Algorithme à gradients multiples pour l'optimisation multiobjectif en simulation de haute fidélité : application à l'aérodynamique compressible

Adrien Zerbinati

UNIVERSITE DE NICE-SOPHIA ANTIPOLIS - UFR Sciences
Ecole Doctorale de Sciences Fondamentales Appliquées (EDSFA)

THESE

pour obtenir le titre de
Docteur en Sciences
de l'Université de Nice-Sophia Antipolis
Discipline : **Mathématiques Appliquées**

présentée et soutenue par
Adrien ZERBINATI

Algorithme à gradients multiples pour l'optimisation multiobjectif en simulation de haute fidélité

Application à l'aérodynamique compressible

Thèse dirigée par **Jean-Antoine DESIDERI**
et codirigée par **Régis DUVIGNEAU**
soutenue le 24 mai 2013

Jury :

M. Patrick Siarry	Rapporteur
M. Mohamed Masmoudi	Rapporteur
M. Olivier Pironneau	Examinateur
M. Luc Bordier	Examinateur
M. Jean-Antoine Désidéri	Examinateur
M. Régis Duvigneau	Examinateur

Remerciements

Je remercie tout d'abord l'Agence Nationale de la Recherche (ANR) ainsi que tous les collaborateurs du projet Optimisation MultiDisciplinaire II (OMD2) sans qui cette thèse n'aurait pas été possible. Je remercie aussi l'Inria et, plus particulièrement, le centre Inria Méditerrannée de Sophia Antipolis pour l'ensemble des moyens mis à ma disposition durant ces trois années.

Je tiens à remercier M. Patrick Siarry et M. Mohamed Masmoudi de m'avoir fait l'honneur d'être rapporteurs et M. Olivier Pironneau et M. Luc Bordier d'avoir accepté d'être examinateurs.

Je remercie tout particulièrement mes deux encadrants, Jean-Antoine Désidéri et Régis Duvigeau. Ils m'ont suivi avec patience tout au long de cette thèse et ont su m'épauler et m'orienter lorsque j'en ai eu besoin. Leurs connaissances scientifiques et techniques associées à un sens pédagogique développé font de ce duo l'encadrement idéal. C'est avec une grande humilité que je parle d'eux. Les enseignements que j'ai retenus durant ces années de collaboration me seront utiles tout au long de ma carrière professionnelle. Merci Jean-Antoine et Régis.

Je tiens aussi à remercier l'ensemble de l'équipe projet OPALE pour ses conseils avisés et pour l'ambiance agréable que j'ai appréciée au quotidien.

Je remercie aussi, tout particulièrement Carole qui partage ma vie. Elle m'a supporté et soutenu durant cette thèse qui fût bien plus douce en sa présence. Je remercie aussi ma famille pour son soutient durant toutes ces années et tout particulièrement mes parents sans qui rien n'aurait été possible.

Résumé

En optimisation multiobjectif, les connaissances du front et de l'ensemble de Pareto sont primordiales pour résoudre un problème. Un grand nombre de stratégies évolutionnaires sont proposées dans la littérature classique. Ces dernières ont prouvé leur efficacité pour identifier les fronts de Pareto. Pour atteindre un tel résultat, ces algorithmes nécessitent un grand nombre d'évaluations. En ingénierie, les simulations numériques sont généralement réalisées par des modèles de haute fidélité. Aussi, chaque évaluation demande un temps de calcul élevé.

A l'instar des algorithmes mono-objectifs, les gradients des critères, ainsi que les dérivées successives, apportent des informations utiles sur la décroissance des fonctions. De plus, de nombreuses méthodes numériques permettent d'obtenir ces valeurs pour un coût modéré. En s'appuyant sur les résulats théoriques obtenus par Inria [Dés12], nous proposons un algorithme basé sur l'utilisation des gradients de descente. Ces travaux résument la caractérisation théorique de cette méthode et la validation sur des cas test analytiques.

Dans le cas où les gradients ne sont pas accessibles, nous proposons une stratégie basée sur la construction de métamodèles de type Krigeage. Ainsi, au cours de l'optimisation, les critères sont évalués sur une surface de réponse et non par simulation. Le temps de calcul est considérablement réduit, au détriment de la précision. La méthode est alors couplée à une stratégie de progression du métamodèle. Nous proposons d'utiliser cette méthode d'optimisation pour résoudre un problème classique en aéronautique de minimisation de trainée et de maximisation de portance. Cette étude est réalisée sur un profil d'aile simplifié d'avion d'affaire de type NACA0012. Nous proposons, aussi, une application à un cas test de nature pré-industrielle. Nous considérons une conduite d'air climatisé utilisée par le constructeur automobile Renault. Il s'agit d'optimiser la forme du coude de cette conduite afin de réduire la perte de charge et d'obtenir une répartition du champs de vitesse homogène, en sortie.

Les résultats obtenus montrent la performance de cette méthode. L'amélioration des critères pour un nombre d'évaluations restreint est significative.

Abstract

In multiobjective optimization, the knowledge of the Pareto set provides valuable information on the reachable optimal performance. A number of evolutionary strategies have been proposed in the literature and proved to be successful to identify Pareto set. Howerver, these derivative free algorithms are very demanding in computational time. Today, in many areas of computational sciences, codes are developed that include the calculation of the gradient, cautiously validated and calibrated. Thus, an alternate method applicable when the gradients are known is introduced presently. Using a clever combination of the gradients, a descent direction common to all criteria is identified. As a natural outcome, the Multiple Gradient Descent Algorithm (MGDA) is defined as a generalization of the steepest-descent method and compared with PAES by numerical experiments. Using MGDA on a multiobjective optimization problem requires the evaluation of a large number of points with regards to criteria, and their gradients. In the particular case of CFD problems, each point evaluation is very costly. Thus here we also propose to construct metamodels and to calculate approximate gradients by local finite difference.

Table des matières

Glossaire

AGs	Algorithmes Génétiques
ANR	Agence Nationale de la Recherche
AoA	Angle Of Attack, angle d'attaque
CMAES	Covariance Matrix Approximation Evolution Strategy
DMOEA	Dynamic Multi Objective Evolutionary Algorithm
ES	Evolution Strategy
KKT	Karush Kuhn Tucker
MGDA	Multiple Gradient Descent Algorithm
MOCMAES	Multi Objective Covariance Matrix Approximation Evolution Strategy
MOGA	Multi Objective Genetic Algorithm
MUSCL	Monotone Upstream-centered Schemes for Conservation Laws
NPGA	Niched Pareto Genetic Algorithm
NSGA	Non-dominated Sorting Genetic Algorithm
OMD	Optimisation Multi Disciplinaire
PAES	Pareto Archived Evolution Strategy
PESA	Pareto Envelope-based Selection Algorithm
PSO	Particle Swarm Optimization
RDGA	Rank-Density based Genetic Algorithm
RWGA	Random Weighted Genetic Algorithm
SPEA	Strength Pareto Evolutionary Algorithm
VEGA	Vector Evaluated Differential Evolution
WBGA	Weighted Based Genetic Algorithm

Chapitre 1

Introduction

Dans les bureaux d'étude et les laboratoires de recherche, les praticiens et les chercheurs construisent des modèles mathématiques plus ou moins fidèles pour représenter les situations physiques qu'ils souhaitent calculer, contrôler ou optimiser. Dans les situation les plus favorables où, la validation est possible, le niveau de précision du moèlle utilisé peut être qualibré suivant le but recherché et le caractère plus ou moins critique que peut avoir la simulation, généralement par ordinateur, du phénomène physique principal. Par exemple, en aéronautique, réduire ne serait-ce que d'un faible pourcentage le coefficient de traînée d'une voilure d'avion est un objectif technique important et difficile. Cependant, l'évaluation précise de ce coefficient, à ce niveau de précision, exige la résolution sur ordinateur des modèles de mécanique des fluides les plus précis (équation de Navier-Stokes moyennées) et le post-traitement des résultats bruts de la simulation par un expert. Cette chaîne de procédures complexes et onéreuses en temps-calcul et en heures d'ingńieur, mais elle seule permet d'atteindre un résultat précis de "haute fidélité". Cependant, à différents stades de la conception à la fabrication, d'autres modèles, dits de "basse fidélité" peuvent s'avérer extrêmement utiles, notamment pour estimer des paramètres plus globaux. Par exemple, en aérodynamique, la composante principale de la traînée, dite traînée d'onde, peut se calculer par les équations d'Euler, beaucoup plus simples, ou même, en l'absence de choc fort, par l'équation du potentiel complet. On voit donc qu'à une physique donnée, on peut souvent associer non pas un modèle mathématique, mais toute une hiérarchie de modèles de plus ou moins grande fidélité. Bien évidemment, les modèles de plus grande fidélité sont plus onéreux et complexes à traiter sur ordinateur.

Une thématique en optimisation multi-disciplinaire est donc naturellement liée à la manière plus ou moins savante de coupler plusieurs disciplines lorsque chacune est

associée à une hiérarchie de modèles et que le couplage des modèles de plus haute-fidélité entre eux est plus généralement hors de portée.

L'évolution de la puissance de calcul, au cours des dernières années, donne une place importante à la simulation numérique. Dans un contexte pré-industriel, la conception optimale intègre la simulation de "haute fidélité" dans sa chaîne de calculs, pour évaluer les critères. En traduisant un problème d'optimisation sous la forme de la minimisation d'une fonction f, on dispose d'un grand nombre de méthodes qui ont été développées. Pour ce type de problèmes, les fonctions $f^{(k)}$ (dérivées à l'ordre k de f) apportent des informations précieuses. Dans la littérature, les méthodes les plus anciennes basées sur les simples valeurs de la fonction (simplexe [DT97]) ont alors été améliorées par l'utilisation des dérivées successives (méthodes du gradient et du gradient conjugué [PGW81, All05, Sny05]). Malheureusement, il existe de nombreux cas où l'évaluation de la fonction f pose des difficultés et ses dérivées successives, si elles existent, ne sont pas calculables aisément. C'est le cas notamment lorsque ces fonctions sont en réalité des fonctionnelles associées à un phénomène physique complexe.

Un problème d'optimisation ne se résume que très rarement à l'étude d'une seule fonction. De manière générale, il faut considérer un ensemble de plusieurs fonctions dont les points optimaux de l'espace des paramètres sont différents. On parle alors d'optimisation multiobjectif. La comparaison entre deux points de l'espace de recherche, par rapport aux critères considérés, ne peut plus se faire aussi simplement qu'avec un seul objectif. Le comportement des algorithmes doit alors être différent. De plus, particulièrement en ingénierie, les critères considérés sont couramment de natures différentes. Par exemple, l'étude d'un problème d'aéro-masse en aéronautique, où l'on cherche à maximiser la portance et à minimiser la masse de la structure, simultanément. On a alors un critère de nature aérodynamique (la portance) et un problème de nature structurelle (la masse). Le traitement numérique d'un problème d'optimisation multiobjectif consiste généralement à trouver un ensemble de points qui satisfont, au mieux, les critères considérés.

Les méthodes proposées pour obtenir cet ensemble sont multiples. L'idée naturelle pour traiter les problèmes d'optimisation multiobjectif est de se ramener à un problème mono-objectif. Pour ce faire, on interprète chaque critère comme un objectif à minimiser et on somme l'ensemble des objectifs. La résolution du problème consiste alors à minimiser cette somme. Celle-ci est souvent réalisée avec une pondération des objectifs. La mise en oeuvre de cette méthode est simple et bénéficie de l'ensemble des résultats

obtenus en optimisation mono-objectif. Cependant, le problème initial est considérablement modifié. La solution ainsi trouvée est très dépendante des poids choisis lors de l'écriture de la somme. De plus, le rapport au sens physique est perdu. En effet, des objectifs de nature très différente sont sommés (structure, aérodynamique, magnétisme, etc.).

Les *stratégies évolutionnaires* (ES) constituent une famille importante des optimiseurs mono et multiobjectifs. Les plus populaires d'entre elles font régulièrement l'objet d'études comparatives [Deb01, EZ99]. Des problèmes-types ont été développés pour les tester et comparer leurs performances [DTLZ05, DTLZ02]. Si l'on considère un échantillon initial suffisamment diversifié, ces méthodes produisent un ensemble discret de solutions. Par diverses techniques, propres à chaque optimiseur, une certaine diversité est maintenue dans la population. Ainsi, ces méthodes sont adaptées pour résoudre des problèmes multimodaux. Cependant, les stratégies évolutionnaires nécessitent un grand nombre d'évaluations pour obtenir un résultat aussi complet.

Il est aussi possible de traiter un problème d'optimisation multiobjectif par une formulation de jeu dynamique. Elle fut introduite par Von Neumann dans les années 30 [Neu28] et exploitée dans les années 50 par Nash [Nas50]. On considère alors plusieurs joueurs associés aux objectifs qui sont mis en compétition. Chaque joueur applique sa stratégie pour améliorer son critère. Le jeu s'arrête lorsqu'aucun joueur ne peut améliorer son critère sans détériorer celui d'un autre. Un équilibre de Nash est alors atteint. La recherche de l'équilibre de Nash se fait par échange d'information, chaque joueur définissant sa stratégie pour optimiser le critère qui lui est propre, en fonction des variables qui lui sont attachées et sous la contrainte de la stratégie des autres. Selon la stratégie d'attribution des variables de l'espace de recherche aux joueurs, l'équilibre obtenu est différent. Ainsi, certaines stratégies n'admettent qu'un seul et unique équilibre et d'autres plusieurs ou aucun [OR94]. La solution du problème d'optimisation proposée par cette méthode est dépendante de la stratégie choisie. Une étude préalable est alors nécessaire pour déterminer le partage de territoire [Dés07, VV04]. En plus de cette difficulté, les méthodes d'optimisation basées sur la théorie des jeux sont généralement plus économiques que les stratégies évolutionnaires, mais néanmoins encore assez coûteuses. En effet, à chaque itération, on résout au moins partiellement un problème d'optimisation mono-objectif pour chaque joueur. La mise en compétition des critères peut aussi conduire vers une solution qui oscille en faveur d'un joueur puis d'un autre, sans jamais trouver d'équilibre.

Cette thèse vise un objectif double, méthodologique et applicatif. Au plan méthodologique, on souhaite d'une part mettre en oeuvre un nouvel algorithme multiobjectif qui limite le coût lié aux approches de type évolutionnaire et valider le procédé sur des cas tests mathématiques. D'autre part, nous souhaitons étendre la méthode en obtenant les gradients par construction de méta-modèles. Au plan applicatif, on souhaite conduire l'expérimentation numérique d'un cas test pré-industriel d'optimisation de la forme d'une conduite de climatisation d'automobile, siège d'un écoulement compressible complexe régi par les équations de Navier-Stokes.

En s'appuyant sur les résultats théoriques de Inria [Dés09, Dés12], nous avons initié une généralisation des méthodes dites de descente [All05, Sny05] au cadre multiobjectif. La méthode MGDA repose sur l'existence, en un point non-stationnaire [Dés12], d'une direction de descente commune à l'ensemble des critères. En ce sens, MGDA peut être considéré comme une méthode d'optimisation coopérative.

Cette étude se décompose en 5 autres chapitres. Dans un premier chapitre, nous introduisons les notions et le vocabulaire usités en optimisation. Nous y décrivons aussi les principales méthodes d'optimisation mono-objectif basées sur l'utilisation des gradients de descente. Dans un le chapitre 2, nous introduisons la notion de dominance de Pareto, nécessaire lors de l'étude de problèmes multiobjectifs. Nous décrivons ensuite les principaux algorithmes évolutionnaires et nous mettons en évidence les avantages et les inconvénients des particularités de chacun. Dans le chapitre 3, nous décrivons le cadre d'application de l'algorithme de descente à gradients multiples (MGDA). Après avoir introduit les éléments théoriques démontrés dans [Dés12], nous testons la méthode sur des cas tests analytiques. Dans le chapitre 4, nous étudions une méthode hybride basée sur l'assistance de MGDA par la construction d'un métamodèle de type *Krigeage* [Kri51]. Nous l'utilisons alors pour résoudre deux problèmes d'optimisation simplifiés : le cas test de Kursawe [Kur91], dont le front de Pareto est non convexe et discontinu, puis un cas classique de profil d'aile, dont la paramétrisation est simplifiée. Enfin, dans le chapitre 5, nous présentons les résultats obtenus par MGDA assisté par métamodèle dans la résolution d'un problème d'optimisation de nature pré-industrielle. Ce cas test est proposé par le constructeur automobile Renault, dans le cadre du projet ANR OMD2 (Optimisation Multidisciplinaire Distribuée). La thèse se termine par une conclusion générale.

Chapitre 2

Méthodes de descente en optimisation unicritère

Ce chapitre aborde les problèmes d'optimisation unicritère. Nous introduisons le vocabulaire propre à cette discipline et nous énonçons les résultats d'existence et d'unicité des solutions de problèmes, sous certaines conditions. Plus particulièrement, nous poursuivons par la description des méthodes de résolution basées sur l'utilisation des gradients. Nous décrivons ensuite les principaux algorithmes basés sur l'utilisation des directions de descente.

2.1 Définitions et notations

L'optimisation possède un vocabulaire particulier. Aussi, nous allons, dans un premier temps, introduire les notations et les définitions courantes de la discipline. Ce chapitre est inspiré du livre de Grégoire Allaire [All05]. Dans ce qui suit, sans perte de généralité, nous limiterons notre étude aux problèmes de minimisation. En effet, il suffit de changer le signe de la fonctionnelle étudiée pour transformer un problème de maximisation en problème de minimisation.

Les problèmes d'optimisation sont généralement traités dans des espaces de Hilbert (ou de Banach, dans de rares cas). Nous noterons V l'espace vectoriel normé dans lequel est posé le problème. L'ensemble des cas traités au cours de cette étude dépendent de variables réelles. Aussi, sauf mention contraire, nous considèrerons que $V = \mathbb{R}^N$. Pour tout couple $(u, v) \in V^2$, on note $\langle u,\, v\rangle$ le produit scalaire entre u et v.

$$\forall (u, v) \in V^2, \quad \langle u,\, v\rangle = \sum_{i=1}^{N} u_i v_i \tag{2.1}$$

Pour tout élément $v \in V$, on note $\|v\|$ la norme associée au produit scalaire défini en 2.1.

$$\forall v \in V, \quad \|v\| = \sqrt{\langle v, v \rangle} \tag{2.2}$$

Nous notons $K \subset V$ le sous-ensemble dans lequel nous cherchons la solution. On dit que K est l'ensemble des éléments **admissibles** du problème, ou bien que K définit les **contraintes** qui s'exercent sur le problème. Enfin, la **fonction coût**, ou le **critère**, ou la fonction **objectif** à minimiser J est une fonction définie sur K et à valeurs dans $\mathbb{R}$. Le problème considéré sera donc :

$$\begin{cases} \inf J(v) \\ v \in K \subset V \end{cases} \tag{2.3}$$

L'utilisation de inf dans la formulation du problème traduit le fait que, à priori, nous ne savons pas s'il existe un élément v^* qui réalise l'infimum. Si l'on souhaite indiquer que la solution du problème appartient à l'ensemble des éléments admissibles, il suffit de remplacer le inf par min.

Définition 2.1.0.1 (Minimum local/global)**.** *On dit que u est un minimum (ou un point de minimum) local de J sur K si et seulement si :*

$$u \in K \text{ et } \exists \delta > 0, \quad \forall v \in K, \quad \|u - v\| < \delta \Rightarrow J(v) \geq J(u).$$

On dit que u est un minimum (ou un point de minimum) global de J sur K si et seulement si :

$$u \in K, \text{ et } J(v) \geq J(u) \quad \forall v \in K.$$

La figure 2.1 illustre différents exemples d'optima locaux et globaux.

Définition 2.1.0.2 (Suite minimisante)**.** *Une suite minimisante sur K est une suite telle que :*

$$\forall n \in \mathbb{N}, u_n \in K \text{ et } \lim_{n \to +\infty} J(u_n) = \inf_{v \in K} J(v)$$

2.2 Optimisation en dimension finie

Dans cette section, nous traitons la question d'existence de **minima** pour des problèmes d'optimisation posés en dimension finie. Le résultat suivant garantit l'existence d'un minimum, en dimension finie.

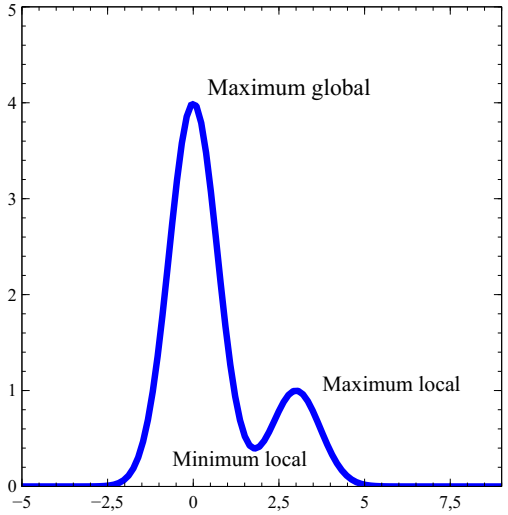


FIGURE 2.1: $f : x \mapsto 4e^{-x^2} + e^{-(x-3)^2}$, exemple de minima et de maxima locaux et globaux.

Théorème 2.2.1 (Existence d'un minimum en dimension finie)**.** *Soient K un ensemble fermé non vide de $\mathbb{R}^N$ et J une fonction continue sur K, à valeurs dans $\mathbb{R}$ qui vérifie la propriété dite* ***infinie à l'infini*** *(cf figure 2.2) suivante :*

$$\forall (u_n)_{n\in\mathbb{N}} \text{ suite dans } K, \quad \lim_{n\to+\infty} \|u_n\| = +\infty \Rightarrow \lim_{n\to+\infty} J(u_n) = +\infty \tag{2.4}$$

Alors, il existe au moins un point de minimum de J sur K. De plus, on peut extraire de toute suite minimisante sur K une sous-suite qui converge vers un point de minimum sur K.

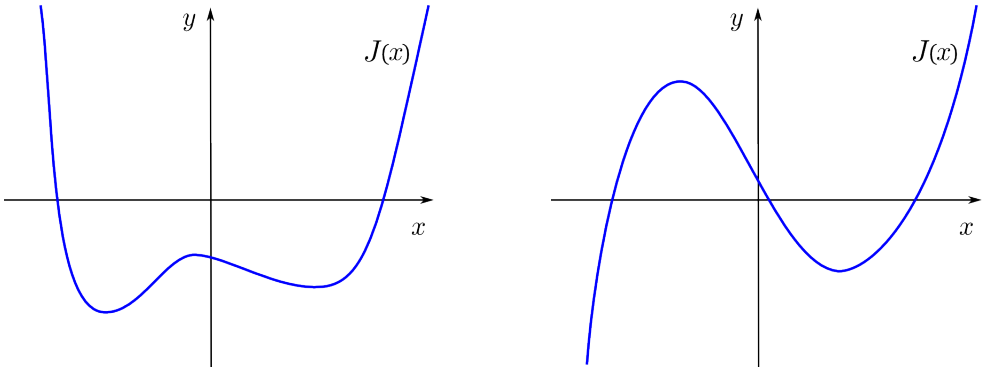


FIGURE 2.2: Exemple de fonction infinie à l'infini (à gauche) et de fonction non infinie à l'infini (à droite).

Preuve. Soit (u_n) une suite minimisante de J sur K. La condition 2.4 implique que (u_n)

est bornée puisque $J(u_n)$ est une suite de réels majorée. La réciproque du théorème de Bolsano-Weierstrass [Wag03] assure l'existence d'une sous-suite (u_{n_k}) de (u_n) qui converge vers un point $u \in \mathbb{R}^n$. Or K est fermé donc $u \in K$ et $J(u_{n_k})$ converge vers $J(u)$, par continuité de J. D'où :

$$J(u) = \inf_{v \in K} J(v)$$

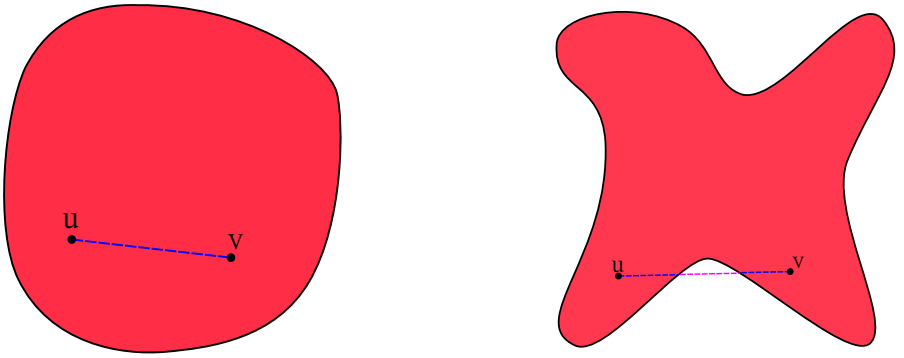
□

2.2.1 Analyse convexe

Rappelons qu'un ensemble K est convexe s'il contient tous les segments qui relient deux de ses points quelconques. Voir illustration figure 2.3.

Définition 2.2.1.1 (Ensemble convexe)**.** *Soit $K \subset \mathbb{R}^n$. K est un ensemble convexe si et seulement si :*

$$\forall (u, v) \in K^2, \quad \forall \lambda \in]0, 1[, \quad \lambda u + (1 - \lambda)v \in K$$



ensemble convexe ensemble non convexe

Figure 2.3: Notion de convexité.

Donnons maintenant quelques propriétés sur les fonctions convexes.

Définition 2.2.1.2 (Fonction convexe)**.** *On dit qu'une fonction J définie sur un ensemble convexe non vide $K \subset V$ et à valeurs dans $\mathbb{R}$ est convexe sur K si et seulement si :*

$$J(\lambda u + (1 - \lambda)v) \leq \lambda J(u) + (1 - \lambda)J(v) \quad \forall u, v \in K, \forall \lambda \in [0, 1]. \tag{2.5}$$

De plus, J est dite strictement convexe si l'inégalité 2.5 est stricte lorsque $u \neq v$ et $\lambda \in]0, 1[$.

Définition 2.2.1.3 (Epigraphe)**.** *Soit J une application définie sur K, à valeurs dans $\mathbb{R}$. On appelle* ***épigraphe*** *de J l'ensemble (Illustration figure 2.4) :*

$$Epi(J) = \{(\theta, v) \in \mathbb{R} \times K, \quad \theta \geq J(v)\}$$

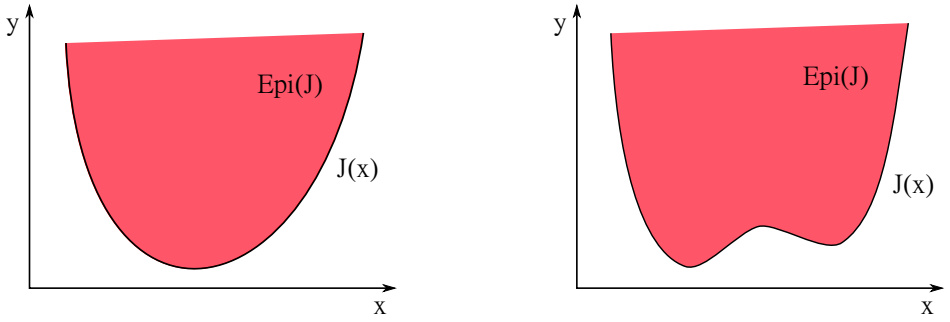


Figure 2.4: Epigraphe d'une fonction J continue et convexe, à gauche, et d'une fonction continue, non convexe, à droite.

Proposition 2.2.2. *Une fonction J définie de K dans $\mathbb{R}$ est convexe si et seulement si $Epi(J)$ est une partie convexe de $\mathbb{R} \times K$.*

Preuve. Supposons que J est une fonction convexe de K dans $\mathbb{R}$ et considérons les éléments (α, u) et (β, v) de $Epi(J)$. Alors, pour tout $\lambda \in [0, 1]$, nous avons :

$$J\left(\lambda v + (1-\lambda)u\right) \leq \lambda J(v) + (1-\lambda)J(u) \leq (1-\lambda)\alpha + \lambda\beta$$

Donc $(1-\lambda)(\alpha, u) + \lambda(\beta, v) \in Epi(J)$, pour tout $\lambda \in [0, 1]$ et $Epi(J)$ est convexe.

Supposons, réciproquement que $Epi(J)$ est convexe et considérons u et v deux éléments de K tels que $J(u) < +\infty$ et $J(v) < +\infty$. Alors il existe $\alpha \in \mathbb{R}$ et $\beta \in \mathbb{R}$ tels que $J(u) \leq \alpha$ et $J(v) \leq \beta$. Comme (α, u) et (β, v) sont dans l'épigraphe qui est convexe, il en est de même pour $(1-\lambda)(\alpha, u) + \lambda(\beta, v) = ((1-\lambda)\alpha + \lambda\beta, (1-\lambda)u + \lambda v)$. Ce qui se traduit par :

$$J((1-\lambda)u + \lambda v) \leq (1-\lambda)\alpha + \lambda\beta, \quad \forall \lambda \in [0, 1].$$

Comme $\alpha \geq J(u)$ et $\beta \geq J(v)$ sont arbitraires, on obtient :

$$J((1-\lambda)u + \lambda v) \leq (1-\lambda)J(u) + \lambda J(v), \quad \forall \lambda \in [0, 1].$$

Donc J est convexe. □

Pour les fonctions convexes, les minima locaux sont aussi les minima globaux.

Proposition 2.2.3. *Soit J une fonction convexe sur un ensemble convexe K. Tout point de minimum local de J sur K est un minimum global et l'ensemble des points de minimum est convexe.*
Si de plus J est strictement convexe, alors il existe au plus un minimum.

Preuve. Soit u un minimum local de J sur K. D'après la définition 2.1.0.1, nous pouvons écrire :

$$\exists \delta > 0, \quad \forall w \in K, \quad \|w - u\| < \delta \Rightarrow J(w) \geq J(u). \tag{2.6}$$

Soit $v \in K$. Pour $\lambda \in]0,1[$ suffisamment petit, alors l'élément w_λ défini par :

$$w_\lambda = \lambda v + (1 - \lambda)u,$$

vérifie :

$$\|w_\lambda - u\| < \delta \text{ et } w_\lambda \in K,$$

puisque K est convexe. Donc

$$J(w_\lambda) \geq J(u)$$

d'après 2.6 et la convexité de J, nous avons :

$$J(u) \leq J(w_\lambda) \leq \lambda J(v) + (1 - \lambda)J(u).$$

Ce qui montre bien que $J(u) \leq J(v)$, c'est-à-dire que u est un minimum global sur K.

D'autre part, si u_1 et u_2 sont deux minima et si $\lambda \in [0,1]$, alors

$$w = \lambda u_1 + (1 - \lambda)u_2$$

est un minimum, puisque $w \in K$, et que

$$\inf_{v \in K} J(v) \leq J(w) \leq \lambda J(u_1) + (1 - \lambda)J(u_2) = \inf_{v \in K} J(v).$$

Le même raisonnement avec $\lambda \in]0,1[$ montre que si J est strictement convexe, alors $u_1 = u_2$. □

Nous introduisons maintenant la notion de **forte convexité**, plus restrictive que celle de convexité :

Définition 2.2.1.4 (Forte convexité)**.** *Une fonction J définie sur un ensemble convexe K est fortement convexe si et seulement s'il existe $\alpha > 0$ tel que :*

$$J\left(\frac{u+v}{2}\right) \leq \frac{J(u) + J(v)}{2} - \frac{\alpha}{8}\|u - v\|^2 \tag{2.7}$$

On dit aussi dans ce cas que J est α-convexe

Le résultat suivant permet de montrer qu'une fonction J fortement convexe et continue sur un ensemble K fermé, convexe et non vide, est infinie à l'infini dans K. Ceci est essentiel à la démonstration d'existence d'un minimum en dimension infinie. Avant d'énoncer ce résultat, nous écrivons le théorème suivant :

Théorème 2.2.4 (Séparation d'un point et d'un convexe)**.** *Soit K une partie convexe non vide et fermée d'un espace de Hilbert V et $x_0 \notin K$. Alors, il existe un hyperplan fermé de V qui sépare strictement x_0 et K, c'est-à-dire qu'il existe une forme linéaire $L \in V^*$ et $\alpha \in \mathbb{R}$ tels que :*

$$L(x_0) < \alpha < L(x) \quad \forall x \in K. \tag{2.8}$$

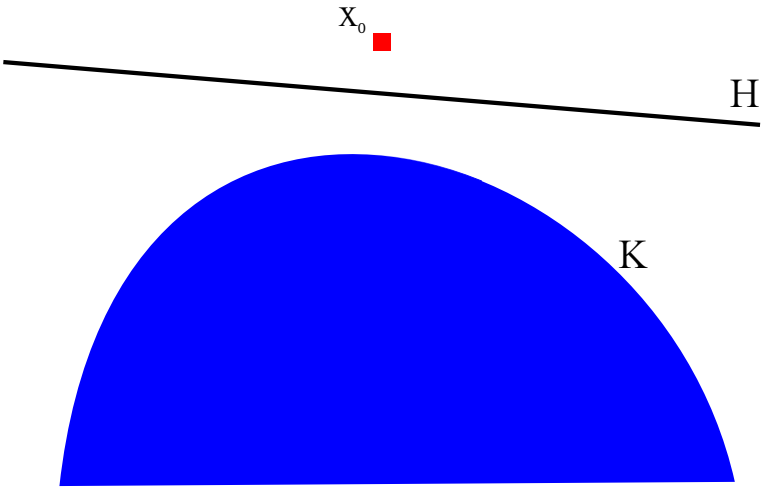


Figure 2.5: Séparation d'un point et d'un convexe

Preuve. Notons x_K la projection de x_0 sur K. Puisque $x_0 \notin K$, $\|x_K - x_0\| \neq 0$. Soit L la forme linéaire définie pour tout $y \in V$ par : $L(y) = (x_K - x_0, y)$ et soit $\alpha = (L(x_K) + L(x_0))/2$. D'après le théorème de projection sur un convexe, on a :

$$L(x) \geq L(x_K) > \alpha > L(x_0), \quad \forall x \in K.$$

Ce qui achève la démonstration. □

Proposition 2.2.5. *Soit J une fonction convexe et continue sur un ensemble K convexe, fermé et non vide. Il existe alors une forme linéaire continue $L \in V^*$ et une constante*

$\delta \in \mathbb{R}$ telles que :

$$J(v) \geq L(v) + \delta \quad \forall v \in K. \tag{2.9}$$

Si, de plus, J est fortement convexe sur K, il existe deux constantes $\gamma > 0$ et $\delta \in \mathbb{R}$ telles que :

$$J(v) \geq \gamma \|v\|^2 - \delta \quad \forall v \in K. \tag{2.10}$$

Preuve. Nous commençons par montrer l'inégalité 2.9. Si J est convexe continue (ou simplement semi-continue inférieurement) sur un ensemble K convexe fermé non vide, alors son épigraphe $Epi(J)$ (cf définition 2.2.1.3) est convexe, fermé et non vide.
Soient $v_0 \in K$ et $\lambda_0 < J(v_0)$. Puisque $(\lambda_0, v_0) \notin Epi(j)$, nous déduisons du théorème 2.8 de séparation d'un point et d'un convexe l'existence de deux réels α et β, ainsi qu'une forme linéaire continue $L \in V^*$, tels que :

$$\beta\lambda + L(v) > \alpha > \beta\lambda_0 + L(v_0), \quad \forall (\lambda, v) \in Epi(J). \tag{2.11}$$

Comme, pour v fixé, on peut prendre λ arbitrairement grand dans le membre de gauche de 2.11, il est clair que $\beta \geq 0$.

De plus, comme l'on peut prendre $v = v_0$ dans le membre de gauche de l'inégalité 2.11, β ne peut pas être nul. Donc $\beta > 0$. On déduit alors de 2.11 que $(J(v)+L(v))/\beta > \alpha/\beta$, pour tout $v \in K$. Ce qui prouve 2.9.

Prouvons maintenant 2.10. On considère $v_0 \in K$ fixé. Pour tout $v \in K$, Les résultats 2.7 et 2.9 impliquent que :

$$\frac{J(v)}{2} + \frac{J(v_0)}{2} \geq L\left(\frac{v+v_0}{2}\right) + \frac{\alpha}{8}\|v - v_0\|^2 \geq \frac{L(v) + L(v_0)}{2} + \frac{\alpha}{8}\|v - v_0\|^2 + \delta$$

On en déduit :

$$J(v) \geq \frac{\alpha}{4}\|v\|^2 - \frac{\alpha}{2}(v, v_0) + L(v) + C_1, \quad C_1 = \frac{\alpha}{4}\|v_0\|^2 + L(v_0) - J(v_0) + 2\delta.$$

D'après l'inégalité de Cauchy-Schwarz [Wag03] appliquée à (v, v_0) et la continuité de L, il vient :

$$J(v) \geq \frac{\alpha}{4}\|v\|^2 - \left(\|L\|_{V^*} + \frac{\alpha\|v_0\|}{2}\right)\|v\| + C_1 \geq \frac{\alpha}{2}\|v\|^2 - C$$

Pour $C \in \mathbb{R}$, choisi. □

2.2.2 Résultats d'existence

Nous commençons dans cette sous-section par montrer un résultat d'existence de minimum dans le cas particulier où J est fortement convexe (i.e. α-convexe).

Théorème 2.2.6 (Existence d'un minimum, cas fortement convexe)**.** *Soit K un convexe fermé non vide d'un Hilbert V et J une fonction α-convexe continue sur K. Alors, il existe un unique minimum u de J sur K et on a :*

$$\|v-u\|^2 \leq \frac{4}{\alpha}(J(v)-J(u)), \quad \forall v \in K. \tag{2.12}$$

En particulier, toute suite minimisante de J sur K converge vers u.

Preuve. Soit (u_n) une suite minimisante de J sur K. D'après 2.10, J est minorée sur K et, pour $n, m \in \mathbb{N}$, la propriété 2.7 de forte convexité entraîne :

$$\frac{\alpha}{8}\|u_n-u_m\|^2+J\left(\frac{u_n+u_m}{2}\right)-\inf_{v\in K} J(v) \leq \frac{1}{2}\left(J(u_n)-\inf_{v\in K} J(v)\right)+\frac{1}{2}\left(J(u_m)-\inf_{v\in K} J(v)\right).$$

La suite (u_n) est donc de Cauchy et converge vers une limite u. Cette limite est nécessairement un minimum de J sur K car K est fermé. L'unicité du point minimum a été montrée dans la proposition 2.2.3. Enfin, si $v \in K$, $(u+v)/2 \in K$ car K est convexe, d'où, par 2.7 il vient que :

$$\frac{\alpha}{8}\|u-v\|^2 \leq \frac{J(u)}{2}+\frac{J(v)}{2}-J\left(\frac{u+v}{2}\right) \leq \frac{J(v)-J(u)}{2},$$

car $J\left(\frac{u+v}{2}\right) \geq J(u)$. □

Le résultat du théorème 2.2.6 peut être généralisé aux fonctions J convexes (et non pas fortement convexes).

Théorème 2.2.7 (Existence d'un minimum, cas convexe)**.** *Soient K un convexe fermé non vide d'un espace de Hilbert V et une fonction J convexe et continue sur K et infinie à l'infini, c'est-à-dire :*

$$\forall (u_n)_{n\in\mathbb{N}} \text{ suite dans } K, \quad \lim_{n\to+\infty}\|u_n\| = +\infty \Rightarrow \lim_{n\to+\infty} J(u_n) = +\infty.$$

Alors, il existe un minimum de J sur K.

Le théorème 2.2.7 assure l'existence d'un minimum, comme le théorème 2.2.6, mais il ne dit rien sur l'unicité ou l'estimation de l'erreur 2.12.
La preuve du théorème 2.2.7 nécessite la définition de la convergence faible, ainsi que les résultats de quelques Lemmes. Avant d'en donner la démonstration, nous allons énoncer ces résultats.

Définition 2.2.2.1 (Convergence faible)**.** *On dit qu'une suite (u_n) de V converge fai-*

blement vers $u \in V$ si et seulement si :

$$\lim_{n\to+\infty}(u_n, v) = (u, v) \quad , \forall v \in V. \tag{2.13}$$

En particulier, si l'on considère la base hilbertienne (e_i) de V et si l'on note $u_{n,i} = (u_n, e_i)$, les composantes dans cette base de la suite u_n, uniformément bornée dans V, la définition 2.2.2.1 de la convergence faible est équivalente à la convergence de chacune des composantes $u_{n,i}$.

Lemme 2.2.8. *Soit (u_n) une suite bornée d'éléments de V. Alors il existe une sous-suite (u_{n_k}) de (u_n) qui converge faiblement dans V.*

Preuve. Comme la suite (u_n) est bornée, chaque suite d'une composante $(u_{n,i})$ est bornée dans $\mathbb{R}$. Pour chaque i, il existe une sous-suite $(u_{n_i,i})$ qui converge vers une limite u_i. On obtient ainsi une sous-suite $u_{n'}$ telle que, pour tout i, $(u_{n',i})$ converge vers u_i. La suite $(u_{n'})$ converge alors faiblement vers $u \in V$. □

Définition 2.2.2.2 (Demi-espace fermé)**.** *Soient L une forme linéaire continue de V^*, non identiquement nulle et $\alpha \in \mathbb{R}$. Alors, les sous-ensembles de V de la forme :*

$$\left\{v \in V, \quad L(v) \leq \alpha\right\}$$

*sont appelés **demi-espaces fermés** de V.*

On peut ainsi caractériser de manière commode les ensembles convexes fermés.

Lemme 2.2.9. *Une partie convexe fermée K de V est l'intersection des demi-espaces fermés qui contiennent K.*

Preuve. Il est clair que K est inclus dans l'intersection des demi-espaces fermés qui le contiennent. Réciproquement, supposons qu'il existe un point u_0 de cette intersection qui n'appartienne pas à K. Le théorème 2.2.4 de séparation entre un point et un convexe assure l'existence d'un demi-espace fermé qui contient K, mais pas u_0. Or ceci contredit la définition de u_0. Donc $u_0 \in K$. □

Lemme 2.2.10. *Soit K un ensemble convexe fermé non vide de V. Alors K est fermé pour la convergence faible.*

De plus, si J est convexe, et semi-continue inférieurement sur K, alors J est aussi semi-continue inférieurement sur K pour la convergence faible.

Preuve. Par continuité de L, si u_n converge faiblement vers u, $L(u_n)$ converge vers $L(u)$. Par conséquent, un demi-espace fermé de V est fermé pour la convergence faible. Le Lemme 2.2.9 permet d'obtenir la même conclusion sur K.

D'après la définition de J, $Epi(J)$ est un convexe fermé de $\mathbb{R} \times V$ donc il est aussi fermé pour la convergence faible. On en déduit alors facilement le résultat suivant : si la suite (v_n) tend faiblement vers v dans K, alors :

$$\lim_{n \to +\infty} \left(\inf_{v_n} J(v_n) \right) \geq J(v)$$

□

Et finalement :

Preuve du théorème 2.2.7. Comme toute suite minimisante est bornée, on déduit du Lemme 2.2.8 qu'il existe une sous-suite $u_{n'}$ qui converge faiblement vers une limite $u \in V$. Mais, d'après le Lemme 2.2.10, $u \in K$ et

$$J(u) \leq \liminf J(u_{n_k}) = \inf_{v \in K} J(v).$$

Le point u définit bien un minimum de $J(u)$ sur K. □

2.3 Généralités sur les algorithmes de descente

On considère un point x_0 quelconque. A partir de ce point, par un algorithme de descente, on cherchera à générer une suite d'éléments $(x_n)_{n \in \mathbb{N}}$ telle que :

$$\forall k \in \mathbb{N}, \quad J(x_{k+1}) \leq J(x_k)$$

Dans un premier temps, nous allons introduire la notion de descente.

2.3.1 Notion de descente

Dans le cadre de l'optimisation par une méthode de descente, il est important d'analyser le comportement de la fonctionnelle à minimiser dans des directions particulières. Il est alors important de préciser la notion de dérivée directionnelle.

Définition 2.3.1.1. *Soit $J : \mathbb{R}^n \to \mathbb{R}$ une application continue. Soit $x \in \mathbb{R}^n$ et $d \in \mathbb{R}^n$. La dérivée directionnelle de J en x, dans la direction d est définie par :*

$$dJ(x; d) = \lim_{t \to 0^+} \frac{J(x + td) - J(x)}{t},$$

si cette limite existe.

Proposition 2.3.1. *Si J est différentiable en un point $x \in \mathbb{R}^n$, alors, pour tout $d \neq 0$, J admet une dérivée dans la direction d en x et :*

$$dJ(x;d) = DJ(x)(d) = \nabla J(x)^\top d$$

On rappelle que la réciproque est fausse. La dérivabilité en un point x, selon toutes les directions, n'implique pas nécessairement la différentiabilité de J en x.

La dérivée directionnelle donne des informations sur la pente de la fonction, en un point, dans une direction donnée. En particulier :

- Si $dJ(x;d) > 0$ alors J est croissante dans la direction d ;
- Si $dJ(x;d) < 0$ alors J est décroissante dans la direction d ;

Dans le second cas, on dira que d est une direction de descente de J, en x.

Définition 2.3.1.2 (Direction de descente)**.** *Soient $J : \mathbb{R}^n \to \mathbb{R}$ et $x \in \mathbb{R}^n$. Le vecteur $d \in \mathbb{R}^n$ est une direction de descente pour J à partir du point x si $t \mapsto J(x+td)$ est décroissante en $t = 0$, c'est-à-dire s'il existe $\eta > 0$ tel que :*

$$\forall t \in]0, \eta[, \quad J(x+td) < J(x) \tag{2.14}$$

Vient alors la proposition suivante :

Proposition 2.3.2. *Soient $J : \mathbb{R}^n \to \mathbb{R}$ différentiable et $x \in \mathbb{R}^n$ tel que $\nabla J(x) \neq 0$. Le vecteur $d \in \mathbb{R}^n$ est une direction de descente pour J à partir du point x si et seulement si la dérivée directionnelle de J en x dans la direction d vérifie :*

$$dJ(x;d) = \nabla J(x)^\top d < 0 \tag{2.15}$$

De plus, pour tout $\alpha < 1$, il existe $\xi > 0$ tel que :

$$\forall t \in]0, \xi[, \quad J(x+td) < J(x) + t\alpha \nabla J(x)^\top d \tag{2.16}$$

La relation 2.16 se traduit de la manière suivante : la décroissance de la fonction J, en effectuant un pas de longueur t dans la direction d, est au moins égale à la longueur du pas multipliée par une fraction α de la pente.

Preuve. On considère d, une direction de descente de la fonction J, au point $x \in \mathbb{R}^n$. Nous avons donc :

$$\nabla J(x)^\top d < 0$$

Le développement de Taylor-Young de l'application $t \mapsto J(x+td)$ à l'ordre 1 s'écrit :

$$J(x+td) = J(x) + t\nabla J(x)^\top d + t\varepsilon(t), \quad \lim_{t \to 0} \varepsilon(t) = 0$$

Sachant que $\mu = -\nabla J(x)^\top d > 0$, il existe $\eta > 0$ tel que, si $|t| < \eta$, $|\varepsilon(t)| < \mu$. D'où :

$$\forall t \in]0, \eta], \quad J(x + td) - J(x) = t(\nabla J(x)^\top d + \varepsilon(t)) < t(\nabla J(x)^\top d + \mu) = 0$$

La démonstration de l'inégalité 2.16 est identique, en choisissant $\mu = (\alpha - 1)\nabla J(x)^\top d$ et $\alpha < 1$. □

Parmi toutes les directions de descente qui existent en un point x donné, la direction donnée par le gradient est particulière. En effet, cette direction offre la plus grande descente. D'où le théorème suivant :

Théorème 2.3.3 (Direction de plus forte descente)**.** *Soient $J : \mathbb{R}^n \to \mathbb{R}$ une fonction différentiable et $x \in \mathbb{R}^n$. Alors, pour toute direction d, de norme constante égale à $\|d\| = \|\nabla J(x)\|$, on a :*

$$(-\nabla J(x))^\top \nabla J(x) \leq d^\top \nabla J(x), \tag{2.17}$$

La direction $d^* = -\nabla J(x)$ est appelée direction de plus forte descente.

Preuve. Soit $d \in \mathbb{R}^n$ telle que $\|d\| = \|\nabla J(x)\|$. L'inégalité de Cauchy-Schwarz [Wag03] donne :

$$(-d)^\top \nabla J(x) \leq \| - d\| \|\nabla J(x)\|$$

On en déduit directement $(-d)^\top \nabla J(x) \leq \|\nabla J(x)\|^2$, soit 2.17. □

2.3.2 Convergence et vitesse de convergence

Etudier la convergence d'un algorithme revient à étudier la suite des itérés de cet algorithme. Un algorithme de descente est dit convergent si la suite des itérés $(x_n)_{n\in\mathbb{N}}$ converge vers un point limite x^* solution du problème :

$$J(x^*) = \min_{x \in \mathbb{R}^n} J(x)$$

La convergence est dite *locale* si elle n'a lieu que pour des points situés dans un voisinage de x^*. Sinon, elle est dite *globale*.

En pratique, l'objectif d'un algorithme de descente est de trouver un point critique ($\nabla J(x^*) = 0$). La notion de *convergence globale* doit alors être définie :

Définition 2.3.2.1. *On considère un algorithme itératif qui génère une suite $(x_n)_{n\in\mathbb{N}}$ d'éléments de $\mathbb{R}^n$, afin de résoudre le problème d'optimisation :*

$$\min_{x \in \mathbb{R}^n} J(x)$$

où $J : \mathbb{R}^n \to \mathbb{R}$ est au moins de classe C^1. L'algorithme est dit globalement convergent si, pour tout point initial $x_0 \in \mathbb{R}^n$,

$$\lim_{k \to +\infty} \|\nabla J(x)\| = 0$$

Bien qu'il soit primordial de garantir la convergence d'un algorithme, sous certaines conditions, l'étude de la vitesse de convergence et la complexité sont à prendre en compte. En effet, nous avons tout intérêt à développer une méthode qui soit à la fois rapide, précise et stable. Pour cela, nous introduisons les notions de vitesse de convergence qui mesurent l'évolution de l'erreur commise : $\|x - x^*\|$.

Définition 2.3.2.2. *On considère $(x_n)_{n\in\mathbb{N}}$, l'ensemble des itérés donnés par un algorithme convergent. On note x^* la limite de la suite $(x_n)_{n\in\mathbb{N}}$ et on suppose que $x_k \neq x^*$, pour tout $k \in \mathbb{N}$. La convergence de l'algorithme est alors dite :*

- *linéaire si l'erreur décroît linéairement i.e. :*

$$\exists \tau \in]0,1[, \quad \lim_{k \to +\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = \tau$$

- *superlinéaire si :*

$$\lim_{k \to +\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0$$

- *d'ordre p si :*

$$\exists \tau \geq 0, \quad \lim_{k \to +\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p} = \tau$$

En particulier, si $p = 2$, la convergence est dite quadratique.

2.4 Premiers algorithmes de descente

Un algorithme de descente est déterminé à la fois par la direction de descente choisie et par le pas effectué dans cette direction, à chaque itération. Nous donnons une description générale de ces algorithmes dans l'algirithme 1.

Données : J, x_0, ε
Resultat : x^*, approximation de la solution de $\nabla J(x) = 0$
Initialisation;
tant que $\|\nabla J(x_k)\| \geq \varepsilon$ **faire**
  a) Calcul de la direction de descente d_k telle que : $\nabla J(x_k)^\top d_k < 0$;
  b) Recherche linéaire du pas s_k : $J(x_k + s_k d_k) < J(x_k)$;
  c) Calcul du nouveau point : $x_{k+1} = x_k - s_k d_k$; $k = k + 1$.
fin
retourner : x_k.

Algorithme 1: Algorithme de descente.

Dans un premier temps, nous allons nous concentrer sur l'étude de la direction de descente.

2.4.1 Algorithmes à pas fixe

On considère l'itéré courant $x_k \in \mathbb{R}^n$. On remplace la valeur de J dans un voisinage de x_k par son développement de Taylor au premier ordre :

$$J(x_k + d) \approx J(x_k) + \nabla J(x_k)^\top d$$

Nous voudrions que la dérivée directionnelle $\nabla J(x_k)^\top d$ soit la plus petite possible dans un voisinage de $d = 0$. Ce qui revient à résoudre :

$$\min_{d \in \mathbb{R}^n} \nabla J(x_k)^\top d \quad s.c. \quad \|d\| = 1$$

La solution de ce problème d'optimisation nous est donnée par le théorème 2.3.3. Il s'agit de la direction de plus grande descente, normalisée :

$$d_k = -\frac{\nabla J(x_k)}{\|\nabla J(x_k)\|}$$

Le choix de cette direction particulière définit une famille d'algorithmes appelée *algorithmes de descente de gradient*. Le schéma de fonctionnement de ces algorithmes est le suivant :

Données : J, x_0, ε
Resultat : x^*, approximation de la solution de $\nabla J(x) = 0$
Initialisation;
tant que $\|\nabla J(x_k)\| \geq \varepsilon$ **faire**
- a) Calcul de la direction de descente : $d_k = -\dfrac{\nabla J(x_k)}{\|\nabla J(x_k)\|}$;
- b) Recherche linéaire du pas s_k : $J(x_k + s_k d_k) < J(x_k)$;
- c) Calcul du nouveau point : $x_{k+1} = x_k - s_k \dfrac{\nabla J(x_k)}{\|\nabla J(x_k)\|}$; $k = k + 1$.

fin
retourner : x_k.

Algorithme 2: Algorithme de descente de gradient.

Théorème 2.4.1 (Convergence de l'algorithme à pas fixe)**.** *On suppose que J est α-convexe (cf définition 2.2.1.4) différentiable et que ∇J est Lipchitzien sur V, c'est-à-dire qu'il existe une constante $C > 0$ telle que :*

$$\|\nabla J(v) - \nabla J(w)\| \leq C\|v - w\|, \quad \forall v, w \in V. \tag{2.18}$$

Alors, si $0 < \mu < 2\alpha/C^2$*, l'algorithme de gradient à pas fixe converge.*

Preuve. On pose $r_n = u_n - u$. Comme $\nabla J(u) = 0$, on a :

$$r_{n+1} = r_n - \mu \left(\nabla J(u_n) - \nabla J(u) \right).$$

D'où il vient :

$$\begin{array}{rcl} \|r_{n+1}\|^2 & = & \|r_n\|^2 - 2\mu \left\langle J(u_n) - \nabla J(u),\, u_n - u \right\rangle + \mu^2 \|\nabla J(u_n - \nabla J(u)\|^2 \\ & \leq & \left(1 - 2\alpha\mu + C^2\mu^2\right) \|r_n\|^2, \end{array} \tag{2.19}$$

d'après 2.18 et l'α-convexité de J. Si $0 < \mu < 2\alpha/C^2$, il est facile de voir que $1 - 2\alpha\mu + C^2\mu^2$ est dans $]0, 1[$. La convergence se déduit alors de 2.19. □

Il reste maintenant à définir un pas de descente. Nous allons étudier différentes possibilités.

2.4.2 Méthode de plus grande descente

De manière naturelle, nous voudrions suivre la direction de plus grande descente et progresser du pas qui offre la plus grande réduction de la fonction J. Cette méthode est appelée *méthode de plus grande descente* (*"steepest descent"*).
Dans ce cas là, l'étape b) de l'algorithme de descente 2 est remplacée par :

$$b) \quad s_k \text{ solution de } \min_{s>0} J(x_k + s d_k) \tag{2.20}$$

Cet algorithme converge, comme l'indique le résultat suivant :

Théorème 2.4.2 (Convergence de l'algorithme à pas optimal)**.** *On suppose que J est α-convexe, différentiable et que ∇J est Lipschitzien sur tout borné de V, c'est-à-dire que :*

$$\forall M > 0, \exists C_M > 0, \|v\| + \|w\| \leq M \Rightarrow \|\nabla J(v) - \nabla J(w)\| \leq C_m \|v - w\| \tag{2.21}$$

Alors, l'algorithme de gradient à pas optimal converge.

Preuve. La fonction f définie par $\mu \mapsto J(u_n - \mu \nabla J(u_n))$ est fortement convexe et dérivable sur $\mathbb{R}$. Le problème de minimisation 2.20 admet alors une solution unique, caractérisée par la condition $f'(\mu) = 0$, ce qui s'écrit aussi :

$$\langle \nabla J(u_{n+1}),\, \nabla J(u_n) \rangle = 0. \tag{2.22}$$

Ceci montre que deux directions de descente consécutives sont orthogonales.

Puisque 2.22 implique que $\langle \nabla J(u_{n+1}),\, u_{n+1} - u_n \rangle = 0$, on déduit de l'$\alpha$-convexité de J que

$$J(u_n) - J(u_{n+1}) \geq \frac{\alpha}{2} \|u_n - u_{n+1}\|^2. \tag{2.23}$$

Puisque $J(u_n)$ est décroissante minorée (par $J(u)$), elle converge et l'inégalit'e 2.23 montre que $u_{n+1} - u_n$ tend vers 0. D'autre part, l'α-convexité de J et le fait que $J(u_n)$ soit bornée montrent que la suite u_n est bornée. Il existe une constante M telle que :

$$\|u_n\| \leq M, \quad \forall n \in \mathbb{N}$$

En écrivant 2.21 pour $v = u_n$ et $w = u_{n_1}$ et en utilisant 2.22, nous avons

$$\|\nabla J(u_n)\| \leq C_M \|u_{n+1} - u_n\|.$$

Ceci montre que $\|\nabla J(u_n)\|$ tend vers 0. L'α-convexité de J donne alors :

$$\begin{aligned} \alpha \|u_n - u\|^2 &\leq \langle \nabla J(u_n) - \nabla J(u),\, u_n - u \rangle \\ &= \langle \nabla J(u_n),\, u_n - u \rangle \\ &\leq \|\nabla J(u_n)\| \|u_n - u\|, \end{aligned}$$

qui implique $\alpha \|u_n - u\| \leq \|\nabla J(u_n)\|$, d'où l'on déduit la convergence de l'algorithme. □

En pratique, nous ne savons pas calculer de manière exacte la valeur d'un tel s_k. De plus, le calcul d'une valeur approchée de s_k peut être très coûteux en temps de calcul. Pour ces raisons, on peut lui préférer la méthode à pas constant.

2.4.3 Méthodes de Newton

On se place ici dans le cas d'un problème d'optimisation en dimension finie, $V = \mathbb{R}^N$. On considère une fonction F de classe C^2 de $\mathbb{R}^N$ dans $\mathbb{R}^N$. Soir u un zéro régulier de F, c'est-à-dire que

$$F(u) = 0 \text{ et } F'(u) \text{ matrice inversible.}$$

Au voisinage de $v \in \mathbb{R}^N$, un développement de Taylor de F s'écrit :

$$F(u) = F(v) + F'(v)(u - v) + \mathcal{O}(\|u - v\|^2)$$

c'est-à-dire

$$u = v - (F'(v))^{-1} F(v) + \mathcal{O}(\|v - u\|^2).$$

On cherche à résoudre ce problème de manière itérative, en négligeant le reste. Pour un point initial $u_0 \in \mathbb{R}^N$ donné, on effectue le calcul suivant :

$$u_{n+1} = u_n - (F'(v))^{-1}F(v), \quad n \geq 0. \tag{2.24}$$

Dans l'équation 2.24, on ne calcule pas réellement l'inverse de $F'(v)$, mais une valeur approchée, obtenue par des méthodes numériques [All05].

La méthode de Newton peut être appliquée à la résolution d'un problème d'optimisation si l'on pose $F = J'$ et que l'on utilise le processus 2.24 itératif pour résoudre l'équation $J'(u) = 0$. On peut aussi aborder la méthode de Newton comme une méthode de minimisation. Grâce au développement de Taylor

$$J(w) = J(v) + J'(v)(w - v) + \frac{1}{2}J''(v)(w - v).(w - v) + \mathcal{O}(\|w - v\|^3)$$

on peut approcher $J(w)$ au voisinage de V par une fonction quadratique. Le minimum de la partie quadratique est atteint pour $w = v - (J''(v))^{-1}J'(v)$, si la matrice $J''(v)$ est définie positive. On retrouve alors la formule itérative proposé 2.24.

Proposition 2.4.3. *Soit F une fonction de $\mathbb{R}^N$ dans $\mathbb{R}^N$, de classe C^2 et u un zéro régulier de F. Alors, il existe $\varepsilon > 0$ tel que si u_0 est tel que $\|u - u_0\| \leq \varepsilon$, la méthode de Newton définie par 2.24 converge. C'est-à-dire que (u_n) converge vers u. De plus, il existe une constante $C > 0$ telle que*

$$\|u_{n+1} - u\| \leq C\|u_n - u\|^2. \tag{2.25}$$

Preuve. Comme F' est continue, par définition de F, il existe $\varepsilon > 0$ tel que F' est inversible en tout point de la boule de centre u et de rayon ε. On suppose que u_n est dans cette boule (i.e. $\|u_n - u\| \leq \varepsilon$). Donc $F'(u_n)$ est inversible. Comme $F(u) = 0$, on déduit de 2.24

$$u_{n+1} - u = u_n - u - (F'(u_n))^{-1}\left(F(u_n) - F(u)\right)$$

dont le développement de Taylor donne :

$$u_{n+1} - u = (F'(u_n))^{-1}\mathcal{O}\left(\|u_n - u\|^2\right)$$

Comme $\|u_n - u\| \leq \varepsilon$, il existe une constante $C > 0$ telle que

$$\|u_{n+1} - u\| \leq C\|u_n - u\|^2. \tag{2.26}$$

Si ε est tel que $C\varepsilon \leq 1$, u_{n+1} est aussi dans la boule de centre u et de rayon ε. Cela

permet de vérifier par récurrence l'hypothèse $\|u_n - u\| \leq \varepsilon$, pour tout $n \geq 0$ et 2.26 est la conclusion désirée. □

Dans le cadre bien précis d'une fonction de classe C^2 et où le point initial u_0 est proche de la solution, la méthode de Newton offre une convergence quadratique vers la solution. Il faut cependant noter que chaque itération de la méthode nécessite la résolution d'un système linéaire. Opération qui peut se révéler très coûteuse pour N grand. De plus, il est nécessaire de connaître le Hessien de la fonctionnelle J''. Il existe cependant des méthodes de résolution approchée des systèmes linéaires, moins précises, mais plus rapides.

2.4.4 Méthodes de Quasi-Newton

Nous venons de voir que les méthodes de Newton possèdent des aspects de vitesse de convergence très intéressants. Cependant, la résolution du système

$$(F'(v))^{-1}F(v), \quad v \in \mathbb{R}^N$$

pose de véritables problèmes lors de la mise en application.

Dans les méthodes dites de *quasi-Newton*, on remplace le calcul de la Hessienne $F'(v)$ et de son inverse par l'algorithme suivant :

$$u_0 \in \mathbb{R}^N, \quad u_{n+1} = u_n - s_n S_n \nabla J_n(u_n) \tag{2.27}$$

avec :

- s_n le pas optimal dans la direction $-S_n \nabla J_n(u_n)$, solution du problème :

$$J(u_{n+1}) = \inf_{\alpha>0} \left\{ J\left(u_n - \alpha S_n \nabla J(u_n)\right) \right\}$$

- S_n est une matrice définie positive, proche de $(F'(v))^{-1}$ que l'on peut calculer à partir d'une formule de la forme :

$$S_{n+1} = S_n + C_n$$

  C_n est une matrice de correction établie afin que S_{n+1} vérifie la relation suivante, dite *condition de quasi-Newton* :

$$S_{n+1}\left(\nabla J(u_{n+1}) - \nabla J(u_n)\right) = u_{n+1} - u_n$$

Il existe alors plusieurs choix pour la matrice S_{n+1}. Nous détaillons ici celle proposée par Davidson, Fletcher et Powell (D.F.P.) résumée par l'algorithme 3. D'autres sont présentées dans [Cul94].

Données : Point initial u_0. Matrice symétrique définie positive S_0.
Resultat : Point final u^*
tant que *Critère d'arrêt* **faire**

a) On définit s_k par :

$$J(u_k - s_k S_k \nabla J(u_k)) = \inf_{\alpha>0} J(u_k - \alpha S_k \nabla J(u_k))$$

b) On pose :

$$\begin{cases} \delta_k &= u_{k+1} - u_k; \\ \gamma_k &= \nabla J(u_{k+1}) - \nabla J(u_k). \end{cases}$$

On calcule alors :

$$\begin{cases} u_{k+1} &= u_k - s_k S_k \nabla J(u_k), \\ S_{k+1} &= S_k + \dfrac{\delta_k \delta_k^t}{\delta_k^t \gamma_k} - \dfrac{(S_k\gamma_k)(S_k\gamma_k)^t}{\gamma_k^t S_k \gamma_k}. \end{cases}$$

fin

Algorithme 3: Algorithme de quasi-Newton, D.F.P..

2.4.5 Conditions sur le pas

En pratique, le choix de la direction de descente ne suffit pas pour assurer la convergence de l'algorithme. L'application de *steepest descent* consiste à trouver le pas optimal, dans la direction de descente donnée. La recherche de ce pas implique la résolution d'un nouveau problème d'optimisation. Cette méthode est rarement utilisée car les calculs sont trop fastidieux et la solution du problème n'est généralement pas aisée à trouver. Les algorithmes de descente classiques évaluent le critère pour un certain nombre de pas distincts jusqu'à l'obtention d'un candidat convenable au regard des conditions. Une condition simple est bien sûr la réduction du critère. Cependant, cette condition n'est pas suffisante pour assurer la convergence de l'algorithme. Certaines conditions sur la décroissance du critère telle que celle d'Armijo [NW99], et sur le gradient du critère, telle que les conditions de Wolfe [NW99], doivent être vérifiées.

2.5 Optimisation sous contrainte, multiplicateurs de Lagrange

On considère maintenant un ensemble de recherche K qui n'est pas convexe. Nous alons commencer par écrire les conditions de minimalité dans ce nouveau cadre. Nous définissons l'ensemble K par des contraintes d'égalité ou des contraintes d'inégalité (ou les deux). Avant d'étudier la minimisation d'une fonction définie sur K, nous donnons la définition des *directions admissibles*.

Définition 2.5.0.1 (directions admissibles)**.** *On considère un point $v \in K$. Alors l'ensemble défini par :*

$$K(v) = \left\{ \begin{array}{l} w \in V, \quad \exists (v_n) \in K^{\mathbb{N}}, \quad \exists (\varepsilon_n) \in (\mathbb{R}_+^*)^{\mathbb{N}}, \\ \lim\limits_{n\to\infty} v_n = v, \lim\limits_{n\to\infty} \varepsilon_n = 0, \lim\limits_{n\to\infty} \dfrac{v_n - v}{\varepsilon_n} = w \end{array} \right\}$$

est appelé le cône des directions admissibles de v.

$K(v)$ représente l'ensemble des vecteurs tangents en v à une courbe contenue dans K. Ainsi, le résultat d'un déplacement infinitésimal, à partir de v, dans une direction contenue dans $K(v)$, reste dans K.

On peut alors écrire le résultat suivant :

Proposition 2.5.1. *Soit u un minimum local de J sur K. Alors, si J est différentiable en u, on a :*

$$\langle \nabla J(u),\, w \rangle \geq 0, \quad \forall w \in K(u).$$

2.5.1 Contraintes d'égalité

On suppose, sans perte de généralité, que l'ensemble K est donné par :

$$K = \{v \in V, \quad F(v) = 0\}. \tag{2.28}$$

Dans 2.28, $F(v) = (F_1(v), \ldots, F_M(v))$ est une application de V dans $\mathbb{R}^M$ où $M \geq 1$ est le nombre de contraintes d'égalité du problème. La condition nécessaire de minimalité est donnée par le théorème suivant, dont la preuve est donnée dans le livre de Grégoire Allaire [All05] :

Théorème 2.5.2. *On considère $u \in K$. On considère une application J définie sur K et dérivable en u. On suppose que les fonctions F_i ($1 \leq i \leq M$) sont continûment dérivables dans un voisinage de u. On suppose de plus que la famille $(\nabla J_i(u))$ est linéairement*

indépendante. Alors, si u est un minimum local de J, sur K, il existe $\lambda_1, \ldots, \lambda_M \in \mathbb{R}$, appelés ***multiplicateurs de Lagrange****, tels que :*

$$\nabla J(u) + \sum_{i=1}^{M} \lambda_i \nabla F_i(u) = 0 \tag{2.29}$$

Lorsque la famille de vecteurs $(\nabla F_i(u))$ est linéairement indépendante, on dit que le cas est **régulier**. Les multiplicateurs de Lagrange sont alors définis de manière unique. Dans le cas contraire, il est **non régulier** et le théorème 2.5.2 ne s'applique pas.

Dans le cas régulier, nous pouvons introduire la fonction $\mathcal{L}$ que l'on définit par :

$$\mathcal{L}(v, \mu) = J(v) + \sum_{k=1}^{M} \mu_i F_i(v) = J(v) + \langle \mu,\, F(v) \rangle \,.$$

Cette fonction est appelée **Lagrangien** du problème d'optimisation de J sur K. Si $u \in K$ est un minimum local de J sur K, le théorème 2.5.2 assure alors que dans le cas régulier, il existe $\lambda \in \mathbb{R}^M$ tel que :

$$\frac{\partial \mathcal{L}}{\partial v}(u, \lambda) = 0, \quad \frac{\partial \mathcal{L}}{\partial \mu}(u, \lambda) = 0.$$

Puisque $\dfrac{\partial \mathcal{L}}{\partial \mu}(u, \lambda) = F(u) = 0$, si $u \in K$ et $\dfrac{\partial \mathcal{L}}{\partial v}(u, \lambda) = \nabla J(u) + \lambda \nabla F(u) = 0$, d'après l'équation 2.29. Ainsi, la satisfaction de la contrainte et la condition d'optimalité s'écrivent comme l'annulation du gradient du Lagrangien. Ce qui revient à la stationnarité du Lagrangien.

La proposition suivante donne une condition d'optimalité du second ordre. La preuve de cette proposition est donnée dans [All05].

Proposition 2.5.3. *Sous les mêmes hypothèses que le théorème 2.5.2 et on suppose que les fonctions J et (F_i) sont deux fois continûment dérivables. Soit $\lambda \in \mathbb{R}^M$ le multiplicateur de Lagrange défini par le théorème 2.5.2. Alors, tout minimum local u de J sur K vérifie :*

$$\left(J^{''}(u) + \sum_{i=1}^{M} \lambda_i F^{''}(u) \right) \langle w,\, w \rangle \geq 0, \quad \forall w \in K(u) = \bigcap_{i=1}^{M} [F_i^{'}(u)]^{\perp}. \tag{2.30}$$

2.5.2 Contraintes d'inégalité

On suppose maintenant que l'ensemble K est donné par :

$$K = \{ v \in V, \quad F_i(v) \leq 0, \quad 1 \leq i \leq M \}. \tag{2.31}$$

Les fonctions F_i sont toujours définies de V dans $\mathbb{R}$. En revanche, il est plus compliqué que dans le cas précédent de déterminer le cône des directions admissibles. En effet, les contraintes définies par l'équation 2.31 ne jouent pas le même rôle selon le cas d'inégalité stricte $F_i(u) < 0$ ou nulle $F_i(u) = 0$. Si $F_i(u) < 0$, pour ε suffisamment petit, on aura aussi $F_i(u + \varepsilon w) < 0$. La contrainte est alors dite *active*. En revanche, si $F_i(u) = 0$, l'existence du vecteur $w \in V$ tel que, pour ε suffisamment petit, $u + \varepsilon w$ satisfait toutes les contraintes, n'est pas évidente. Il faut alors introduire des conditions de qualification sur les contraintes. Ces conditions garantissent que l'on peut se déplacer dans le voisinage d'un point $u \in K$, tout en respectant les contraintes, afin de garantir son optimalité dans le voisinage.

Définition 2.5.2.1. *Soit $u \in K$. L'ensemble défini par :*

$$I(u) = \{i \in \{1, \dots, M\},\ F_i(u) = 0\},$$

est appelé l'ensemble des contraintes actives en u.

Définition 2.5.2.2. *On dit que les contraintes de l'équation 2.31 sont **qualifiées** en $u \in K$ si et seulement si il existe une direction $\overline{w} \in V$ telle que l'on ait, pour tout $i \in I(u)$,*

$$\begin{cases} \text{ou bien} & \langle \nabla F_i(u), \overline{w} \rangle < 0, \\ \text{ou bien} & \langle \nabla F_i(u), \overline{w} \rangle = 0 \text{ et } F_i \text{ est affine.} \end{cases} \tag{2.32}$$

Les conditions nécessaires d'optimalité sur l'ensemble défini en 2.31 sont alors :

Théorème 2.5.4. *Soit K l'ensemble donné par 2.31, J et (F_i) sont des fonctions dérivables en u et les contraintes sont qualifiées en u. Alors, si u est un minimum local de J sur K, il existe $\lambda_1, \dots, \lambda_M \geq 0$, appelés multiplicateurs de Lagrange, tels que :*

$$\nabla J(u) + \sum_{i=1}^{M} \lambda_i \nabla F_i(u) = 0, \quad \lambda_i \geq 0, \quad \lambda_i = 0, \text{ si } F_i(u) < 0 \quad \forall i \in \{1, \dots, M\}. \tag{2.33}$$

2.5.3 Théorème de Kuhn-Tucker

On considère un problème de minimisation sous contraintes d'inégalité. L'ensemble K est donné par l'équation 2.31. Le théorème 2.5.4 donne des conditions nécessaires d'optimalité pour le problème. Nous allons maintenant voir que, sous certaines conditions sur la fonction J et les fonctions (F_i), ces conditions sont aussi suffisantes

Tout d'abord, il est nécessaire de définir un point-selle.

Définition 2.5.3.1 (Point-selle). *Soient V et Q deux espaces de Hilbert réels et U (resp P) une partie de V (resp Q). On dit que $(u,p) \in U \times P$ est un point-selle de $\mathcal{L}$ sur $U \times P$ si :*

$$\forall q \in P, \quad \mathcal{L}(u,q) \leq \mathcal{L}(u,p) \leq \mathcal{L}(v,p), \quad \forall v \in U \tag{2.34}$$

Proposition 2.5.5. *On suppose que les fonctions J, $F_1, \ldots, F_M$ sont continues sur V et que l'ensemble K est défini par 2.28 ou par 2.31. On note $P = \mathbb{R}^M$ dans le cas des contraintes d'égalité 2.28 et $P = (\mathbb{R}_+)^M$ dans le cas des contraintes d'inégalité 2.31. Soit U un ouvert de V qui contient K. Pour $(v,q) \in U \times P$, on pose $\mathcal{L}(v,q) = J(v) + qF(v)$.*

Soit (u,p) un point-selle de $\mathcal{L}$ sur $U \times P$. Alors $u \in K$ et u est un minimum global de J sur K. De plus, si J et $F_1, \ldots F_M$ sont dérivables en u, on a :

$$\nabla J(u) + \sum_{i=1}^{M} p_i \nabla F_i(u) = 0. \tag{2.35}$$

Le théorème de Kuhn-Tucker affirme que, dans le cas convexe, la condition nécessaire d'optimalité du théorème 2.5.4 est en fait une condition nécessaire et suffisante.

Théorème 2.5.6 (Kuhn-Tucker). *On suppose que les fonctions J, $F_1, \ldots, F_M$ sont convexes et continues sur V et dérivables sur l'ensemble K 2.31. Le Lagrangien $\mathcal{L}$ associé s'écrit alors :*

$$\mathcal{L}(v,q) = J(v) + \langle q,\, F(v) \rangle, \quad \forall (v,q) \in V \times (\mathbb{R}_+)^M.$$

Soit $u \in K$ un point où les contraintes sont qualifiées (définition 2.5.2.2). Alors u est un minimum global de J sur K si et seulement si il existe $p \in (\mathbb{R}_+)^M$ tel que (u,p) soit un point-selle du Lagrangien $\mathcal{L}$ sur $V \times (\mathbb{R}_+)^M$ ou, de manière équivalente, tel que :

$$F(u) \leq 0, \quad p \geq 0, \quad \langle p,\, F(u) \rangle, \quad \nabla J(u) + \sum_{i=1}^{M} p_i \nabla F_i(u) = 0. \tag{2.36}$$

2.6 Bilan

Nous venons de voir dans ce chapitre les principales méthodes d'optimisation mono-objectif basées sur l'utilisation des gradients. Comme nous l'avons montré, une fois le gradient du critère évalué, il existe plusieurs choix de direction. Les méthodes d'optimisation qui suivent ce schéma sont à la fois rapides et performantes si l'on souhaite minimiser une fonction régulière et convexe. De plus, ce sont, pour la majorité, des méthodes déterministes. Une fois que la direction de descente est choisie, la recherche

du pas peut se révéler difficile. Il s'agit en effet de résoudre un nouveau problème d'optimisation. Si la fonction que l'on cherche à minimiser est de classe au moins $\mathcal{C}^2$, convexe et que la matrice hessienne associée est accessible, la convergence vers le minumum est assurée. Malheureusement, ces conditions d'application ne sont valables que pour une infime partie des problèmes d'optimisation que nous devons traiter.

Le principal inconvénient des algorithmes de type gradient concerne les minima locaux. En effet, les méthodes que nous venons d'étudier tendent vers le point stationnaire dont la direction est donnée par la pente. Aussi, l'optimisation de critères dont le graphe comporte plusieurs minima locaux (problème multimodal), n'est pas réalisable. De plus, ces méthodes ne sont pas adaptées au traitement des problèmes multiobjectif.

Chapitre 3

Méthodes évolutionnaires en optimisation multicritère

Après avoir introduit le vocabulaire et les principaux algorithmes d'optimisation unicritère de descente, nous abordons dans ce chapitre la notion d'optimisation multiobjectif [DS03]. Il s'agit ici de résoudre un problème d'optimisation dans lequel interviennent plusieurs fonctions dites *objectifs*, simultanément. Nous présentons une branche importante des algorithmes d'optimisation, les méthodes qui utilisent une stratégie d'évolution (ES). Parmi ces méthodes, on passe en revue les plus usitées et nous mettons en évidence leurs principales caractéristiques.

3.1 Optimisation multiobjectif

Dans le chapitre précédent, nous abordons la résolution de problèmes d'optimisation unicritères, par des méthodes basées sur les gradients de descente. De manière plus générale, nous étudions ici les problèmes d'optimisation soumis à plusieurs critères. En effet, dans un contexte pré-industriel, on est confronté à des systèmes qui doivent répondre à plusieurs phénomènes. Ces derniers peuvent être de natures diverses, telles que aérodynamique, de structure, magnétiques, etc. On parle alors d'optimisation multicritère ou multiobjectif. Alors qu'il est aisé de comparer deux points de l'espace de recherche par rapport à un critère unique, cela n'est pas évident lorsque l'on en considère plusieurs. Il faut alors définir un opérateur de comparaison dans un espace multiobjectif.

Nous considérons ici un problème d'optimisation arbitraire avec n objectifs $(J_i)_{1\leq i\leq n}$ qui sont, sans perte de généralité, à minimiser. De plus, nous supposons que tous les critères sont de même importance. Nous supposons que l'espace admissible des N paramètres est un sous-espace de $\mathbb{R}^N$. Nous notons K cet espace. Nous notons $\mathbb{J}$ l'opérateur

défini de $\mathbb{R}^N$ dans $\mathbb{R}^n$ par :

$$\mathbb{J} : \begin{array}{ccc} \mathbb{R}^N & \rightarrow & \mathbb{R}^n \\ Y = (y_1, \ldots, y_N) & \mapsto & \mathbb{J}(Y) = (J_1(Y), \ldots, J_n(Y)) \end{array}$$

Comme illustré sur la figure 3.1, chaque point Y de l'espace admissible K possède une image et une seule dans l'espace des critères. Notons que la dimension de l'espace de recherche est généralement différente du nombre de critères.

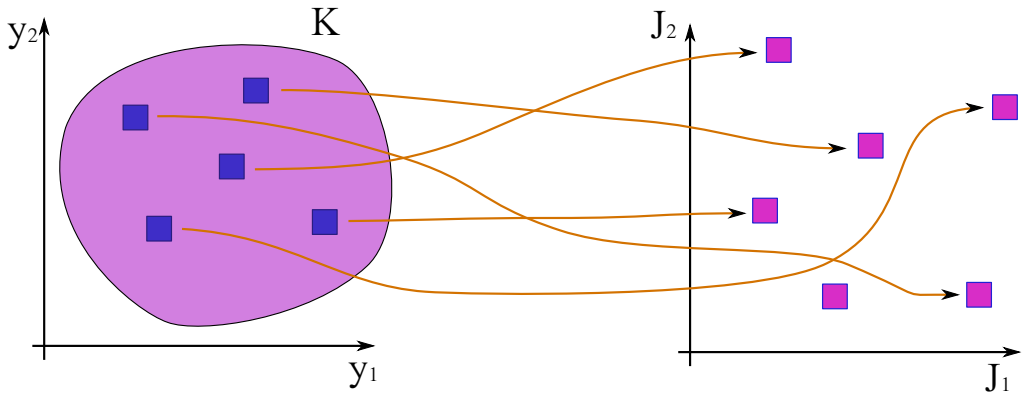


FIGURE 3.1: A gauche l'espace des paramètres, à droite l'espace des critères. A chaque point $Y = (y_1, \ldots, y_N)$ de K est associé un point image $\mathbb{J}(Y) = (J_1(Y), \ldots, J_n(Y))$ de l'espace des critères.

On considère 2 points Y_1 et Y_2 dans $\mathbb{R}^N$ et leurs images respectives par la fonction $J : \mathbb{R}^N \rightarrow \mathbb{R}$, $J(Y_1)$ et $J(Y_2)$. Si le problème d'optimisation était de minimiser J seulement, il serait simple de comparer les points Y_1 et Y_2 par rapport à J. En effet, Y_1 serait plus efficace que Y_2 si et seulement si $J(Y_1) < J(Y_2)$.

Dans le cas où l'on considère plusieurs critères simultanément, $\mathbb{J} : \mathbb{R}^N \rightarrow \mathbb{R}^n$, $n > 1$, la comparaison entre les images de Y_1 et de Y_2, deux éléments de $\mathbb{R}^N$, par $\mathbb{J}$ se complique. Comment comparer deux points par rapport à plusieurs objectifs ? La définition de la dominance de Pareto permet de répondre à cette question. Le point de conception $\mathbb{J}(Y_1)$ *domine en efficacité* le point $\mathbb{J}(Y_2)$ si et seulement si toutes les valeurs de critères y sont inférieures ou égales et si au moins une inégalité est stricte [Mie99, CS02]. On note alors $\mathbb{J}(Y_1) \succ \mathbb{J}(Y_2)$. Si l'on considère un point $Y \in K$, réalisable, on peut alors définir des sous-espaces particuliers de l'espace des critères, relativement à $\mathbb{J}(Y)$. Ces sous espaces contiennent respectivement les points qui dominent $\mathbb{J}(Y)$, les points dominés par $\mathbb{J}(Y)$ et les points qui ne peuvent être comparés à $\mathbb{J}(Y)$. Ceci est résumé par la figure 3.2.

Si $\mathbb{J}(Y_1) \succ \mathbb{J}(Y_2)$, on peut dire que la solution Y_1 est meilleure que la solution Y_2,

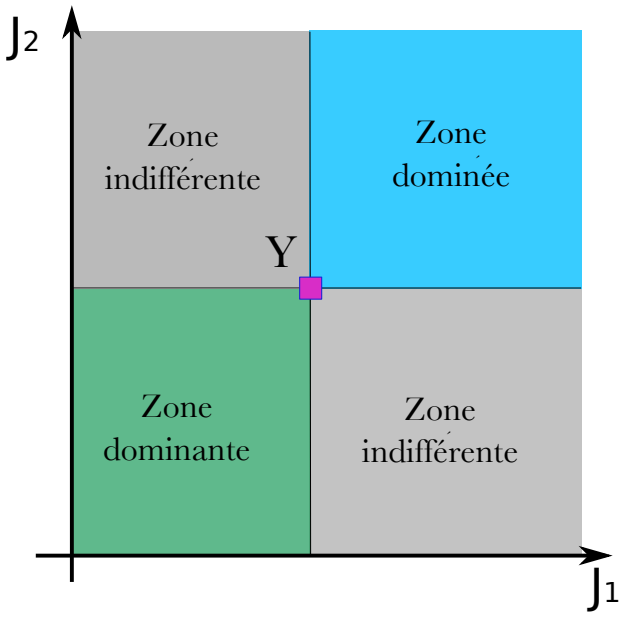


FIGURE 3.2: Description de la dominance au sens de Pareto par rapport à l'image par $\mathbb{J}$ d'un point Y [CS02].

pour le problème :

$$\min_{Y \in K} \mathbb{J}(Y)$$

A partir de la dominance de Pareto, nous pouvons définir l'ensemble des points solution du problème de minimisation. On dira qu'un point Y_1 est optimal s'il n'est dominé par aucun autre point de l'espace admissible K. L'ensemble des points de $\mathbb{J}(K)$ *non dominés* est appelé **front de Pareto** du problème. L'ensemble des points de K dont l'ensemble image représente le front de Pareto est appelé **ensemble de Pareto**. Ceci est illustré par la figure 3.3. Résoudre un problème d'optimisation multiobjectif revient donc à chercher l'**ensemble de Pareto**.

Une famille d'algorithmes d'optimisation, basée sur la théorie de l'évolution des espèces de Charles Darwin, permet, dans certains cas de trouver cet ensemble. Nous allons introduire les bases de fonctionnement des ces algorithmes et présenter succinctement les plus connus d'entre eux.

3.2 Principe des méthodes évolutionnaires

Pour surmonter les diffiultés inhérentes aux algorithmes de descente, notamment dans le contecte de problèmes multi-modaux ou multiobjectif, des approches alterna-

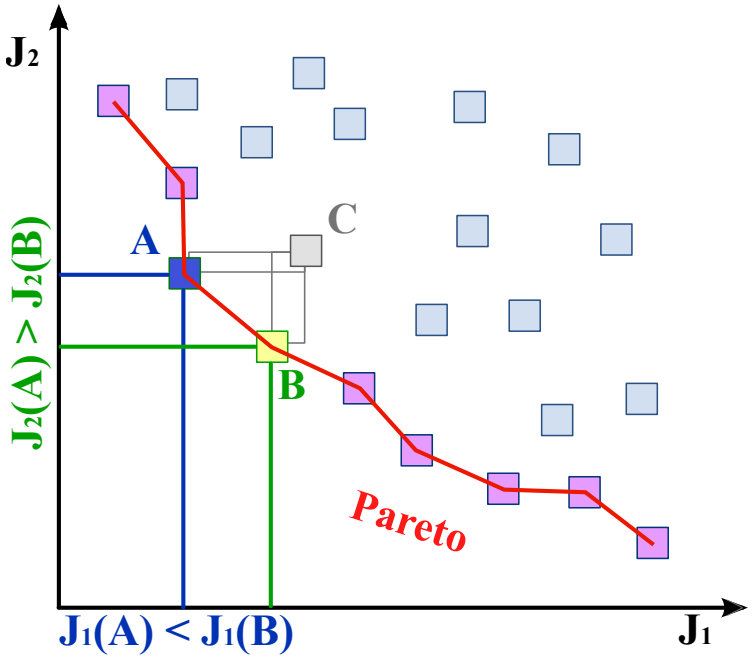


Figure 3.3: Dominance de Pareto pour deux objectifs J_1 et J_2

tives ont été développées, de nature semi-stochastiques et souvent qualifiées de meta-heuristiques. Ces m/'ethodes se caractérisent généralement par un taux de convergence faible, mais ont démontré leur capacité à réaliser une optimisation globale, pour une large variété de problèmes. A ce titre, ces méthodes apparaissent comme complémentaires aux méthodes de descente et sont souvent utilisées dans un contexte industriel à l'exploration de l'espace de conception.

Les approches méta-heuristiques sont souvent d'inspiration biollogique. Les plus connus de ces algorithmes génétiques proposés par Holland [Hol75] et popularisées par Goldberg [Gol89], qui s'appuient sur une analogie avec les théories de l'évolution et cherchent la solution par combinaison de blocs (codage des variables d'optimisation). Les stratégies d'évolution [Mic92] peuvent être considérées comme une généralisation, qui repose sur des distributions de variables aléatoires réelles plutôt que sur des combinaisons pour définir la progression de l'algorithme. Plus récemment, des méthodes inspirées des théories de l'intelligence collective ont vu le jour. Par exemple, l'optimisation par colonies de fourmis [Mic92] ou l'optimisation par essains de particules [**?**]. Ces approches se caractérisent par l'utilisation d'un ensemble d'agents qui communiquent pour concilier recherche locale et exploration.

Les Méthodes Evolutionnaires sont des algorithmes stochastiques d'optimisation globale qui s'inspirent de la théorie d'évolution des populations biologiques de Charles Dar-

win [Dar69]. Celle ci tend à produire des organismes plus adaptés à leur environnement que la génération précédente. Les Algorithmes Génétiques (AGs) sont les plus connus d'entres eux. Ils ont été introduits dans les années 1970 par le professeur John Holland de l'Université du Michigan [Hol92]. Ils constituent une grande famille d'algorithmes reconnus pour leur simplicité, leur généralité et leur robustesse. De plus, les méthodes génétiques peuvent facilement être adaptées au calcul parallèle. Cette particularité est primordiale car l'utilisation de ressources distribuées devient rapidement indispensable. En effet, la quantité de calculs à effectuer croît rapidement en fonction de la dimension de l'espace de recherche. Il en est de même lorsque l'évaluation des critères est coûteuse en temps de calcul. Ils sont généralement préférés aux autres algorithmes dans la résolution de problèmes multi-modaux ou bruités, pour leur robustesse.

3.2.1 Introduction

La recherche de l'ensemble de Pareto est très coûteuse en termes de temps de calcul et même irréalisable, la plupart du temps. A cause de la complexité des critères à minimiser, les méthodes exactes sont rarement applicables. Pour cette raison, les algorithmes évolutionnaires, la recherche avec tabous [FGM97], le recuit simulé [SKV83] ont été développés. Aucun de ces algorithmes ne garantit de trouver la solution exacte du problème, mais chacun essaie de s'en approcher. Par analogie avec l'évolution naturelle, chaque point $Y = (y_1, \ldots, y_N)$ de K est appelé *individu*. Un ensemble d'individus de K est appelé *population*. La valeur des critères obtenue pour un individu Y, $\mathbb{J}(Y)$ nous permet de comparer Y par rapport aux autres solutions. La population considérée à l'itération k de l'algorithme est appelée k-ième *génération*.

Les algorithmes génétiques (AGs) débutent par la création d'une population initiale, échantillon de l'espace de recherche K. Celle-ci est généralement créée aléatoirement, sur le domaine admissible K. Nous pouvons citer notamment l'équi-répartition et le LHS (*Latin Hypercube Sampling*) introduit par McKay [MMC00] en 1979. L'objectif est d'avoir un échantillon initial représentatif de l'espace de recherche K. Après évaluation de cette population par les critères, l'algorithme commence son évolution. Nous pouvons alors écrire l'algorithme 4.

Avant d'expliquer chaque étape de l'algorithme, nous précisons que la taille de la population reste constante à chaque génération. On note N_p la taille de la population à chaque génération. Nous pouvons expliquer ces trois étapes de la manière suivante :

- Sélection : On choisit $N_{sel} < N_p$ individus de la population considérée qui doivent *survivre*. La sélection est principalement basée sur la valeur des critères obtenue

Données : Population initiale évaluée
Resultat : Population finale évaluée
tant que *Critère d'arrêt* **faire**
  a) Sélection ;
  b) Croisement ;
  c) Mutation.
fin

Algorithme 4: Algorithme d'optimisation génétique (AGs).

par chaque individu. Après la phase de sélection, la taille de la population a diminué.

- Croisement : On ne considère alors que les N_{sel} individus sélectionnés au cours de l'étape précédente. A partir de ces individus, on accroît la population jusqu'à sa taille normale N_p. Pour cela, on effectue des croisements entre individus dits *parents* pour créer des individus *enfants*. Les variables $(y_1, y_2, \ldots, y_N)$ qui définissent chaque individu sont alors codées, par exemple en binaire.
- Mutation : On introduit une perturbation aléatoire chez certains individus enfants. Cette opération introduit de nouvelles caractéristiques génétiques dans la population. On utilise le code *génétique* de chaque individu pour introduire aléatoirement des changements.

La figure 3.4 représente un schéma de principe des algorithmes génétiques. Elle illustre l'évolution de la population étape par étape.

Les nouveaux individus sont alors évalués par les critères et l'algorithme se poursuit avec une nouvelle génération. Nous allons maintenant voir un peu plus en détail chacune des étapes importantes de l'optimisation par algorithme génétique. Avant cela, nous devons expliquer une étape primordiale dans la mise en place d'algorithmes génétiques, le codage. Nous ne présentons ici que le codage binaire, mais il en existe d'autres types, comme le codage sous forme d'arbres ou le codage à caractères multiples.

3.2.2 Le codage

Les AGs se distinguent des autres algorithmes évolutionnaires par le codage des paramètres de l'espace de recherche. Cette étape décrit l'individu par un code qui représente son *génome*. L'individu est alors identifié par un *chromosome*, lui-même composé de *gènes*.

$$Y = \underbrace{(y_1, y_2, \ldots, y_N)}_{\text{individu}} \in K \longmapsto \underbrace{(\overbrace{a_1}^{\text{gène}}, a_2, \ldots, a_l)}_{\text{chromosome}}. \tag{3.1}$$

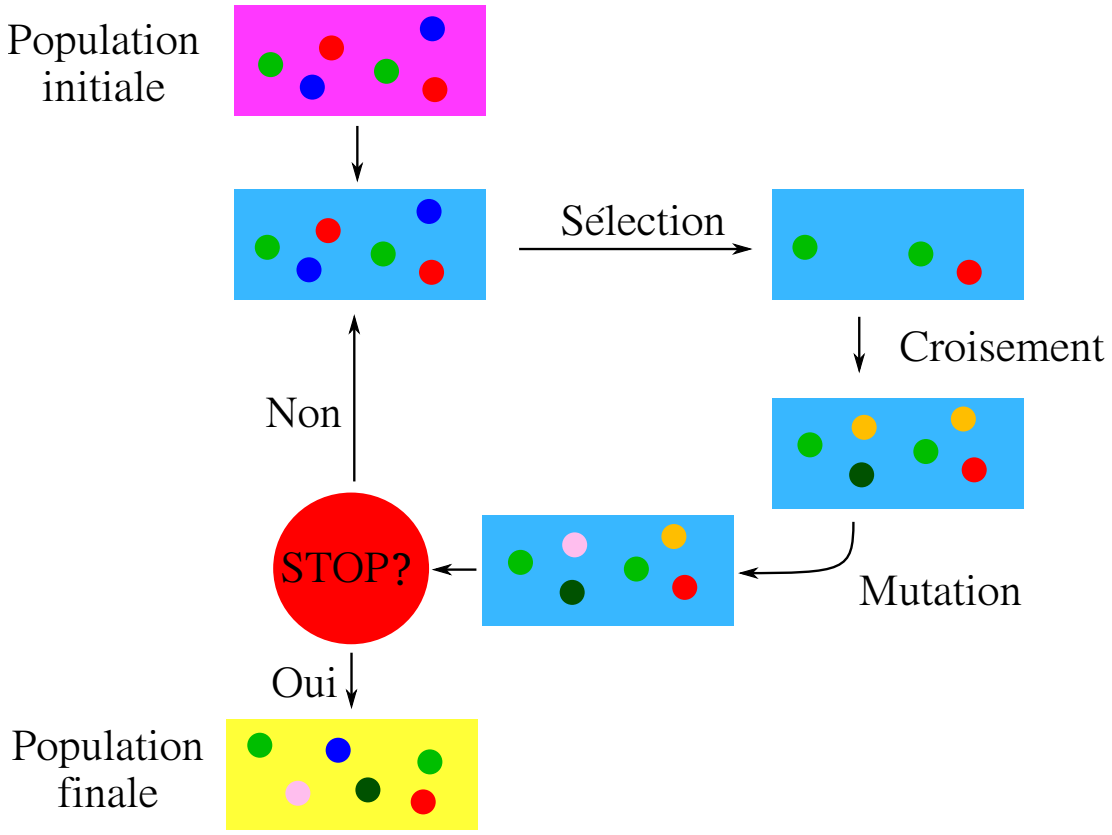


FIGURE 3.4: Schéma de fonctionnement d'un algorithme génétique.

Les *gènes* a_i appartiennent à un *alphabet* choisi pour le codage. Par exemple, pour le codage binaire, $a_i \in \{0, 1\}$. Ce cas particulier de codage est illustré par la figure 3.5.

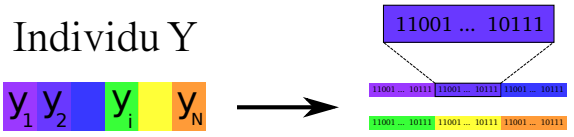


FIGURE 3.5: Codage d'un individu $Y \in K$ en binaire.

Les (a_i) sont organisés de telle manière qu'ils décrivent, par groupes successifs, chacune des variables. Les gènes $(a_1, \ldots a_{l_1})$ décrivent la première composante y_1 du vecteur $Y \in \mathbb{R}^N$, $(a_{l_1+1}, \ldots a_{l_1+l_2})$ la seconde, et ainsi de suite jusqu'à y_N.

3.2.3 La sélection

L'objectif de la sélection est de conserver les individus qui présentent de bonnes caractéristiques génétiques. Ainsi, nous pouvons espérer créer des individus meilleurs au fil des générations, sans régression. Naturellement, nous serions tentés de conserver les N_{sel} meilleurs individus et d'éliminer les $N_p - N_{sel}$ autres. Cette opération peut être contre productive car nous souhaitons conserver, pour les générations futures, les meilleures caractéristiques et non nécessairement les meilleurs individus. En effet, un individu peut posséder une partie du génome de la solution optimale tout en obtenant de mauvaises valeurs par rapport aux critères. Malgré tout, il est important de garder cet individu. Cependant, la probabilité de trouver de bonnes caractéristiques est plus élevée chez les individus qui obtiennent de bons résultats par rapport aux critères. Il faut alors pondérer le choix des individus en fonction de la probabilité que l'individu possède de bons gènes. Dans la théorie de l'évolution des espèces, cette pondération est appelée *pression de sélection*. Le dosage de cette pression est crucial. En effet, une pression trop forte ne permet de sélectionner que les meilleurs individus, au détriment de la diversité. L'algorithme risque alors de converger trop rapidement et la solution proposée n'est que partielle. Au contraire, une pression trop faible risque de ralentir l'évolution de la population. La performance de l'algorithme est alors réduite.

Parmi les choix existants de sélection, nous pouvons citer la *sélection par roulette* décrite par Bäck dans [BÖ6], *la sélection par rang*, la *sélection par tournoi* ou encore *l'élitisme*, qui sera traité ultérieurement.

3.2.3.1 Sélection par roulette

C'est la méthode la plus connue des sélections stochastiques. On attribue une probabilité de sélection á chaque individu. Cette probabilité est évaluée en fonction de la valeur obtenue par les critères au point considéré. Ainsi, un individu considéré comme meilleur qu'un autre par rapport aux critères obtient une probabilité supérieure d'être conservé. Comme illustré par la figure 3.6, plus la surface sur laquelle se trouve l'individu est grande, plus la probabilité qu'il soit sélectionné est élevée. On organise ensuite un tirage aléatoire, par exemple sur $[0, 2\pi[$ et on choisit l'individu dont l'angle correspond. Le procédé est alors répété jusqu'à ce que l'on obtienne les N_p individus cherchés. Cependant, cette méthode souffre de nombreux inconvénients. En effet, si l'écart entre les valeurs obtenues par les critères en deux points est important, le meilleur point est largement favorisé. Cela conduit à une convergence prématurée de l'algorithme. Afin de pallier ce défaut, certains ouvrages [DG89] proposent une mise à l'échelle des critères.

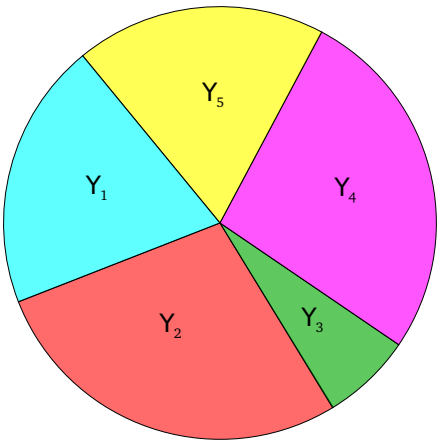


Figure 3.6: Sélection des individus par roulette

Mais cette technique ne se révèle pas toujours efficace. C'est pourquoi, elle est très peu utilisée aujourd'hui.

3.2.3.2 Sélection par rang

La sélection par rang est une méthode de sélection qui se décompose en deux étapes. Tout d'abord, les individus doivent être rangés par ordre croissant (décroissant si l'on maximise les critères) de leur performance. On obtient alors des classes d'équivalences de points, selon l'ordre du front auquel les solutions appartiennent. L'organisation de ces fronts est illustré par la figure 3.7. Ensuite, une sélection similaire à la sélection par

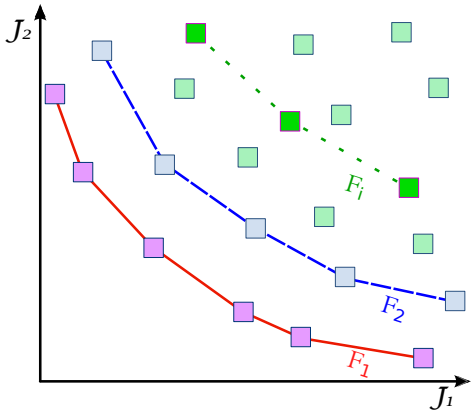


Figure 3.7: Classement des solutions par rang de fronts

roulette est utilisée. Cette procédure permet d'attribeur une probabilité de sélection en fonction du rang. La probabilité de sélection est alors proportionnelle à la valeur des critères obtenue par la solution, dégradée en fonction du rang auquel elle appartient.

L'avantage de cette méthode est qu'elle n'est pas soumise au problème de mise à l'échelle de la sélection par roulette. En effet, la probabilité de sélection ne dépend que de la variante utilisée pour le calcul de la performance des points.

3.2.3.3 Sélection par tournoi

Cette méthode utilise la comparaison entre individus, comme le fait la sélection par rang. En revanche, aucun tri n'est à effectuer dans la population. Le principe est le suivant :

- on sélectionne aléatoirement un échantillon de I ($I \leq N_p$) individus ;
- on conserve le meilleur individu parmi l'échantillon.

En cas de litige (si plusieurs individus appartiennent au même rang), plusieurs issues sont possibles. On peut alors dégrader la performance en fonction de la densité de population de chaque individu ou effectuer des tirages aléatoires.

3.2.4 Les croisements

On considère ici les N_{sel} individus sélectionnés à la première étape de l'algorithme 4, que nous appelons *parents*. Après avoir classé aléatoirement les parents par paires, l'algorithme opère des croisements entre eux. Pour cela, on considère chaque gène qui compose les chromosomes d'une paire d'individu. Chaque gène est séparé du suivant en un point dit de *coupe*. Il est alors possible d'échanger deux gènes qui occupent la même position dans le chromosome de chaque individu de la paire. La figure 3.8 illustre le fonctionnement de divers opérateurs de croisement entre deux chromosomes, en un ou deux points de coupe et le croisement uniforme. Cet échange a lieu selon une probabilité fixée par l'utilisateur.

Une probabilité d'échange élevée favorise le brassage d'informations mais augmente le risque de pertes des bons individus.

3.2.5 Les mutations

L'étape de mutation sert à introduire de nouveaux gènes dans la population. Pour cela, une partie des gènes des individus enfants est modifiée avec une probabilité p_m. La figure 3.9 illustre la mutation d'un gène chez l'enfant 1.

La mutation permet à l'algorithme d'explorer l'espace de recherche K aléatoirement. Cependant, la probabilité de mutation doit rester faible pour ne pas transformer l'optimisation en une simple recherche aléatoire.

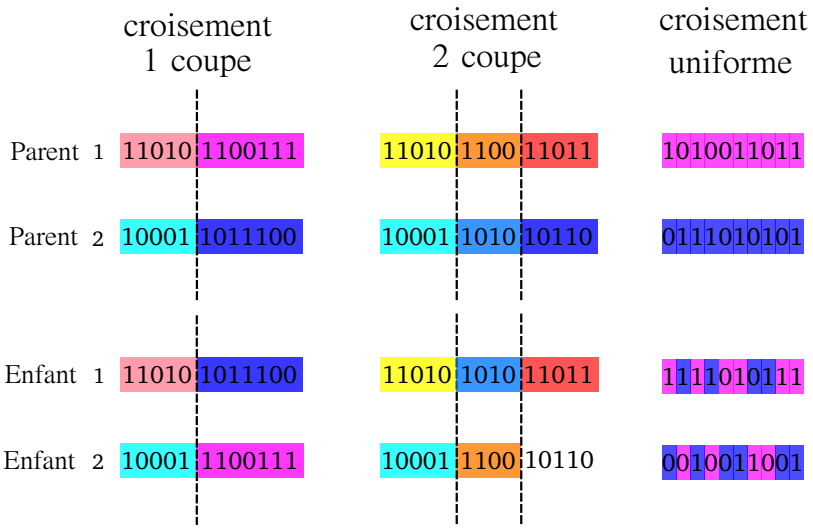


Figure 3.8: Croisement entre deux *parents* pour créer 2 nouveaux individus *enfants*. Les lignes en pointillés représentent les points de coupe.

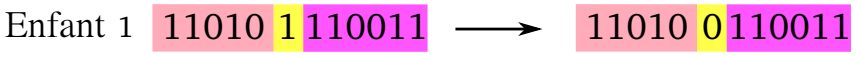


Figure 3.9: Mutation d'un gène d'un individu *enfant*

.

3.3 Optimisation multiobjectif par méthodes évolutionnaires

Nous présentons ici les principaux algorithmes de type évolutionnaire dont les caractéristiques sont résumées dans le tableau 3.1. Comme nous pouvons le constater, malgré un principe général de construction, il existe beaucoup de variantes. Chaque algorithme présenté dans le tableau 3.1 se distingue des autres par l'utilisation de techniques propres. Nous retrouvons cependant des propriétés communes à plusieurs méthodes. Nous résumons dans ce qui suit les plus courantes. Nous les divisons alors en plusieurs catégorie afin de mettre en évidence les avantages et les inconvénients de chacune. Ainsi, nous mettons en évidence la notion de performance d'un individu, les mécanismes qui maintiennent la diversité au sein de la population, la notion d'élitisme, utilisée dans certaines méthodes. Nous résumons aussi les avantages et inconvénients des méthodes évolutionnaires les plus connues. Cette méthode utilise l'élitisme développé dans sa version mono-objectif, CMA-ES [HK04] et un schéma de sélection de

Algorithme	Utilisation de la performance	Mécanisme de diversité	Elitisme	Population extérieure	Avantages	Inconvénients
VEGA [Sch85]	Chaque sous population est comparée par rapport à chaque objectif	Non	Non	Non	Premier MOGA [1] Implémentation simple	Tend à converger vers les extremum de chaque objectif
MOGA [FF93]	Classement par fronts	Répartition par niches	Non	Non	Extension simple d'AG mono-objectif	Convergence lente Problèmes relatifs à la taille des niches
WGBA [HL92]	Moyenne pondérée des objectifs normalisés	niches ; poids prédéfinis	Non	Non	Extension simple d'AG mono-objectif	Difficultés dans les espaces non convexes
NPGA [HNG94]	Sélection par tournoi	Sélection par tournoi	Non	Non	Procédé de sélection simple par tournoi	Problèmes liés aux paramètres de taille des niches Paramètres supplémentaires pour la sélection par tournoi
RWGA [MI95]	Moyenne pondérée des objectifs normalisés	Attribution aléatoire des poids	Oui	Oui	Efficace et facile à implémenter	Difficultés dans les espaces non convexes
PESA [DCO00]	Aucun	Basé sur la densité de cellules	Elitisme pur	Oui	Facile à implémenter et efficace	La performance dépend de la taille des cellules Une information a priori sur l'espace objectif est nécessaire
PAES [KC00]	Remplacement des parents par les enfants basé sur la dominance de Pareto	Basé sur la densité de cellules Tournoi parents/enfants	Oui	Oui	Mutations aléatoires Facile à implémenter Efficace Convergence rapide	La performance dépend de la taille des cellules La population n'est pas constante
NSGA [SD94]	Classement basé sur l'ensemble non dominé	Partage de performance	Non	Non	Convergence rapide	Problèmes liés à la taille des niches
NSGA-II [KDM00]	Classement basé sur l'ensemble non dominé	Distance avec le voisinage	Oui	Non	Un seul paramètre Beaucoup testé Efficace	La distance avec le voisinage ne fonctionne que dans l'espace des critères
SPEA [EZ99]	Classement basé sur une archive de points non dominés	Regroupement pour tronquer la population extérieure	Oui	Oui	Beaucoup testé Aucun paramètre pour le regroupement	Complexité de l'algorithme de regroupement
SPEA-2 [ZLT02]	Renforcement des points dominants	Classement de voisinage	Oui	Oui	Les meilleurs points sont préservés	Beaucoup de calculs pour la performance et la densité
RDGA [LY03]	Limitation du problème à deux objectifs Utilisation du classement par rang et densité des objectifs	Densité de cellules et régions interdites	Oui	Oui	Robuste	Plus difficile à implémenter que les autres
DMOEA [YL03]	Note basée sur les cellules	Densité de cellules adaptatif	Oui (implicite)	Non	Techniques efficaces d'actualisation de calcul de densité des cellules. Approche adaptative	Plus difficile à implémenter que les autres

Table 3.1: Tableau récapitulatif des AGs les plus connus et de leurs caractéristiques

type $(1 + \lambda)$ des algorithmes évolutionnaires [BS02]. Certaines de ses caractéristiques très particulières en font un cas à part.

3.3.1 Performance

Afin d'orienter efficacement les recherches de nouveaux points, certains algorithmes dégradent la valeur des critères des points étudiés. Nous pouvons alors parler de *perfor-*

mance d'individu. Les trois principales méthodes d'évaluation de performance sont les suivantes :

- Partage de performance (*fitness sharing*) ;
- Distance d'encombrement (*crowding distance*) ;
- Par densité de cellule (*cell based density*).

Ici, nous résumons chacune de ces méthodes par des illustrations simples. De plus, nous donnons, pour chaque exemple de calcul de performance, un algorithme de type évolutionnaire du tableau 3.1 qui l'utilise.

3.3.1.1 Partage de performance

Le partage de performance favorise la recherche des zones inexplorées du front de Pareto en pénalisant des solutions situées dans les parties à haute densité de population. Pour obtenir ce résultat, les zones denses sont identifiées et une méthode de pénalisation locale y est appliquée.

Cette idée fut initiée par Goldberg et Richardson en 1987 dans [GR87] dans le but de chercher les optima locaux de fonctions multimodales. Fonseca et Fleming [FF93] ont utilisé cette idée pour pénaliser les solutions concentrées de même rang, comme détaillé dans l'algorithme 5. Dans la procédure 5, σ définit un voisinage de chaque

Données : $\mathbb{J}(Y_i), Y_i \in P$
Resultat : $p(Y_i), Y_i \in P$
tant que *Critère d'arrêt* **faire**
  Step 1 : **pour tous les** $(Y_i, Y_j) \in P^2$, $i < j$ **faire**

$$d\mathbb{J}(Y_i, Y_j) = \sqrt{\sum_{k=1}^{n} \left(\frac{J_k(Y_i) - J_k(Y_j)}{J_k^{max} - J_k^{min}} \right)^2}$$

  fin
  Step 2 : **pour tous les** $Y_i \in P$ **faire**

$$nc(Y_i) = \sum_{X \in P,\, r(X)=r(Y)} \max \left\{ \frac{\sigma - d\mathbb{J}(Y_i, X)}{\sigma}, 0 \right\} \tag{3.2}$$

  fin
  Step 3 : **pour tous les** $Y_i \in P$ **faire**

$$p(Y) = \frac{\mathbb{J}(Y_i)}{nc(Y_i)}$$

  fin
fin

Algorithme 5: Procédure d'évaluation des performances p des solutions en fonction des critères $\mathbb{J}$ et de la densité des solutions.

solution, dans l'espace des critères. Ceci est illustré par la figure 3.10. Les solutions

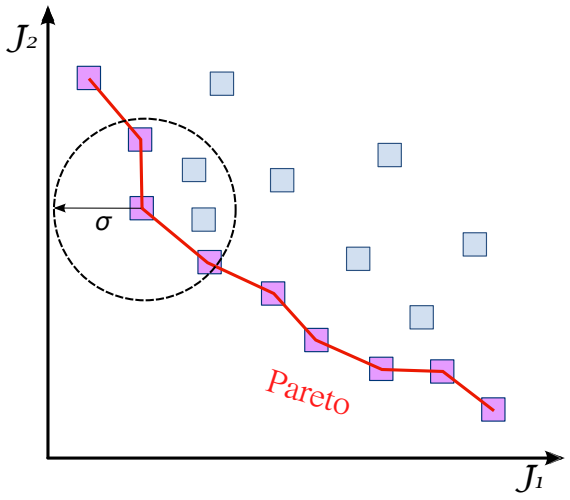


FIGURE 3.10: Définition de niches de solutions basée sur la distance entre solutions dans l'espace des critères

contenues dans un même voisinage sont comptabilisées dans la fonction "niche". Ainsi, une solution contenue dans un voisinage dense aura une probabilité plus faible d'être choisie comme parent. Cette méthode limite donc la prolifération de solutions dans une zone particulière de l'espace des critères.

Une alternative est d'utiliser la distance entre deux solutions Y et Y', dans l'espace des paramètres :

$$dy(Y,Y') = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - y_i')^2}, \quad Y = (y_1,\ldots,y_N), \quad Y' = (y_1',\ldots,y_N')$$

Si les critères sont suffisamment réguliers, deux solutions doivent être proches dans l'espace des critères si elles sont proches dans l'espace des paramètres. Cependant, la réciproque est fausse. Le calcul de performance basé sur les distances dans l'espace des critères devrait réduire la diversité dans l'espace des paramètres. Malgré tout, Deb et Goldberg ont vérifié dans [DG89] que le calcul de performance basé sur une distance évaluée dans l'espace des paramètres est moins efficace. L'inconvénient majeur de cette méthode est l'introduction d'un nouveau paramètre σ que l'utilisateur doit fixer. Ce problème est abordé par Deb et Goldberg [DG89], ainsi que par Fonseca et Flemming [FF93]. Une approche systématique est proposée pour estimer et mettre à jour dynamiquement le paramètre σ. L'autre inconvénient de cette méthode est le surcoût imposé par le calcul de la fonction niche.

3.3.1.2 Distance d'encombrement

L'approche de la distance d'encombrement (*crowding distance* en anglais) vise à obtenir un échantillon de solutions uniformément réparties sur le front de Pareto sans utiliser de paramètre σ, contrairement à la méthode de création de niches. Ici, on attribut une performance à chaque individu en fonction de celle obtenue par ses voisins de même front. La notion d'organisation de fronts par rang est illustrée par la figure 3.7. La figure 3.11 illustre cette méthode. Nous détaillons la procédure de calcul de distance

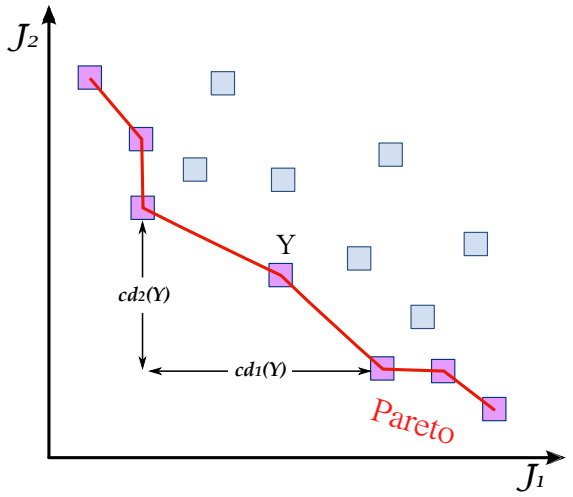


Figure 3.11: Méthode de création de niches sans paramètre.

d'encombrement dans l'algorithme 6. Le principal avantage de cette approche est que la mesure de densité de population autour d'un individu est évaluée sans que l'utilisateur n'ait besoin de définir de paramètre σ ou le λ-ième plus proche voisin. Dans NSGA-II [KDM00], cette distance est utilisée pour définir un système de tournoi entre individus. Pour deux individus sélectionnés aléatoirement, si leur rang est égal, le gagnant est celui dont la distance d'encombrement est la plus grande.

3.3.1.3 Densité de cellules

Le calcul de la performance basé sur la densité de cellules nécessite la subdivision de l'espace des critères en cellules de tailles égales. Pour cela, on se place dans l'espace des critères normalisés. Ceci est illutré par la figure 3.12. Le nombre de solutions contenues dans chaque cellule en définit la densité. La cellule considérée est un hypercube. Si l'on considère 2 critères, ce sera un carré, 3 critères, un cube, etc. La densité d'une solution est égale à la densité de la cellule dans laquelle elle se trouve. Cette information de densité est utilisée pour introduire de la variété dans la population. En effet, comme

Données : P, N_P
Resultat : P, $cd(P)$
On identifie les fronts $F_1, F_2, \ldots, F_j$ de P (cf figure 3.7);
pour $k = 1$ **à** j **faire**
    On organise les individus de F_k selon $J_1(Y)$, $Y \in F_k$ croissant;
    $l = |F_k|$, $F_k = (Y_1, \ldots, Y_l)$;
    On pose : $cd(Y_1) = cd(Y_l) = +\infty$;
    pour $i = 2$ **à** $l - 1$ **faire**
        $$cd(Y_i) = \sum_{q=1}^{N} \frac{J_p(Y_{i+1}) - J_p(Y_{i-1})}{J_p^{max} - J_p^{min}};$$
        où $J_p^{max} = \max_{Y \in P} J_p(Y)$;
    fin
fin

Algorithme 6: Procédure de calcul de la distance d'encombrement d'un individu par rapport à la population.

pour les méthodes d'attribution de performance que nous avons présentées, l'information apportée par le voisinage d'une solution, dans l'espace des critères, nous renseigne sur la concentration des solutions dans une zone. Ainsi, nous pouvons utiliser les performances des individus pour réaliser des tournois, si le critère de dominance n'est pas utilisable.

Comme pour la méthode de partage de performance, la méthode de densité de cellules nécessite que l'utilisateur fixe un paramètre. En effet, la taille des cellules doit être définie dans une étape initiale. Par exemple, dans PESA [DCO00], entre deux solutions, celle de plus faible densité est préférée. La procédure de PESA est décrite par l'algorithme 7.

3.3.2 Elitisme

Dans un contexte mono-objectif, l'utilisation de l'élitisme se traduit par le maintien de la meilleure solution au fil des générations. De la même manière, dans le contexte multi-objectif, l'ensemble des solutions non dominées sont considérées comme les élites de la génération. Ces solutions sont alors conservées d'une génération à l'autre, tant qu'elles occupent ce statut. Cependant, la mise en place de l'élitisme est alors plus complexe que dans le cadre mono-objectif. A cause de la multitude de solutions non-dominées, beaucoup d'algorithmes génétiques ne l'utilisent pas. Plusieurs études [EZ99], [Deb01], [VL00] montrent que les algorithmes génétiques qui utilisent l'élitisme sont plus performants que les autres. L'élitisme peut être implémenté de deux manières [Jen03] :

1. Conserver les solutions *Elites* de la population ;

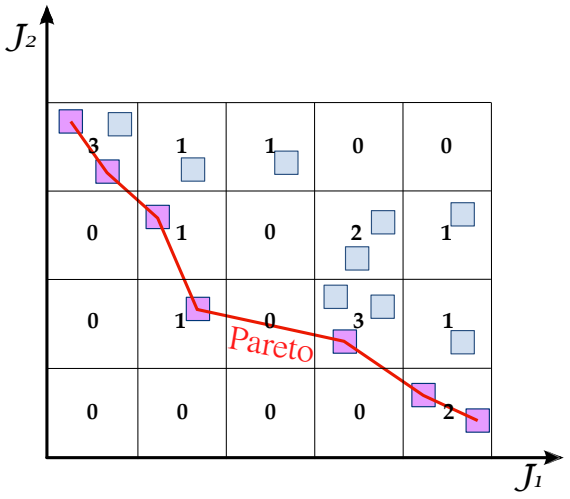


FIGURE 3.12: Calcul de la densité du voisinage d'un individu basé sur des cellules

2. Stocker les solutions *Elites* dans une archive externe et les réintroduire dans la population.

Une sélection aléatoire ne peut assurer le maintien des solutions d'élite à la génération suivante. Aussi, il est nécessaire d'utiliser d'autres stratégies pour conserver les meilleurs individus.

3.3.2.1 Elitisme sans population extérieure

La méthode la plus simple est de copier toutes les solutions *élites* d'une génération P_k à la suivante P_{k+1}. La population P_{k+1} est complétée par une sélection d'individus dominés de la population P_k. Cette approche ne peut plus fonctionner lorsque la taille de la population non-dominée dépasse la taille maximale N_P. Plusieurs solutions sont proposées pour résoudre ce problème.

Konak et Smith proposent dans [KS02] et [KS04] un algorithme génétique à taille de population dynamique qui utilise une stratégie d'élitisme pur [1]. En effet, la population n'est composée que d'invividus non-dominés. Si la taille de la population dépasse une valeur maximale N_{max}, $N_{max} - N_{min}$ solutions sont éliminées de la population. Cette opération est effectuée en valorisant la répartition sur le front non-dominé courant. Pour cela, une sélection par tournoi basée sur la Pareto-dominance est effectuée [HNG94]. Ainsi :

– on choisit deux solutions aléatoirement ;

1. L'élitisme pur consiste à ne conserver que des individus non-dominés pour créer la génération suivante

Données : P, N_P, N_E, n_g
Resultat : $\widetilde{P}$
Initialisation : $k = 0$, $P_0 = P$, $E_0 = \emptyset$. On génère n_g^n hypercubes;
tant que *Critère d'arrêt* **faire**
  Construction de E_k ;
  pour $p = 1$ **à** N_p **faire**
    si $\exists \overline{Y} \in E_k$, $\overline{Y} \succ Y_p$ **alors**
      Y_p est éliminé.
    sinon si $\exists e_k \subset E_k$, $\forall \overline{y} \in e_k, Y_p \succ \overline{y}$ **alors**
      e_k est éliminé.
    sinon
      $E_k = E_k \cup \{Y_p\}$
    fin
    si $|E_k| = n_g + 1$ **alors**
      On élimine un élément choisi aléatoirement dans la cellule de plus grande densité.
    fin
  fin
  Nouvelle génération ;
  si *Le critère d'arrêt est satisfait* **alors**
    On retourne $\widetilde{P} = E_k$.
  sinon
    Par croisements et mutations des individus de E_k, on crée G_k.
    $|G_k| = N_p - |E_k|$;
    $P_{k+1} = E_k \cup G_k$;
  fin
fin

Algorithme 7: Procédure de l'algorithme PESA [DCO00]. P est la population initiale, de taille N_P. E est l'archive de taille N_E, n_g le nombre de grilles et n le nombre de critères.

– on ne retient que celle dont la fonction de niche est la plus faible.

Cette opération est effectuée dès que l'ensemble des solutions sont non-dominées.

NSGA-II utilise une population dont la taille N_p est fixée. A la génération k, une population enfants E_k, de taille N_p est créée à partir de la population parents P_k. Les fronts non dominés $F_1, F_2, \ldots, F_R$ sont alors identifiés dans la population combinée $P_k \cup E_k$. La puissance de NSGA-II vient de son algorithme de tri rapide des fronts successifs. La population P_{k+1} est alors complétée à partir des solutions de F_1, puis de F_2 et ainsi de suite. Si l'on note j est l'indice du front F_j non-dominé de la population courante P_k tel que $|F_1 \cup F_2 \cup \cdots \cup F_j| \leq N_p$ et $|F_1 \cup F_2 \cup \cdots \cup F_{j+1}| > N_p$. Les solutions contenues dans les fronts $F_1, F_2, \ldots, F_j$ sont alors copiées dans P_{k+1}. La population est alors complétée par $N_p - |P_{k+1}|$ individus du front F_{j+1}. Cet algorithme assure la conservation de l'ensemble des solutions non-dominées tant que $|F_1| \leq N_p$ et la seconde

sélection basée sur la distance d'encombrement encourage la diversité. Notons que,

Données : Population initiale P_0, $N_P = |P_0|$
Resultat : Population finale de taille N_p
tant que *Critère d'arrêt* **faire**
  Croisement et mutations de P_k donnent : E_k, $|E_k| = N_p$;
  $R_k = P_k \cup E_k$;
  On identifie $F_1, F_2, \ldots, F_j$ dans R_k;
  pour $i = 1$ **à** j **faire**
    pour tous les $Y \in F_i$ **faire**
      On évalue $cd(Y)$ (voir algorithme 6);
      Génération de P_{k+1};
      si $|P_{k+1}| + |F_i| \leq N_p$ **alors**
        $P_{k+1} = P_{k+1} \cup F_i$
      sinon
        On sélectionne $N_p - |P_{k+1}|$ de F_i par tournoi sur $cd(Y)$
    fin
  fin
fin

Algorithme 8: Procédure de l'algorithme NSGA-II.

lorsque les populations combinées de parents et d'enfants de la génération k possèdent plus de N_p solutions non-dominées, NSGA-II devient un algorithme génétique d'élitisme pur. La simplicité de mise en oeuvre de la conservation des solutions non-dominées constitue un avantage par rapport à d'autres algorithmes génétiques. La taille de la population N_p est un paramètre important pour les algorithmes de ce type, car aucune archive extérieure ne contient les solutions non-dominées.

3.3.2.2 Elitisme avec population extérieure

Lorsque l'on utilise une archive extérieure à la population pour conserver les solutions d'élite, certaines précisions doivent être apportées. La première consiste à préciser quelles solutions sont stockées dans l'archive. La majeure partie des algorithmes génétiques y conservent les solutions non-dominées identifiées au cours de l'optimisation [EZ99]. L'archive A_k est alors mise à jour pour chaque solution trouvée en éliminant les solutions dominées par le nouvel individu ou en ajoutant ce nouvel individu s'il n'est dominé par aucun élément d'élite de A_k. Cette opération demande beaucoup de calculs, car l'archive est parcourue pour chaque nouvel individu. De plus, il peut exister un nombre très grand de solutions non dominées. La taille de l'archive doit alors être maitrisée. L'algorithme 9 propose une solution de gestion de taille d'archive. Notons que la distance d du calcul de distance entre les ensembles c peut être évaluée dans l'espace des critères ou

Données : Ensemble de solutions non dominées A, $N_A = |A|_{max}$
Resultat : Ensemble de solutions non dominées de taille N_A
tant que $|A| > N_A$ **faire**
  pour tous les $Y \in A$ **faire**
    $\exists c_i$, $Y \in c_i$ et $A = \{c_1, \ldots, c_M\}$
  fin
  tant que $M \geq N_A$ **faire**
    pour tous les $(i, j), i, j \leq M$ **faire**
      $d_{c_i,c_j} = \dfrac{1}{|c_i||c_j|} \displaystyle\sum_{y \in c_i, y' \in c_j} d(y, y')$
    fin
    (i_0, j_0), $d_{c_{i_0},c_{j_0}} = \min\limits_{(i,j)} \left\{ d_{c_i,c_j} \right\}$, $\{c_i\} = \{c_i\} \cup \{c_j\}$;
  fin
  pour $i = 1$ **à** M **faire**
    $c_i = \{Y_i\}$, Y_i minimise la distance moyenne entre les individus de c_i
  fin
fin

Algorithme 9: Description de la gestion de la taille de la population d'une archive élitiste.

dans l'espace des paramètres. Les algorithmes SPEA [EZ99] et SPEA2 [ZLT02] sont deux très bons exemples d'algorithmes génétiques qui utilisent une archive d'élites. Ce sont aussi de bons exemples d'algorithmes qui utilisent une population extérieure. L'algorithme SPEA2 est résumé par l'algorithme 10. SPEA [EZ99] utilise la méthode décrite par l'algorithme 9 pour contrôler la taille de l'archive courante. L'objectif de cette opération est de sélectionner les solutions d'élite de A_k que l'on réintroduit dans la population. Dans SPEA [EZ99], les individus de la population P_{k+1} sont sélectionnés parmi la réunion $P_k \cup A_k$. On attribue à chaque élément de la réunion d'ensembles une performance à partir de laquelle, on sélectionne N_p individus pour créer la population P_{k+1}.

Une autre solution est proposée par Ishibuchi et Murata [IM96]. Leur stratégie consiste à attribuer des places pour n_A solutions d'élite dans population suivante P_{k+1} Auquel cas, $N_P - n_A$ solutions sont choisies parmi les parents de la population P_k et des enfants E_k. Cette population est complétée par n_A solutions choisies dans l'archive élitiste A_k.

D'autres algorithmes utilisent des populations externes comme PAES [KC00], PESA [DCO00], RDGA [LY03], RWGA [MI95] et DMOEA [YL03].

Outre les méthodes évolutionnaires, il existe aussi des méthodes d'approche de fronts de Pareto de type géométrique. Nous citons en exemple la méthode NBI (*Normal Boun-*

Données : P_0, N_P, $A_0 = \emptyset$, $N_A = \max|A|$, $\lambda = \sqrt{N_P + N_A}$.
Resultat : Population P non-dominée.
tant que *Critère d'arrêt* **faire**
  Calcul de performance;
  pour tous les $Y \in P_k \cup A_k$ **faire**
    $D_Y = \left\{\overline{Y} \in P_k \cup A_k, \quad Y \succ \overline{Y}\right\}$;
    $r(Y) = \sum_{\overline{Y}, Y \in D_{\overline{Y}}} s(\overline{Y})$, où $s(\overline{Y}) = |D_{\overline{Y}}|$;
    $m(Y) = \left(\sigma_Y^{\lambda} + 1\right)^{-1}$;
    On pose alors : $\widetilde{\mathbb{J}}(Y) = r(Y) + m(Y)$.
  fin
  $A_{k+1} = \text{nondom}\{A_k \cup P_k\}$;
  si $|A_{k+1}| > N_A$ **alors**
    On élimine les $|A_{k+1}| - N_A$ solutions au σ^{λ} maximal;
    En cas de litige, on regarde σ^l, pour $l = \lambda - 1, \lambda - 2, \ldots$, successivement.
  sinon
    On copie dans A_{k+1} les $N_A - |A_{k+1}|$ solutions de
    $A_k \cup P_k - \text{nondom}\{A_k \cup P_k\}$.
  fin
  si *Critère d'arrêt* **alors**
    $P = \text{nondom}\{A_{k+1}\}$
  sinon
    On sélectionne des parents dans A_{k+1} par tournoi avec remise : P_{k+1};
    On crée N_P enfants par croisements et mutations des parents : E_{k+1};
    $P_{k+1} = P_{k+1} \cup E_{k+1}$
  fin
fin

Algorithme 10: Procédure de l'algorithme SPEA2. σ_Y^{λ} est la distance entre Y et son λ-ième plus proche voisin.

dary Intersection) proposée par Das et Dennis [DD98]. Celle-ci consiste en la construction d'agrégation de points dans l'espace des critères, à partir desquels l'on construit de front de Pareto. Des méthodes basées sur l'évolution d'essaims comme, par exemple PSO (Particule Swarm Optimization) [KE95] se montrent aussi efficaces pour le traitement de problèmes multimodaux.

3.4 Bilan sur les AGs

Nous venons de voir que les algorithmes génétiques présentent l'avantage d'être affranchis de toute étude de sensibilité. Leur mise en place nécessite le calibrage de quelques paramètres, tels que la taille de la population étudiée, la taille de l'archive (selon les cas), les coefficients de diversité (voisinage, taille des cellules, etc ...) et d'autres, en fonction de la méthode choisie. Cependant, ils restent robustes et un choix approximatif de ces paramètres reste efficace. Grâce à cette particularité, ils s'avèrent efficaces pour traiter des problèmes qui possèdent plusieurs optima locaux, là où les méthodes déterministes sont défaillantes. Cette situation est courante en ingénierie, par exemple dans les études de phénomènes non linéaires ou qui comportent des interactions entre variables de conception. De plus, les algorithmes génétiques peuvent évoluer dans des espaces quelconques : discontinus, discrets, etc .... Enfin, nous avons vu au travers de plusieurs exemples que ces méthodes sont généralement simples à implémenter. L'utilisation, proche du principe d'évolution des espèces, est aussi facile à comprendre.

Cependant, ce type d'algorithme d'optimisation n'est pas parfait. En effet, l'inconvénient majeur de ces méthodes est le coût en termes de temps de calcul nécessaire pour obtenir une solution. Cette particularité peut s'avérer prohibitive, mais l'amélioration des moyens informatiques tend à réduire l'impact de cette difficulté. De plus, comme nous l'avons indiqué, ces méthodes s'adaptent facilement au calcul parallèle.

Chapitre 4

Proposition de l'Algorithme de Descente à Gradients Multiples : MGDA

Dans le chapitre précédent, nous avons présenté quelques optimiseurs connus, de type évolutionnaire. Ces derniers sont efficaces et appréciés pour leur robustesse et leur simplicité. De plus, les techniques utilisées dans ces algorithmes pour maintenir une population variée favorisent l'obtention d'un échantillon représentatif du front de Pareto. Cependant, pour atteindre un tel résultat, ces méthodes nécessitent un grand nombre d'évaluations. Dans un contexte pré-industriel, les critères sont évalués par simulation de "haute fidélité". Ainsi, chaque appel solveur requiert un temps de calcul et une utilisation de ressources significatifs.

On reprend ici les résultats théoriques établis par J.A. Désidéri dans les publications [Dés12, Dés09] qui a observé un résultat très simple d'analyse convexe selon lequel l'élément de plus petite norme de l'enveloppe convexe des gradient associés à plusieurs critères fournit une direction de descente commune. Ce résultat que nous allons reprendre en détail permet de généraliser la méthode classique du gradient au contexte de l'optimisation multiobjectif. Dans ce chapitre, on rappelle ces acquis théoriques sur lesquels on s'appuie pour établir une méthode numérique pratique, robuste et adaptée au contexte de l'optimisation de "haute fidélité", que l'on valide ensuite sur des cas-tests de la littérature.

4.1 Description générale

Soit $\mathcal{H}$ un espace de Hilbert réel. Dans la pratique, $\mathcal{H}$ est un sous-espace de $\mathbb{R}^N$. Comme dans le chapitre précédent, on note K le sous-espace de $\mathcal{H}$ que l'on appelle domaine de recherche ou encore espace des paramètres. On considère n fonctions $(J_i)_{1\leq i\leq n}$ définies sur K et à valeurs dans $\mathbb{R}$. On note, comme dans le chapitre précédent :

$$\mathbb{J} : \begin{array}{ccc} \mathbb{R}^N & \longrightarrow & \mathbb{R}^n \\ Y & \longmapsto & \mathbb{J}(Y) = (J_1(Y), \ldots, J_n(Y)) \end{array}$$

Sans perte de généralité, nous considérons le problème d'optimisation qui consiste à trouver l'élément de K qui minimise l'ensemble des fonctions (J_i), simultanément.

$$\min_{Y\in K} \left\{ (J_i(Y))_{1\leq i\leq n} \right\}$$

L'algorithme d'optimisation MGDA est une généralisation de l'algorithme de plus grande descente [All05]. Nous avons décrit cet algorithme d'optimisation dans le chapitre 2, algorithme 2, page 19. Il s'agit de l'extension de ce dernier au cas multiobjectif. Le rapport de recherche de Jean-Antoine Désidéri [Dés12] propose une version plus détaillée des résultats suivants. Cette méthode a pour but de développer une coopération entre les critères. En effet, MGDA vise la réduction, à chaque pas, de l'ensemble des critères (dans le cas d'un problème de minimisation), simultanément. Cette particularité repose principalement sur l'existence d'une direction particulière dans laquelle cette réduction est réalisée, en tout point de K. Nous allons donc donner quelques définitions importantes, avant d'énoncer et de démontrer les résultats d'existence. Nous verrons ensuite quelques exemples d'application de MGDA.

4.1.1 Définitions et théorèmes

Nous invitons le lecteur à se reporter au livre de K. Miettinen [Mie99] pour plus de précisions en optimisation non-linéaire. Nous allons simplement énoncer les théorèmes et lemmes qui sont indispensables à la mise en place de l'algorithme MGDA.

Définition 4.1.1.1 (Pareto-optimalité)**.** *On considère un point Y^0 réalisable de K et un ensemble $(J_i)_{1\leq i\leq n}$ de critères réguliers que nous souhaitons minimiser. La solution Y^0 du problème de minimisation des critères (J_i) est dite Pareto-optimale si elle n'est dominée, au sens de Pareto, par aucune autre solution de K.*

Si l'on consière un point Y^0 de $\mathbb{R}^N$ qui n'est pas optimal au sens de Pareto, on souhaite trouver une direction, si elle existe, dans laquelle l'ensemble des critères est diminué localement.

Considérons n critères réguliers $J_i(Y)$, où Y est un vecteur de paramètres, $Y \in \mathcal{H}$. L'espace des paramètres $\mathcal{H}$ est un espace de Hilbert, généralement égal à $\mathbb{R}^N$. En pratique, chaque fonction coût est supposée de classe C^2, dans une boule ouverte de l'espace des paramètres $\mathcal{H}$, centrée en Y^0 Dans la suite, sauf mention contraire, N désignera la dimension de l'espace des paramètres $\mathcal{H}$, lorsque cette dernière est finie.

Lemme 4.1.1 (J.A. Désidéri, [Dés12])**.** *On considère Y^0 un point Pareto-stationnaire relativement aux critères réguliers $J_i(Y)$ $(1 \leq i \leq n)$. On note $u_i^0 = \nabla J_i(Y^0)$ le gradient de chaque critère J_i, au point Y^0. Alors, il existe une combinaison convexe nulle de ces gradients :*

$$\exists (\alpha_i)_{1\leq i\leq n}, \quad \alpha_i \geq 0, \quad \sum_{i=0}^{n} \alpha_i = 1, \qquad \sum_{i=1}^{n} \alpha_i u_i^0 = 0 \tag{4.1}$$

Ainsi, une solution Pareto-optimale est Pareto-stationnaire.

Preuve. Sans perte de généralité, nous supposons que tous les critères sont nuls en Y^0.

$$\forall i, 1 \leq i \leq n, \quad J_i(Y^0) = 0$$

Si l'on considère une boule ouverte $\mathcal{B}_0$, centrée en Y^0 dans laquelle les critères (J_i) sont convexes, Y^0 est une solution du problème d'optimisation sous contraintes suivant :

$$Y^0 \in \left\{ \underset{Y\in\mathcal{B}}{\operatorname{Argmin}}\, J_n(Y), \qquad \text{s.c. } J_i(Y) \leq 0, \quad 1 \leq i \leq n-1 \right\} \tag{4.2}$$

On note $\overline{\mathcal{U}}_{n-1}$ l'enveloppe convexe de la famille $\{(u_i^0)_{1\leq i\leq n-1}$ et on pose :

$$\omega_{n-1} = \underset{u\in\overline{\mathcal{U}}_{n-1}}{\operatorname{Argmin}} \|u\|.$$

La convexité de l'ensemble ω_{n-1} assure l'existence et l'unicité du vecteur ω_{n-1}. De plus, il vérifie les inégalités :

$$\forall i, 1 \leq i \leq n-1, \quad \langle \omega_{n-1},\, u_i^0 \rangle \geq \|\omega_{n-1}\|^2.$$

Deux cas sont alors possibles :

- $\omega_{n-1} = 0$ et la solution Y^0 est Pareto-stationnaire pour l'ensemble des critères $(J_i)_{1\leq i\leq n-1}$. La condition est alors satisfaite pour l'ensemble des critères.
- $\omega_{n-1} \neq 0$. On définit alors pour tout i, $1 \leq i \leq n-1$, la fonction j_i par :

$$j_i : \begin{array}{ccc} \mathbb{R}^{+*} & \longrightarrow & \mathbb{R} \\ \varepsilon & \longmapsto & J(Y^0 - \varepsilon\omega_{n-1}) \end{array}$$

Alors, pour tout i, $1 \leq i \leq n-1$, la fonction $j_i(0) = 0$ et :

$$j_i^{'}(0) = -\left\langle u_i^0, \omega_{n-1} \right\rangle \leq \|\omega_{n-1}\|^2 < 0.$$

La condition de qualification des contraintes de Slatter [Cea71] est alors vérifée pour le problème 4.2. Ainsi, le Lagrangien augmenté défini par :

$$\mathcal{L} = J_n(Y) + \sum_{i=1}^{n-1} \lambda_i J_i(Y),$$

est stationnaire en Y^0, i.e. :

$$u_n^0 + \sum_{i=1}^{n-1} \lambda_i u_i^0 = 0. \tag{4.3}$$

Tant que les contraintes ne sont pas saturées ($J_i(Y^0) = 0$), les conditions de Karush-Kuhn-Tucker (KKT) assurent que $\lambda_i > 0$. L'équation 4.3 normalisée répond à la définition de la Pareto-stationnarité.

□

Ce résultat nous mène naturellement à la définition suivante de point *Pareto stationnaire*.

Définition 4.1.1.2 (Pareto stationnaire, J.A. Désidéri, [Dés12])**.** *Un ensemble de critères réguliers $J_i(Y)$, $1 \leq i \leq n \leq N$, est dit Pareto stationnaire en un point de l'espace des paramètres Y^0 si et seulement si il existe une combinaison convexe nulle des gradients $u_i^0 = \nabla J_i(Y^0)$:*

$$\exists (\alpha_i)_{1 \leq i \leq n}, \quad \alpha_i \geq 0, \quad \sum_{i=i}^{n} \alpha_i = 1, \qquad \sum_{i=1}^{n} \alpha_i u_i^0 = 0. \tag{4.4}$$

En général, pour des critères réguliers, sans contrainte, la Pareto stationnarité est une condition nécessaire à la Pareto optimalité. Dans le cas contraire, si l'ensemble de critères réguliers $J_i(Y)$ n'est pas Pareto stationnaire en un point donné de l'espace des paramètres Y^0, une direction de descente commune à tous les critères existe. Nous allons étudier comment une telle direction peut être caractérisée. Nous avons le Lemme suivant :

Lemme 4.1.2. *Soit $\mathcal{H}$ un espace de Hilbert, de dimension finie ou infinie N et $\{u_i\}$, une famille de n vecteurs de $\mathcal{H}$ Soit $\mathcal{U}$ l'enveloppe convexe de cette famille :*

$$\mathcal{U} = \left\{ w \in \mathcal{H}, \quad \exists (\alpha_i)_{1 \leq i \leq n}, \quad \alpha_i > 0, \quad \sum_{i=i}^{n} \alpha_i = 1, \quad w = \sum_{i=1}^{n} \alpha_i u_i \right\}$$

et $\overline{\mathcal{U}}$ *la clôture algébrique de* $\mathcal{U}$*. Alors l'élément de norme minimale* ω *de* $\mathcal{U}$ *est unique et vérifie :*

$$\forall u \in \overline{\mathcal{U}}, \quad \langle \omega, u \rangle \geq \langle \omega, \omega \rangle = \|\omega\|^2 \tag{4.5}$$

Preuve. L'enveloppe convexe $\overline{\mathcal{U}}$ est un sous-espace convexe et fermé de $\mathcal{H}$. Ceci implique l'existence et l'unicité de l'élément ω de plus petite norme de l'ensemble.

On considère un élément $u \in \overline{\mathcal{U}}$ et on pose $r = u - \omega$, soit $u = r + \omega$. Comme $\overline{\mathcal{U}}$ est convexe :

$$\forall \varepsilon \in [0, 1], \quad \omega + \varepsilon r \in \overline{\mathcal{U}}$$

Comme ω est le vecteur de norme minimale de l'ensemble $\overline{\mathcal{U}}$, $\|\omega + \varepsilon r\| \geq \|\omega\|$ Ce qui implique :

$$\|\omega + \varepsilon r\| - \|\omega\| = \langle \omega + \varepsilon r, \omega + \varepsilon r \rangle - \langle \omega, \omega \rangle = 2\varepsilon \langle r, \omega \rangle + \varepsilon^2 \langle r, r \rangle \geq 0$$

Pour ε assez petit, cette inégalité impose :

$$\langle r, \omega \rangle = \langle u - \omega, \omega \rangle \geq 0$$

Ce qui donne le résultat recherché. □

En combinant la définition 4.1.1.2 et le Lemme 4.1.2 nous obtenons le théorème suivant :

Théorème 4.1.3 (J.A. Désidéri, [Dés12])**.** *Soit* $\mathcal{H}$ *un espace de Hilbert de dimension* N*, finie ou infinie. Soit* $\{J_i\}$*,* n *fonctions régulières de variable* $Y \in \mathcal{H}$ *et* Y^0 *un point admissible de l'espace des paramètres. En ce point, les gradients des fonctions sont notés* $u_i^0 = \nabla J_i(Y^0)$*. On note* $\mathcal{U}$ *l'enveloppe convexe des* u_i^0*, définie par :*

$$\mathcal{U} = \left\{ w \in \mathcal{H}, \quad \exists (\alpha_i)_{1 \leq i \leq n}, \quad \alpha_i > 0, \quad \sum_{i=i}^{n} \alpha_i = 1, \quad w = \sum_{i=1}^{n} \alpha_i u_i \right\}$$

Soit ω *le vecteur de norme minimale de* $\overline{\mathcal{U}}$*, clôture algébrique de* $\mathcal{U}$*. Il vient alors que :*

1. *Si* $\omega = 0$*, l'ensemble des critères* $\{J_i\}$ *est Pareto stationnaire en* Y^0 *;*
2. *Sinon,* $-\omega$ *est une direction de descente commune à l'ensemble des critères* $\{J_i\}$*. De plus, si* $\omega \in \mathcal{U}$*, les dérivées directionnelles* $\langle u_i^0, \omega \rangle$ $(1 \leq i \leq n)$ *sont égales à* $\|\omega\|^2$*.*

Preuve. Tous les éléments de ce théorème sont des variantes de résultats énoncés précédemment. Seul le résultat sur les dérivées directionnelles du second cas où $\omega \in \mathcal{U}$ est à démontrer. On se place donc dans le cas où $\omega \neq 0$ et $\omega \in \mathcal{U}$. On définit la fonction

$j : \mathcal{U} \to \mathbb{R}$ par :

$$j : \begin{array}{ccc} \mathcal{U} & \longrightarrow & \mathbb{R}_+ \\ u & \longmapsto & \langle u, u \rangle \end{array}$$

Le point qu'il nous reste à établir se résume à la résolution du problème d'optimisation suivant :

$$\begin{cases} \alpha = (\alpha_i)_{1 \leq i \leq n} = \underset{u \in \mathcal{U}}{\text{Argmin}} \{j(u)\} \\ \text{s.c.} \quad \left| \begin{array}{l} \sum_{i=1}^{n} \alpha_i = 1 \\ \alpha_i > 0 \end{array} \right. \end{cases} \tag{4.6}$$

Par définition de $\mathcal{U}$, aucune des contraintes $\alpha_i > 0$ n'est saturée. Par conséquent, en utilisant le vecteur $\alpha \in \mathbb{R}^n$ comme variable, le Lagrangien augmenté s'écrit :

$$\mathcal{L}(\alpha, \lambda) = j + \lambda \left(\sum_{i=1}^{n} \alpha_i - 1 \right)$$

La condition d'optimalité satisfaite par le vecteur α s'écrit :

$$\begin{cases} \dfrac{\partial \mathcal{L}}{\partial \alpha_i} &= 0, \quad 1 \leq i \leq n \\ \\ \dfrac{\partial \mathcal{L}}{\partial \lambda} &= 0 \end{cases}$$

Ce qui implique, pour tout i :

$$\frac{\partial j}{\partial \alpha_i} + \lambda = 0$$

Or, $j(u) = (u, u)$ et pour $u = \omega = \sum_{i=1}^{n} \alpha_i u_i^0$, nous avons :

$$\frac{\partial j}{\partial \alpha_i} = 2 \left\langle \frac{\partial u}{\partial \alpha_i}, u \right\rangle = 2 \left\langle u_i^0, \omega \right\rangle = -\lambda \Rightarrow \left\langle u_i^0, \omega \right\rangle = -\lambda/2$$

qui est constant. □

En résumé, nous avons identifié le vecteur :

$$\omega = \sum_{i=1}^{n} \alpha_i u_i^0,$$

en résolvant le problème de minimisation sous contrainte dans $\mathbb{R}^n$ de la forme quadratique suivante :

$$\min_{\alpha \in \mathbb{R}^n} \left\| \sum_{i=1}^{n} \alpha_i u_i^0 \right\|^2$$

sous la contrainte :

$$\left| \begin{array}{l} \forall i, \quad \alpha_i \geq 0, \\ \sum_{i=1}^{n} \alpha_i = 1. \end{array} \right.$$

4.1.2 Cas d'un problème biobjectif, $n = 2$

Dans ce qui suit, nous traitons le cas particulier d'un problème d'optimisation à deux critères, $n = 2$. On pose $u = u_1^0$ et $v = u_2^0$, $\alpha = \alpha_1$ et $1 - \alpha = \alpha_2$, afin de simplifier les notations. Considérons la forme quadratique suivante :

$$j(\alpha) = \|\alpha u + (1-\alpha)v\|^2 = \langle \alpha u + (1-\alpha)v,\, \alpha u + (1-\alpha)v \rangle$$

Le cas trivial $u = v$ écarté, la minimum est atteint pour :

$$\alpha = \frac{v.(u-v)}{\|u-v\|^2} = \frac{\|v\|^2 - u.v}{\|u\|^2 + \|v\|^2 - 2u.v} \tag{4.7}$$

En effet, la projection orthogonale du vecteur nul $\vec{0}$ sur l'enveloppe convexe décrite par $\alpha u + (1-\alpha)v$, sans restriction sur α est donné par :

$$\langle \alpha u + (1-\alpha)v,\, u - v \rangle = 0$$

ce qui donne l'expression de α de l'équation 4.7.

Si les vecteurs u et v sont normalisés ($\|u\| = \|v\| = 1$), nous avons $\alpha = 1/2$. Dans le cas contraire, il n'est pas assuré que $\alpha \in [0, 1]$. La figure 4.1 illustre les différentes configurations possibles rencontrées dans le cas $n = 2$ et le vecteur ω choisi dans chaque cas.

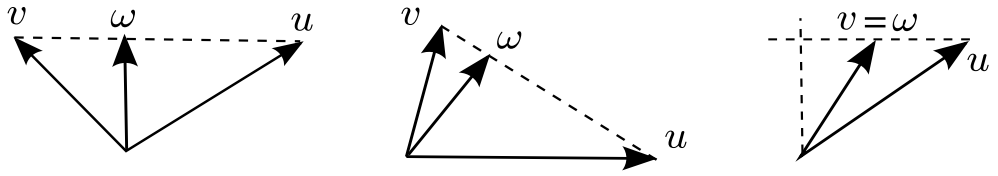


Figure 4.1: Illustration des différentes configurations possibles dans la résolution d'un problème d'optimisation à deux critères, sans normalisation des gradients.

La solution est donc la suivante :

$$\alpha = \begin{cases} \dfrac{v.(u-v)}{\|u-v\|^2} & \text{si } u.v < \min(\|u\|^2, \|v\|^2) \\ \\ 0 \text{ ou } 1 & \text{sinon. Selon } \min(\|u\|^2, \|v\|^2) = \|u\|^2 \text{ ou } \|v\|^2 \end{cases}$$

Finalement, lorsque $\alpha \in [0, 1]$, nous posons :

$$\theta = \widehat{(u, v)}, \quad \gamma = \langle u, v \rangle = \|u\|.\|v\| \cos\theta$$

Nous avons alors :

$$\omega = \frac{\|v\|^2 - \langle u, v \rangle}{\|u\|^2 + \|v\|^2 - 2\langle u, v \rangle} u + \frac{\|u\|^2 - \langle u, v \rangle}{\|u\|^2 + \|v\|^2 - 2\langle u, v \rangle} v$$

Il vient alors :

$$\langle u, \omega \rangle = \frac{\left(\|v\|^2 - \gamma\right)\|u\|^2 + \left(\|u\|^2 - \gamma\right)\gamma}{\|u - v\|^2} = \frac{\|u\|^2\|v\|^2 - \gamma^2}{\|u - v\|^2} = \frac{\|u\|^2\|v\|^2}{\|u - v\|^2} \sin^2(\theta) \geq 0$$

Cette relation reste identique si l'on intervertit les rôles de u et de v donc :

$$\langle u, \omega \rangle = \langle v, \omega \rangle .$$

4.2 Utilisation pratique de l'optimiseur MGDA

Dans cette section, nous allons aborder les détails pratiques du fonctionnement de l'optimiseur MGDA. Nous allons commencer par décrire globalement l'algorithme puis nous décrirons en détail les étapes importantes.

Nous considérons un problème d'optimisation multiobjectif. Les critères $(J_i)_{1 \leq i \leq n}$ que nous considérons sont supposés réguliers. L'algorithme est initialisé à partir d'un

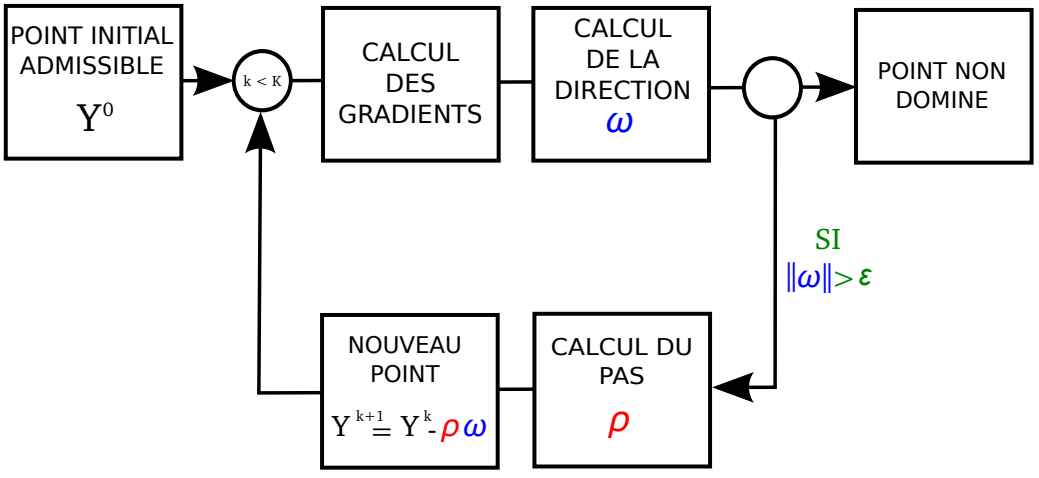


Figure 4.2: Schéma fonctionnel de l'algorithme MGDA.

point admissible de l'espace des paramètres K. On le note Y^0. Les gradients des fonc-

tions objectifs sont évalués en Y^0. Comme précédemment, nous adoptons la notation suivante :

$$u_i^0 = \nabla J_i(Y^0)$$

D'après le théorème 4.1.3, si l'ensemble des critères $(J_i)_{1\leq i\leq n}$ n'est pas Pareto-stationnaire en Y^0, il existe une direction $-\omega$ dans laquelle chaque critère J_i décroît, localement. On cherche donc le vecteur de norme minimale de l'enveloppe convexe $\mathcal{U}$ de la famille $(u_i^0)_{1\leq i\leq n}$. Si $\|\omega\| \leq \varepsilon$ (où ε est un réel strictement positif dont la valeur est fixée par l'utilisateur), l'ensemble des critères est supposé Pareto-stationnaire au point courant Y^k et l'algorithme s'arrête. Si $\|\omega\| > \varepsilon$, nous cherchons le pas ρ que l'algorithme doit effectuer dans la direction $-\omega$ pour atteindre le point suivant : Y^{k+1} défini par :

$$Y^{k+1} = Y^k - \rho\omega$$

De plus, une condition qui limite du nombre maximal d'évaluations est imposé. Si le nombre de boucles k atteint la valeur K (K est fixé par l'utilisateur), l'algorithme s'arrête au point Y^k.

4.2.1 Calcul du pas

Cette partie traite de la recherche du pas d'avancée, dans la direction $-\omega$. En optimisation multiobjectif, il est difficile de trouver un pas d'avancée qui convienne à l'ensemble des objectifs. Il faut donc adopter une méthode adaptative pour calculer un pas global.

En un point donné, dans la direction $-\omega$, les dérivées de Frechet de l'ensemble des critères sont strictement négatives. En particulier, elles sont toutes égales si $\omega \in \mathcal{U}$. En plus de l'évaluation du point courant, nous évaluons chaque critère en 2 points, dans la direction $-\omega$. A partir de ces trois points, nous construisons un modèle quadratique de chaque fonction objectif. Pour chaque critère J_i, nous calculons alors un pas ρ_i correspondant au minimum du modèle quadratique. Cette étape est illustrée dans le cas d'un problème d'optimisation à deux critères sur la figure 4.3.

Nous prenons ensuite le plus petit pas parmi les ρ_i pour pas global ρ :

$$\rho = \min_{1\leq i\leq n} \rho_i$$

Par définition de ω nous avons $\rho_i \geq 0$ et $\rho \geq 0$.

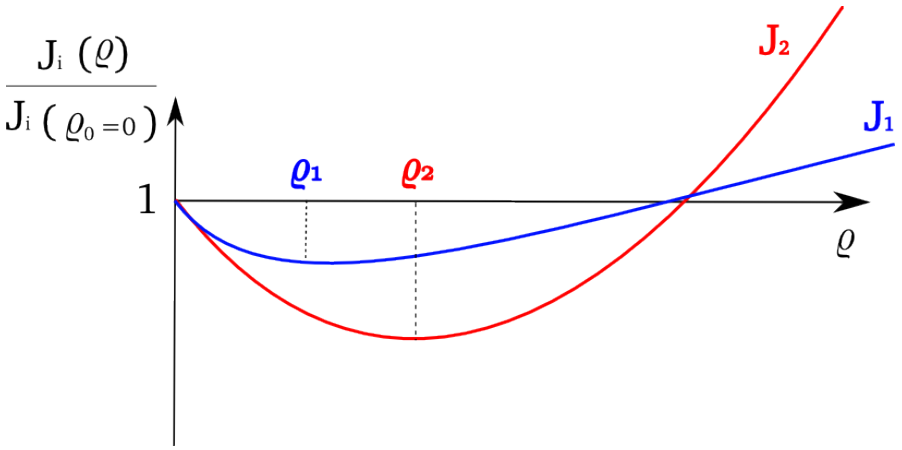


FIGURE 4.3: Approximation quadratique des critères J_1 et J_2, dans la direction $-\omega$ et calcul du pas optimal ρ_i.

4.3 Application sur un cas analytique

4.3.1 Description du cas test

Nous étudions dans cette section l'application de l'algorithme MGDA à un cas test analytique. Il s'agit de minimiser simultanément les deux fonctions de $\mathbb{R}^2 \to \mathbb{R}$, notées f et g, définies par :

$$\begin{cases} f(x,y) = 4x^2 + y^2 + xy \\ \\ g(x,y) = (x-1)^2 + 3(y-1)^2. \end{cases} \tag{4.8}$$

La figure 4.4 représente les surfaces de réponse des fonctions f et g. Les coordonnées des points non dominés du problème d'optimisation considéré sont représentées par la courbe entre les points de coordonnées respectives $(0,0)$ et $(1,1)$. Chaque fonction a un unique minimum facilement identifiable. De plus, il est aisé de déterminer l'ensemble des points où les critères f et g à minimiser sont Pareto-stationnaires. Les gradients de f et g sont donnés par :

$$\begin{cases} \nabla f(x,y) &= \begin{pmatrix} 8x+y \\ 2y+x \end{pmatrix}, \\ \\ \nabla g(x,y) &= \begin{pmatrix} 2(x-1) \\ 6(y-1) \end{pmatrix}. \end{cases}$$

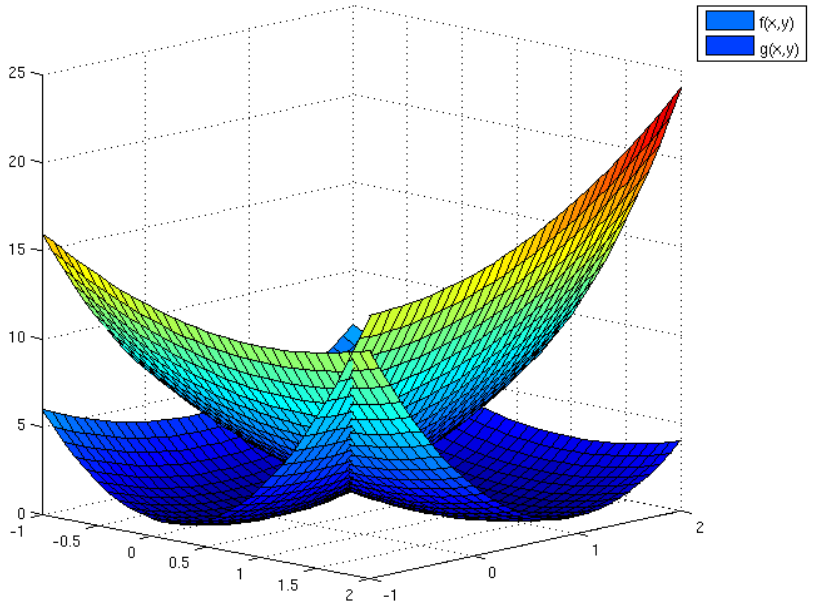


FIGURE 4.4: Représentation des surfaces de $f(x,y)$ et $g(x,y)$, pour $(x,y) \in [-1,2]^2$.

Si les critères f et g sont Pareto-stationnaires en $(x,y) \in \mathbb{R}^2$, il existe $\alpha > 0$ tel que $\nabla f(x,y) = -\alpha \nabla g(x,y)$. Nous avons alors :

$$\begin{cases} 8x + y &= -2\alpha(x-1) \\ \\ 2y + x &= -6\alpha(y-1) \end{cases}$$

$$\Rightarrow \quad \begin{cases} \dfrac{8x+y}{2(x-1)} = \dfrac{2y+x}{6(y-1)}, \\ x \neq 1, \quad y \neq 1. \end{cases}$$

$$\Rightarrow \quad \begin{cases} 3 = \dfrac{(2y+x)(x-1)}{(8x+y)(y-1)}. \\ x \neq 1, \quad y \neq 1. \end{cases}$$

De plus le point de coordonnées $(0,0)$ (resp $(1,1)$) correspond au minimum de la fonction f (resp g). L'ensemble des points de coordonnées (x,y), antécédents des points du front de Pareto du problème de minimisation considéré sont :

$$P_{f,g} = \left\{ (x,y) \in]0,1[^2, \quad 3 = \frac{(2y+x)(x-1)}{(8x+y)(y-1)} \right\}.$$

Il s'agit donc de la ligne de niveau d'égalité 3 de la fonction F définie par :

$$F : \begin{array}{ccc}]0,1[^2 & \longrightarrow & \mathbb{R}^2 \\ (x,y) & \longmapsto & \dfrac{(2y+x)(x-1)}{(8x+y)(y-1)} \end{array}$$

La figure 4.5 représente des lignes de niveaux des fonctions f et g sur l'ensemble $[-0.5, 1.5]^2$ dans l'espace des paramètres $(x, y) \in \mathbb{R}^2$. L'ensemble de Pareto y est aussi représenté, en ligne continue.

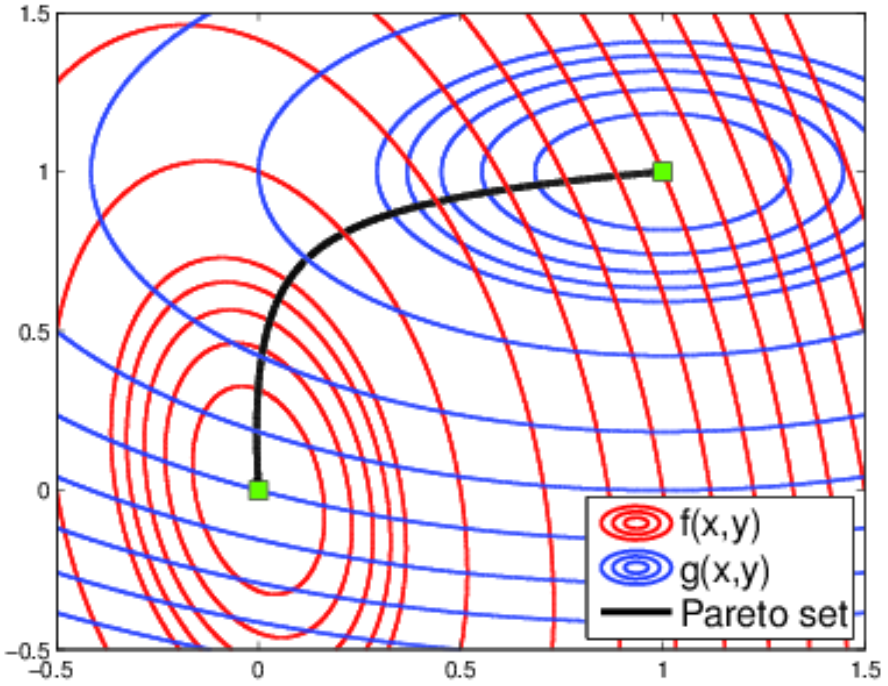


Figure 4.5: Lignes de niveaux des fonctions f et g et ensemble de Pareto du problème d'optimisation considéré, dans l'espace des paramètres (x, y).

4.3.2 Application de MGDA

Nous allons utiliser l'algorithme MGDA afin de minimiser simultanément les fonctions f et g à partir d'un point initial donné. La figure 4.6 représente la convergence pas à pas de MGDA vers un point Pareto optimal, dans trois cas différents. Nous considérons les points de coordonnées $(-1.5, -2.5)$, $(0.5, 2)$ et $(1.5, 2.5)$, dans l'espace des paramètres. Ces points particuliers ont été choisis pour illustrer la convergence de

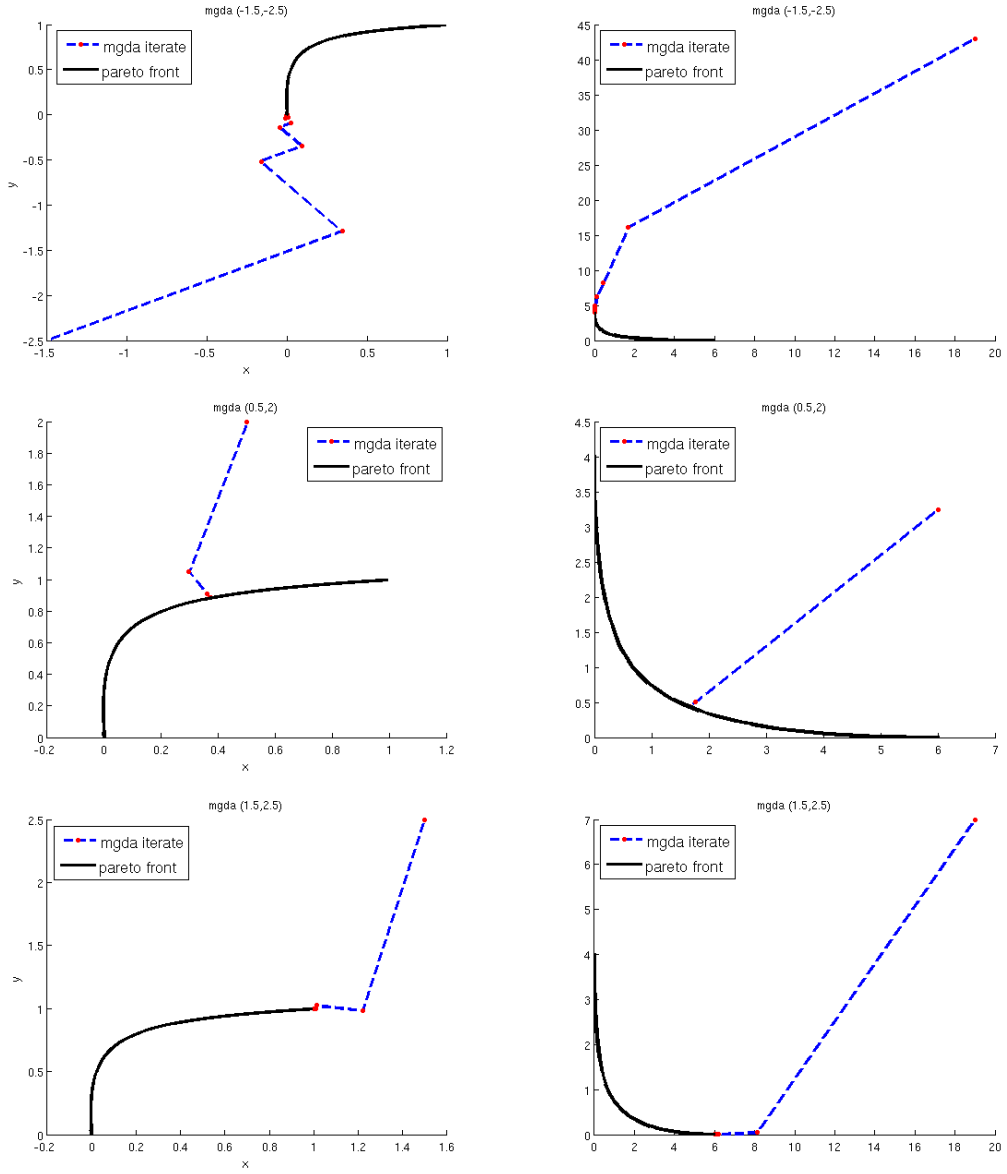


Figure 4.6: Convergence vers l'ensemble et le front de Pareto grâce à MGDA pour trois points initiaux distincts. Dans l'espace des paramètres (x, y), à gauche et dans l'espace des critères $(f(x, y), g(x, y))$, à droite.

l'algorithme vers différentes parties du front de Pareto. Si l'on considère le point de coordonnées $(0.5, 2)$, MGDA converge vers le milieu du front. Dans les deux autres cas, l'algorithme converge vers les extrémités du front. Chaque cas correspond à un point initial particulier. La colonne de gauche de la figure correspond à la convergence dans l'espace des paramètres ($(x, y) \subset \mathbb{R}^2$). La colonne de droite correspond à la convergence dans l'espace des critères ($(f(x, y), g(x, y)) \subset \mathbb{R}^2$). Chaque segment en pointillés entre

deux points correspond à une itération de MGDA. Selon les cas, le point final de la convergence peut être le minimum de f, le minimum de g ou un point intermédiaire situé sur l'ensemble de Pareto. Dans ce cas précis, le front de Pareto est continu et convexe, ce qui correspond à un contexte d'optimisation très favorable. De plus, comme les fonctions considérées sont quadratiques, le pas ρ calculé par l'algorithme doit correspondre au pas optimal. Afin de valider la convergence de l'algorithme MGDA sur ce cas test, nous considérons un ensemble de points répartis de manière homogène sur le cercle de centre $\Omega(1, 0)$ et de rayon $r = 1$. Nous appliquons la méthode MGDA à chaque point de l'ensemble jusqu'à ce que la condition d'arrêt $\omega < \varepsilon$ soit atteinte. La figure 4.7 représente l'ensemble des itérations (points) ainsi que le résultat final, à convergence. La convergence est illustrée dans l'espace de recherche K et dans l'espace des critères.

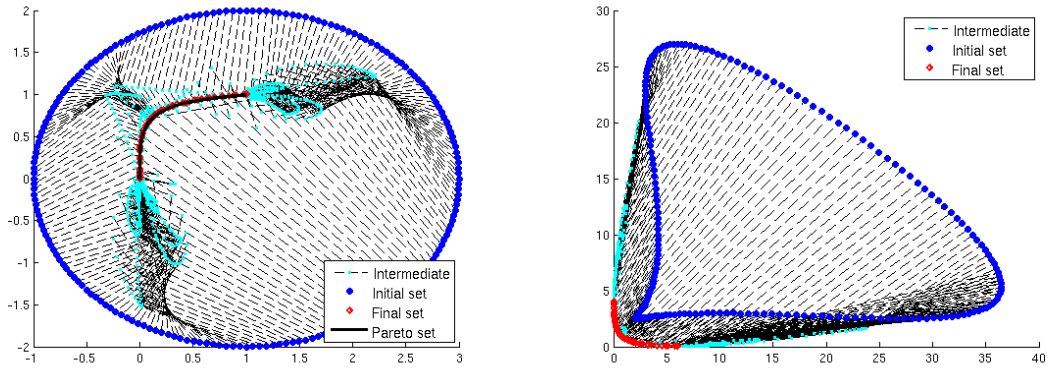


Figure 4.7: Convergence de MGDA vers l'ensemble Pareto optimal, depuis un ensemble de points répartis sur un cercle.

Les points intermédiaires entre un point initial du cercle et son point de convergence sur l'ensemble de Pareto sont reliés en trait pointillés. En moyenne, il est nécessaire de faire 5 itérations pour atteindre le critère d'arrêt de l'algorithme. De plus, comme nous l'avons indiqué, l'ensemble de Pareto est identifiable analytiquement, dans l'espace des paramètres. Il est représenté sur la figure en trait continu. Nous pouvons alors constater que les points issus de la convergence de MGDA sont disposés avec précision sur l'ensemble de Pareto. De plus, la répartition dans l'espace de recherche des points initiaux permet d'obtenir un échantillon représentatif du front final. La comparaison entre l'ensemble Pareto optimal analytique et le résultat de la convergence de MGDA à partir d'un ensemble de points initiaux décrit ci-dessus est représentés sur la figure 4.8. La comparaison des deux ensembles démontre la grande précision de l'algorithme sur cet exemple d'optimisation multiobjectif.

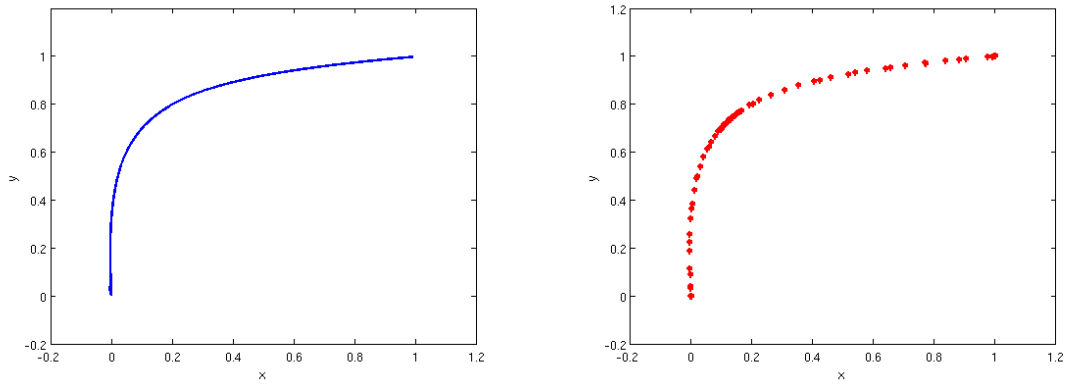


Figure 4.8: Ensemble Pareto optimal dans l'espace des paramètres : tracé analytique, à gauche et résultat de la convergence de MGDA, à droite.

4.3.3 Comparaison avec un algorithme évolutionnaire

La même expérience a été conduite en utilisant PAES (*Pareto Archived Evolution Strategy*) [KC00] à la place de MGDA. Dans cet exemple, nous générons une population de 100 individus. Seul les individus non dominés sont retenus par l'algorithme. Le résultat de l'approximation du front de Pareto est illustré par la figure 4.9.

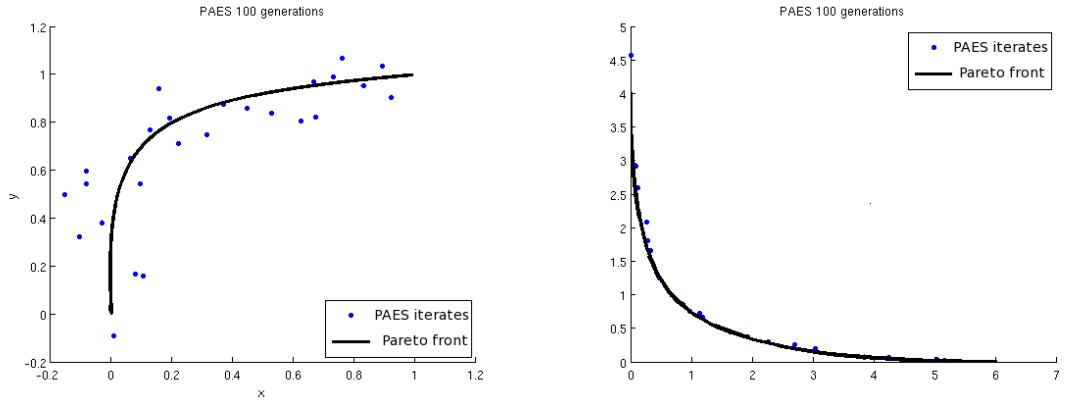


Figure 4.9: Les lignes continues représentent le front de Pareto dans l'espace des paramètres et dans l'espace des critères (à gauche et à droite, respectivement). Les points représentent le résultat obtenu en utilisant l'algorithme PAES.

Bien évidemment, dans un cas aussi simple avec un front de Pareto continu et convexe, où les gradients sont aisément calculables et réguliers, la méthode d'optimisation basée sur l'utilisation des gradients est plus rapide et plus précise. De plus, la méthode de calcul de pas utilisée approche les fonctions par des quadratiques. Comme les fonctions sont des quadratiques, le pas calculé est très proche de *l'optimal*.

4.4 Cas test de Fonseca

4.4.1 Description du cas test

Le cas test suivant est proposé par Fonseca et utilisé par Deb avec NSGA-II dans [KDM00]. Il consiste en la minimisation simultanée de deux fonctions f^- et f^+ définies de $\mathbb{R}^3 \to \mathbb{R}$ par :

$$\begin{cases} f^-(Y) = 1 - \exp\left(-\sum_{i=1}^{3}\left(y_i - \frac{1}{\sqrt{3}}\right)^2\right) \\ f^+(Y) = 1 - \exp\left(-\sum_{i=1}^{3}\left(y_i + \frac{1}{\sqrt{3}}\right)^2\right) \end{cases} \tag{4.9}$$

où $Y = (y_1, y_2, y_3) \in \mathbb{R}^3$. Ce cas test est connu pour avoir un front de Pareto continu mais concave, dans l'espace des critères. Au delà de cette particularité, ce problème ne présente pas de difficulté supplémentaire par rapport au cas précédent. En effet, les deux fonctions objectifs sont continues et régulières. De plus, les gradients peuvent être obtenus analytiquement. Il est par ailleurs possible de décrire le front de Pareto du problème par une courbe paramétrée. En effet, comme les deux fonctions objectifs sont régulières partout, les solutions *Pareto optimales* sont aussi *Pareto stationnaires*. Soit :

$$\exists \alpha \in [0,1], \quad \alpha \nabla f^+ + (1-\alpha) f^- = 0$$

Or :

$$\begin{cases} \frac{\partial}{\partial y_j} f^+ = 2\left(y_j + \frac{1}{\sqrt{3}}\right) \exp\left(-\sum_{i=1}^{3}\left(y_i + \frac{1}{\sqrt{3}}\right)^2\right) \\ \frac{\partial}{\partial y_j} f^- = 2\left(y_j - \frac{1}{\sqrt{3}}\right) \exp\left(-\sum_{i=1}^{3}\left(y_i - \frac{1}{\sqrt{3}}\right)^2\right) \end{cases}$$

Donc, sur le front de Pareto, pour tout j :

$$\alpha\left(y_j + \tfrac{1}{\sqrt{3}}\right) e^{\left(-\sum_{i=1}^{3}\left(y_i + \frac{1}{\sqrt{3}}\right)^2\right)} + (1-\alpha)\left(y_j - \tfrac{1}{\sqrt{3}}\right) e^{\left(-\sum_{i=1}^{3}\left(y_i - \frac{1}{\sqrt{3}}\right)^2\right)} = 0$$

L'équation est la même pour tous les y_j donc :

$$\forall j, \quad y_j = y = y(\alpha)$$

Soit :

$$\alpha\left(y+\frac{1}{\sqrt{3}}\right)e^{-\frac{12}{\sqrt{3}}y}+(1-\alpha)\left(y-\frac{1}{\sqrt{3}}\right)=0$$

D'où :

$$\begin{aligned}\alpha &= \frac{y-a}{y-a-(y+a)e^{-12ay}}\\ &= \frac{1}{1+\frac{a+y}{a-y}e^{-12ay}}\end{aligned}$$

Avec :

$$a=\frac{1}{\sqrt{3}}$$

Nous avons les valeurs particulières suivantes :

$$\begin{cases} y=-a & : \quad \alpha=1 \\ y=+a & : \quad \alpha=0 \end{cases}$$

Quand $y \in]-a,a[$, $\alpha(y)$ est continue (le rapport $\frac{a+y}{a-y}$ reste positif). Par conséquent, toutes les valeurs de α entre 0 et 1 sont atteintes au moins une fois. A l'inverse, toutes les valeurs de y extérieures à l'intervalle $[-a,a]$ sont telles que le rapport $\frac{a+y}{a-y}$ est négatif, nous avons alors $\alpha<0$ ou $\alpha>1$. L'ensemble du front de Pareto est donc atteint pour $y \in]-a,a[$. Le front de Pareto est donc décrit par la courbe paramétrée :

$$\begin{cases} f^+(y)=1-\exp(-3(y+a)^2), \\ f^-(y)=1-\exp(-3(y-a)^2). \end{cases}, \quad y \in]-a,a[$$

La figure 4.10 illustre le front de Pareto du cas test proposé par Fonseca, dans l'espace des critères (f^-, f^+).

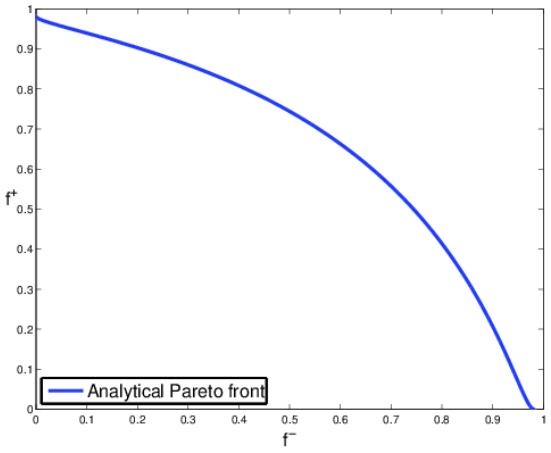


Figure 4.10: Front de Pareto analytique du cas test proposé par Fonseca.

4.4.2 Résultats

Afin de valider la convergence de MGDA vers le front de Pareto, nous considérons un ensemble de points répartis sur une sphère de centre $\Omega(0, 0, 0)$ et de rayon $r = 1.5$ qui englobe l'ensemble des points Pareto optimaux de notre problème. La figure 4.11 illustre la convergence de chacun des points de cet ensemble vers le front de Pareto obtenu analytiquement. Pour plus de lisibilité, les points intermédiaires de chaque séquence ne sont pas représentés. Comme dans le cas précédent, nous illustrons la convergence de

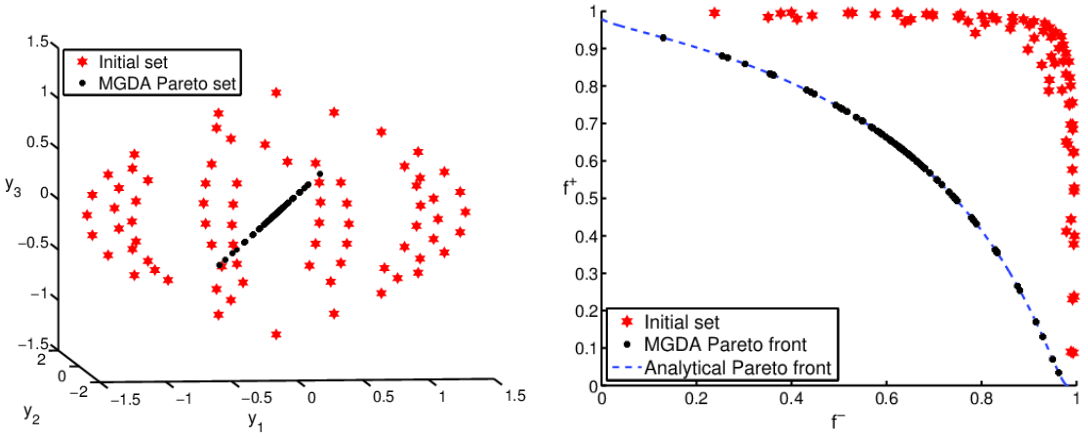


FIGURE 4.11: Convergence de MGDA vers l'ensemble Pareto optimal, depuis un ensemble de points répartis sur une sphère.

MGDA vers le front de Pareto, pour quelques cas. La figure 4.12 illustre l'évolution point par point de l'état initial vers le front, pour les trois cas proposés. Dans l'espace des critères, colonne de droite, le résultat est comparé avec le front obtenu analytiquement. La figure 4.13 compare les résultats obtenus avec MGDA et ceux obtenus avec PAES sur le cas test proposé par Fonseca. Pour MGDA, nous considérons 12 points initiaux pour lesquels le front de Pareto est atteint, en moyenne en 6 itérations. Pour PAES, à partir de deux points initiaux distincts, nous avons créé 2 générations de 50 points. Selon le fonctionnement de PAES [KC00], seuls les points de l'ensemble non-dominé, parmi ceux générés, sont retenus. Le coût en termes de calcul est résumé dans le tableau suivant :

Optimiseur	Nombre de points calculés
MGDA	72
PAES	100

Il faut cependant noter que le calcul des gradients n'est pas pris en compte dans le résumé. En effet, MGDA, comme tout algorithme basé sur l'utilisation des gradients,

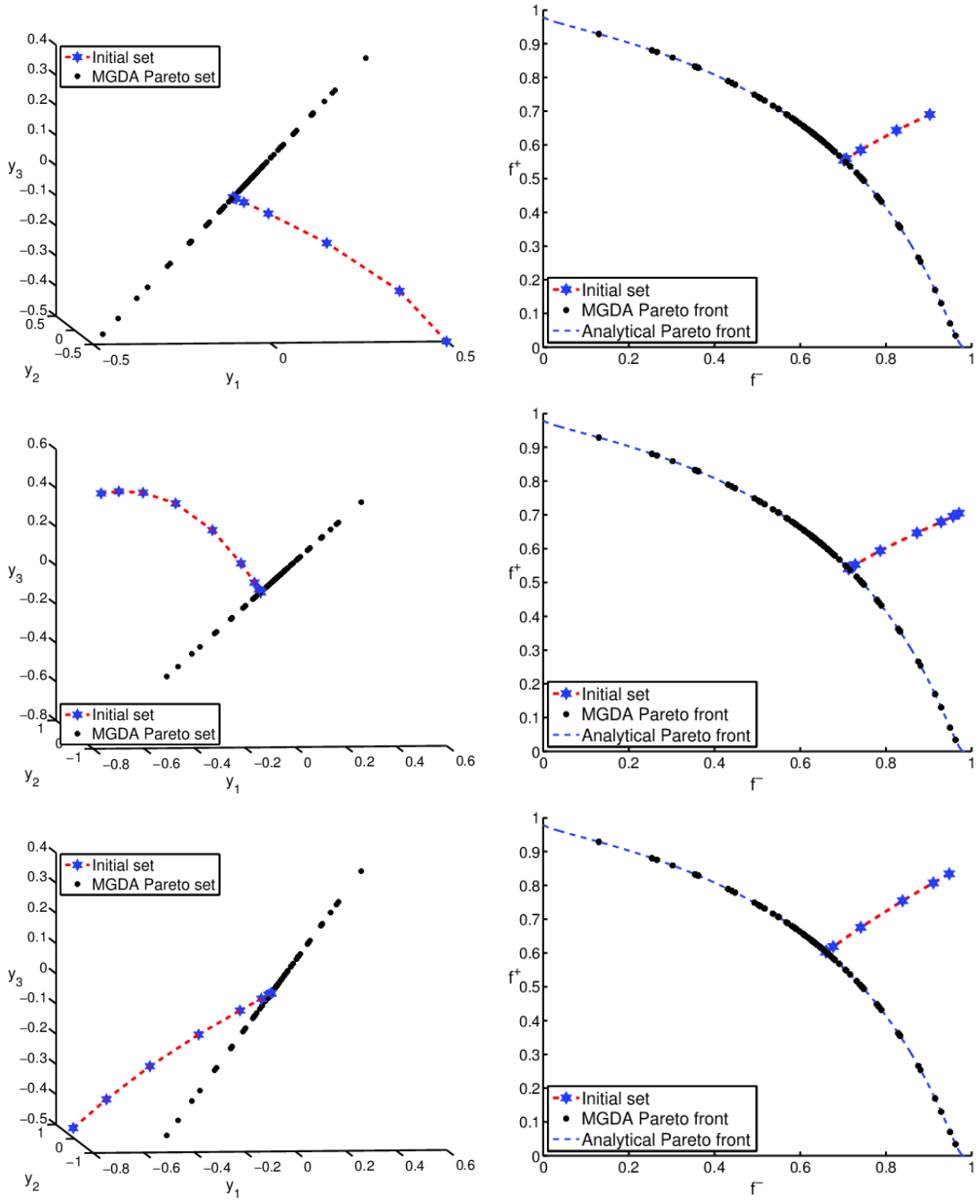


Figure 4.12: Convergence vers l'ensemble et le front de Pareto grâce à MGDA pour trois points initiaux distincts.

reste efficace pour un coût raisonnable si l'évaluation des gradients ne demande qu'un faible temps de calcul supplémentaire. Le coût de calcul des fonctions objectifs est le même pour les deux méthodes. De plus, les points qui résultent de la convergence de MGDA représentés sont volontairement choisis pour couvrir une large partie du front

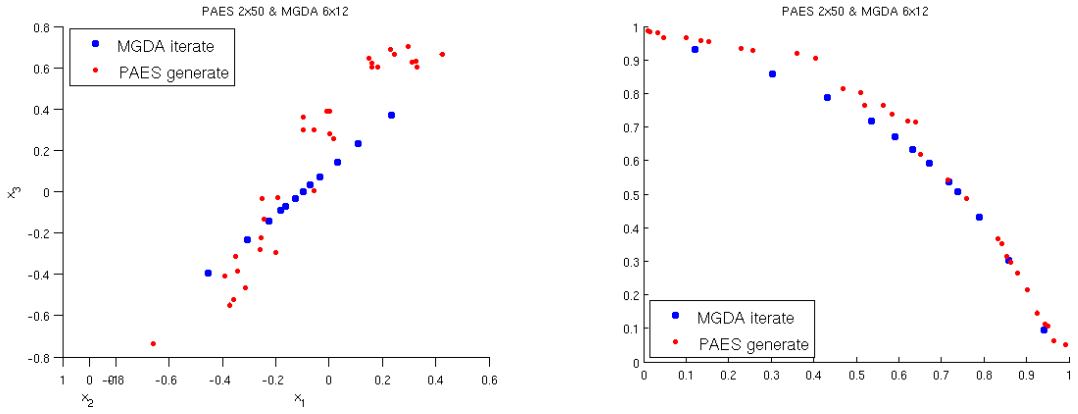


Figure 4.13: Comparaison entre le résultat obtenu avec MGDA et celui obtenu avec PAES.

analytique. En comparaison, PAES offre naturellement cette variété.

4.5 Bilan

Dans ce chapitre, nous avons considéré un problème de minimisation simultanée de n critères réguliers $(J_i)_{1\leq i\leq n}$. Nous avons introduit la notion de Pareto-stationnarité qui peut être abordée comme une formulation plus faible de la Pareto-optimalité. Nous avons montré que lorsque l'ensemble des critères $(J_i)_{1\leq i\leq n}$ n'est pas Pareto-stationnare en un point de l'espace de recherche, il existe une direction commune $-\omega$ dans laquelle tous les critères diminuent, localement. Le vecteur ω appartient à l'enveloppe convexe de l'ensemble des critères. Cette caractéristique nous mène à la définition de MGDA (*Multiple Gradient Descent Algorithm*). Cet algorithme est une extension au cadre multiobjectif des méthodes classiques d'optimisation par descente de gradient. Il permet la mise en place d'une *coopération* entre les critères. En effet, au cours de la phase de descente, tous les critères considérés sont réduits, simultanément. Cette application est réalisable tant qu'un point Pareto-stationnaire n'est pas atteint.

Nous avons ensuite montré que cet algorithme permet d'obtenir un échantillon représentatif du front de Pareto, pour des problèmes de minimisation de fonctions régulières. Cette méthode déterministe est rapide et efficace. En effet, à partir d'un point quelconque de l'espace de recherche, MGDA converge vers un point du front. De plus, l'approximation obtenue est meilleure qu'avec les méthodes évolutionnaires. Comme nous l'avons montré dans le chapitre 3, ces dernières montrent des faiblesses à l'approche du front. La possibilité d'utilisation de cet algorithme pour un nombre illimité de critères

constitue un avantage sur les algorithmes génétiques, limités à quelques critères.

Cependant, l'application de MGDA nécessite la connaisance des gradients, au point courant. Dans le cadre que nous venons d'étudier, les fonctions sont régulières et nous pouvons évaluer les gradients analytiquement. Cependant, si nous sommes confrontés à un problème d'ingénierie où les critères sont évalués par le biais de simulations numériques, l'obtention des gradients devient problématique. En effet, si nous devons évaluer ces gradients par différences finies, pour un problème composé de N paramètres, cette opération demande $2N$ évaluations supplémentaires à l'évaluation des critères. Dans le cadre de critères de type aéronautique, une évaluation peut prendre plusieurs heures, même si le calcul est distribué sur plusieurs processeurs. Il n'est donc pas envisageable d'utiliser cette méthode. Si les gradients ne sont pas fournis au cours de la simulation, par l'utilisation d'une méthode adjointe, par exemple, l'application directe de MGDA n'est pas envisageable.

Chapitre 5

MGDA assisté par métamodèle

Nous avons introduit, dans le chapitre précédent, un algorithme d'optimisation multiobjectif basé sur l'utilisation des gradients. Comme nous l'avons montré, il se révèle d'une grande efficacité dans la résolution de problèmes de minimisation de critères analytiques réguliers. Les principaux avantages de cette méthode sont le caractère déterministe de l'algorithme, la coopération entre les critères et la non régression de la phase d'optimisation. En effet, l'algorithme évolue en améliorant à chaque étape les critères.

Nous avons aussi mis en évidence les principales faiblesses de cette méthode. Comme tous les optimiseurs basés sur les directions de descente, MGDA n'est pas adapté á la résolution de problèmes multimodaux. De plus, dans un contexte industriel, l'obtention des gradients associés aux critères considérés devient rapidement un sacerdoce. La définition d'une nouvelle stratégie devient alors indispensable pour étendre l'usage de MGDA à l'ingénierie. Nous proposons ici d'associer la méthode MGDA à la construction d'un métamodèle de type *Krigeage* [CD07, RW06, GM97]. A partir d'une base de donnée des critères, nous entrainons un métamodèle pour chaque objectif. L'optimisation est alors réalisée sur les surfaces de réponse. Ainsi, le calcul des gradients est effectué sur la base de fonctions régulières du modèle de *Krigeage* (un autre type de métamodèle peut être utilisé). Il est alors possible d'évaluer les gradients par différences finies ou de dériver directement la base pour obtenir les gradients analytiques du métamodèle. Les points obtenus sont alors évalués et viennent enrichir la base de données. La nouvelle base de données est utilisée pour réaliser un métamodèle plus précis. On applique ce processus plusieurs fois afin d'obtenir un métamodèle qui se précise sur les zones optimales du problème.

Nous allons expliquer comment nous associons la méthode MGDA à la construction d'un métamodèle. Nous évaluerons ensuite les performances de ce couplage. Dans un premier temps, nous traiterons un cas classique d'optimisation multiobjectifs présenté

par Kursawe [Deb01]. Ce problème consiste à minimiser simultanément deux fonctions régulières. Les gradients peuvent être obtenus analytiquement. La difficulté de ce problème est due à son caractère multimodal. Ce qui constitue le principal handicap des optimiseurs basés sur l'utilisation des gradients de descente. Nous traiterons ensuite un problème d'optimisation plus proche du milieu de l'ingénierie, avec un jeu de paramètres simplifié. Nous allons considérer un profil d'aile d'avion d'affaire. En utilisant l'algorithme MGDA assisté par la construction d'un métamodèle, nous traiterons un problème d'optimisation biobjectif et bipoint. En condition subsonique, nous cherchons à maximiser le coefficient de portance. En condition transonique, nous cherchons à minimiser le coefficient de traînée.

5.1 Description de la méthode

L'application de l'algorithme MGDA demande la connaissance des gradients de chaque critère, relativement aux paramètres considérés. Or, il n'est que rarement possible de les obtenir avec précision. De plus, lorsque cela est possible, il s'agit d'opérations coûteuses, en termes de temps de calcul et de ressources informatiques. Nous proposons ici une variante de l'algorithme d'optimisation MGDA basée sur la construction itérative de métamodèles. Le fonctionnement de l'algorithme est résumé par la figure 5.1. Afin d'éviter les problématiques complexes induites par l'obtention des gradients, nous les évaluons ici par différences finies sur le métamodèle. Cette opération est rapide et le calcul ne mobilise que peu de ressources. Pour cela, nous devons disposer d'une base de données initiale. Afin d'obtenir une répartition équilibrée sur l'ensemble du domaine de recherche, cette dernière peut être réalisée, par exemple, selon un hypercube latin (LHS : *Latin Hypercube Sample* [IRD80]). Cette base de données est alors utilisée pour entraîner un métamodèle de type *Krigeage*. Nous appliquons alors MGDA à chacun des points de la base de données sur le métamodèle. Chaque évaluation des fonctions objectif et de leurs gradients, au cours de la phase d'optimisation, est réalisée sur le métamodèle. Lorsque cette phase est terminée, nous disposons d'un ensemble de points supposés Pareto-stationnaires par rapport à l'ensemble des métamodèles des critères. Nous soumettons cet ensemble de points à un filtre afin de n'évaluer par le solveur numérique que des points significatifs. Si deux points sont proches dans l'espace des paramètres K, un seul sera considéré. Les points qui se situent en dehors du domaine de recherche sont éliminés. Les points sélectionnés sont alors évalués par simulation et nous complétons la base de données considérée par ces points. Cette nouvelle base vient remplacer la précédente pour l'itération suivante.

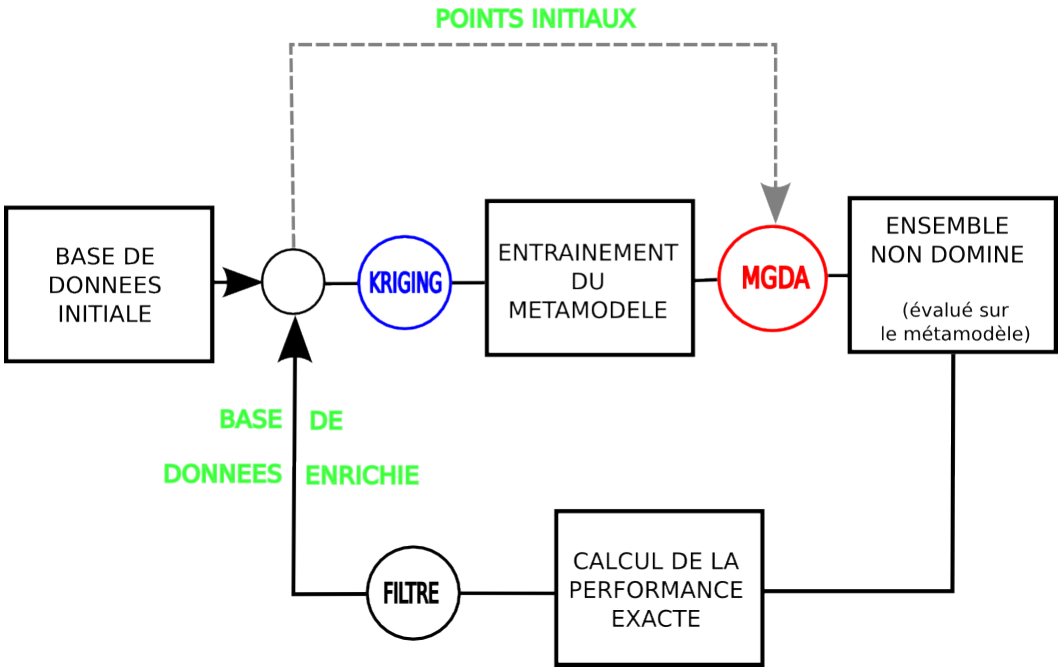


Figure 5.1: Schéma récapitulatif du fonctionnement de l'algorithme MGDA assisté par métamodèle.

5.2 Cas test de Kursawe

5.2.1 Description du cas test

Le cas test suivant est proposé par Kursawe et utilisé par Deb dans [Deb01] avec NSGA-II. Il consiste en la minimisation simultanée de deux fonctions f_1 et f_2 définies de $\mathbb{R}^3$ dans $\mathbb{R}$ par :

$$\begin{cases} f_1(Y) = -\sum_{i=1}^{2} 10 \exp\left(-0.2\sqrt{y_i^2 + y_{i+1}^2}\right) \\ f_2(Y) = \sum_{i=1}^{3} \left(|y_i|^{0.8} + 5\sin y_i^3\right) \end{cases} \tag{5.1}$$

où $Y = (y_1, y_2, y_3) \in \mathbb{R}^3$. Ce cas test est connu pour avoir un front de Pareto continu par morceaux et non convexe. Les deux fonctions à minimiser sont continues sur $\mathbb{R}^3$ et dérivables en tout point où aucune des coordonnées n'est nulle. Nous avons appliqué l'algorithme évolutionnaire multiobjectif PAES [KC00] en utilisant plusieurs points initiaux et nous ne conservons que les points non dominés parmi l'ensemble des points des archives issus de chaque point initial. L'ensemble final constitue un échantillon représentatif du front de Pareto. Le résultat que nous obtenons est très proche de celui

obtenu avec NSGA-II [Deb01]. Ce résultat est illustré par la figure 5.2. Nous pouvons

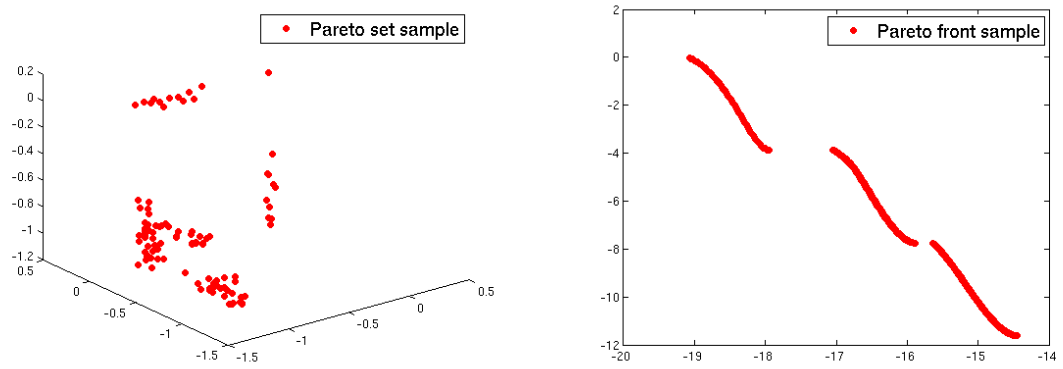


FIGURE 5.2: Représentation d'un échantillon représentatif de l'ensemble de Pareto et du front de Pareto obtenu par application de PAES.

constater les discontinuités du front de Pareto ainsi que la non connexité de l'ensemble correspondant. La présence de la fonction $x \mapsto \sin x^3$ dans la fonction f_2 introduit des optima locaux. Aussi, notre algorithme, qui est une extension de la méthode de plus grande descente aux problèmes multiobjectif, ne pourra pas être efficace.

5.2.2 Collaboration entre PAES et MGDA

Pour ce cas test dont les gradients sont accessibles analytiquement :

$$\nabla f_1(Y) = -2 \begin{pmatrix} \frac{y_1}{\sqrt{y_1^2+y_2^2}} \exp\left(-0.2\sqrt{y_1^2+y_2^2}\right) \\ \frac{y_2}{\sqrt{y_1^2+y_2^2}} \exp\left(-0.2\sqrt{y_1^2+y_2^2}\right) + \frac{y_2}{\sqrt{y_2^2+y_3^2}} \exp\left(-0.2\sqrt{y_2^2+y_3^2}\right) \\ \frac{y_3}{\sqrt{y_2^2+y_3^2}} \exp\left(-0.2\sqrt{y_2^2+y_3^2}\right) \end{pmatrix}$$

$$\nabla f_2(Y) = \begin{pmatrix} 0.8\mathrm{sgn}(y_1)|y_1|^{-0.2} + 15y_1^2 \cos y_1 \\ 0.8\mathrm{sgn}(y_2)|y_2|^{-0.2} + 15y_2^2 \cos y_2 \\ 0.8\mathrm{sgn}(y_3)|y_3|^{-0.2} + 15y_3^2 \cos y_3 \end{pmatrix}$$

Il nous est donc encore possible d'appliquer la méthode MGDA pour un coût raisonnable. Le caractère fortement multimodal du problème contrarie cependant une application directe de cet algorithme. Nous avons noté, par ailleurs, que les algorithmes de type évolutionnaire sont performants pour résoudre ce type de problème. Outre le coût de calcul, ces méthodes souffrent d'imprécision, localement, à l'approche du front de Pareto. Au contraire, MGDA est inapplicable globalement pour résoudre ce type de problème. La figure 5.3 illustre cette collaboration. Nous couplons les deux méthodes

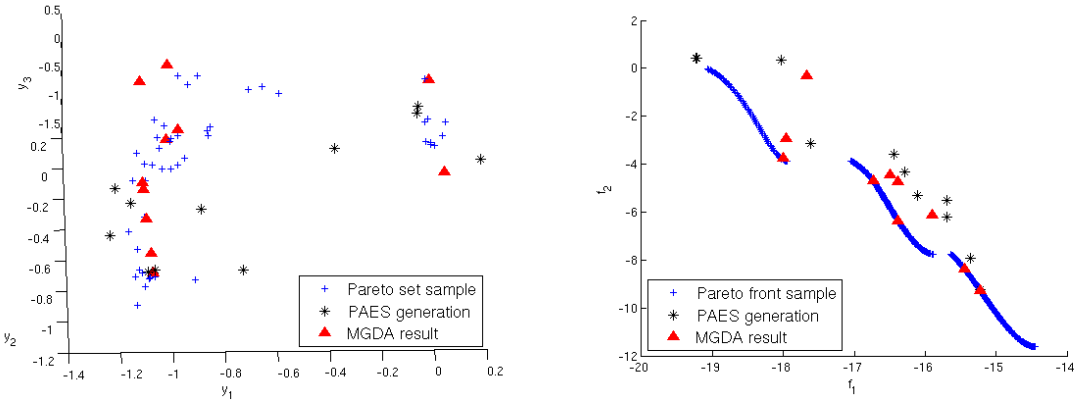


FIGURE 5.3: Application de l'algorithme MGDA à l'archive donnée par PAES appliqué au cas test de Kursawe.

pour en tirer les avantages respectifs. Nous appliquons PAES en un point de l'espace de coordonnées $(2, -1, 1)$, choisi arbirairement, afin de générer 100 points. L'archive finale obtenue comporte 11 points non-dominés, dans l'espace des critères. PAES offre un échantillon de points proches du front de Pareto et répartis sur sa longueur. Ainsi, la méthode évolutionnaire donne un échantillon de points qui se rapproche de la solution et présente une grande variété. Nous appliquons alors MGDA en chacun de ces points, en utilisant les gradients calculés analytiquement. MGDA permet, pour certains points, de converger vers le front de Pareto avec une grande précision. En revanche, d'autres points sont améliorés sans se placer sur le front. Le couplage des deux méthodes se révèle efficace, mais trop de points ne subissent aucune modification. La forte multimodalité de ce problème est certainement la cause de cet immobilisme local.

5.2.3 Application de MGDA assisté par métamodèle

Nous allons appliquer la méthode MGDA assistée par un métamodèle de *Krigeage*. Pour ce faire, nous considérons une base de données initiale constituée de 30 points répartis selon un hypercube latin [MMC00] dans l'espace de recherche $K = [-5, 5]^3$. Comme expliqué dans la section 5.1, nous utilisons la base de données pour réaliser un métamodèle de type *Krigeage* pour f_1 et une autre pour f_2. Nour notons alors $\tilde{f}_1(Y)$ et $\tilde{f}_2(Y)$ les valeurs des métamodèles de f_i en Y. Nous appliquons alors MGDA à chaque point de la base de données afin de résoudre le problème de minimisation :

$$\min \left\{ \left(\tilde{f}_1(Y), \tilde{f}_2(Y) \right), \quad Y \in K \right\} \tag{5.2}$$

Les points de l'ensemble $\tilde{Y}$, solution du problème 5.2, sont alors soumis à un filtre, afin d'éliminer les points trop proches (dans K) entre eux et de la base de données initiale. Les points sélectionnés sont évalués par f_1 et f_2 afin de compléter la base de données. La base de données est alors utilisée pour réaliser deux nouveaux métamodèles de type *Krigeage* pour effectuer un second cycle. La figure 5.4 illustre l'application de la méthode à 5 reprises. L'échantillon du front de Pareto obtenu par PAES y est aussi représenté, à titre de comparaison. Les fronts qui correspondent à la base de données initiale et finale sont aussi représentés. A partir de la troisième étape, les fronts successifs

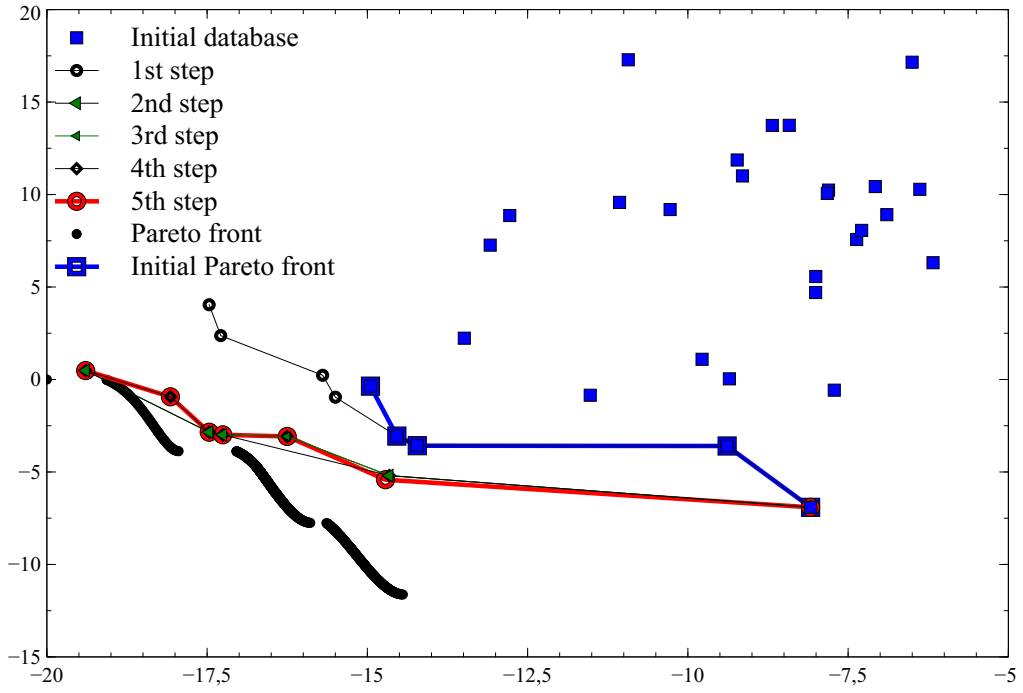


Figure 5.4: Evolution de MGDA assisté par métamodèle appliqué au problème de Kursawe, dans le plan (f_1, f_2). Comparaison entre les ensembles non dominés initial et final.

obtenus à l'aide de MGDA assisté par métamodèle sont très proches. La méthode n'est alors plus apte à réduire les objectifs significativement. De plus, nous constatons un déséquilibre dans le traitement des critères. En effet, l'algorithme est proche du front pour la minimisation de f_1 tandis que f_2 n'est pas amélioré par rapport à la base de données initiale. Nous résumons dans le tableau 5.1 l'évolution des valeurs optimales des critères au fil des itérations, ainsi que le nombre de points qui composent les fronts successifs. Sur la partie haute du front, quelques points, issus de la convergence de l'algorithme, sont proches du front de Pareto, mais MGDA ne peut les améliorer au

Itération	0	1	2	3	4	5
Taille Base de données	30	45	57	65	69	79
Points non-dominés	5	8	5	6	7	7
minimum f_1	-14.95	-17.47	-19.39	-19.39	-19.39	-19.39
minimum f_2	-6.91	-6.91	-6.91	-6.91	-6.91	-6.91

Table 5.1: Récapitulatif des valeurs minimales des critères et de la taille des bases de données au fil des itérations.

fil des étapes suivantes. L'échec de l'application de la méthode à ce cas test peut être provoqué par plusieurs facteurs :

- la base de données initiale ;
- la mauvaise qualité du modèle de *Krigeage* utilisé ;
- le caractère multimodal du problème étudié.

L'expérience a été réalisée successivement, en considérent différentes bases de données initiales. Ormis pour qelques points, le résultat obtenu reste similaire. Peu de points issus de la convergence de MGDA sont proches du front de Pareto et l'algorithme cesse de progresser à partir d'une dizaine d'itérations.

5.3 Optimisation d'un profil d'aile d'avion d'affaires

5.3.1 Condition d'étude

Le profil d'une aile d'avion est conçu de telle manière que l'écoulement autour de l'aile engendre une différence de pression entre la partie supérieure (*extrados*) et la partie inférieure (*intrados*). Ceci est illustré par la figure 5.5. La force qui résulte de cette

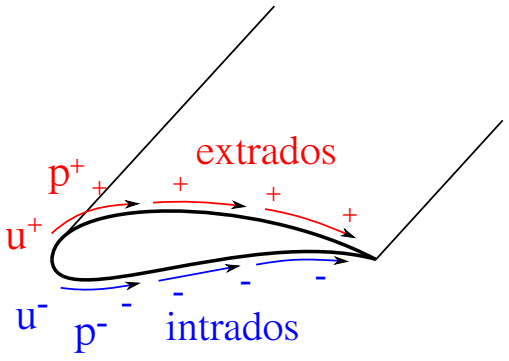


Figure 5.5: Distribution de pression et de vitesse autour d'une aile.

différence de pression est appelée *portance*. Le direction de cette force est perpendiculaire à la direction donnée par l'écoulement $\overrightarrow{U_\infty}$ et son amplitude sera notée L (*lift* en anglais). Si L est assez grand pour compenser le poids de l'avion, celui-ci s'envole. La forme d'une aile peut être définie de plusieurs manières. Ici, nous n'exprimerons que la conception à partir d'un profil d'aile extrudé homothétiquement. Le profil de l'aile correspond à une coupe de l'aile effectuée orthogonalement au plan d'envergure. En plus de la force de portance, les distributions de pression et de frottements exercent sur l'aile une force dont la direction est identique à celle de l'écoulement $\overrightarrow{U_\infty}$. Cette force résultante est appelée *traînée* et son intensité est notée D (*drag*, en anglais). L'ensemble des points de l'aile qui entrent en contact avec le fluide en premier est appelé *bord d'attaque*. Inversement, l'ensemble des points de l'aile qui entrent en contact avec le fluide en dernier est appelé *bord de fuite*. s correspond à l'abscisse curviligne du profil. Ceci est illustré par la figure 5.6. Comme nous venons de l'indiquer, les forces

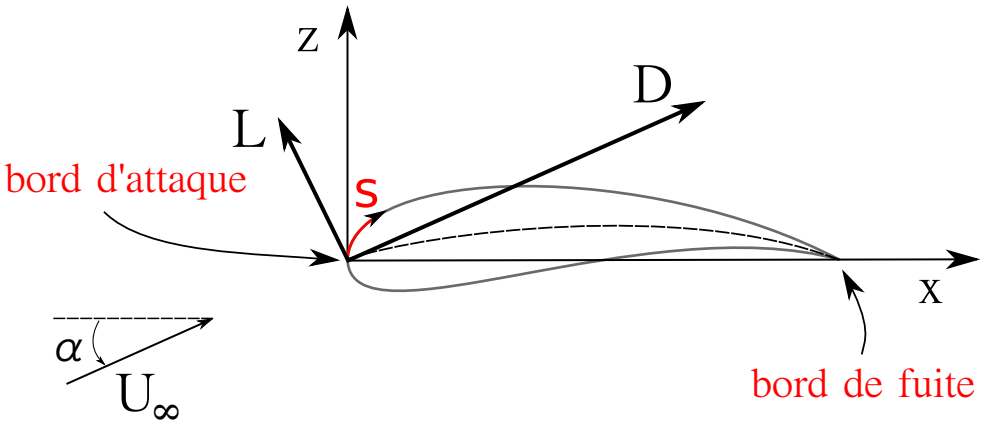


Figure 5.6: Définition d'un profil d'aile et représentation des efforts aérodynamiques.

de traînée et de portance sont des forces résultantes. En effet, elles sont la somme de forces surfaciques exercées sur les surfaces élémentaires dont la réunion recouvre l'aile. Nous allons exprimer ces forces élémentaires, en fonction de l'angle d'attaque (AoA) noté α. Comme illustré par la figure 5.7, nous distinguons les forces exercées sur une surface élémentaire de l'extrados et de l'intrados. Nous notons $p_e(s)$ (resp $p_i(s)$) la résultante des forces de pression exercées par le fluide sur la surface élémentaire ds_e (resp ds_i) située sur l'extrados (resp intrados) de l'aile. Nous notons $\tau_e(s)$ (resp $\tau_i(s)$) la résultante des forces de frottement visqueux exercées par le fluide sur la surface élémentaire ds_e (resp ds_i) située sur l'extrados (resp intrados) de l'aile. Nous notons N

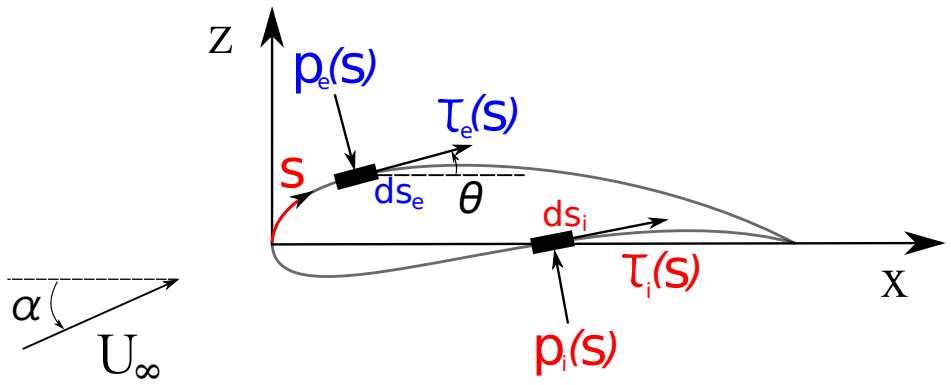


Figure 5.7: Distribution de pression sur l'intrados et l'extrados d'un profil d'aile.

la résultante des forces aérodynamiques dans la direction normale au plan qui contient les bords d'attaque et de fuite de l'aile, dirigée positivement selon (Oz). Nous notons A la résultante des forces aérodynamiques axiales, dirigée positivement selon (Ox). Sur un profil donné, nous notons b_a le point qui correspond au bord d'attaque et b_f le point qui correspond au bord de fuite. Alors, le calcul des forces résultantes s'effectue de la manière suivante :

– Sur l'extrados, sur une surface élémentaire ds_e :

$$\begin{aligned} dN_e &= -p_e ds_e \cos\theta + \tau_e ds_e \sin\theta \\ dA_e &= p_e ds_e \sin\theta + \tau_e ds_e \sin\theta. \end{aligned}$$

– Sur l'intrados, sur une surface élémentaire ds_i :

$$\begin{aligned} dN_i &= p_i ds_i \cos\theta + \tau_i ds_i \sin\theta \\ dA_i &= -p_i ds_i \sin\theta + \tau_i ds_i \sin\theta. \end{aligned}$$

Les forces normales et axiales résultantes, par unité d'envergure, s'obtiennent par intégration des forces élémentaires côté extrados et côté intrados, entre le bord d'attaque et le bord de fuite :

$$N = \int_{b_a}^{b_f} \left(-p_e \cos\theta + \tau_e \sin\theta\right) ds_e + \int_{b_a}^{b_f} \left(p_i \cos\theta + \tau_i \sin\theta\right) ds_i$$

$$A = \int_{b_a}^{b_f} \left(p_e \sin\theta + \tau_e \cos\theta\right) ds_e + \int_{b_a}^{b_f} \left(-p_i \sin\theta + \tau_i \cos\theta\right) ds_i.$$

On en déduit alors la portance et la traînée, par unité d'envergure, en fonction de l'angle d'attaque α :

$$L = N\cos\alpha - A\sin\alpha, D = N\sin\alpha + A\cos\alpha.$$

On note ρ la densité du fluide dans lequel pénètre l'aile. On note S la surface totale de l'aile. Cette surface est appelée voilure de l'aile. On définit la pression dynamique du fluide considéré par :

$$q_\infty = \frac{1}{2}\rho U_\infty^2 \tag{5.3}$$

A l'aide de cette valeur caractéristique de l'écoulement, nous pouvons définir les coefficients aérodynamiques de l'aile :

- le coefficient de portance :

$$C_L = \frac{L}{q_\infty S} \tag{5.4}$$

- le coefficient de trainée :

$$C_D = \frac{D}{q_\infty S} \tag{5.5}$$

La forme d'une aile d'avion est étudiée de telle manière que la portance (L) exercée sur la voilure soit maximale et l'intensité de la traînée (D) soit minimale. Durant les phases d'atterrissage et de décollage, le régime d'écoulement de l'air sur la surface de l'aile est subsonique. A un tel régime, l'intensité de la traînée, proportionelle au carré de la vitesse, est forcément faible. La forme d'aile optimale correspond alors à celle qui offre la meilleure portance. Inversement, durant la phase de vol de croisière, le régime d'écoulement d'air sur la surface de l'aile est transonique (ou supersonique, selon le type d'avion étudié). A un tel régime, l'intensité de la portance, proportionelle au carré de la vitesse, est nécessairement élevée. La forme de l'aile optimale correspond alors à celle qui offre une intensité de traînée minimale. Cependant, comme indiqué, une bonne aile doit subir une faible traînée, pour réduire la consommation de carburant et une portance élevée, pour faire voler l'appareil. Il est alors impossible de trouver une forme unique qui satisfasse parfaitement ces deux critères. Afin de pallier ce problème, les avions de ligne sont équipés d'ailes à géométrie variable. Ainsi, par le biais de volets situés sur les bords d'attaque et de fuite de l'aile, la surface de la voilure est augmentée au décollage et à l'atterrissage afin d'amplifier la portance et les volets sont ensuite rentrés afin de réduire la traînée durant la phase de vol. Le profil d'aile présenté ici ne possède pas de volets rétractables. La géométrie de l'aile doit alors être adaptée aux conditions subsoniques et transoniques simultanément, sans recourir à ces modifications. Nous allons donc nous intéresser au problème d'optimisation biobjectif et bipoint, que nous écrivons comme

un problème de minimisation, résumé par l'équation 5.6.

$$\begin{cases} \min\limits_{Y \in K} \Big\{ C_D(Y), \text{ en régime transonique} \Big\} \\ \min\limits_{Y \in K} \Big\{ -C_L(Y), \text{ en régime subsonique} \Big\} \end{cases} \tag{5.6}$$

5.3.2 Description du cas test

Dans cette section, nous étudions l'optimisation du profil d'aile d'un avion d'affaires, plongé dans un écoulement de type Euler compressible. Nous définissons une section d'aile à l'aide de points de contrôle. L'aile est ensuite réalisée par un prolongement homothétique de cette section. Seule la forme de cette section est donc optimisée. La forme de référence est définie par une courbe B-Spline pour la surface supérieure de l'aile et une courbe B-Spline pour la surface inférieure. Le point d'attaque et le point de traînée sont fixés et 10 points de contrôle mobiles constituent les variables d'optimisation. Ces variables sont initialisées de manière à obtenir un profil d'aile initial proche de celui du NACA0012. Chaque point peut se déplacer verticalement, dans un intervalle donné. La figure 5.8 résume la construction du profil 3D de l'aile. Nous proposons ici la résolution d'un problème d'optimisation bicritère et bipoint. Celui-ci est décrit par le système 5.6. L'espace de recherche K est un sous-espace de $\mathbb{R}^{10}$. Chaque point de contrôle peut se déplacer sur l'axe vertical, dans un intervalle centré sur le point correspondant à la configuration nominale. Nous allons chercher une approximation du front de Pareto du problème d'optimisation que nous venons de décrire en utilisant l'optimiseur multiobjectif MGDA assisté par la construction d'un métamodèle de type *Krigeage*. La base de données initiale est composée d'un ensemble de 40 points répartis sur l'ensemble de recherche selon un hypercube latin [MMC00]. A partir de cette base de données, nous construisons un métamodèle pour chaque critère considéré. Nous appliquons alors la méthode MGDA à chaque point de la base générée afin de minimiser les fonctions $\tilde{C_D}$ et $-\tilde{C_L}$ sur le domaine K. Afin de simplifier la paramétrisation de l'optimiseur, chaque variable est normalisée afin de réduire l'espace de recherche au domaine $[-1, 1]^{10}$. Comme précédemment, l'ensemble de points obtenu est soumis à un filtre de répartition sur le domaine de recherche. La forme d'aile définie par les paramètres sélectionnés est alors évaluée dans les conditions subsonique puis transonique décrites afin d'évaluer sa performance en terme de portance et de traînée, respectivement. La base de données est alors complétée par ces nouveaux points et une nouvelle itération est effectuée. Dans le cas présent, nous considérons que les régimes correspondent aux conditions suivantes :

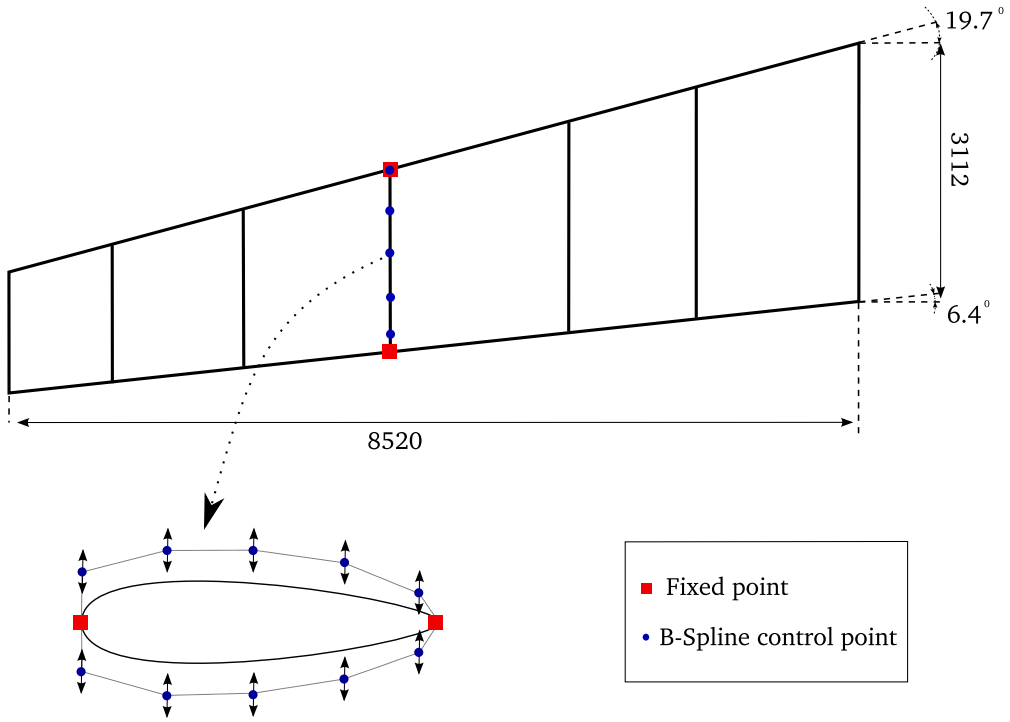


FIGURE 5.8: Paramétrisation avec des B-Splines de la forme de l'aile et points de contrôle.

– régime subsonique :

$$M_\infty = 0.3, AoA = 8^o$$

– régime transonique :

$$M_\infty = 0.8, AoA = 2^o$$

où M_∞ est une grandeur sans dimension caractéristique de l'écoulement, appelée *Nombre de Mach*. Si l'on note c la célérité du son dans le fluide considéré, M_∞ est défini par l'équation :

$$M_\infty = \frac{U_\infty}{c}$$

5.3.3 Résultats

Nous avons réalisé six itérations de la méthode que nous venons de décrire. Nous allons en présenter les résultats. La figure 5.9 illustre la convergence pas à pas de MGDA assisté par métamodèle. Seules les performances exactes (calculées par le solveur Euler) sont illustrées. De plus, les ensembles non dominés sont tracés pour la base de données initiale et pour la base de données finale. Pour chaque itération, nous représentons sur la figure 5.9 les performances calculées avec le solveur Euler compressible obtenues par les points finaux de MGDA. Les points rejetés par le filtre de répartition au sein

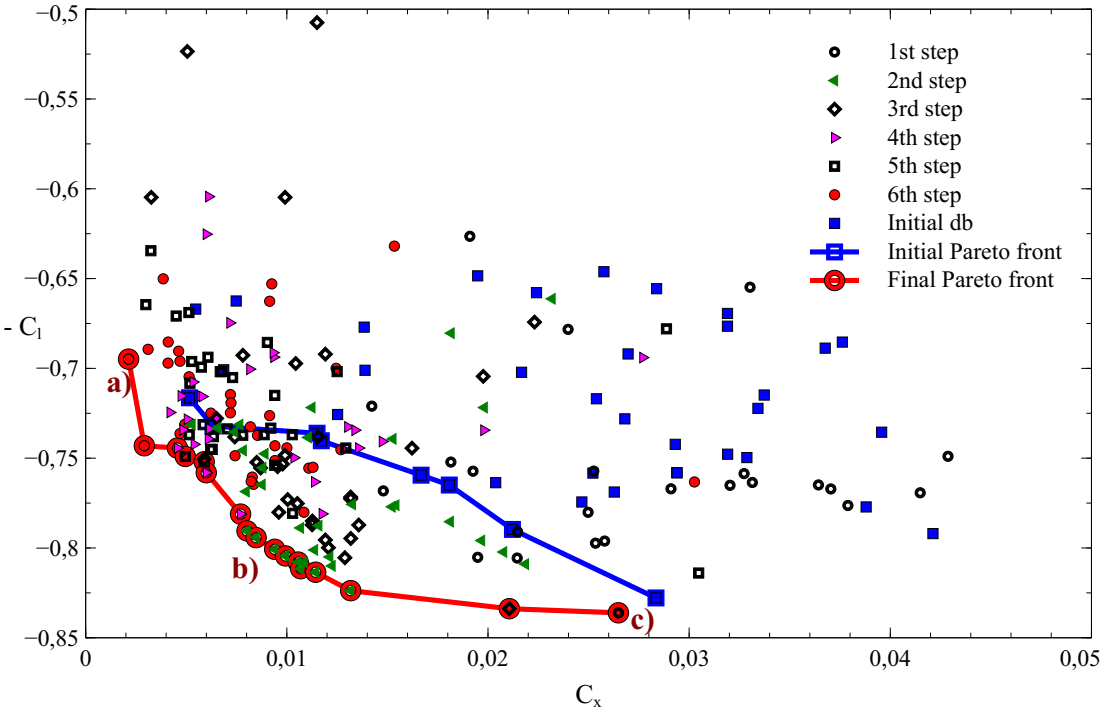


Figure 5.9: Evolution de MGDA assisté par métamodèle appliqué au problème classique de minimisation du coefficient de traînée et maximisation du coefficient de portance simultanément. Comparaison entre les ensembles non dominés initial et final.

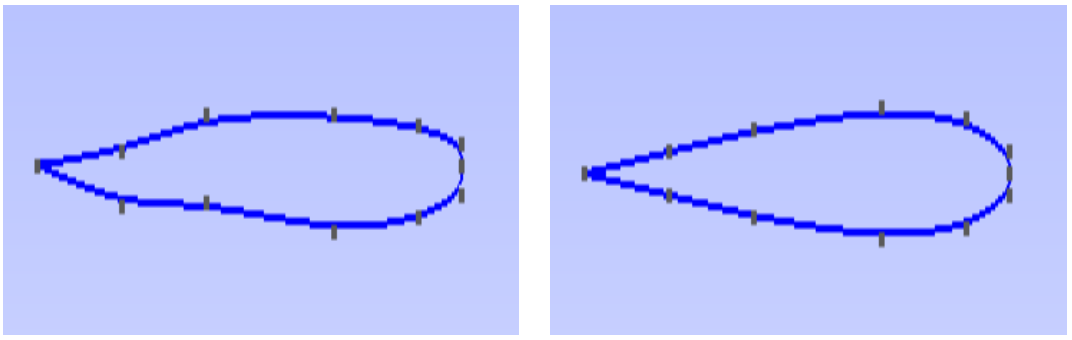
du domaine ne sont pas évalués. Ainsi, nous pouvons remarquer une progression de l'ensemble des nouveaux points vers le front de Pareto. De plus, la longueur du front final nous indique que MGDA assisté par métamodèle conserve la variété de points fournie par la répartition, de type LHS, initiale, au fil des itérations. Le nombre de points évalués, le meilleur coefficient de traînée et le meilleur coefficient de portance obtenus sont indiqués dans le tableau 5.10. Contrairement au coefficient de portance,

Itération	0	1	2	3	4	5	6
Taille Base de données	40	64	98	129	158	189	223
Points non-dominés	8	8	16	15	15	17	17
minimum traînée $\times 10^{-3}$	5.16	5.16	5.16	3.26	3.26	2.99	2.13
maximum portance $\times 10^{-1}$	8.28	8.36	8.36	8.36	8.36	8.36	8.36

Figure 5.10: Récapitulatif des valeurs minimales des critères et de la taille des bases de données au fil des itérations.

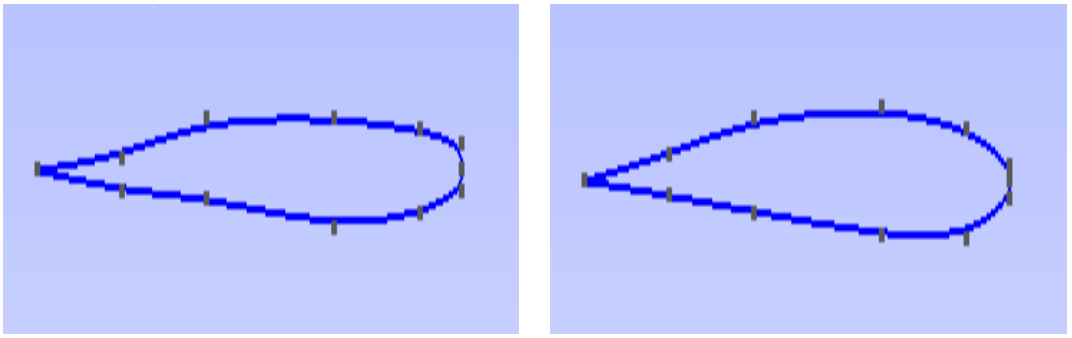
dont la performance maximale est presque atteinte lors de la création de la base de données initiale, la performance en traînée est fortement améliorée au fil des itérations.

En effet, on note une réduction de presque 60% de la meilleure performance en traînée entre la première et la dernière génération. Nous représentons sur la figure 5.11 les profils



a) Traînée optimale

Configuration initiale



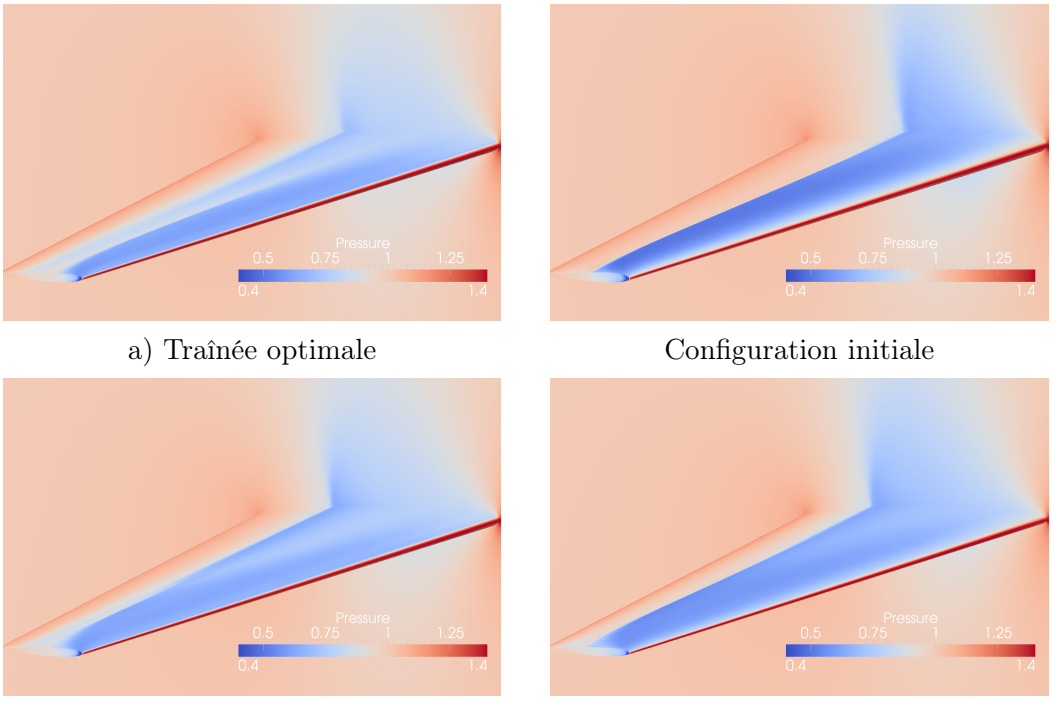
b) Configuration intermédiaire

c) Portance optimale

Figure 5.11: Profil d'aile pour diverses configurations optimales et initiale.

obtenus pour trois configurations du front de Pareto obtenu et pour la configuration initiale. Nous adoptons les mêmes notations que celles utilisée sur la figure 5.9. Ainsi, la forme a) correspond à la meilleure performance obtenue sur le coefficient de traînée, b) à un point intermédiaire du front de Pareto de la dernière génération et c) à la meilleure performance obtenue sur le coefficient de portance. Nous pouvons alors constater que la forme qui offre le meilleur coefficient de traînée (a)) est plus effilée que la forme initiale. En effet, la partie proche du bord d'attaque est plus mince. Il en est de même pour la partie proche du bord de fuite. Au contraire, la forme qui offre le meilleur coefficient de portance (c)) est plus bombée que la forme initiale. En particulier, on constate une augmentation de la surface du profil de l'aile par une évolution affirmée de l'extrados de l'aile. Afin d'observer les résultats en termes de portance et de traînée, nous proposons un comparatif des champs de pression répartis sur les extrados des différentes formes obtenues. Sur chaque comparatif, le champ de pression de la configuration initiale est aussi présent. Ainsi, nous pouvons noter l'évolution apportée par l'application de la méthode. La figure 5.12 compare les résultats obtenus en régime transonique où nous

évaluons les coefficients de traînée. Nous constatons alors que la réduction du coefficient



a) Traînée optimale

Configuration initiale

b) Configuration intermédiaire

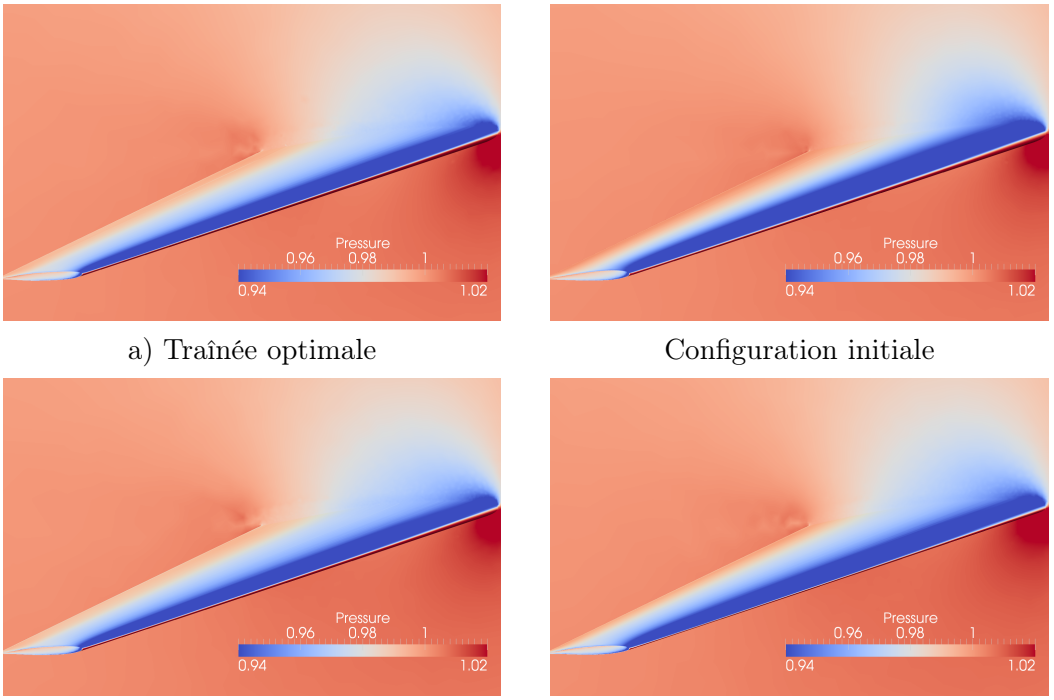
c) Portance optimale

Figure 5.12: Champ de pression sur l'extrados en régime transonique.

de traînée est associée à la réduction de l'intensité du choc sur l'extrados de l'aile. En effet, l'intensité observée est la plus faible sur le profil a). Au contraire, l'intensité du choc observé est plus élevée sur le profil c). L'intensité du choc est aussi importante sur le profil initial. En revanche, la position du choc est déplacée vers l'arrière de l'aile. La figure 5.13 compare les résultats obtenus en régime subsonique où nous évaluons les coefficients de portance. La dépression observée sur l'extrados occupe une large portion de la surface dans le cas du profil où le coefficient de portance est maximal. Au contraire, la surface occupée par la zone de dépression est très faible sur le profil qui donne le coefficient de traînée le plus faible. Nous retrouvons dans ces observations l'antagonisme entre les deux critères.

Comme nous avons défini les coefficients de portance et de traînée par les équations 5.4 et 5.5, nous allons définir un autre coefficient caractéristique de l'écoulement autour du profil d'aile. En tout point j de la voilure, le coefficient de pression exercée par le fluide sur la voilure est défini par :

$$C_{p_j} = \frac{p_j - p_\infty}{q_\infty} \tag{5.7}$$



a) Traînée optimale

Configuration initiale

b) Configuration intermédiaire

c) Portance optimale

Figure 5.13: Champ de pression sur l'extrados en régime subsonique.

Les grandeurs qui interviennent dans cette définition sont :

- q_∞ : pression dynamique, définie précédemment 5.3,
- p_j : pression statique du fluide au point j situé sur la voilure,
- p_∞ : pression du fluide hors du sillage de l'aile.

Afin d'illustrer l'évolution des performances après six itérations de la méthode, nous donnons les représentations des courbes de distribution de la pression sur le profil d'aile, en différentes sections de l'envergure de l'aile. Pour chaque régime étudié, nous donnons les courbes de répartition pour trois formes représentatives de l'ensemble de Pareto de la dernière génération. Nous adoptons encore les notations de la figure 5.9. Les figures 5.14 et 5.15 représentent la distribution de la pression sur le profil d'aile pour le régime transsonique et pour le régime subsonique, respectivement. Tous les points correspondent à des évaluations par simulation Euler compressible réalisée à le solveur développé par Inria : *Num3sis*. Sur chaque figure, le tracé de la courbe C_p est comparé avec celui qui correspond à la forme nominale, à la même position d'envergure. Nous pouvons ainsi remarquer, sur la figure 5.14, que la réduction du coefficient de traînée correspond à une forme qui réduit le choc, en le décomposant en deux plus petits. Nous retrouvons cette caractéristique sur la configuration intermédiaire. Au contraire,

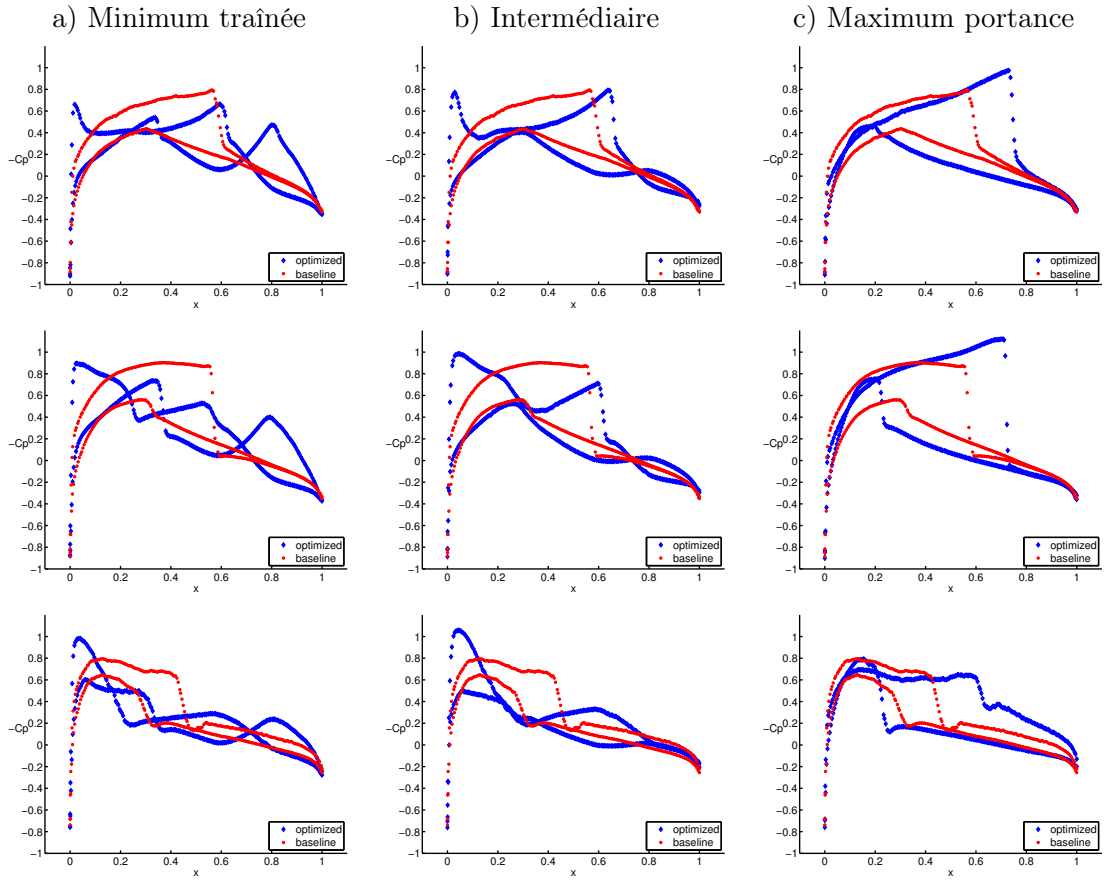


Figure 5.14: Distribution du coefficient de pression en régime transsonique, AoA = 2^o et Mach = 0.8. Sur chaque image, la distribution du coefficient de pression est comparée à celle de la forme initiale. (haut / milieu / bas : 5% / 50 % / 95% envergure).

la configuration qui donne le meilleur coefficient de portance augmente l'intensité du choc. Ainsi, le coefficient de traînée correspondant est lui aussi augmenté. De plus, nous pouvons observer que celui-ci est déplacé vers la droite. La surface délimitée par les courbes C_p de l'extrados et de l'intrados est alors augmentée. Nous constatons sur la figure 5.15 que la valeur du coefficient de portance est directement liée à la surface délimitée par les courbes C_p de l'intrados et de l'extrados. Ainsi, nous retrouvons une surface plus importante dans le cas c) que dans les autres et une surface plus faible dans le cas a).

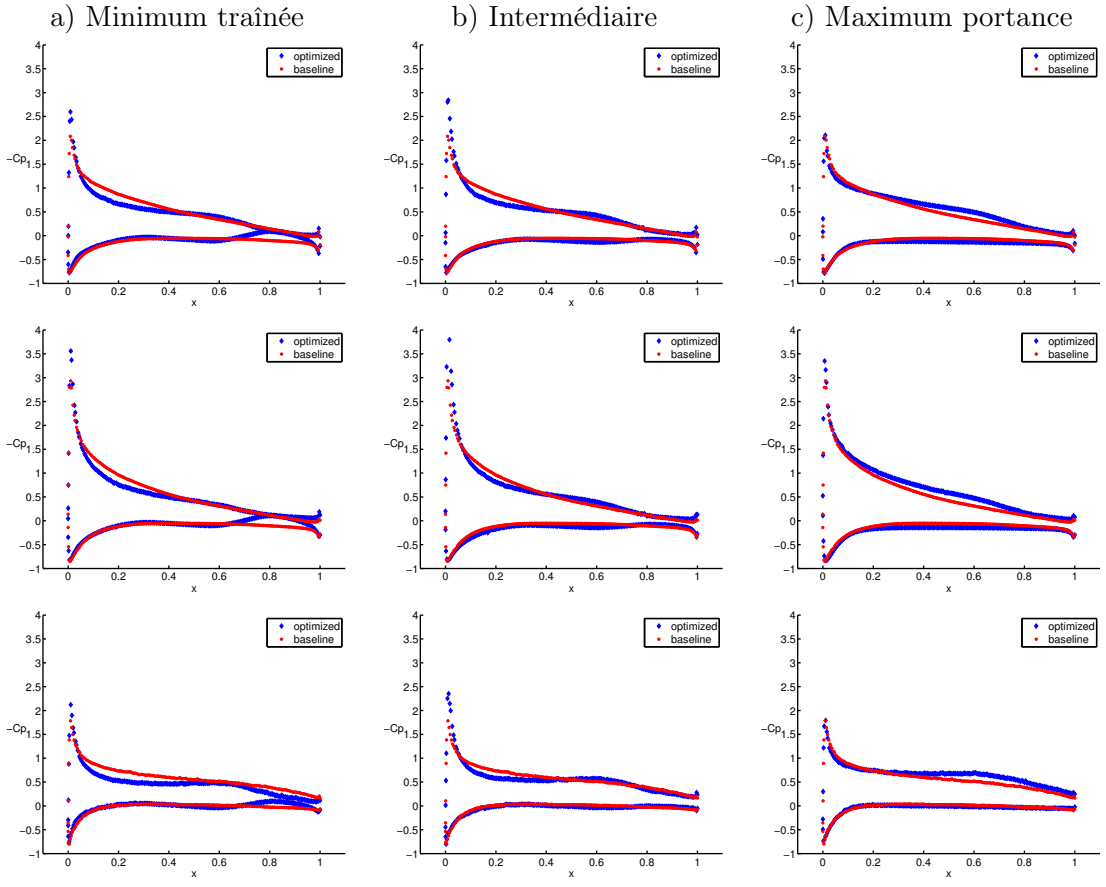


Figure 5.15: Distribution du coefficient de pression en régime subsonique, AoA = 8^o et Mach = 0.3. Sur chaque image, la distribution du coefficient de pression est comparée à celle de la forme initiale. (haut / milieu / bas : 5% / 50 % / 95% envergure).

5.4 Bilan

Dans ce chapitre, nous avons introduit une méthodologie de couplage de l'algorithme d'optimisation multiobjectif MGDA et la construction de métamodèles de type *Krigeage*. Comme nous l'avons précisé, l'objectif de cette association n'est pas de construire une surface de réponse précise sur l'ensemble de l'espace de recherche K. Il s'agit d'affiner la précision en des zones où l'on pense pouvoir localiser les points optimaux. Si l'application de cette méthode s'avère laborieuse sur des cas tests fortement multimodaux (Kursawe), nous obtenons de très bons résultats sur des cas tests industriels simplifiés (Profil d'aile type NACA0012). L'efficacité du couplage pourrait être accrue par une amélioration de l'exploitation du métamodèle, en utilisant des fonctions de mérite [PWG, Gin09, Jon01].

Comme nous l'avons observé dans un chapitre précédent, les algorithmes évolutionnaires attribuent une *performance* aux individus (points) qui est une dégradation de la valeur obtenue par les critères en ce point. Cette modification est réalisée suivant diverses stratégies, selon la méthode étudiée. Ainsi, l'application d'une pénalisation pour les points situés dans une zone dense favorise la diversité, qui fait défaut dans le traitement du cas test de Kursawe par la méthode MGDA associée à la construction d'un métamodèle.

Chapitre 6

Application à un cas test industriel

Nous avons introduit, dans le chapitre 5, une méthode de couplage entre l'algorithme d'optimisation multiobjectif MGDA et la construction de métamodèles de type *Krigeage*. Nous avons ensuite testé cette association sur un cas test analytique classique en optimisation multicritère. Nous avons ensuite appliqué la méthode à la résolution d'un problème d'optimisation aérodynamique simplifié. Les résultats obtenus montrent que la méthode s'avère rapide et précise pour un coût de calcul modéré, pour la résolution de cas relativement simples. En revanche, elle souffre de faiblesses pour le traitement de problèmes multimodaux. Cette défaillance est courante chez les algorithmes basés sur l'utilisation des gradients.

Afin de valider les performances de cet algorithme, nous allons le tester sur un problème à caractère industriel. Le cas test que nous traiterons dans ce chapitre nous a été proposé par RENAULT, dans le cadre du projet ANR, OMD2 (Optimisation Multi Disciplinaire Distribuée).

6.1 Description du problème

Aujourd'hui, la climatisation est devenue la norme sur la majorité des véhicules. En plus du confort d'utilisation accru, ceci permet de maintenir les fenêtres fermées, même durant les périodes de fortes chaleurs. Ainsi, la forme aérodynamique du véhicule n'est pas perturbée. Afin que les usagers profitent au maximum de cet agrément, les constructeurs automobiles veillent à l'optimisation du fonctionnement du système de climatisation et du réseau de diffusion dans l'habitacle. Cet ensemble doit répondre à plusieurs critères. D'une part, le groupe frigo en charge de la production de froid doit fonctionner de manière à maintenir une température constante. D'autre part, les conduites et les diffuseurs doivent tempérer l'habitacle du véhicule de façon homogène.

On s'intéresse ici à la forme d'une conduite de climatisation dont la fonction est de diffuser l'air conditionné dans l'habitacle du véhicule. La figure 6.1 représente la conduite qui fait l'objet de notre étude. Sur cette représentation, l'air circule de la gauche vers la

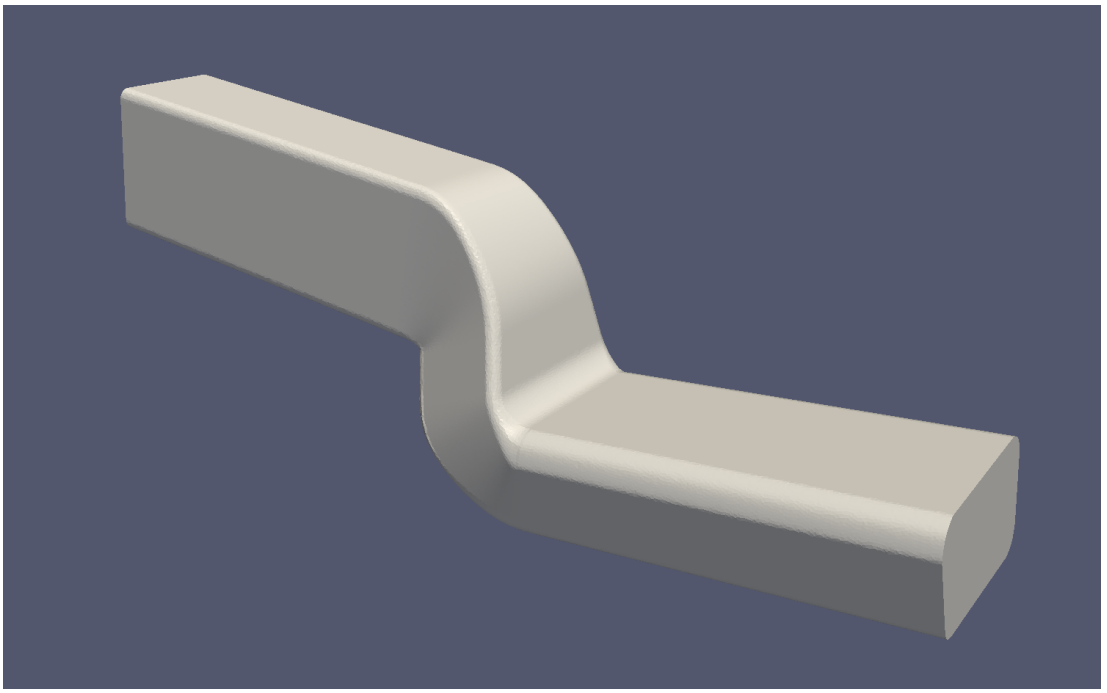


Figure 6.1: Forme de la conduite d'air climatisé.

droite. Cette conduite est utilisée par RENAULT pour diffuser l'air conditionné depuis la pompe vers l'habitacle. Des contraintes d'encombrement imposent les positions des sections droites d'entrée et de sortie. Aussi, seule la forme du coude intermédiaire entre ces deux sections pourra être modifiée. Avant de tester nos méthodes d'optimisation afin d'adapter la forme du coude de la conduite, nous définissons un jeu de paramètres qui décrit sa forme.

6.1.1 Définition de la paramétrisation

Comme nous l'avons précisé, notre étude se résume à la recherche de la forme optimale de la jonction entre les deux sections droites. Le coude est défini par un jeu de 8 paramètres répartis de la manière suivante :

- 3 angles : X_2, X_5 et X_6 ;
- 5 longueurs : X_1, X_3, X_4, X_7 et X_8.

La géométrie de la conduite ainsi que les paramètres qui la définissent sont représentés par la figure 6.2. Ainsi, nous utilisons à la fois des longueurs et des angles. De plus,

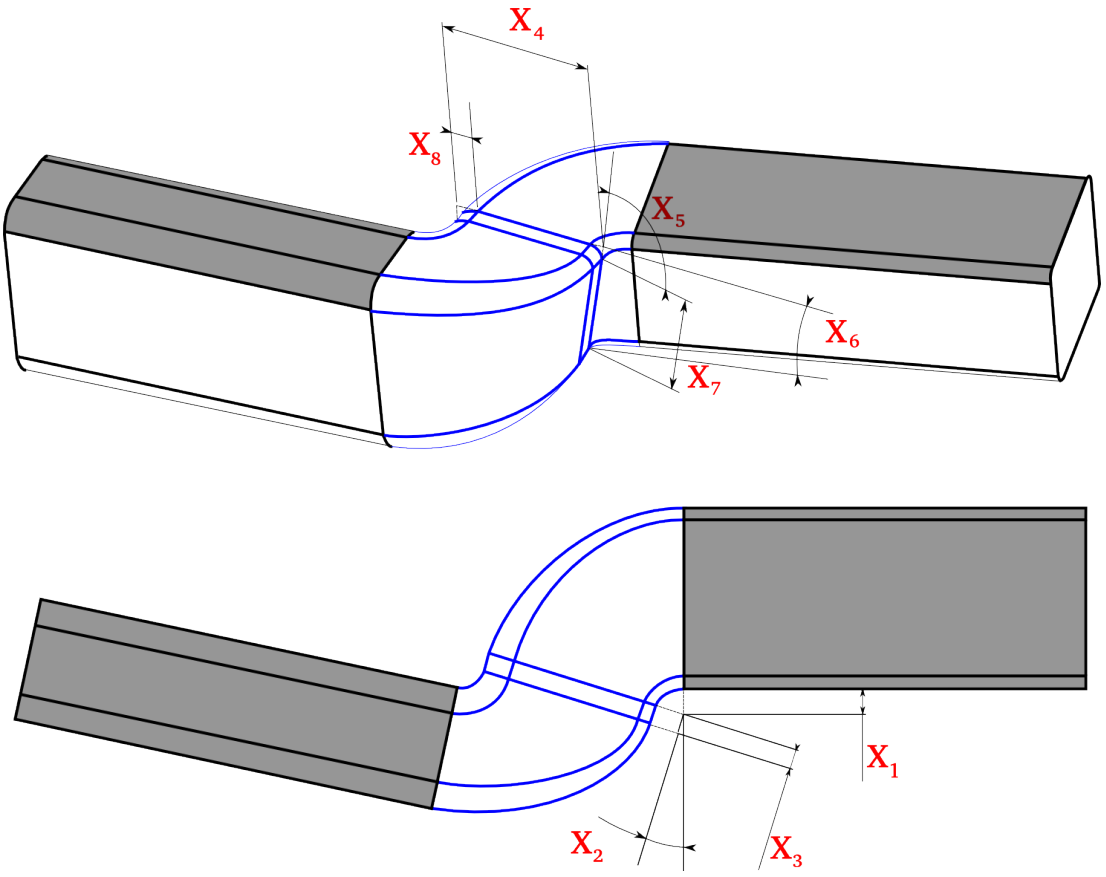


Figure 6.2: Conduite d'air climatisé de véhicule RENAULT. L'entrée et la sortie sont fixées. La géométrie est définie par 8 paramètres notés $X_i, i = 1, \ldots, 8$ et des B-splines.

les longueurs sont d'échelles différentes. Par exemple, le rapport entre X_4 et X_8 est de l'ordre de 10. Ces paramètres sont utilisés pour positionner des points qui définissent la forme de la partie intermédiaire du coude. Des points de contrôle sont alors positionnés afin de définir des BSplines entre les sections droites (entrée et sortie) et la partie intermédiaire. La construction d'une BSpline est illustrée par la figure 6.3. La forme finale est définie par 8 fonctions BSpline qui lient la conduite d'entrée à la conduite intermédiaire et 8 autres lient la conduite intermédiaire à la conduite de sortie. Afin de pallier la différence de sensibilité entre les paramètres, chacun évolue dans un intervalle normalisé $[-1, 1]$.

6.1.2 Définition des critères

Nous cherchons ici l'ensemble de paramètres qui définit le coude de la conduite en minimisant :

– la perte de charge entre le plan d'entrée et le plan de sortie ;

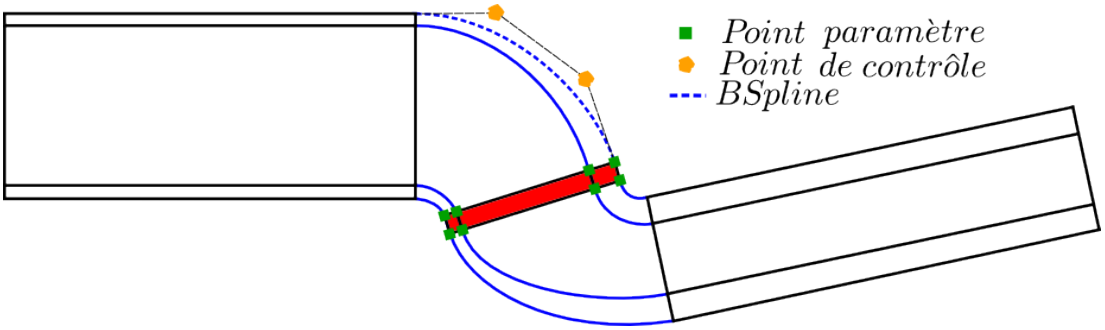


FIGURE 6.3: Construction d'une BSpline entre la section droite d'entrée et la forme intermédiaire.

– la variance de vitesse en sortie.

La réduction de perte de charge permet de diminuer la puissance du système de soufflerie. La réduction de la variance du champ de vitesse en sortie homogénéise la diffusion d'air dans l'habitacle.

6.1.2.1 Calcul de la perte de charge

La perte de charge est calculée en résolvant l'équation donnée par le théorème de Bernoulli généralisé. Ce dernier se résume en une relation qui lie la vitesse du fluide à l'entrée d'une conduite u_i, la vitesse du fluide à la sortie de la conduite u_o, les densités du fluide en entrées ρ_i et en sortie ρ_o et l'accélération de la pesanteur. Au regard des dimensions de la conduite, nous ne prenons pas en compte la différence d'altitude. La perte de charge est alors définie par l'équation 6.1.

$$\Delta p = p_i - p_o + \frac{\rho_i u_i^2}{2} - \frac{\rho_o u_o^2}{2} \tag{6.1}$$

En pratique, nous considérons une conduite dont les deux sections droites sont fixes. Les sections d'entrées et de sorties sont définies comme illustré par la figure 6.4. Les vitesses d'entrée et de sortie sont définies comme des moyennes du champ de vitesse calculées sur l'ensemble des points de maillage situés à proximité du plan de section considéré. On note respectivement $\mathcal{P}_i$ et $\mathcal{P}_o$ les plans d'entrée et de sortie. Ainsi, la vitesse moyenne en sortie de conduite est définie par l'équation 6.2

$$u_k = \|U_m^k\| = \left\| \frac{1}{V_{tot}} \sum_{d(cel,\mathcal{P}_o)\leq\varepsilon} u(cel) \times Vol(cel) \right\|, \tag{6.2}$$

dont les termes correspondent à :

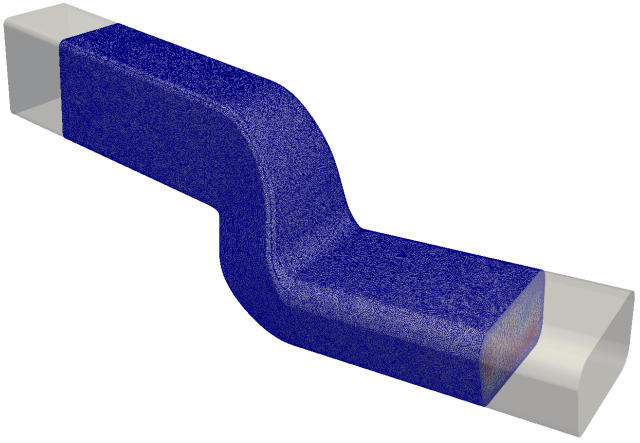


FIGURE 6.4: Sections planes d'entrée et de sortie de la conduite de climatisation.

- $u(cel)$: vitesse moyenne de la cellule courante ;
- $Vol(cel)$: volume de la cellule courante ;
- V_{tot} : volume total des cellules considérées.

Les cellules considées sont celles dont le centre de gravité se trouve à une distance inférieure à ε du plan $\mathcal{P}_o$. De même, nous calculons la pression moyenne en sortie de conduite par l'équation 6.3.

$$p_k = \left|P_m^k\right| = \left|\frac{1}{V_{tot}} \sum_{d(cel,\mathcal{P}_o)\leq\varepsilon} p(cel) \times Vol(cel)\right|. \tag{6.3}$$

Le critère de perte de charge est alors évalué selon l'équation 6.1.

6.1.2.2 Calcul de la variabilité de vitesse

Le second critère considéré est la variabilité de la vitesse en sortie de conduite. Comme pour le calcul de la perte de charge, nous considérons les vitesses moyennes des cellules situées dans un voisinage du plan de section illustré par la figure 6.5. La vitesse moyenne est évaluée selon l'équation 6.2. Les variances directionnelles sont évaluées en fonction des vitesses moyennes directionnelles selon l'équation 6.4

$$\sigma^2_{velx_i} = \frac{1}{V_{tot}} \sum_{d(cel,\mathcal{P}_o)\leq\varepsilon} (U_{mx_i} - u_{x_i}(cel))^2 * Vol(cel). \tag{6.4}$$

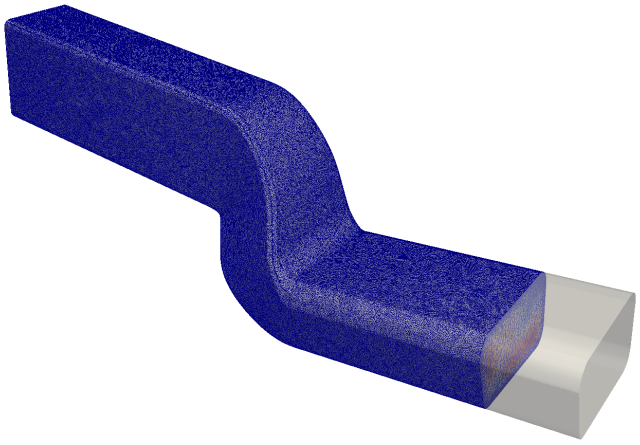


Figure 6.5: Section plane de sortie de la conduite de climatisation.

La variabilité de la vitesse est alors égale à la somme des trois variabilités directionnelles (équation 6.5).

$$\sigma^2 = \sigma^2_{velx_1} + \sigma^2_{velx_2} + \sigma^2_{velx_3}. \tag{6.5}$$

6.2 Etude de l'écoulement

Après avoir défini les critères de notre problème d'optimisation, nous étudions l'écoulement à l'intérieur de la conduite. Les simulations sont réalisées à partir du solveur Navier-Stokes 3D NUM3SIS développé par Inria. Il est basé sur un maillage non-structuré tétraédrique et une méthode *vertex centered* en formulation mixte volumes finis, éléments finis. La précision du second ordre est obtenue en utilisant la technique MUSCL (*Monotone Upstream- centered Schemes for Conservation Laws*) pour extrapoler les termes convectifs. L'intégration temporelle est effectuée en utilisant une approche implicite basée sur une approximation de la matrice Jacobienne et un pas de temps local. Pour de plus amples informations, nous invitons le lecteur à consulter [DD92], [Klo08].

Les caractéristiques du fluide considéré sont :

- Densité : $\rho = 1.2$;
- Rapport entre les capacités isobare et isochore : $\gamma = 1.4$;
- Vitesse initiale : $u = 1\ m.s^{-1}$.

Ainsi, l'écoulement à l'intérieur de la conduite est caractérisé par :

- Nombre de Mach : $M_\infty = 0.1$;
- Nombre de Reynolds : $Re = 2800$.

L'écoulement interne d'air est donc en régime transitionnel ($Re \approx 2800$) et subsonique ($M_\infty = 0.1$). On résout cependant le problème en régime laminaire. Afin de produire un maillage adapté au modèle Navier-Stokes, ce dernier est plus fin à proximité des parois et plus grossier vers l'intérieur de la conduite. La géométrie et le maillage sont réalisés avec le logiciel GMSH [GR]. Pour chaque évaluation, le script qui définit la géométrie est modifié, en fonction des paramètres décrits précédemment. Le maillage est alors généré par GMSH, puis décomposé afin de répartir la simulation sur plusieurs processeurs. La figure 6.6 illustre l'écoulement dans la conduite définie sous sa forme initiale. Nous y

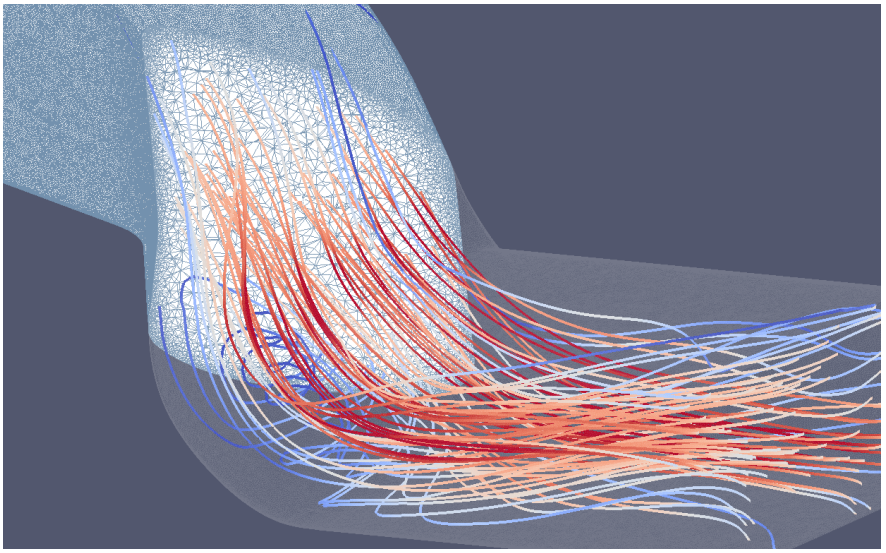


Figure 6.6: Ecoulement dans la conduite définie sous sa forme nominale. Représentation des lignes de courant selon l'amplitude de la vitesse.

illustrons également des lignes de courant, colorées en fonction de l'amplitude de la vitesse. Nous pouvons également voir le raffinement de maillage proche du bord de la conduite. Pour chaque évaluation, le maillage généré est composé d'environ 600 000 points.

6.3 Résultats

6.3.1 Résultats numériques

Nous considérons initialement une base de données de 30 points répartis selon un hypercube latin (LHS [MMC00]) dans l'espace des paramètres, inclus dans $\mathbb{R}^8$. Comme indiqué, l'espace de recherche K est normalisé pour se réduire à $[-1, 1]^8$. Ainsi, la sensibilité intrinsèque à la nature des variables est réduite. L'algorithme MGDA assisté

par métamodèle est utilisé sur 9 cycles pour obtenir les résultats suivants. A la fin de chaque cycle, les points issus de l'application de MGDA sur le métamodèle sont évalués par le solveur. Après 9 cycles, nous obtenons une base de données de 232 points et un *front de Pareto* composé de 4 points. L'évolution de la base de données au fil des itérations est illustré par la figure 6.7. Seuls les points ajoutés aux itérations 3, 6 et 9 ainsi que les points des bases de données initiale et finale sont représentés afin d'obtenir une meilleure lisibilité. Les ensembles de points non dominés des bases de données initiales et finales (neuvième cycle) sont mise en évidence, afin de pouvoir les comparer. Nous pouvons noter une évolution progressive des points ajoutés d'un cycle à l'autre vers le front de Pareto. Les résultats représentés sur cette figure sont normalisés, afin d'obtenir une performance égale à $(1, 1)$ pour le point nominal. Ce dernier est représenté par un carré plein sur la figure. Le nombre de points évalués, le nombre de

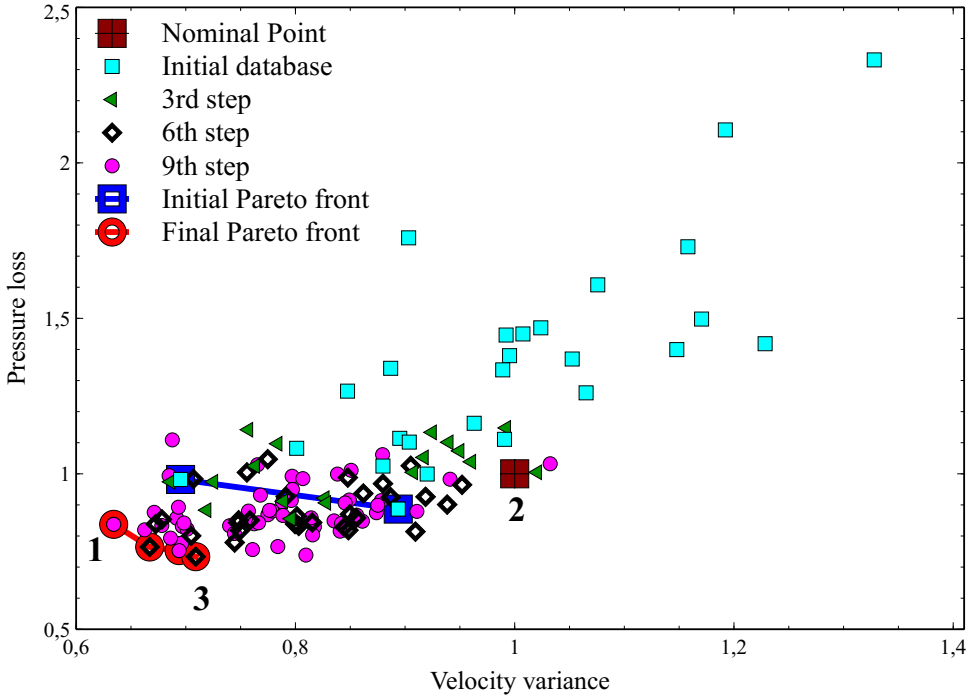


Figure 6.7: Evolution de la base de données enrichie par MGDA assisté par métamodèle. Comparaison entre les ensembles non dominés initial et final (après 9 cycles).

points non-dominés ainsi que les meilleures performances obtenues vis-à-vis des critères, sont résumés pour chaque itération dans le tableau 6.8. Après 9 itérations de la méthode, les points donnés à convergence de MGDA, sur le métamodèle, ont des performances suffisamment proches de celles obtenues par évaluation par le solveur des mêmes points. Nous pouvons alors supposer que les résultats obtenus sont proches du front de Pareto

Itération	0	1	2	3	4	5	6	7	8	9
Taille DB	27	38	44	61	73	84	114	142	179	232
Taille front	2	3	3	3	3	1	2	2	2	4
min σ_v^2 ($\times 10^{-1}$)	5.2	5.2	5.2	5.26	5.2	4.9	4.9	4.9	4.9	4.8
min Δp	1.08	1.03	0.99	0.99	0.99	0.91	0.89	0.89	0.89	0.89

Figure 6.8: Récapitulatif des valeurs minimales des critères et de la taille des bases de données (DB), au fil des itérations.

du problème d'optimisation. Nous pouvons constater, sur la figure 6.7, que les points non dominés sont concentrés. Ce manque de diversité du front de la dernière base de données peut être expliqué par plusieurs facteurs :

- Ce problème peut être de nature multimodale et l'algorithme MGDA accumule les points de convergence dans un bassin ;
- Les deux critères proposés ne sont pas fortement antagonistes.

La première hypothèse est peu probable. D'une part, la base de données initiale est réalisée de telle manière que l'ensemble du domaine y est représenté équitablement. D'autre part, nous avons effectué plusieurs essais, à partir de différentes bases de données composées de nombres supérieurs et inférieurs de points. Par ailleurs, la perte de charge est amplifiée par le phénomène de recirculation dans la partie basse du coude que nous observons sur la figure 6.6. De même, cette recirculation favorise l'écoulement du fluide sur un côté de la section droite de sortie. Ainsi, une forme du coude qui diminue la recirculation favorise une perte de charge moindre et une distribution homogène du champ de vitesse d'écoulement en sortie de conduite.

6.3.2 Comparaison entre les formes optimales et initiales

Après l'étude des résultats numériques du traitement de ce problème d'optimisation, nous réalisons une étude comparative des formes obtenues. Nous comparons ici les coudes des conduites situées aux extrémités du front et le coude initial. Sur la figure 6.9, nous comparons la forme qui obtient la meilleure performance en termes de réduction de variance de vitesse à la forme nominale. La forme nominale y est représentée en transparence afin de faciliter la visualisation. Ainsi, nous constatons que la forme du coude tend vers la diminution de son épaisseur et une augmentation de la largeur. La forme concave de la partie supérieure du coude est alors remplacée par une forme convexe. Comme nous l'avons supposé, les formes du coude qui obtiennent les meilleures performances en variance de vitesse et en perte de charge sont relativement proches. Nous comparons ces deux formes sur la figure 6.10. La conduite qui correspond au mini-

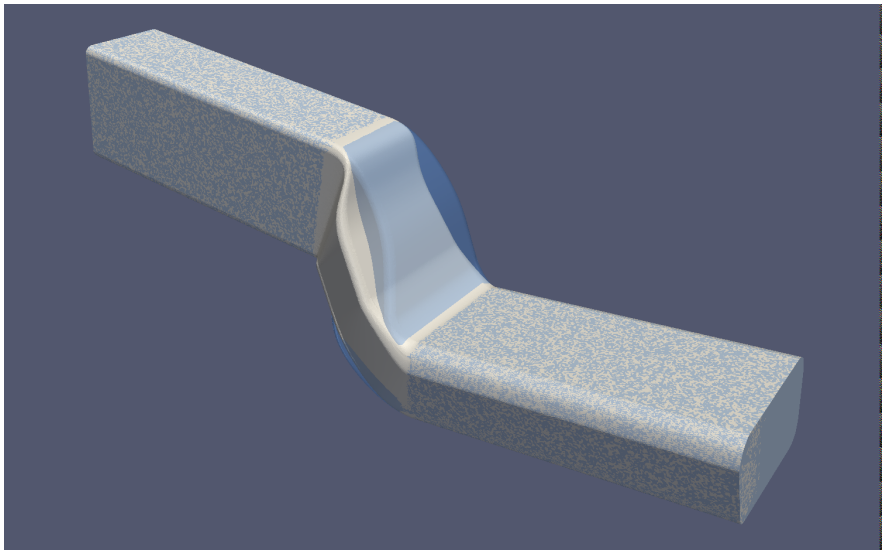


Figure 6.9: Comparaison entre la forme nominale et la forme qui correspond à la meilleure variance de vitesse obtenue. La forme nominale est représentée en transparence.

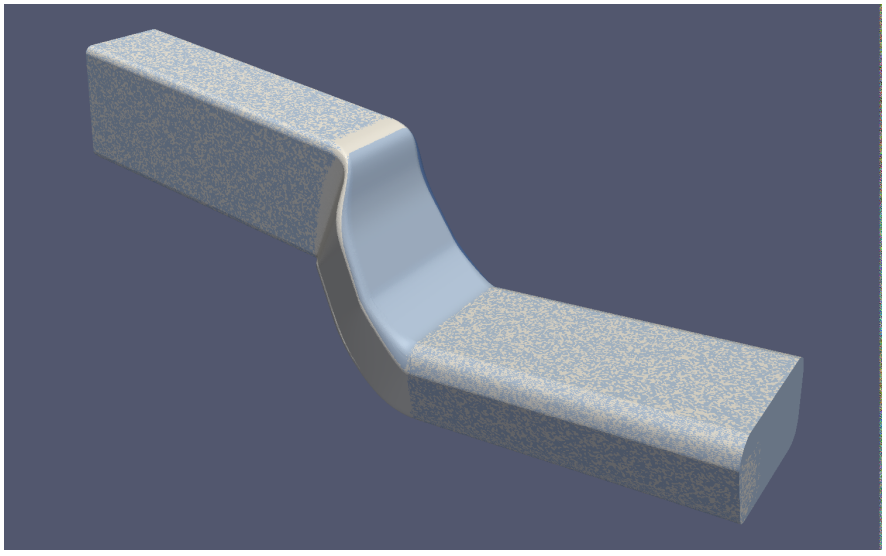


Figure 6.10: Comparaison entre les deux formes optimales obtenues. La forme de conduite qui correspond à la perte de charge minimale est représentée en transparence.

mum de perte de charge est représentée en transparence afin de faciliter la comparaison. Ainsi, nous pouvons constater que le coude le plus performant en termes de perte de charge est légèrement moins large. En revanche, la convexité de la partie supérieure est moins accentuée. Ces différences restent très sensibles.

Afin de comparer les profils des formes des solutions situées aux extrémités du front de Pareto, nous illustrons sur la figure 6.11 des coupes longitudinales des conduites. Comme nous venons de le faire pour les conduites complètes, nous superposons deux à deux les résultats obtenus. Afin de faciliter la comparaison, les deux conduites optimales situées dans la colonne de gauche sont superposées à une coupe similaire du profil nominal. Ce dernier est représenté en transparence. Ainsi, nous pouvons constater la différence notable entre les formes optimales et la forme nominale. Sur la colonne de droite, nous illustrons une superposition des deux conduites optimales obtenues. La forme du coude qui offre la meilleure performance en réduction de perte de charge est représentée en transparence. Nous constatons alors la différence minime qui distingue les coupes des conduites situées aux extrémités du front.

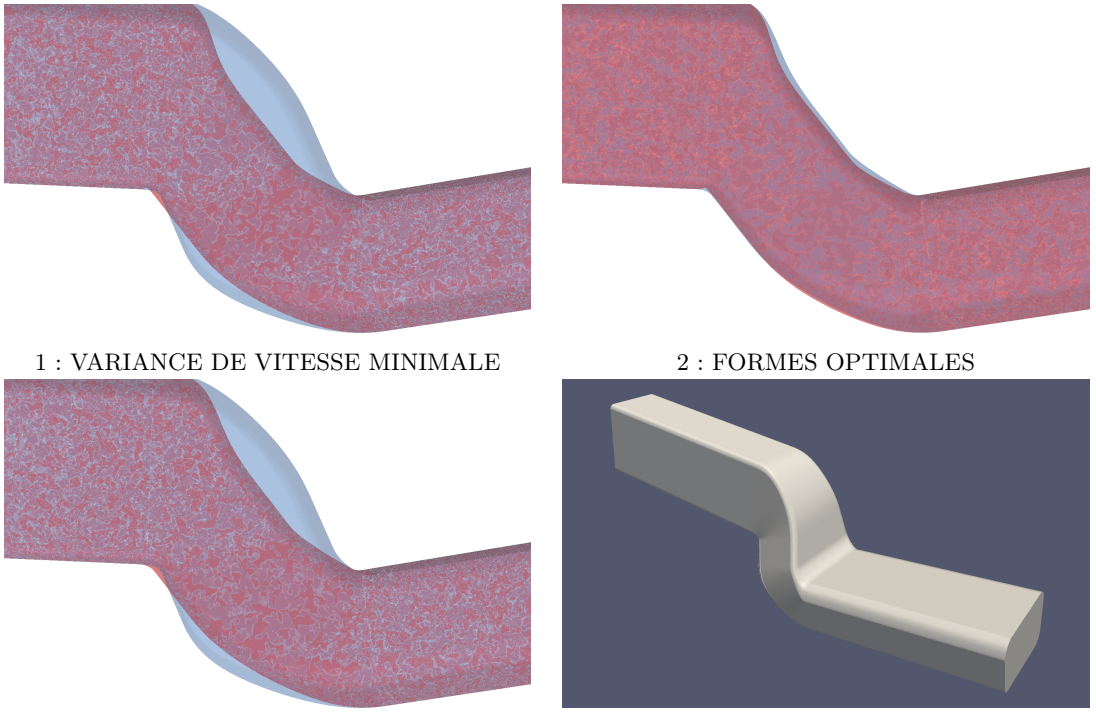
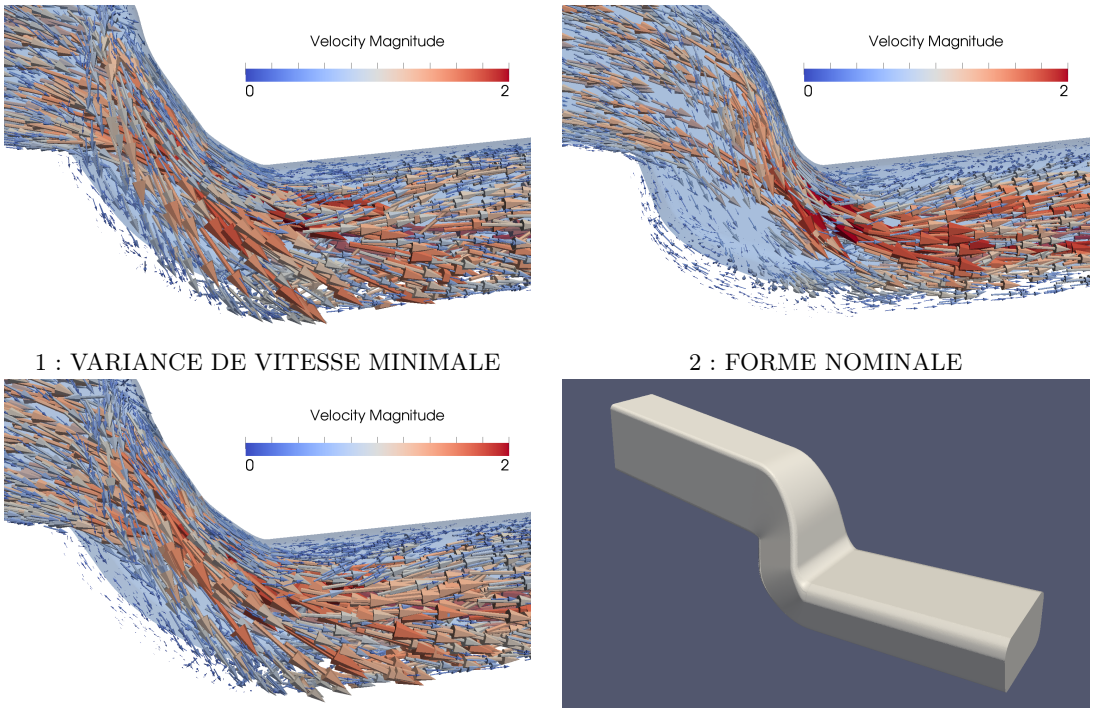


Figure 6.11: Coupes longitudinales des solutions situées aux extrémités du front et de la solution nominale.

6.3.3 Comparaison entre les écoulements

Aprés avoir comparé les formes optimales obtenues, nous comparons ici les écoulements à l'intérieur des conduites correspondantes. La figure 6.12 compare la répartition du champ des vecteurs de vitesse à l'intérieur du coude, pour les deux sections extrêmes du front et pour la solution nominale. Les vecteurs vitesses de la forme nominale mettent en évidence un phénomène de recirculation du fluide dans la partie basse du coude, pour la forme nominale. En revanche, cela ne se produit pas dans les conduites optimales. En effet, la forme moins bombée du coude canalise l'écoulement de la section droite



1 : VARIANCE DE VITESSE MINIMALE

2 : FORME NOMINALE

3 : PERTE DE CHARGE MINIMALE

FORME NOMINALE

Figure 6.12: Comparaison entre les champs de vitesse. La numérotation correspond à celle utilisée pour la figure 6.7.

d'entrée vers celle de sortie. La recirculation peut être provoquée par la forme concave de la partie supérieure du coude et la partie convexe de la partie inférieure. La figure 6.13 permet de comparer la répartition du champ de vitesse (en norme) de l'écoulement sur deux sections de coupe sigificatives. Les résultats observés sur la colonne de gauche correspondent à une section de coupe située à la sortie du coude. Les deux écoulements à l'intérieur des formes optimales se distinguent de la forme nominale par l'apparition en sortie du coude d'un tourbillon longitudinal. La présence du vortex permet d'uniformiser le champ de vitesse. Les deux formes optimales sont proches, car les deux critères

proposés dans ce cas test d'optimisation ne sont pas à proprement parler antagonistes. En effet, deux formes proches du coude sont amenées à produire un écoulement qui va minimiser la variance de vitesse et la perte de charge.

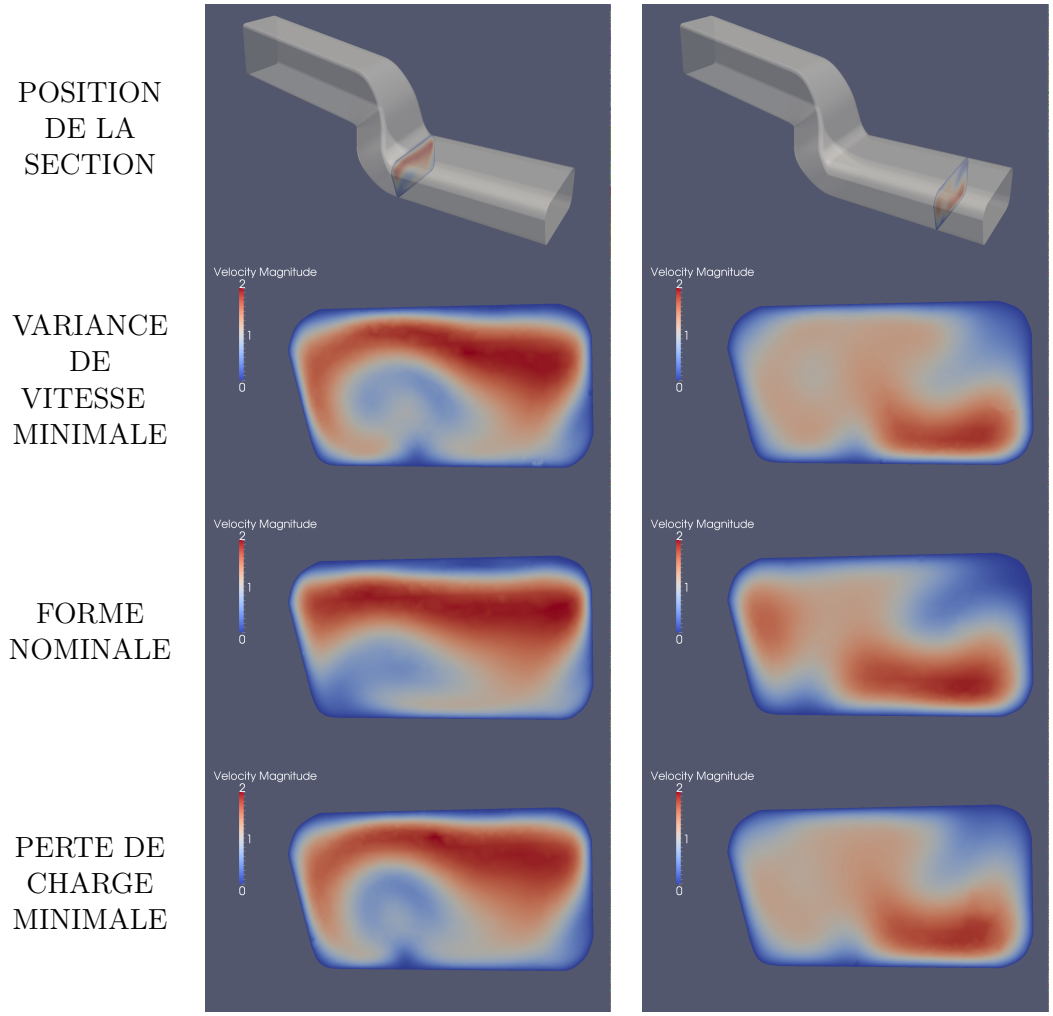


Figure 6.13: Répartition du champ de vitesse (en norme) des écoulements sur deux sections significatives.

6.4 Bilan

Dans ce chapitre, nous appliquons la méthode MGDA, assistée par la construction de métamodèles de type *Krigeage* à un problème pré-industriel. Ce cas test comporte plusieurs caractéristiques intéressantes. Premièrement, les paramètres qui définissent la géométrie de la conduite sont de natures différentes. En effet, les 8 paramètres se composent en trois angles et cinq longueurs. Cette diversité induit une forte différence de

sensibilité entre eux. Deuxièmement, les paramètres de longueur ne varient pas selon la même échelle. Nous pouvons noter un rapport de un sur dix pour l'écart le plus important. Nous pallions partiellement cet handicap par une normalisation globale des paramètres. Ainsi, notre espace de recherche devient $[-1, 1]^8$. Finalement, l'écoulement étudié prend en compte la viscosité du fluide en résolvant les équations de Navier-Stokes à chaque évaluation. Ces équations génèrent des écoulements beaucoup plus complexes que les équations d'Euler. En effet, elles prennent en considération des phénomènes locaux qui produisent des recirculations et des décollements de l'écoulement. Cette particularité augmente considérablement le temps nécessaire à l'évaluation des critères. Les surfaces de réponse produites par ce modèle sont moins régulières que celles produites par un modèle de type Euler et comportent plus de minima locaux. De plus, la prise en compte de la viscosité nécessite la construction automatique d'un maillage plus fin sur les bords de la conduite.

La progression par étapes illustrée par la figure 6.7 montre l'efficacité de la méthode. De plus, le coût en terme de calcul nécessaire pour obtenir ce résultat reste modéré (232 appels solveur). Ce résultat pourrait être amélioré par l'utilisation d'une méthode de construction de métamodèle plus performante. De plus, l'utilisation de fonctions de mérite [Jon01] pourrait apporter une plus grande diversité aux échantillons.

Chapitre 7

Conclusion

Au cours du premier chapitre de ce mémoire, nous avons introduit, dans une première section, les définitions et notations classiques de l'optimisation. Dans une seconde section, nous avons énoncé les principaux algorithmes basés sur l'utilisation des gradients de descente. Dans le cadre d'applications à des fonctions convexes, nous avons donné les preuves de convergence. Après une étude des problèmes non contraints, nous avons introduit les multiplicateurs de Lagrange ainsi que le théorème de Karush-Kuhn-Tucker (KKT) qui traduit la condition d'optimalité d'un problème différentiable sous contraintes.

Dans un second chapitre, nous avons abordé les problèmes d'optimisation multiobjectif. Nous avons débuté ce chapitre par la définition de la dominance au sens de Pareto. Celle-ci nous permet de définir l'optimalité par rapport à plusieurs critères. Nous avons consacré la suite de ce chapitre aux méthodes dites évolutionnaires. Ces algorithmes inspirés de la théorie de l'évolution des espèces, énoncée par Charles Darwin [Dar69] en 1869, sont les plus couramment utilisés dans la pratique. En effet, leur simplicité de mise en oeuvre et leur robustesse en font l'outil idéal pour un usage à large spectre. Cependant, les paramètres d'initialisation de ces méthodes peuvent être délicats à déterminer pour un problème précis. Par ailleurs, l'obtention d'un échantillon représentatif des solutions du problème demande un grand nombre d'évaluations. Nous avons fait, dans ce chapitre, un bref résumé des différentes techniques qui caractérisent certaines des méthodes évolutionnaires les plus connues.

Dans un troisième chapitre, nous avons introdui un algorithme multiobjectif basé sur l'utilisation des gradients des critères à minimiser. Il s'agit d'une généralisation de la méthode de plus grande descente, présentée dans le premier chapitre pour traiter les

cas de minimisation d'un seul critère. Nous avons résumé, dans une première section, les bases théoriques sur lesquelles repose la méthode, détaillées par Jean-Antoine Désidéri dans [Dés12]. Ensuite, nous avons développé la mise en oeuvre de l'algorithme. Afin de le valider, nous l'avons testé sur des problèmes de minimisation de fonctions analytiques d'une complexité progressive. Ces études sont comparées aux résultats obtenus sur les mêmes problèmes par une méthode évolutionnaire, PAES [KC00]. L'utilisation de MGDA nécessite la connaissance des gradients des critères à minimiser, au point courant. Lorsque cela est possible, un tel calcul constitue généralement un frein à son utilisation.

Aussi, dans un quatrième chapitre, nous avons proposé d'associer l'algorithme de descente MGDA à la construction d'un métamodèle de type *Krigeage*. Le calcul des gradients et des critères au point courant s'effectue sur la surface de réponse et non par simulation numérique. Les étapes de la méthode sont détaillées dans la première section. Nous avons appliqué ensuite la méthode à un cas test analytique proposé par Kursawe [Kur91], fréquemment utilisé par les concepteurs d'algorithmes multiobjectif. Nous constatons alors que le couplage de MGDA avec la construction d'un métamodèle ne suffit pas à pallier le manque d'efficacité des méthodes de gradients au traitement des problèmes d'optimisation multimodaux. Nous avons proposé ensuite la résolution d'un problème simplifié d'optimisation de forme d'un profil d'aile d'avion d'affaires. La méthode a donné de bons résultats en termes de rapidité, pour un nombre d'évaluations restreint, sur un problème multiobjectif et multipoint.

Nous avons conclu ce travail par le traitement d'un cas test pré-industriel proposé par Renault, dans le cadre du projet ANR, OMD2. Ce dernier consiste à étudier la forme du coude d'une conduite de climatisation. Les critères à minimiser sont la variabilité de la vitesse et la perte de charge. Les écoulements sont simulés par un solveur Navier-Stokes compressible. La complexité de ce problème vient de la durée nécessaire pour évaluer les critères en un point et de la nature hétérogène des critères. En effet, l'espace de recherche est décrit par des angles et des longueurs. La sensibilité de variation n'est alors pas la même selon le paramètre perturbé. Les résultats obtenus par la méthode MGDA assistée par métamodèle sur ce problème montrent son efficacité.

Nous avons consacré cette étude à la mise en place d'un algorithme multiobjectif basé sur les gradients. Nous y avons validé l'efficacité sur des cas tests de difficulté croissante. Les résultats obtenus sont satisfaisants, mais quelques pistes devraient être exploitées.

Pour les problèmes mono-objectif, la direction de descente donnée par le gradient n'est pas celle qui mène à la solution le plus rapidement. En effet, les méthodes de Newton montrent qu'il est préférable d'utiliser la direction inverse de l'image du gradient par la réciproque du hessien de la fonction objectif. A l'instar de ces algorithmes, il est légitime de s'intéresser aux images des gradients des objectifs par l'inverse de leurs matrices hessiennes respectives. Cette étude s'est limitée à l'utilisation des gradients uniquement. Nous avons montré que le vecteur choisi donne une direction de descente, même si ce n'est pas la meilleure. Est-ce toujours le cas si l'on considère l'enveloppe convexe des projections hessiennes ?

Par ailleurs, même si la méthode est prévue pour un nombre indéterminé de critères, seuls des problèmes à deux objectifs y sont traités. Nous avons montré que, pour deux critères, le vecteur de norme minimale ω peut être calculé directement. Lorsque le nombre de critères est supérieur, nous proposons actuellement de résoudre un problème de minimisation sur l'enveloppe convexe des gradients par une méthode de type évolutionnaire. L'existence d'une méthode de calcul direct, similaire au cas $n = 2$ est en cours d'étude. Des recherches sont actuellement réalisées sur l'utilisation de procédés d'orthonormalisation de type Gram-Schmidt sur le sous-espace vectoriel des gradients de critères. Par ailleurs, une méthode inspirée des algorithmes de descente de Newton où l'on considère l'image inverse de la hessienne du gradient des critères pour établir la direction de progression, est à l'étude.

Références

[All05] G. Allaire. *Analyse numérique et optimisation : Une introduction à la modélisation mathématique et à la simulation numérique.* Mathématiques appliquées. École Polytechnique, 2005. 2, 4, 5, 22, 25, 26, 54

[BÖ96] T. Bäck. *Evolutionary algorithms in theory and practice : evolution strategies, evolutionary programming, genetic algorithms.* Oxford University Press, Oxford, UK, 1996. 38

[BS02] H.G. Beyer and H.P. Schwefel. Evolution strategies - a comprehensive introduction. *Natural Computing*, 1(1) :3–52, 2002. 42

[CD07] P. Chandrashekarappa and R. Duvigneau. Radial Basis Functions and Kriging Metamodels for Aerodynamic Optimization. Rapport de recherche RR-6151, INRIA, 2007. 75

[Cea71] J. Cea. *Optimisation : Théorie et algorithmes.* Méthodes mathématiques de l'informatique. Dunod, Paris, 1971. BIBLIOGR. PP. 219-227. 56

[CS02] Y. Collette and P. Siarry. *Optimisation multiobjectif.* Algorithmes (Paris). Eyrolles, 2002. 32, 33

[Cul94] J.C. Culioli. *Introduction à l'optimisation.* Ellipses, 1994. 24

[Dar69] C. Darwin. *On the Origin of the Species by Means of Natural Selection : Or, The Preservation of Favoured Races in the Struggle for Life.* John Murray, 1869. 35, 109

[DCO00] J.D. Knowles D.W. Corne and M.J. Oates. *The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization.* Springer, 2000. 42, 46, 48, 50

[DD92] A. Dervieux and J.A. Désidéri. Compressible flow solvers using unstructured grids. INRIA Research Report 1732, June 1992. 100

[DD98] I. Das and J.E. Dennis. Normal-boundary intersection : A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM J. on Optimization*, 8(3) :631–657, mar 1998. 51

[Deb01] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley-Interscience Series in Systems and Optimization. John Wiley & Sons, Chichester, 2001. 3, 46, 76, 77, 78

[Dés07] J.A. Désidéri. Split of Territories in Concurrent Optimization. Rapport de recherche RR-6108, INRIA, 2007. 3

[Dés09] J.A. Désidéri. Multiple-Gradient Descent Algorithm (MGDA). Rapport de recherche RR-6953, INRIA, June 2009. 4, 53

[Dés12] J.A. Désidéri. Multiple gradient descent algorithme (mgda) for multiobjective optimization. *compte rendu de l'académie des siences de Paris*, pages 313–318, 2012. v, 4, 53, 54, 55, 56, 57, 110

[DG89] K. Deb and D.E. Goldberg. An investigation of niche and species formation in genetic function optimization. In *Proceedings of the third international conference on Genetic algorithms*, pages 42–50, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc. 38, 44

[DS03] J. Dréo and P. Siarry. *Métaheuristiques pour l'optimisation difficile : recuit simulé, recherche tabou, algorithmes évolutionnaires, colonies de fourmis...* Algorithmes (Paris). Eyrolles, 2003. 31

[DT97] G.B. Dantzig and M.N. Thapa. *Linear programming 1 : introduction*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997. 2

[DTLZ02] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Multi-Objective Optimization Test Problems. In *Congress on Evolutionary Computation (CEC 2002)*, pages 825–830. IEEE Press, 2002. 3

[DTLZ05] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multi-Objective Optimization. In A. Abraham, R. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization : Theoretical Advances and Applications*, chapter 6, pages 105–145. Springer, 2005. 3

[EZ99] L. Thiele E. Zitzler. Multiobjective evolutionary algorithms : A comparative case study and the strength pareto evolutionary algorithm. *IEEE Transactions on Evolutionary Computation*, 3(4) :257–271, 1999. 3, 42, 46, 49, 50

[FF93] C.M. Fonseca and P.J. Fleming. *Genetic Algorithms for Multiobjective Optimization : Formulation, Discussion and Generalization.* UK : IEE, 1993. 42, 43, 44

[FGM97] M. Laguna F. Glover and R. Martí. Tabu search, 1997. 35

[Gin09] D. Ginsbourger. *Multiples métamodèles pour l'approximation et l'optimisation de fonctions numériques multivariables.* PhD thesis, École Nationale Supérieure des Mines de Saint-Étienne, March 2009. 93

[GM97] M. Gibbs and D.J.C. MacKay. Efficient implementation of gaussian processes. 1997. 75

[Gol89] D.E. Goldberg. *Genetic Algorithm in Search, Optimization and MAchine Learning.* Addison Wesley Company Inc., 1989. 34

[GR] C. Geuzaine and J.F. Remacle. *A three-dimensional finite element mesh generator with built in pre- and post-processing facilities.* 101

[GR87] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 41–49, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc. 43

[HK04] N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In X. Yao et al., editors, *Parallel Problem Solving from Nature PPSN VIII*, volume 3242 of *LNCS*, pages 282–291. Springer, 2004. 41

[HL92] P. Hajela and C.Y. Lin. *Genetic search strategies in multicriterion optimal design*, volume 4. Springer Berlin / Heidelberg, 1992. 10.1007/BF01759923. 42

[HNG94] J. Horn, N. Nafpliotis, and D.E. Goldberg. *A Niched Pareto Genetic Algorithm for Multiobjective Optimization.* USA : IEEE, 1994. 42, 47

[Hol75] J. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, 1975. 34

[Hol92] J. Holland. *Adaptation in Natural and Artificial Systems : An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence.* MIT Press, Cambridge, MA, USA, 1992. 35

[IHR07] C. Igel, N. Hansen, and S. Roth. Covariance matrix adaptation for multi-objective optimization. *Evolutionary Computation*, 15(1) :1–28, 2007.

[IM96] H. Ishibuchi and T. Murata. Multi-objective genetic local search algorithm. In *International Conference on Evolutionary Computation*, pages 119–124, 1996. 50

[IRD80] Davenport J.M. Iman R.L. and Zeigler D.K. *Latin Hypercube Sampling (program User's Guide).* SAND ; 79-1473. United States Department of Energy, Sandia Laboratories, 1980. 76

[Jen03] M.T. Jensen. Reducing the run-time complexity of multiobjective eas : The nsga-ii and other algorithms. *Trans. Evol. Comp*, 7(5) :503–515, October 2003. 46

[Jon01] D.R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21 :345–383, 2001. 93, 108

[KC00] J. Knowles and D. Corne. *Approximating the nondominated front using the Pareto Archived Evolution Strategy. Evolutionary Computation.* MIT press, 2000. 42, 50, 67, 70, 77, 110

[KDM00] S. Agarwal K. Deb, A. Pratap and T. Meyrivan. *A Fast and Elitist Multiobjective Genetic Algorithm-NSGA-II.* KanGAL Report Number 2000001, 2000. 42, 45, 68

[KE95] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948 vol.4. IEEE, November 1995. 51

[Klo08] T. Kloczko. Concept, architecture and performance study for a parallel code in cfd. In *Parallel CFD Conference, May 19-22, Lyon, France*, 2008. 100

[Kri51] D.G. Krige. *A statistical approach to some mine valuation and allied problems in the Witwatersrand*. PhD thesis, Master's thesis of the University of Witwatersrand, 1951. 4

[KS02] A. Konak and A.E. Smith. Multiobjective optimization of survivable networks considering reliability. In *The 10th International Conference on Telecommunication Systems*, Monterey, CA, 2002. 47

[KS04] A. Konak and A.E. Smith. Capacitated network design considering survivability : An evolutionary approach. *Journal of Engineering Optimization*, 36, 2004. 47

[Kur91] F. Kursawe. A variant of evolution strategies for vector optimization. In *Proceedings of the 1st Workshop on Parallel Problem Solving from Nature*, PPSN I, pages 193–197, London, UK, 1991. Springer-Verlag. 4, 110

[LY03] H. Lu and G.G. Yen. Rank-density-based multiobjective genetic algorithm and benchmark test function study. *IEEE Trans. Evolutionary Computation*, pages 325–343, 2003. 42, 50

[MI95] T. Murata and H. Ishibuchi. *MOGA : Multi-objective genetic algorithms*. IEEE Service Center, Piscataway, NJ, 1995. 42, 50

[Mic92] Z. Michalewics. *Genetic algorithm + data structures = evolutionnary programs*. AI Series. Springer-Verlag, New York, 1992. 34

[Mie99] K. M. Miettinen. *Nonlinear Multiobjective Optimization*. Kluver Academic publishers, 1999. 32, 54

[MMC00] R.J. Beckman M.D. Mckay and W.J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1) :202–208, 2000. 35, 79, 85, 101

[Nas50] J.F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences of the United States of America*, 36(1) :48–49, January 1950. 3

[Neu28] J.V. Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100(1) :295–320, 1928. 3

[NW99] J.A. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Series in Operations Research Series. Springer-Verlag GmbH, 1999. 24

[OR94] M.J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994. 3

[PGW81] W. Murray P.E. Gill and M.H. Wright. *Practical optimization*. Academic Press, 1981. 2

[PWG] V. Picheny, T. Wagner, and D. Ginsbourger. A benchmark of kriging-based infill criteria for noisy optimization. 93

[RW06] C.E. Rasmussen and C.K.I. Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, MA, 2006. 75

[Sch85] J.D. Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1985. 42

[SD94] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2 :221–248, 1994. 42

[SKV83] C.D. Gelatt S. Kirkpatrick and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220 :671–680, 1983. 35

[Sny05] J. Snyman. *Practical Mathematical Optimization : An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*. Applied Optimization. Springer, 2005. 2, 4

[VL00] D.A. Van Veldhuizen and G.B. Lamont. *Multiobjective Evolutionary Algorithms : Analyzing the State-of-the-Art*. Evol Comput, 2000. 46

[VV04] F. Vermolen and C. Vuik. Multi-objective design strategies using deterministic optimization with different parametrizations in aerodynamics. In P. Neittaanmäki, T. Rossi, K. Majava, and O. Pironneau, editors, *Proceedings of the 4^{th} European Congress on Computational Methods in Applied Sciences and Engineering, Volume I, ECCOMAS 2004, Jyväskylä, Finland, July 24-28, 2004*, page CD, Jyväskylä, 2004. University of Jyväskylä. ISBN 951-39-1868-8. 3

[Wag03] C. Wagschal. *Topologie et analyse fonctionnelle : Exercices corrigés*. Collection Méthodes. Hermann, 2003. 8, 12, 17

[YL03] G.G. Yen and H. Lu. Dynamic multiobjective evolutionary algorithm : adaptive cell-based rank and density estimation. *Trans. Evol. Comp*, 7(3) :253–274, June 2003. 42, 50

[ZLT02] E. Zitzler, M. Laumanns, and L. Thiele. Spea2 : Improving the strength pareto evolutionary algorithm for multiobjective optimization. In K.C. Giannakoglou et al., editors, *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE), 2002. 42, 50