Si x^* est solution du problème d'optimisation multi-objectif, alors il existe $\omega\in\mathbb{R}^k_+, \mu\in\mathbb{R}^m_+, \lambda\in\mathbb{R}^p$ Avec $(\omega,\mu,\lambda)\neq(0,0,0)\subset\mathbb{R}^k_+ * \mathbb{R}^m_+ * \mathbb{R}^p$ tels que :

$$\begin{cases} \sum_{i=1}^k \omega_i\nabla f_i(x^*)+\sum_{j=1}^m \mu_j\nabla g_j(x^*)+\sum_{l=1}^p \lambda_l\nabla h_l(x^*) \\ \mu_j g_j(x^*)=0 \qquad\qquad j=1,....,m. \\ i=(1,....,k). \end{cases} \tag{2.3}$$

Les coordonnées du point x^* optimal au sens de Pareto et les coefficients de Lagrange λ et μ satisfont donc les $(n+m+p)$ équations :

$$\begin{cases} \sum_{j=1}^k \omega_j\dfrac{\partial f_j(x^*)}{\partial x_i}+\sum_{l=1}^m \mu_l\dfrac{\partial g_l(x^*)}{\partial x_i}+\sum_{q=1}^p \lambda_q\dfrac{\partial h_q(x^*)}{\partial x_i} \\ \mu_j g_j(x^*)=0 \qquad\qquad j=(1,....,m). \\ h_l(x^*)=0 \qquad\qquad l=(1,....,p). \\ i=(1,....,k). \end{cases} \tag{2.4}$$

Il existe des poids $\omega_1^{'},...,\omega_k^{'}$(qui dépendent de x^*) tels que l'on peut appliquer le théorème de Fritz John mono-objectif à la fonction pondérée $f_{\omega'}(x)=\sum_{i=1}^k \omega_i^{'} f_i(x)$. Le théorème de Fritz John fait alors apparaitre un coefficient λ_0 tel que $\lambda_0\omega_i^{'}=\omega_i$ pour $i=1,...,k$. On revient alors à de l'optimisation monobjectif. Cependant, le théorème donne simplement l'existence de ces poids, sans donner de moyen de les trouver. La première égalité peut s'écrire à l'aide des différentielles des fonctions :
$df^T_{x^*}(\omega)+dg^T_{x^*}(\mu)+dh^T_{x^*}(\lambda)=0.$

soit sous forme matricielle :

$$\begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x^*) & \ldots\ldots & \frac{\partial f_k}{\partial x_1} \\ \vdots & \ldots\ldots & \vdots \\ \vdots & & \vdots \\ \frac{\partial f_1}{\partial x_n}(x^*) & \ldots\ldots & \frac{\partial f_k}{\partial x_n} \end{pmatrix} \begin{pmatrix} \omega_1 \\ \vdots \\ \vdots \\ \omega_k \end{pmatrix} + \begin{pmatrix} \frac{\partial g_1}{\partial x_1}(x^*) & \ldots\ldots & \frac{\partial g_m}{\partial x_1} \\ \vdots & \ldots\ldots & \vdots \\ \vdots & & \vdots \\ \frac{\partial g_1}{\partial x_n}(x^*) & \ldots\ldots & \frac{\partial g_m}{\partial x_n} \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \vdots \\ \lambda_m \end{pmatrix} +$$
$$\begin{pmatrix} \frac{\partial h_1}{\partial x_1}(x^*) & \ldots\ldots & \frac{\partial h_m}{\partial x_1} \\ \vdots & \ldots\ldots & \vdots \\ \vdots & & \vdots \\ \frac{\partial h_1}{\partial x_n}(x^*) & \ldots\ldots & \frac{\partial h_p}{\partial x_n} \end{pmatrix} \begin{pmatrix} \mu_1 \\ \vdots \\ \vdots \\ \mu_k \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ \vdots \\ 0 \end{pmatrix}.$$

2.5.2 Conditions suffisantes d'optimalité :

Pour que les solutions du système d'équations obtenues grâce au théorème de Fritz John soient les solutions du problème d'optimisation multi-objectif sous contraintes, les fonctions coût et les contraintes doivent être convexes. On obtient alors le théorème de Karush-Kuhn et Tucker :

Théorème (Théorème de Karush-Kuhn-Tucker) :

Soient $f : \mathbb{R}^n \longrightarrow \mathbb{R}^k, g : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ et $h : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ des fonctions de classe C^1. et telles que les fonctions f_i, g_j, h_l sont convexes. S'il existe $\omega \in \mathbb{R}^k_+, \lambda \in \mathbb{R}^p_+, \mu \in \mathbb{R}^m$ tels que

$$\begin{cases} \sum_{i=1}^{k} \omega_i \nabla f_i(x^*) + \sum_{j=1}^{m} \mu_j \nabla g_j(x^*) + \sum_{l=1}^{p} \lambda_l \nabla h_l(x^*) \\ \mu_j g_j(x^*) = 0 \quad , \mu_j \geq 0 \quad / j = 1, ..., m. \end{cases} \tag{2.5}$$

alors x est solution du problème multi-objectif :

$$P : \begin{cases} optimiser f(x) \\ h(x) = 0 \\ g(x) \leqslant 0 \\ où \\ x \in \mathbb{R}^n, f(x) \in \mathbb{R}^k, g(x) \in \mathbb{R}^m, h(x) \in \mathbb{R}^p \end{cases} \tag{2.6}$$

Puisque dans ce cas, les fonctions coût f_i et les contraintes sont convexes, le minimum x^* est un minimum global du problème d'optimisation.

2.6 Les différents méthodes de résolution des problèmes d'optimisation multiobjectif :

Les méthodes d'optimisation multicritères s'intéressent aux problèmes essayant d'optimiser simultanément plusieurs objectifs.
Il existe un nombre important de méthodes de résolution des problèmes d'optimisation multi-objectif.
Ces méthodes peuvent être classées de différentes façons, selon la manière dont le concepteur va prendre en compte les critères d'optimisation dans la résolution du problème. Par exemple, Van Veldhuizen et al[6]. propose de trier les méthodes en 3 grandes familles de stratégies :

2.6.1 Les méthodes à préférence a priori :

Dans ces méthodes, l'utilisateur définit le compromis qu'il désire réaliser (il fait part de ses préférences) avant de lancer la méthode d'optimisation.

On retrouve dans cette famille la plupart des méthodes par agrégation (où les fonctions objectif sont fusionnées en une seule).

2.6.2 Les méthodes à préférence progressive ou interactive :

Dans ces méthodes, l'utilisateur affine son choix de compromis au fur et à mesure du déroulement de l'optimisation.
On retrouve dans cette famille les méthodes interactives.

2.6.3 Les méthodes à préférence a posteriori :

Dans ces méthodes,l'utilisateur choisit une solution de compromis en examinant toutes les solutions extraites par la méthode l'optimisation.
Les méthodes de cette famille fournissent, à la fin de l'optimisation, une surface de compromis.

Aussi, Colette et al. propose de classer ces mêmes méthode d'optimisation en cinq catégories :

2.6.4 les méthodes scalaires :

ces méthodes consistent à transformer le problème d'optimisation multiobjectif en un problème d'optimisation mono-objectif. Ces méthodes sont des méthodes de résolution à priori.
Quelque méthodes/

La méthode de pondération des fonctions objectif

Principe : Cette approche de la résolution d'un problème d'optimisation multi-objectif est la plus évidente.
D'ailleurs, on appelle aussi cette méthode "l'approche naïve" de l'optimisation multi-objectif. Le but, ici, est de revenir à un problème d'optimisation monobjectif, dont il existe de nombreuses méthodes de résolution. La manière la plus simple de procéder consiste à prendre chacune des fonctions objectif, à leur appliquer un coefficient de pondération et à faire la somme des fonctions objectif pondérées. On obtient alors une nouvelle et unique fonction objectif.

La méthode du compromis

Principe : Après avoir étudié les méthodes qui permettent de fusionner les fonctions objectif en une seule(on parle aussi "d'agrégation" des fonctions objectif),nous allons étudier des méthodes qui permettent de transformer un problème d'optimisation multi-objectif en un problème d'optimisation monobjectif comportant quelques contraintes supplémentaires.
La démarche est la suivante :
– on choisit un objectif à optimiser prioritairement.
– on choisit un vecteur de contraintes initial.
– on transforme le problème en conservant l'objectif prioritaire et en transformant les autres objectifs en contraintes d'inégalité.
On appelle aussi cette méthode la méthode de la contrainte ε

2.6.5 les méthodes interactives :

Les méthodes interactives permettent de chercher une et une seule solution. Elles forment la famille des méthodes progressives et permettent à l'utilisateur de déterminer ses préférences vis-à-vis d'un compromis entre objectifs au cours de l'optimisation.
Ces méthodes sont à comparer aux méthodes à préférence a priori et à préférence a posteriori.

2.6.6 les méthodes floues :

Dans la vie courante, tout ne peut pas être décrit de manière binaire. Par exemple, la transition entre le jour et la nuit se fait progressivement, l'action sur l'embrayage d'un véhicule est, elle aussi, progressive.
Longtemps, le seul outil de description en logique était binaire. Tout en logique a été décrit en termes de VRAI ou FAUX. Le problème de cette description simpliste en logique est qu'elle ne permet pas de traiter l'incertitude et l'imprécision des connaissances humaines. L'automaticien L. A. Zadeh a élaboré une nouvelle logique basée sur les ensembles flous.
Elle permet de traiter l'imprécision et l'incertitude dans la connaissance humaine ainsi que les transitions progressives entre états. La différence principale entre une logique classique et cette logique floue est l'existence d'une transition progressive entre le VRAI et le FAUX Parmi les méthodes floues, nous pouvons citer la méthode de Sakawa ou encore la méthode de Reardon

2.6.7 les méthodes exploitant une méta-heuristique :

Ces méthodes de résolution sont dédiées aux problèmes d'optimisation « difficiles » . Les méta-heuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global. Les méthodes utilisant une méta-heuristique les plus connues sont l'algorithme du recuit simulé , la recherche tabou et l'algorithme génétique .
Dans le chapitre suivant Nous concentrant et présentons plus en détaille les méthodes exploitant une méta-heuristique

2.6.8 les méthodes d'aide à la décision :

Les méthodes que nous avons présentées jusqu'ici étaient basées sur la relation de dominance. Cette relation (que l'on peut définir de plusieurs manières : la dominance de Pareto par exemple) permet de filtrer les éléments d'un ensemble, et de ne retenir que les éléments incomparables entre eux. Cependant, il existe une autre approche pour obtenir un ensemble de solutions, qui repose sur l'établissement d'une relation d'ordre entre les différents objectifs du problème d'optimisation. . Ainsi, on peut, en fonction de la relation d'ordre définie, obtenir un ensemble de solutions (relation d'ordre partiel) ou une et une seule solution (ordre complet). L'autre différence majeure, par rapport aux méthodes d'optimisation multi-objectif "classiques", vient du fait que les méthodes d'aide à la décision ne travaillent que sur des ensembles discrets de points (les méthodes d'optimisation multi-objectif "classiques" peuvent travailler, elles, sur des ensembles continus).

2.7 Classification des Méthodes d'optimisation multiobjectif :

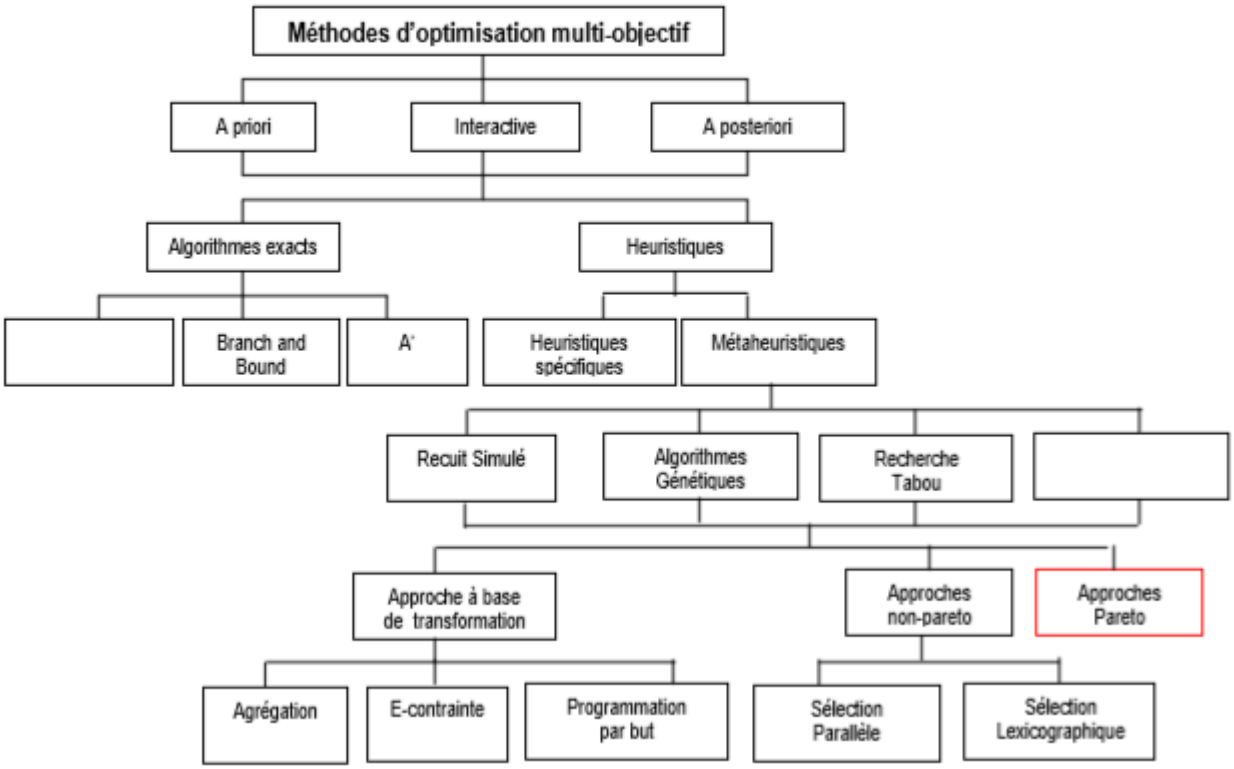


FIGURE 2.9 – Classification des Méthodes d'optimisation multi-objectif

Chapitre 3

LES MÉTHODE EXPLOITANT UNE MÉTA-HEURISTIQUE

Ce chapitre a pour objet de présenter les différentes ensembles des définitions très importantes liées au les méta-heuristique qui seront très utiles pour la suite de ce travail, en nous concentrant sur l'aspect résolution de problèmes d'optimisation combinatoires multi-objectif à l'aide de méta-heuristiques, nous retenons celle nous basée sur la recuit simulé .
Ensuite, il existe plusieurs versions du recuit simulé multi-objectif.
Parmi lesquels, on va présenter et concentrant sur la méthodes P.A.S.A (Pareto Archived Simulated Annealing).
Ainsi, nous avons présenté dans ce chapitre une description générale, plus en détail de l'algorithme de P.A.S.A et donnons quelques exemples et test pour ce dernier .

3.1 Heuristique / Méta-heuristique d'optimisation :

3.1.1 Définition de la heuristique

En optimisation combinatoire, une heuristique est un algorithme approché, ou méthode approximative qui permet d'identifier en temps polynomial au moins une solution réalisable rapide, pas obligatoirement optimale , pour un problème d'optimisation NP-difficile. On oppose les méthodes approchées aux méthodes exactes, qui trouvent toujours l'optimum si on leur en laisse le temps (énumération complète, méthodes arborescentes, programmation dynamique). ou Une heuristique est une technique de résolution spécialisée à un problème. Elle ne garantit pas la qualité du point obtenu.

L'usage d'une heuristique est efficace pour calculer une solution approchée d'un problème et ainsi accélérer le processus de résolution exacte. Généralement une heuristique est conçue pour un problème particulier, en s'appuyant sur sa structure propre sans offrir aucune garantit quant à la qualité de la solution calculée.
Les heuristiques peuvent être classées en deux catégories :
– Méthodes constructives qui génèrent des solutions à partir d'une solution initiale en essayant de ajouter petit à petit des éléments jusqu'à ce qu'une solution complète soit obtenue,
– Méthodes de fouilles locales qui démarrent avec une solution initialement complète (probablement moins intéressante), et de manière répétitive essaie d'améliorer cette solution en explorant son voisinage.
Une méthode heuristique est généralement conçue pour un problème particulier, elles ne sont en général applicables qu'a un seul type de problèmes. en s'appuyant sur sa structure propre.Ce qui a poussé les chercheurs à proposer de nouvelles méthodes générales, On parle de méta-heuristique pour les méthodes approximatives générales, pouvant s'appliquer à différents problèmes.

3.1.2 Définition d'une méta-heuristique

Le terme méta-heuristique vient des mots grecs méta (au delà) et heuriskein (trouver) signifiant niveau supérieur méthodologie. Les méta-heuristiques sont un ensemble d'algorithmes d'optimisation visant à résoudre les problèmes d'optimisation difficiles. elles reprennent des idées que l'on retrouve parfois dans la vie courante. Les principales méta-heuristiques sont le recuit simulé, la recherche tabou et les algorithmes génétiques
Ces algorithmes sont plus complets et complexes qu'une simple heuristique, et permettent généralement d'obtenir une solution de très bonne qualité pour des problèmes issus des domaines de la recherche opérationnelle ou de l'ingénierie dont on ne connait pas de méthodes efficaces pour les traiter ou bien quand la résolution du problème nécessite un temps élevé ou une grande mémoire de stockage.
Pour résumer ces définitions, on peut dire que les propriétés fondamentales des méta-heuristiques sont les suivantes :

1. Les méta-heuristiques sont des stratégies qui permettent de guider la recherche d'une solution optimale
2. Le but visé par les méta-heuristiques est d'explorer l'espace de recherche efficacement afin de déterminer des solutions (presque) optimales.
3. Les techniques qui constituent des algorithmes de type méta-heuristique vont de la simple procédure de recherche locale à des processus d'apprentissage complexes.
4. Les méta-heuristiques sont en général non-déterministes et ne donnent aucune garantie d'optimalité
5. Les méta-heuristiques peuvent contenir des mécanismes qui permettent d'éviter d'être bloqué dans des régions de l'espace d recherche.
6. Les concepts de base des méta-heuristiques peuvent être décrit de manière abstraite, sans faire appel à un problème spécifique.
7. Les méta-heuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problème traité, mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.
8. Les méta-heuristiques peuvent faire usage de l'expérience accumulée durant la recherche de l'optimum, pour mieux guider la suite du processus de recherche.

3.2 Classification des méthodes :

Il existe plusieurs façons de classer les méta-heuristiques.
On en donne quelques une, et nous adopterons celle faisant la différence entre les méthodes de trajectoire et les méthodes basées sur une population.
1/ Méthodes de trajectoire :
Manipulent un seul point à la fois et tentent itérativement d'améliorer ce point.
Elles construisent une trajectoire dans l'espace des points en tentant de se diriger vers des solutions.
Par exemple :
La recherche locale.
Le recuit simulé .
La recherche tabou .
La recherche à voisinages variables .
2/ Méthodes qui travaillent avec une population de points :
en tout temps on dispose d'une "base" de plusieurs points, appelée population.
L'exemple le plus connu est l'algorithme génétique.

3.2.1 Les méta-heuristiques à solution unique

Dans cette section, nous présentons les méta-heuristiques à base de solution unique, aussi appelées méthodes de trajectoire.
Contrairement aux méta-heuristiques à base de population, les méta-heuristiques à solution unique commencent avec une seule solution initiale et s'en éloignent progressivement, en construisant une trajectoire dans l'espace de recherche.
Les méthodes de trajectoire englobent essentiellement la méthode de descente, la méthode du recuit simulé, la recherche tabou, la méthode GRASP, la recherche à voisinage variable, la recherche locale itérée, et leurs variantes.
Alors ona quelque méthodes :

Méthode de descente :

La méthode de descente (DM : Descent method) est l'une des méthodes les plus simples de la littérature.
Elle est également appelée hill climbing dans les problèmes de maximisation. Son principe consiste, à partir d'une solution initiale, à choisir à chaque itération un point dans le voisinage de la solution courante qui améliore strictement la fonction objectif. Il existe plusieurs moyens de choisir ce voisin : soit par le choix aléatoire d'un voisin parmi ceux qui améliorent la solution courante (first improvement) ; soit en choisissant le meilleur voisin qui améliore la solution courante (best improvement).
Dans tous les cas, le critère d'arrêt est atteint lorsque plus aucune solution voisine n'améliore la solution courante.
Le principal inconvénient de la DM est qu'elle reste piégée dans le premier optimum local rencontré. Les méthodes de ce type ne présentent aucune forme de diversification.
Une amélioration de cet algorithme consiste à redémarrer plusieurs fois, lorsqu'un optimum local est trouvé, à partir d'une nouvelle solution générée aléatoirement. On parle alors d'algorithme de descente avec relance (multiple start random hill climbing).

Recherche Tabou (Tabu Search) :

La recherche tabou est une méthode itérative générale d'optimisation combinatoire combinée à la recherche locale, est conçue en vue de surmonter les minima locaux de la fonction objectif. Cette méthode a montré une grande efficacité pour la résolution des problèmes d'optimisation difficiles.
En effet, à partir d'une solution initiale, Tabou engendre une succession de configurations qui doit aboutir à la configuration optimale. Le mécanisme de passage d'une configuration x_n vers une solution voisine x_{n+1} est le suivant :
– La construction d'un ensemble voisin de x_n(configurations accessibles en un seul mouvement à partir dex_n) : soit Voisinage(x_n) l'ensemble (ou le sous-ensemble) envisagé.
– Évaluation de la fonction objectif f du problème pour chacune des configurations appartenant au Voisinage(x_n),la configuration qui succède à x_n(est la configuration de Voisinage(x_n) en laquelle f prend sa valeur minimale.
Pour éviter de retourner à une configuration déjà retenue lors d'une itération précédente, à chaque itération, on met à jour une "liste Tabou" contenant les mouvements inverses $(x_{n+1}) \longrightarrow x_n)$.
La procédure peut être stoppée dès que l'on a effectué un nombre donné d'itérations, sans améliorer la meilleure solution atteinte jusqu'ici.

Le recuit simulé (simulated annealing) :

Cela nous ferons un tour rapide sur cette algorithme et puis dans la suite de ce travail nous basée sur la recuit simulé et Nous donnerons aussi plus en détail des définitions et principaux concepts de ce méthode
.

Définition :

L'idée principale du recuit simulé tel qu'il a été proposé par Metropolis en 1953 est de simuler le comportement de la matière dans le processus du recuit très largement utilisé dans la métallurgie.
Le but est d'atteindre un état d'équilibre thermodynamique, cet état d'équilibre (où l'énergie est minimale) représente
- dans la méthode du recuit simulé
– la solution optimale d'un problème
L'énergie du système sera calculé par une fonction coût (ou fonction objectif) spécifique à chaque problème (Kendall).
La méthode va donc essayer de trouver la solution optimale en optimisant une fonction objectif, pour cela, un paramètre fictif de température a été ajouté par Kirkpatrick, Gelatt et Vecchi.
En gros le principe consiste à générer successivement des configurations à partir d'une solution initiale S_0 et d'une température initiale T_0 qui diminuera tout au long du processus jusqu'à atteindre une température finale ou un état d'équilibre (optimum global).

3.2.2 Les méta-heuristiques à population de solutions :

Contrairement aux algorithmes partant d'une solution singulière, les méta-heuristiques à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions. On distingue dans cette catégorie, les algorithmes évolutionnaires, qui sont une famille d'algorithmes issus de la théorie de l'évolution par la sélection naturelle, énoncée par Charles Darwin .
Alors parmi les différents algorithme de set section on va faire un petit étude sur les algorithmes génétiques.

Les algorithmes génétiques :

Les algorithmes génétiques, et plus généralement les algorithmes révolutionnaires ont permis de résoudre un grand nombre de problèmes, notamment en optimisation. Dans le contexte multiobjectif, l'approche révolutionnaire offre, par le biais de la notion de population, des mécanismes pertinents pour approcher la solution optimale (front Pareto).

Principe des algorithmes génétiques :

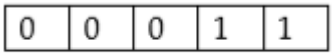
On démarre avec une population initiale. Après avoir déterminé l'efficacité de chacun de ses individus, on opère une reproduction.
On copie les individus proportionnellement à leur efficacité.
Par exemple, l'individu 2 de l'exemple précédent est recopié quatre fois. On va sélectionner les individus suivant leur efficacités, puis les fusionner avec d'autres individus choisis aléatoirement.
Cette opération s'appelle « croisement » et consiste à :
• Sélectionner aléatoirement une position de croisement
• A cette position, on sépare le codage puis on échange les morceaux de droite entre les deux individus.

Exemple de croisement :

Individu 1 :

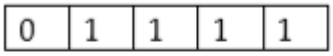
0	1	1	1	0

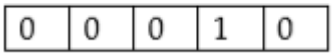
Individu 2 :



Position de croisement = 3

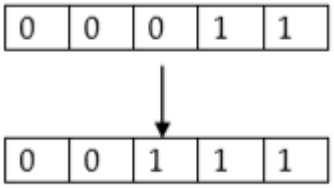
Résultat :





Après cette opération de croisement, la population est diversifiée, on sélectionne, au hasard, un certain nombre d'individus auxquels on applique des mutations. La « mutation » consiste à inverser un des gènes de l'individu.
Exemple de mutation :

Individu :



Algorithme génétique :

Initialisation de la population
Évaluation des fonctions objectif
Calcul de l'efficacité
For i= 1 to MaxIter
Sélection aléatoire,
Sélection proportionnelle à l'efficacité
Croisement
Mutation
Évaluation des fonctions objectif
Calcul de l'efficacité
Fusion des populations
End For

Remarque :

L'avantage de cette méthode est que, dans certains cas, elle peut procurer un jeu de solutions optimales. Un algorithme génétique recherche le ou les extremum d'une fonction définie sur un espace de données.

La figure 3.1 illustre les principales étapes d'un algorithme génétique.

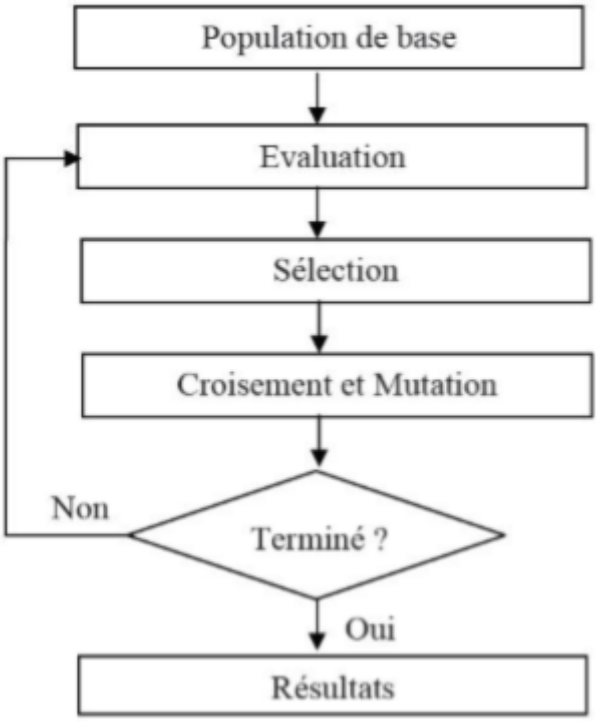


FIGURE 3.1 – Fonctionnement d'un algorithme génétique.

3.3 Le recuit simulé :

3.3.1 Les origines :

La méthode du recuit simulé est une généralisation de la méthode Monte-Carlo ; son but est de trouver une solution optimale pour un problème donné.
La méthode du recuit simulé est basée sur un processus très utilisé en métallurgie pour obtenir un alliage sans défaut, ce processus est appelé « le recuit ».
On commence d'abord par chauffer le métal jusqu'à une certaine température où il devient liquide (les atomes peuvent donc circuler librement).
Après avoir atteint ce stade, on abaisse la température très lentement de sorte à obtenir un solide.
Si cette baisse de température est brusque on obtient alors du verre ; si au contraire cette baisse de température est très lente (laissant aux atomes le temps d'atteindre l'équilibre statistique), nous obtiendrons des structures de plus en plus régulières, jusqu'à atteindre un état d'énergie minimale correspondant à la structure parfaite d'un Cristal, on dit alors que le système est « gelé ».
Au cas où cet abaissement de température ne se ferait pas assez lentement, il pourrait apparaitre des défauts.
Il faudrait alors les corriger en réchauffant de nouveau légèrement la matière de façon à permettre aux atomes de retrouver la liberté de mouvement, leur facilitant ainsi un éventuel réarrangement conduisant à une structure plus stable.

3.3.2 Définition :

L'idée principale du recuit simulé tel qu'il a été proposé par Metropolis en 1953 est de simuler le comportement de la matière dans le processus du recuit très largement utilisé dans la métallurgie.
Le but est d'atteindre un état d'équilibre thermodynamique, cet état d'équilibre (où l'énergie est minimale) représente - dans la méthode du recuit simulé – la solution optimale d'un problème ; L'énergie du système sera calculé par une fonction coût (ou fonction objectif) spécifique à chaque problème .
La méthode va donc essayer de trouver la solution optimale en optimisant une fonction objectif, pour cela, un paramètre fictif de température a été ajouté par Kirkpatrick, Gelatt et Vecchi.
En gros le principe consiste à générer successivement des configurations à partir d'une solution initiale S_0 et d'une température initiale T_0 qui diminuera tout au long du processus jusqu'à atteindre une température finale ou un état d'équilibre (optimum global).

3.3.3 Algorithme :

L'algorithme de Metropolis :

Dans l'algorithme de Metropolis, on part d'une configuration donnée, et on lui fait subir une modification aléatoire.
Si cette modification fait diminuer la fonction objectif (ou énergie du système), elle est directement acceptée ; Sinon, elle n'est acceptée qu'avec une probabilité égale à $exp(DE/T)$ (avec E=énergie, et T=température), cette règle est appelée critère de Metropolis.

L'algorithme de recuit simulé :

Le recuit simulé applique itérativement l'algorithme de Metropolis[2], pour engendrer une séquence de configurations qui tendent vers l'équilibre thermodynamique :

1. Choisir une température de départ $T = T_0$ et une solution initiale $S = S_0$;
2. générer une solution aléatoire dans le voisinage de la solution actuelle ;
3. comparer les deux solutions selon le critère de Metropolis ;
4. répéter (2) et (3) jusqu'à ce que l'équilibre statistique soit atteint ;
5. décroitre la température et répéter jusqu'à ce que le système soit gelé.

Dans un premier temps, T étant généralement choisi très grand, beaucoup de solutions - même celles dégradant la valeur de f - sont acceptées, et l'algorithme équivaut à une visite aléatoire de l'espace des solutions.
Mais à mesure que la température baisse, la plupart des solutions augmentant l'énergie sont refusés, et l'algorithme se ramène à une amélioration itérative classique.
A température intermédiaire, l'algorithme autorise de temps en temps des transformations qui dégradent la fonction objectif. Il laisse ainsi une chance au système de s'extraire d'un minima local
Notons aussi que si la température est égale à 0, seules les solutions optimisant f sont acceptées. L'algorithme se comportera donc comme la méthode de la descente du gradient.

État initial de l'algorithme :

La solution initiale peut être prise au hasard dans l'espace des solutions possibles, elle peut aussi être générée par une heuristique classique, telle que la descente du gradient ou l'algorithme glouton (dans le cas du voyageur du commerce).

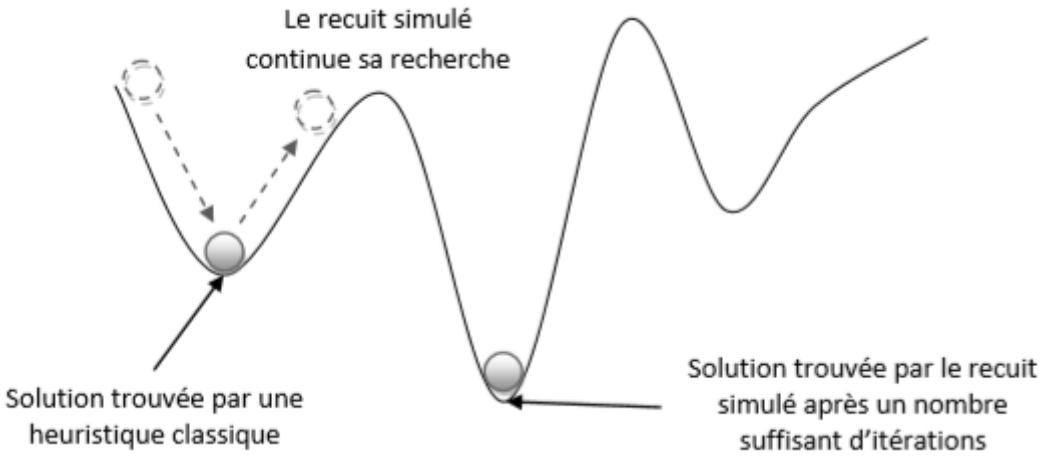


FIGURE 3.2 – Comparaison entre le recuit simulé et une heuristique classique .

La température initiale doit être assez élevée, car c'est elle qui fixe la probabilité d'accepter ou de refuser les solutions défavorables à l'optimisation de la fonction f.

Variation de la température :

Deux approches sont possibles pour décroitre la température :

Décroissance par paliers :

Pour chaque valeur de la température, on itère l'algorithme de Metropolis jusqu'à atteindre un équilibre statistique, puis on diminue la température.

Décroissance continue :

On fait baisser la température d'une façon continue, le plus courant est d'utiliser la loi suivante :
$T_{i+1} = \alpha T_i / \alpha < 1$ en général $\alpha =$ (0.9à 0.99)

Remarque : Le paramètre α est à choisir avec précaution ; En effet, s'il est choisi trop grand, la température baissera très rapidement et l'algorithme pourra être bloqué dans un minima local ;
Si au contraire il est choisi trop petit, la température baissera très lentement et le temps de calcul sera très grand.

Amélioration :

Cet algorithme est parfois amélioré en ajoutant une variable qui mémorise la meilleure valeur rencontrée jusqu'à présent ; sans cela, l'algorithme pourrait converger vers une certaine solution, alors qu'on avait visité auparavant une solution meilleure.

3.4 Domaines d'applications :

Comme pour toute méta-heuristique, la méthode du recuit simulé peut être appliquée dans de nombreux problèmes d'optimisation, les chercheurs l'ont utilisée essentiellement dans :

— La conception des circuits intégrés (problème de placement et de répartition) ;
— Le routage des paquets dans les réseaux ;

— La segmentation d'images ;
— Le problème du voyageur de commerce ;
— Et, le problème du sac à dos.

3.5 Avantages et Inconvénients :

3.5.1 Avantages :

— Facile à implémenter ;
— Donne généralement de bonnes solutions par rapport aux algorithmes de recherche classiques ;
— Peut être utilisé dans la plupart des problèmes d'optimisation ;
— Il converge vers un optimum global (lorsque le nombre d'itérations tend vers l'infini Cela fait de lui une option attrayante pour les problèmes d'optimisation difficiles.

3.5.2 Inconvénient :

Le principal inconvénient du recuit simulé est qu'une fois l'algorithme piégé à basse température dans un minimum local, il lui est impossible de s'en sortir.
Plusieurs solutions ont été proposées pour tenter de résoudre ce problème, par exemple en acceptant une brusque remontée de la température de temps en temps, pour relancer la recherche sur d'autres régions plus éloignées.
cela on peut citer quelques autres inconvénients comme :

1. La difficulté de déterminer la température initiale :
   — Si elle est trop basse, la qualité de recherche sera mauvaise.
   — Si elle est trop haute, le temps de calcul sera élevé.
2. L'impossibilité de savoir si la solution trouvée est optimale ;
3. Dégradation des performances pour les problèmes où il y a peu de minimas locaux (comparé avec les heuristiques classiques comme la descente du gradient par exemple).

Donc on général les principaux inconvénients du recuit simulé résident dans le choix des nombreux paramètres, tels que la température initiale, la loi de décroissance de la température, les critères d'arrêt ou la longueur des paliers de température. Ces paramètres sont souvent choisis de manière empirique.

il existe plusieurs versions du recuit simulé multi-objectif.
Parmi lesquels, on va présenter et concentrant sur la méthodes P.A.S.A (Pareto Archived Simulated Annealing).

3.6 La méthode P.A.S.A(Pareto Archived Simulated Annealing)

3.6.1 Principe :

Cette méthode, développée à EDF dans Engrand et al. 98[16], utilise une fonction d'agrégation des fonctions objectif couplée avec un système d'archivage de solutions non dominées.

3.6.2 Présentation de la méthode :

On suppose que les fonctions objectif du problème d'optimisation multi-objectif sont positives.
Cette hypothèse permet d'utiliser la fonction d'agrégation suivante, pour se ramener à un problème de minimisation mono-objectif :

$G(x) = \sum_{i=1}^{n} ln(f_i(x))$ (5.1)
De cette manière, l'expression suivante :
$\Delta G = G(x') - G(x) = \sum_{i=1}^{n} ln(\frac{f_i(x')}{f_i(x)})$
représente la variation relative moyenne des fonctions objectif entre le point courant (x) et le point à tester (x') :
- si $\Delta G > 0$, la nouvelle solution x' détériore, en moyenne relative, l'ensemble des fonctions objectif ;
– si $\Delta G < 0$,la nouvelle solution x' améliore, en moyenne relative, l'ensemble des fonctions objectif.
Ensuite, comme pour le recuit simulé original, on définit une probabilité d'acceptation p :
$p = exp(\frac{-\Delta G}{T})$
Où T désigne la température de l'algorithme de recuit.
Cette température a la même signification que dans la méthode du recuit simulé monobjectif.
Pour l'archivage des solutions non dominées, on utilise une "archive" de taille variable (entre 0 et N_{max} solutions dans l'archive).
La gestion de l'archivage se conforme aux règles suivantes :
– si la solution x' est dominée par au moins une solution de l'archive, alors la solution x' n'est pas archivée ;
– si la solution x' domine une solution y de l'archive, alors la solution x' remplace la solution y dans l'archive ;
– si la solution x' n'est pas dominée par l'archive, alors on place cette solution dans l'archive et on retire de l'archive les solutions qui sont dominées par la solution x'.
Certaines autres règles peuvent être ajoutées pour améliorer la gestion des individus de l'archive.
Par exemple, on peut ajouter un critère restrictif pour l'ajout d'une solution dans l'archive :
– la solution x' doit dominer toutes les solutions de l'archive et être à une distance minimale de solutions adjacentes de l'archive.
Pour que la recherche se fasse sur toute la surface de compromis, il est nécessaire de relancer régulièrement la recherche à partir d'un point choisi au hasard, au sein de l'archive.
Cette étape du choix de l'individu est appelée return to base. L'algorithme de la méthode P.A.S.A est celui décrit dans l'algorithme président :

3.6.3 Discussion :

Dans cet algorithme, la population initiale présente dans l'archive peut être issue d'une première optimisation multi-objectif.
La qualité des solutions initialement présentes dans l'archive influencera la qualité du résultat obtenu à la fin de l'optimisation courante.

3.6.4 Algorithme La méthode P.A.S.A :

1. Choix de P_0, une population d'archive initiale, éventuellement réduite à un seul élément x_0, et choix de T_q initiale.
2. Soit x'_n un voisin de x_n
3. Calcul de $G(x'_n)$ et de $\Delta G = G(x'_n) - G(x_n)$
4. Si$\Delta G < 0$ alors $x_{n+1} = x'_n$
5. Si $\Delta G > 0$ alors soit $p = exp(\frac{-\Delta G}{T_q})$
- $x_{n+1} = x'_n$ une probabilité p
- $x_{n+1} = x_n$ avec une probabilité 1- p
6. Si on sort de la boucle interne, alors $T_{q+1} = \alpha.Tq$ avec $\alpha < 1$
7. Principe de non-dominance pour l'archivage de x_{n+1}
– si x_{n+1} domine une des solutions archivées y_p, x_{n+1} remplace y_p dans la population archivée ;
– si x_{n+1} est dominée par au moins une des solutions archivées, alors on ne l'archive pas ;
– six_{n+1} n'est pas dominée par la population archivée alors on l'ajoute à la population archivée, et on retire de celle-ci les solutions dominées par x_{n+1}.

8. Retourne to base : lorsque c'est nécessaire, $x_{n+1} = y_i$ (i choisi aléatoirement dans la population archivée)
9. Si (non convergence) alors retour en 2.

Remarque :

Un autre avantage de cette méthode est la facilité avec laquelle on peut inclure des règles heuristiques pour la gestion de l'archive.
Cette "flexibilité" peut permettre d'intégrer un "savoir ingénieur", qui est susceptible d'améliorer la qualité de la population obtenue à la fin de l'optimisation, mais surtout la vitesse de convergence vers cette population d'individus archivés.
Un défaut de cette méthode que l'on peut souligner est que la forme d'agrégation de la fonction objectif (l'équation 5.1) utilisée ne supporte que les valeurs positives de fonctions objectif.

3.7 L'équilibre souhaité entre l'intensification et la diversification

Deux objectifs doivent être pris en compte dans la résolution d'un problème d'optimisation multi-objectif : **l'intensification** et **la diversification**.
Intensification (exploitation) : converger vers la true Pareto.
Diversification (exploration) : trouver les solutions diversifiées le long de la true Pareto : Dans la figure 3.3 :

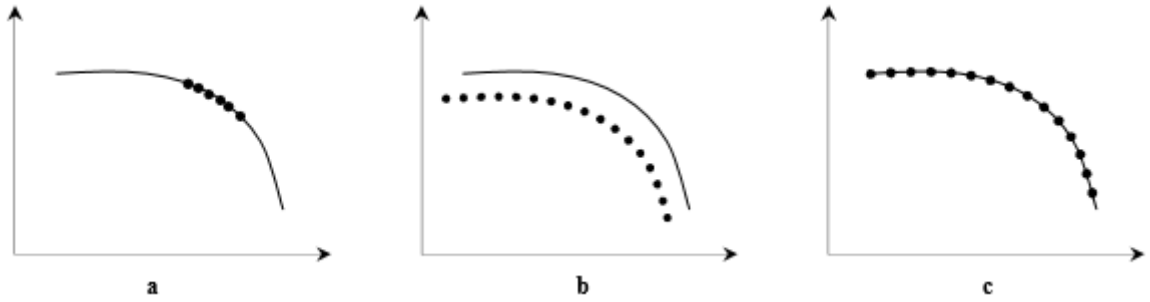


FIGURE 3.3 – Illustration de la convergence et la diversité en multi-objectif

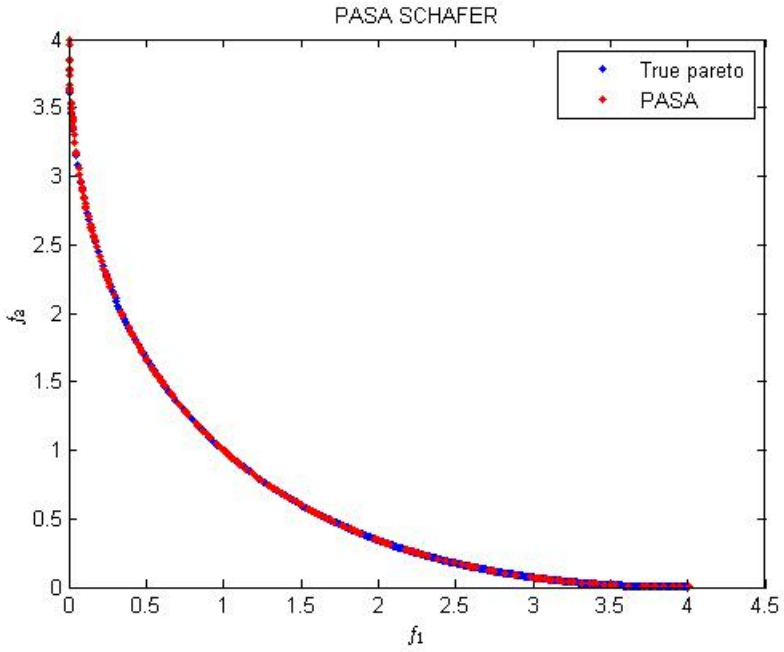
(a) présente une solution de bonne qualité en terme de convergence mais mauvaise pour la diversité. (b) une solution de bonne diversité mais de mauvaise qualité pour la convergence. (c) Une disposition idéale des solutions (l'équilibre souhaité).

Quelques test sur la méthode P.A.S.A (Pareto Archived Simulated Annealing) :

Nous décrivons d'abord les problèmes de test utilisés pour comparer différents multi-objectif . Les problèmes de test sont choisis parmi un certain nombre d'études antérieures significatives dans ce domaine. Veldhuizen a cité un certain nombre de problèmes de test qui ont été utilisés dans le passé. Parmi eux, nous choisissons trois problèmes :
L'étude de Schaffer (SCH) , l'étude de Fonseca et Fleming (FON) , l'étude de Poloni (POL) .
En 1999, le premier auteur a suggéré une manière systématique de développer des problèmes de test pour l'optimisation multi-objective . Zitzler et al. ont suivi ces lignes directrices et suggéré six problèmes de test. Nous choisissons quatre de ces six problèmes ici et les appelons ZDT1, ZDT2, ZDT3 et ZDT4. Tous les problèmes ont deux fonctions objectives. Aucun de ces problèmes n'a de contrainte. Nous décrivons ces problèmes dans les exemple suivants. les exemples montre également le nombre de variables, leurs limites, les solutions Pareto-optimales et la nature du front Pareto-optimal pour chaque problème.

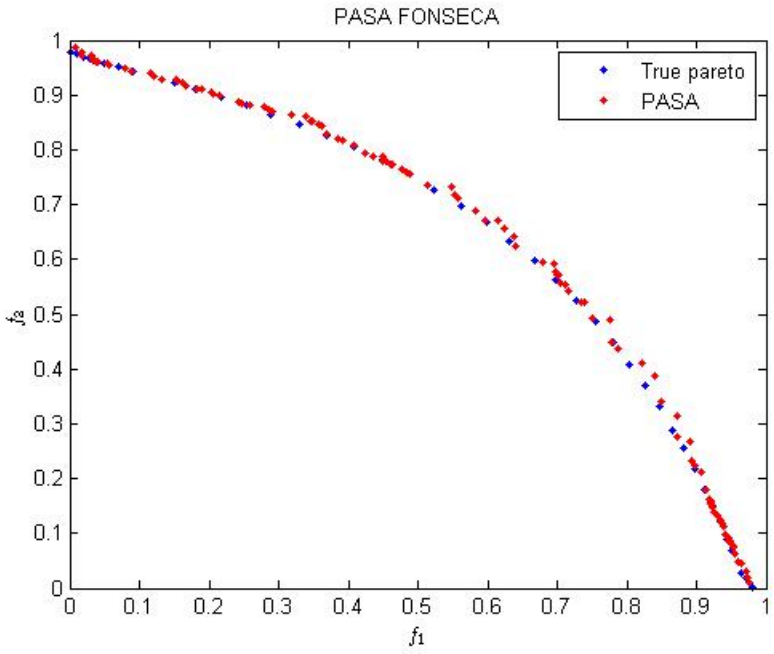
Dans tout ces test tout les fonctions objectif sont de type de minimisation
Exemple 1 : Le problème de Schaffer

$$\begin{cases} \min f_1(x) = x^2 \\ \min f_2(x) = (x-2)^2 \\ x \in [-10^3, 10^3] \end{cases} \tag{3.1}$$



Exemple 2 : Le problème de Fonseca and Fleming

$$\begin{cases} \min f_1(x) = 1 - exp(-\sum_{i=1}^{3}(x_i - \frac{1}{\sqrt{3}})^2) \\ \min f_2(x) = 1 - exp(-\sum_{i=1}^{3}(x_i + \frac{1}{\sqrt{3}})^2) \\ x_i \in [-4,4] \qquad\qquad i = \{1,...,3\} \end{cases} \tag{3.2}$$

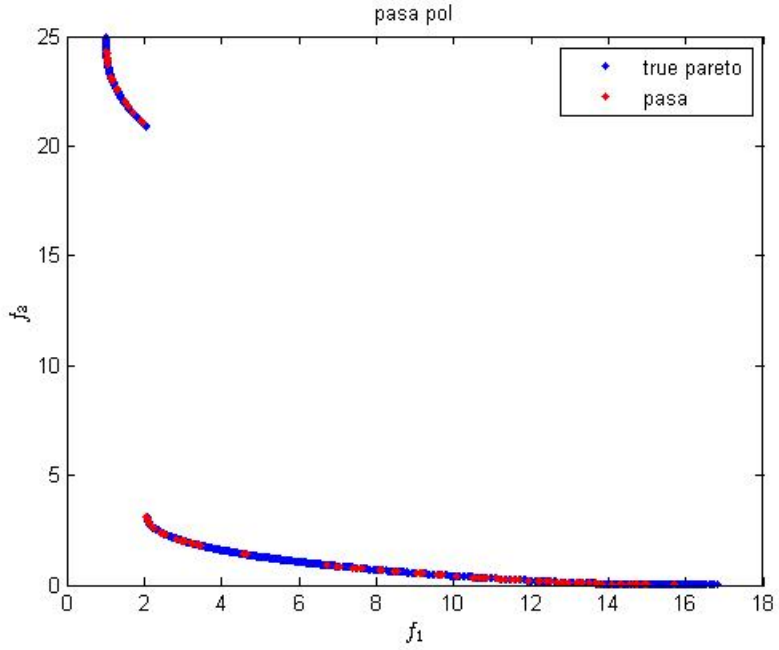


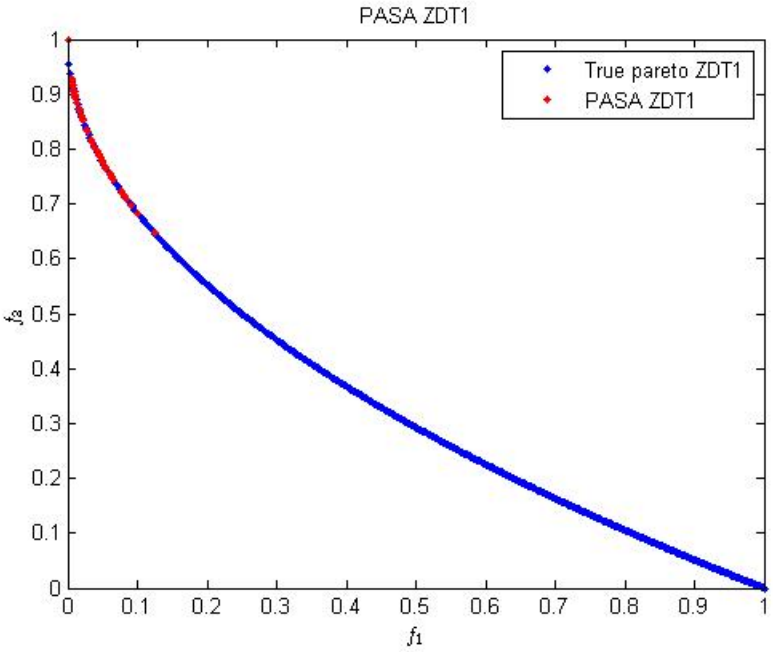
Exemple 3 : Le problème de poloni

$$\begin{cases} \min f_1(x) = [1 - (A_1 - B_1)^2 + (A_2 - B_2)^2] \\ \min f_2(x) = [(x_1 + 3)^2 + (x_2 + 1)^2] \\ A_1 = 0.5sin(1) - 2cos(1) + sin(2) - 1.5cos(2) \\ A_2 = 1.5sin(1) - cos(1) + 2sin(2) - 0.5cos(2) \\ B_1 = 0.5sin(x_1) - 2cos(x_1) + sin(x_2) - 1.5cos(x_2) \\ B_2 = 1.5sin(x_1) - cos(x_1) + 2sin(x_2) - 0.5cos(x_2) \\ x_i \in [-\Pi, \Pi] \qquad\qquad i = \{1,2\} \end{cases} \tag{3.3}$$

Exemple 4 : Le problème de ZDT1

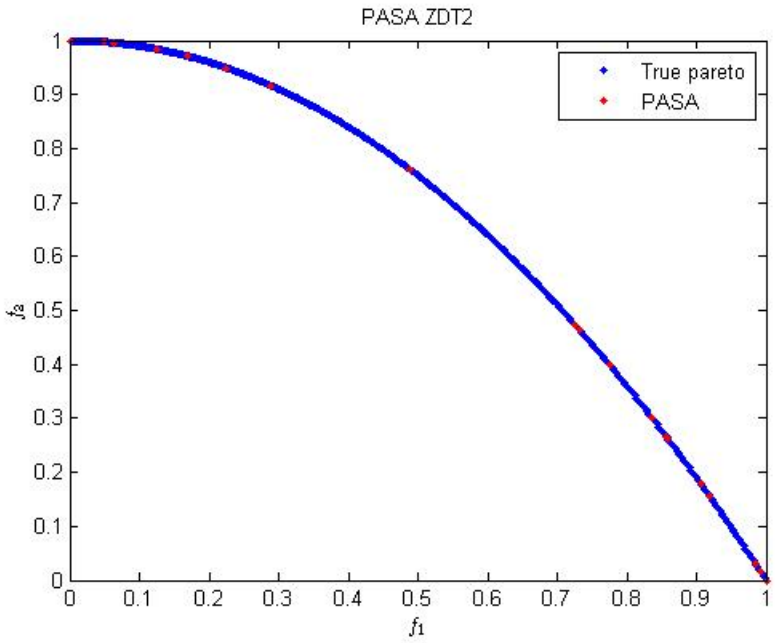
$$\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x)[1 - \sqrt{x_1 / g(x)}] \\ g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1) \\ x_i \in [0,1] \qquad\qquad i = \{1,...,30\} \end{cases} \tag{3.4}$$


pasa pol
true pareto
pasa
f_2
f_1
0
5
10
15
20
25
0
2
4
6
8
10
12
14
16
18


PASA ZDT1
True pareto ZDT1
PASA ZDT1
f_2
f_1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Exemple 5 : Le problème de ZDT2

$$\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x)[1-(x_1/g(x))^2] \\ g(x) = 1+9(\sum_{i=2}^{n} x_i)/(n-1) \\ x_i \in [0,1] \qquad\qquad i = \{1,...,30\} \end{cases} \tag{3.5}$$

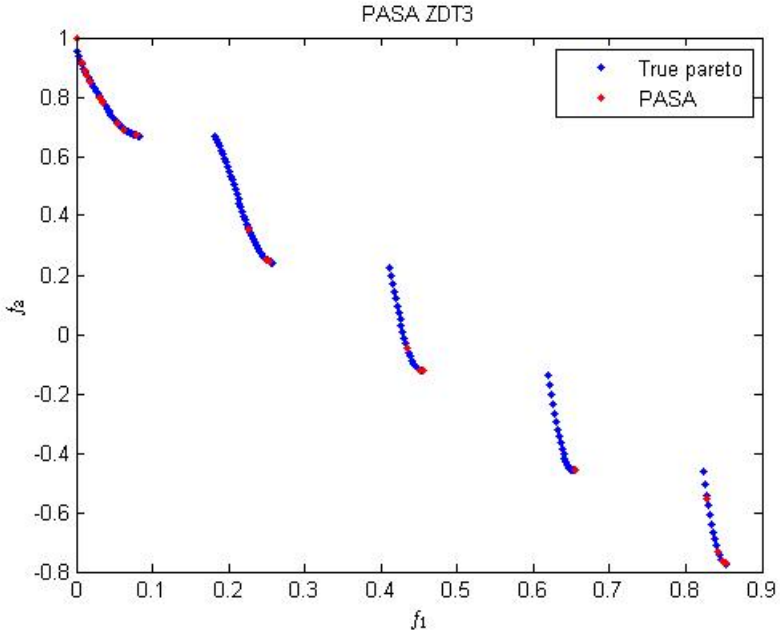


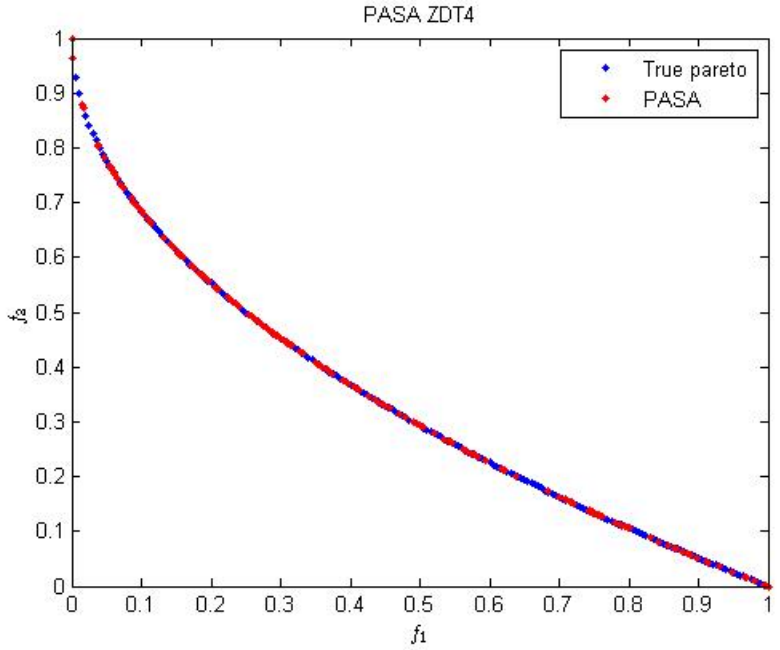
Exemple 6 : Le problème de ZDT3

$$\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x)[1-\sqrt{x_1/g(x)} - \frac{x_1}{g(x)} sin(10\pi x_1)] \\ g(x) = 1+9(\sum_{i=2}^{n} x_i)/(n-1) \\ x_i \in [0,1] \qquad\qquad i = \{1,...,30\} \end{cases} \tag{3.6}$$

Exemple 7 : Le problème de ZDT4

$$\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x)[1-\sqrt{x_1/g(x)}] \\ g(x) = 1+10(n-1)+\sum_{i=2}^{n}[x_i^2 - 10\cos(4\pi x_i)] \\ x_1 \in [0,1] \\ x_i \in [-5,5] \qquad\qquad i = \{2,...,10\} \end{cases} \tag{3.7}$$


PASA ZDT3
True pareto
PASA
f_2
f_1
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9


PASA ZDT4
True pareto
PASA
f_2
f_1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

	cdmean	cdstd	cdmeann	cdstdd
Fonsica and Fleming	0.0362	0.0240	0.0969	0.0370
Schafer	0.0091	0.0097	0.0098	0.0098
Poloni	0.0030	0.0296	0.0036	0.0303
ZDT1	0.0031	0.0057	0.0079	0.0040
ZDT2	0.0314	0.0239	0.0296	0.0426
ZDT3	0.0147	0.0228	0.0080	0.0023
ZDT4	0.0125	0.0181	0.0198	0.0090

TABLE 3.1 – Les Paramètres de diversité

	Diststd	DistMean
Fonsica and Fleming	8.2218e-04	7.7688e-05
Schafer	0.0015	7.3812e-05
Poloni	7.4856e-05	2.0549e-06
ZDT1	3.0440e-05	6.9854e-07
ZDT2	2.249e-05	5.05977e-07
ZDT3	3.5016e-04	2.1928e-05
ZDT4	2.9821e-05	1.6723e-06

TABLE 3.2 – Les Paramètres de comparaison de la distance de convergence

Commentaire de TABLE 3.1 :
cdmean, cdstd représente les paramètres de la diversité de la long de la courbe de P.A.S.A.
cdmeann et cdstdd représente les paramètres de la diversité de la long de la courbe de true Pareto.
On remarque que cdmean de P.A.S.A converge vers cdmeann de la true Pareto et aussi pour cdstd de P.A.S.A et cdstdd de la true Pareto.
Alors pour une meilleur diversité on prend les solutions ayant le plus grand paramètre "diversité". Donc, plus le paramètre est grand, plus on a une bonne diversité

Commentaire de TABLE 3.2 :
Diststd et Distmean représente la convergence vers la true Pareto par la méthode P.A.S.A.
On remarque que Diststd et Distmean entre P.A.S.A et la true Pareto est tend vers zéro alors ces exemples de bonne qualité en terme de convergence .

On a utilise la distance de Crowding pour calculer les paramètres de la diversité.

3.8 Procédure de Crowding :

Une procédure de crowding, basée sur un calcul de distance (distance de crowding) qui ne nécessite aucun paramétrage et qui est également d'une complexité algorithmique moindre que celle de sharing. La distance de crowding d'une solution particulière i se calcule en fonction du périmètre de l'hypercube ayant comme sommets les points les plus proches de i sur chaque objectif. Sur la figure 3.4, est représenté l'hypercube en deux dimensions associé au point i. Le calcul de la distance de crowding nécessite, avant tout, le tri des solutions selon chaque objectif, dans un ordre ascendant. Ensuite, pour chaque objectif, les

individus possédant des valeurs limites se voient associés une distance infinie. Pour les autres solutions intermédiaires, on calcule une distance de crowding égale à la différence normalisée des valeurs des fonctions objectifs de deux solutions adjacentes. Ce calcul est réalisé pour chaque objectif. La distance de crowding d'une solution est obtenue en sommant les distances correspondantes à chaque objectif

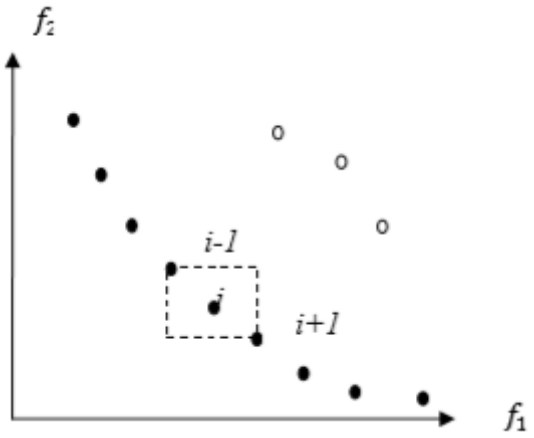


FIGURE 3.4 – Distance de Crowding (les points noirs sont des Solutions appartenant au même front)

Conclusion

Malgré le nombre important de méthodes d'optimisation multi-objectif disponibles, il n'existe pas encore des méthodes puissantes pouvant surpasser les difficultés des problèmes d'optimisation multi-objectif. Ce domaine reste toujours ouvert à de nouvelles idées et philosophies de résolutions. Nous pouvons conclure que les méta-heuristiques présentent une classe de méthodes approchées adaptables à une grande variété de problèmes d'optimisation multi-objectif et mènent à des résultats pertinents. Mais il existe assez peu de contributions permettant de comprendre la raison de cette efficacité, et aucune preuve ne peut montrer qu'une méta-heuristique sera plus efficace qu'une autre sur un problème donné. Certaines méta-heuristiques présentent l'avantage d'être simples à mettre en œuvre, comme le cas de la méthode utilisant recuit simulé, qui s'appelle P.A.S.A(pareto archived simulated annealing).
On a commencer par donnée des notions de base sur l'optimisation mono-objectif au premier lieu
En suite on a présenté les principaux de la méthode exacte/méta-euristique traitons les problémes d'optimisation multi-objectif
Enfin nous avons intéressées a une méthode d'optimisation basée sur le principe du recuit simulé dit P.A.S.A. Cette méthode à été testée sur une ensemble des problèmes multi-objectif très connus. Le résultat confirme l'habitude de cette méthode pour la résolution des problèmes convexe, non convexe, continus, discontinu et aussi bien des problèmes combinatoire comme sac à dos.

Bibliographie

[1] A Fast and Elitist Multiobjective Genetic Algorithm : NSGA-II [Article]/Kalyanmoy Deb, Associate Member, IEEE, Amrit Pratap, Sameer Agarwal, and T. Meyarivan.

[2] BENDAHMANE Amine – Master2 RFIA 25 octobre 2011, Le recuit simulé.

[3] Benjamin Monmege.Cours optimisation.29 février 2012.

[4] Bernard Fortz.Recherche opérationnelle et applications.2012-2013.

[5] Denis Robilliard. Recherche opérationnelle et Optimisation Master 1 — Informatique. Lisic — Univ Littoral-Côte d'opale 2010.

[6] D. A. Van Veldhuizen, G. B. Lamont, On Measuring Multiobjective Evolutionary Algorithms Performances, 2000 Congress on Evolutionary Computation, volume 1, pages 205-211, Piscataway, New Jersey, USA, juillet 2000.

[7] François Rouvière. Petit guide de calcul différentiel à l'usage de la licence et de l'agrégation. Cassini, 2003. Deuxième édition revue et augmentée

[8] Guy Cohen. Convexité et Optimisation. Engineering school. 1999-2001 Ecole Nationale des Ponts et Chaussées, 2000, pp.139. <cel-00356686>

[9] Hafid Zidani. Représentation de solution en optimisation continue, multi-objectif et applications. Autre. INSA de Rouen, 2013. Français. <NNT : 2013ISAM0027>. <tel-00939980>

[10] Ilhem Boussaid. Perfectionnement de méta-heuristiques pour l'optimisation continue. Autre. Université Paris-Est, 2013. Français. NNT : 2013PEST1075 .

[11] I.Sau et C. Molle. Mathématiques pour L'optimisation.LP SIL

[12] Jérôme Picard[1], Optimisation et optimisation multidisciplinaire, [1]je.picard@gmail.com

[13] SAADI LEILA[Rapport] : Mémoire de Magister/Optimisation MultiObjectifs par Programmation Génétique, 08/07 /2007

[14] Lyes Belhoul. Résolution de problèmes d'optimisation combinatoire mono et multi-objectifs par énumération ordonnée. Autre [cs.OH]. Université Paris Dauphine - Paris IX, 2014. Français. <NNT : 2014PA090060>. <tel-01127114>

[15] MAHDI SAMIR[Rapport] : Mémoire de Magister/ Optimisation Multiobjectif Par Un Nouveau Schéma De Coopération Méta/Exacte.

[16] Nicolas Jozefowiez. Optimisation combinatoire multi-objectif : des méthodes aux problèmes, de la Terre à (presque) la Lune. Automatique / Robotique. Institut National Polytechnique de Toulouse (INP Toulouse), 2013. <tel-01104895>

[17] P. Engrand, X. Mouney, Une méthode originale d'optimisation multiobjectif, Note interne EDFDER numéro $HT-14/97/035/A, mars1998$.

[18] Sidi Mohamed Douiri, Souad Elbernoussi, Halima Lakhbab. Cours des Méthodes de Résolution Exactes Heuristiques et Méta-heuristiques.MASTER CODES, CRYPTOGRAPHIE ET SÉCURITÉ DE L'INFORMATION

[19] Vincent Gardeux. Conception d'heuristiques d'optimisation pour les problèmes de grande dimension : application à l'analyse de données de puces à ADN. Autre [cs.OH]. Université Paris-Est, 2011. Français. <NNT : 2011PEST1022>. <tel-00676449>

[20] Xavier ANTOINE[1] [2] [3], Pierre DREYFUSS[4] et Yannick PRIVAT2[3].Introduction à l'optimisation : aspects théoriques,numériques et algorithmes.ENSMN-ENSEM 2A (2006-2007)

[21] Yann Collette[1], Patrick Siarry[2], Optimisation Multiobjectif, ÉDITIONS EYROLLES 61, Bld Saint-Germain 75240 Paris Cedex 05 www.editions-eyrolles.com.