Table des matières

Introduction générale **5**

I Introduction **9**

1 Optimisation multi-objectif **11**
1.1 Position du problème . . . 12
1.2 Dominance et optimalité de Pareto . . . 13
1.2.1 Concept de dominance et solutions Pareto-optimales . . . 13
1.2.2 Procédures de recherche de l'ensemble non-dominé . . . 14
1.2.3 Vecteur idéal et vecteur de Nadir . . . 15
1.2.4 Conditions d'optimalité . . . 16
1.3 Equilibre de Nash . . . 17
1.4 Méthodes déterministes . . . 18
1.4.1 Agrégation pondérée . . . 18
1.4.2 Méthode de ϵ-contraintes . . . 20
1.4.3 Autres approches . . . 21
1.4.4 Moralité . . . 22
1.5 Si le but justifie les moyens . . . 22

2 Evolution Artificielle **25**
2.1 Algorithmes Evolutionnaires . . . 25
2.1.1 Vocabulaire et principe de fonctionnement . . . 26
2.1.2 Historique . . . 28
2.2 Représentations et opérateurs correspondants . . . 30
2.2.1 Représentation binaire . . . 30
2.2.2 Représentation réelle . . . 31
2.3 Darwinisme artificiel . . . 33
2.3.1 Sélection . . . 33
2.3.2 Remplacement . . . 35
2.3.3 Schémas d'évolution . . . 35
2.4 Préservation de la diversité génétique . . . 36
2.4.1 Partage (*sharing*) . . . 37
2.5 Prise en compte des contraintes . . . 37
2.5.1 Méthodes basées sur le concept de pénalité . . . 37

2.5.2 Et les autres . . . 39
2.6 Algorithmes Evolutionnaires vs méthodes d'optimisation "classiques" . . . 40

II Algorithmes Evolutionnaires Multi-Objectif 41

3 Etat de l'art 43
3.1 Introduction . . . 43
3.2 Premiers travaux . . . 44
3.2.1 VEGA . . . 44
3.2.2 VOES . . . 46
3.2.3 HLGA . . . 47
3.2.4 Et les autres . . . 49
3.3 Algorithmes basés sur la dominance de Pareto . . . 50
3.3.1 MOGA . . . 50
3.3.2 NSGA . . . 53
3.3.3 NPGA . . . 55
3.4 Elitisme au sens de la dominance . . . 56
3.4.1 SPEA . . . 57
3.4.2 SPEA2 . . . 60
3.4.3 NSGA-II . . . 61
3.4.4 PAES & PESA . . . 63
3.4.5 Faites à votre guise . . . 65
3.5 Optimisation Multi-Objectif Evolutionnaire sous contraintes . . . 66
3.5.1 Pénalisation et AEMO . . . 66
3.5.2 Tournois "indirects" . . . 67
3.5.3 Mesures d'infaisabilité . . . 68

4 Problèmes clés 71
4.1 Représenter l'ensemble des solutions non-dominées . . . 71
4.2 Évaluer la performance d'un AEMO . . . 76
4.2.1 Mesures de qualité pour la surface de Pareto approchée . . . 76
4.2.2 Performance statistique par Fonseca et Fleming . . . 81
4.2.3 Performance en dynamique (*Running metrics*) . . . 82
4.2.4 Performance de l'algorithme vs qualité des résultats . . . 83
4.2.5 Construction des problèmes tests . . . 84

5 Critère d'arrêt "par désespoir" 87
5.1 Interruption de l'évolution "inefficace" . . . 88
5.1.1 Situation révolutionnaire . . . 88
5.1.2 Moment de désespoir . . . 92
5.2 Le critère d'arrêt "par désespoir" . . . 94
5.3 Leçons et perspectives . . . 97

6 Croisement basé sur la dominance de Pareto 99
6.1 Croisement biaisé vers le parent dominant . . . 99
6.1.1 Les parents . . . 100
6.1.2 Tel père, tel fils . . . 101
6.1.3 Le croisement élitiste . . . 102
6.2 Résultats expérimentaux . . . 102
6.3 Discussion et directions de recherche . . . 105

III Application des AEMO à deux problèmes de l'industrie automobile 109

Introduction 111

7 L'optimisation d'un pare-choc de voiture 113
7.1 Modélisation du pare-choc et critères de l'optimisation . . . 114
7.2 Évaluation approchée . . . 115
7.2.1 Fonctions d'approximation . . . 116
7.2.2 Plan d'expérience . . . 116
7.3 Le problème d'optimisation . . . 117
7.3.1 Solution de référence . . . 117
7.3.2 Optimisation sous contraintes . . . 117
7.4 Méthode de résolution et résultats . . . 118
7.4.1 Prise en compte des contraintes . . . 118
7.4.2 Paramètres de l'algorithme évolutionnaire . . . 119
7.4.3 Applicabilité du croisement élitiste . . . 121
7.4.4 Résultats finaux . . . 122
7.5 Conclusions et perspectives . . . 125

8 L'optimisation de l'injection de carburant pour un moteur diesel Common Rail 127
8.1 Introduction "technique" . . . 127
8.1.1 Moteurs diesel . . . 128
8.1.2 Fonctionnement du système Common Rail . . . 128
8.2 Problème de l'optimisation de l'injection de carburant . . . 130
8.2.1 Critères d'optimisation et paramètres de contrôle . . . 130
8.2.2 Position du problème d'optimisation . . . 131
8.3 Approches "globale" et "locale" . . . 132
8.3.1 Résolution du problème "global" . . . 132
8.3.2 Considération "point par point" . . . 134
8.3.3 Optimisation "point par point" vs optimisation "globale" . . . 134
8.3.4 Discussion . . . 136

Conclusions et perspectives 139

A Tests bi-objectif de Zitzler, Deb et Thiele 141

Introduction générale

De très nombreux problèmes du monde réel impliquent l'optimisation simultanée de plusieurs critères qui sont en général contradictoires. Dans les problèmes de conception, par exemple, il faut le plus souvent trouver un compromis entre des besoins technologiques et des objectifs de coût. La notion de solution optimale unique disparaît pour les problèmes de ce type au profit de la notion de *compromis* entre les divers objectifs, ou de celle d'*équilibre* dans l'espace des objectifs. L'étude des compromis a donné lieu à la définition des *solutions optimales au sens de Pareto*. La notion d'équilibre a fait la jonction avec la théorie des jeux, débouchant notamment sur l'équilibre de Nash. Le travail présenté dans cette thèse se concentre sur le premier de ces aspects, c'est-à-dire, se base sur la notion de dominance au sens de Pareto.

Les Algorithmes Évolutionnaires (AE) sont maintenant reconnus comme une technique particulièrement adaptée à la recherche de *surface de Pareto* puisqu'ils travaillent sur une "population" de solutions-candidats, à la différence de la plupart des méthodes traditionnelles, qui manipulent en général un unique point de l'espace de recherche. Cette particularité des Algorithmes Evolutionnaires rend possible l'obtention d'un ensemble de Pareto approché en un seul essai de l'algorithme, et sans obliger l'utilisateur de définir des paramètres subjectifs supplémentaires tels les poids relatifs des critères d'optimisation. C'est un avantage important car même si ces paramètres sont définis en conformité avec des informations issues du problème, ceci ne garantit pas du tout que les solutions du problème paramétré vont correspondre parfaitement aux priorités introduites lors de la fixation des poids. Notamment, nous verrons que ceci ne sera pas le cas pour les problèmes dont la surface des compromis contient des régions concaves.

Un autre avantage des AEs, mais qui n'est pas spécifique aux problèmes multi-objectif, est que ce type d'algorithmes peut être appliqué aussi bien aux problèmes discrets que continus ou mixtes, et peut travailler sur des espaces de recherche encore plus complexes et exotiques, dans lesquels par exemple il n'existe pas de notion de dérivées.

Tout cela constitue une forte motivation pour étudier le comportement des Algorithmes Evolutionnaires Multi-Objectif dans divers cas d'utilisation d'autant plus que ce type d'approche est très récent et ouvre des perspectives d'applications extrêmement riches et diversifiées.

Cette thèse est constituée de trois parties. La première commence par l'introduction des notions et des concepts de base du domaine de l'Optimisation Multi-Critère. Ensuite, le deuxième chapitre présente brièvement l'état de l'art dans le domaine des Algorithmes Évolutionnaires. Il contient également le vocabulaire particulier utilisé dans ce domaine, qui donne la clef pour traduire dans le langage de l'optimisation certains termes et expressions employés par la suite.

La deuxième partie (chapitres 3 à 6) est entièrement consacrée aux Algorithmes Évolutionnaires Multi-Objectif (AEMO). Le chapitre 3 présente l'état de l'art dans ce domaine très jeune et qui

se trouve actuellement dans une période de très forte et rapide croissance : la plupart des travaux cités dans ce chapitre sont parus après l'année 1995. Bien que les AEMO sont reconnus avoir des avantages significatifs par rapport aux techniques d'optimisation multi-critère connues auparavant, un certain nombre de difficultés se présentent lors de leur utilisation. Dans le chapitre 4, nous nous arrêtons sur deux questions importantes, pour lesquelles la réponse est évidente quand il s'agit des Algorithmes Évolutionnaires mono-objectifs, mais qui posent un problème dans le cadre multi-objectif, que ce soit dans le cadre théorique ou applicatif. Ce sont la représentation des résultats, notamment quand le nombre des objectifs est élevé, et l'évaluation et la comparaison de la performance de ces algorithmes. Nombre de solutions possibles à ces problèmes ont été proposées jusqu'à aujourd'hui, et sont présentées dans ce chapitre.

Un autre problème qui se pose lors de toute application des Algorithmes Évolutionnaires est le choix du critère d'arrêt. Si dans le cas de l'optimisation mono-objectif, ce critère se base souvent sur la valeur de la fonction optimisée, dans le domaine des AEMO, seul l'arrêt après un nombre d'itérations fixe était employé jusqu'à présent. Dans le chapitre 5, nous proposons un critère d'arrêt utilisable avec certains AEMO. Il se base, d'une part, sur la stabilisation de la performance des individus de la population, les comparaisons étant comprises au sens de la *dominance de Pareto*, et d'autre part, sur la stabilisation de l'uniformité de la distribution des *solutions non-dominées* de la population. Le critère proposé peut être vu comme un analogue d'un critère très utilisé dans le cadre mono-objectif qui est l'arrêt après un certain nombre de générations sans amélioration (*Steady Fitness Stopping Criterion*).

Le critère d'arrêt choisi de façon appropriée permet de rendre l'utilisation de l'algorithme la plus efficace possible dans la limite de la performance de cet algorithme. Par contre, un des facteurs qui peuvent améliorer la performance d'une méthode évolutionnaire, notamment, accélérer la recherche de bonnes solutions, ce sont les opérateurs de variation (croisement et mutation). Pourtant, jusqu'à maintenant, il y eu très peu de tentatives d'adapter les opérateurs stochastiques utilisés dans les Algorithmes Evolutionnaires au but particulier de la recherche de l'ensemble de compromis de Pareto. Le sixième chapitre (qui conclue la partie consacrée aux AEMO) introduit une façon particulière de choisir les "partenaires" pour l'opérateur de croisement, qui se base sur la notion de dominance de Pareto, et se démarque ainsi de l'approche "variations aveugles" en introduisant des préférences "sexuelles". L'accélération de la progression de la population de l'AEMO vers la surface de Pareto due à l'application de cet opérateur est évaluée sur trois problèmes test standards.

La troisième et dernière partie de ce travail porte sur l'étude de deux problèmes réels de conception de l'industrie automobile. Les chapitres 7 et 8 décrivent respectivement le problème de l'optimisation de la forme d'un pare-choc de voiture et le problème de l'optimisation des paramètres de contrôle de l'injection du carburant pour un moteur Diesel Poids Lourds. Ces deux applications ont été proposées par le Centre de Recherche de Fiat. Il s'agit, dans tous les deux cas, d'une optimisation multi-objectif sous contraintes.

L'étude du problème du pare-choc a été effectuée dans le cadre du projet européen INGENET. La qualité des solutions trouvées à l'aide d'un des AEMO les plus performants aujourd'hui dépasse largement la solution de référence obtenue précédemment par les chercheurs de Fiat. Une section de ce chapitre présente la méthodologie et les résultats de l'adaptation de l'opérateur de croisement introduit dans le chapitre 6, au cas de l'optimisation sous contraintes.

A la différence de ce premier problème qui peut être vu comme un cas test industriel, la résolution du problème de l'injection du carburant a fait partie d'un travail de développement qui était en cours au Centre de Recherche de Fiat (en collaboration avec Iveco). Il ne s'agit donc

pas, dans ce cas, de l'évaluation des résultats par comparaison avec une solution de référence ni, même de la possibilité de leur présentation détaillée. Le chapitre 8 présente donc simplement les résultats obtenus durant la période de la collaboration directe avec les chercheurs de Fiat qui a été relativement courte. L'intérêt principal de cette application est dans l'analyse de divers avantages qu'on peut tirer en posant le problème d'optimisation de différentes façons.

Première partie

Introduction

Chapitre 1

Optimisation multi-objectif

Il faut vous dire que les shadoks n'avaient pas encore
l'échelle dite intégrale que nous connaissons maintenant.
Ils étaient donc obligés d'avoir deux sortes d'échelles :
d'une part, des échelles pour monter et, d'autre part,
des échelles pour descendre. Et c'étaient pas les mêmes!
Les Shadoks, Num.ZO

Comme le suggère le nom, un problème d'optimisation multi-critère consiste à optimiser plusieurs fonctions objectif simultanément. Les objectifs de l'optimisation sont, en général, contradictoires. Par exemple, pour le problème

$$\begin{cases} min \quad f_1(x) = x_1, \\ min \quad f_2(x) = \frac{1+x_2}{x_1}, \\ 0.1 \leq x_1 \leq 1, \quad 0 \leq x_2 \leq 5, \end{cases} \qquad (\star)$$

l'amélioration du premier critère implique la dégradation du second. Ce conflit lors de l'optimisation des fonctions objectif se traduit par le fait qu'il n'existe pas de solution qui serait optimale du point de vue de tous les deux critères.

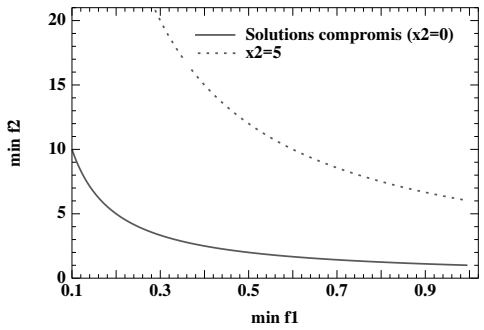


FIG. 1.1 – *Les bornes du domaine de recherche du problème* $(\star)$

Dans cet exemple simple, nous pouvons expliciter le rapport entre les fonctions optimisées : $f_2 = (1 + x_2)/f_1$. Les bornes du domaine de recherche correspondent aux valeurs extrémales du numérateur, c'est-à-dire à $x_2 = 0$ et $x_2 = 5$. La figure 1.1 montre les bornes du domaine de

recherche partiel ($f_2 \leq 20$) pour ce problème dans l'espace des critères. La courbe correspondante à $x_2 = 0$ représente l'image de l'ensemble des solutions compromis dans le plan $< f_1, f_2 >$. Ces solutions sont équivalentes entre elles au sens des deux critères d'optimisation et il n'y a aucune solution dans le domaine de recherche qui serait meilleure qu'une solution compromis sur les deux critères à la fois.

Ce sont ces solutions compromis (ou une ou certaines d'entre elles) qu'on recherche quand il s'agit de résoudre un problème d'optimisation multi-objectif. Notons qu'en pratique, il y a le plus souvent un certain nombre de contraintes qui doivent être satisfaites par les solutions du problème d'optimisation.

1.1 Position du problème

Dans tout ce qui suit, nous considérons que toutes les fonctions objectif sont à minimiser, car le principe de dualité utilisé dans le contexte d'optimisation [88], [86], permet de transformer la maximisation en la minimisation en multipliant la fonction objectif par -1. Le problème d'optimisation multi-critère peut alors être posé sous forme générale suivante:

$$\begin{cases} min \quad f_m(x), & m = 1,...,M; \\ g_j \geq 0, & j = 1,...,K; \\ h_j = 0, & j = K+1,...,L; \\ x_i^l \leq x_i \leq x_i^u, & i = 1,...,n. \end{cases} \tag{1.1}$$

Terminologie

Le vecteur $x = (x_1,...,x_n)$ est le vecteur des n *variables de décision*. Les x_i^l et x_i^u sont les bornes inférieure et supérieure respectivement de la variable x_i. Ces bornes définissent l'*espace de décision* **D**. Le terme *domaine/espace de recherche* sera utilisé comme synonyme de l'espace de décision ainsi que pour désigner l'image de l'espace de décision dans l'espace des critères. Généralement, par une *solution* nous allons entendre un élément de l'espace de recherche.

Notons qu'en tant que contraintes d'inégalité nous considérons seulement des contraintes de type "supérieur-ou-égal" car les contraintes de type "inférieur-ou-égal" peuvent être transformées en ces premières en appliquant le même principe de dualité, qui nous a permis de transformer les objectifs de maximisation en ceux de minimisation. Si une solution x ne satisfait pas au moins une des K contraintes d'inégalité ou une des L contraintes d'égalité, elle est dite *solution infaisable*, contrairement aux *solutions faisables*, qui vérifient l'ensemble des $(K+L)$ contraintes du problème. L'ensemble des solutions faisables constitue la *région faisable* de l'espace de recherche ou encore l'*espace faisable* **S**.

Convexité

Nous citons ici quelques définitions et faits liés à la convexité, car ces notions nous seront utiles par la suite.

- La *fonction* $f : R^n \rightarrow R$ est *convexe* si est seulement si pour tout pair de variables $x^{(1)}, x^{(2)} \in R^n$ et pour tout $\lambda \in [0,1]$ a lieu l'inégalité suivante:

$$f(\lambda x^{(1)} + (1-\lambda)x^{(2)}) \leq \lambda f(x^{(1)}) + (1-\lambda) f(x^{(2)}).$$

Si dans cette définition, nous remplaçons le signe d'inégalité $\leq$ par $>$, nous obtenons la définition de la *fonction non-convexe*. Pour tester si une fonction f est convexe dans un intervalle, il faut calculer sa matrice Hessienne $\nabla^2 f$ et vérifier si elle est définie positive dans chaque point de l'intervalle. Le même test est fait avec la matrice $-\nabla^2 f$ pour vérifier la non–convexité de f. Remarquons que si une fonction $g(x)$ est non–convexe, l'ensemble de x tels que $g(x) \geq 0$ est un ensemble convexe.

– Le *problème d'optimisation multi-objectif* (1.1) est appelé *convexe* si est seulement si toutes les fonctions objectif f_m sont convexes et la région faisable est un ensemble convexe (c'est-à-dire toutes les fonctions g_j sont non-convexes et toutes les fonctions h_j sont linéaires).

1.2 Dominance et optimalité de Pareto

Dans cette section nous allons introduire les notions liées à l'optimisation multi-objectif sur lesquelles toute la suite de ce travail va se baser.

1.2.1 Concept de dominance et solutions Pareto-optimales

Définition de la dominance

La solution $x^{(i)}$ du problème (1.1) est dite *dominer* une autre solution $x^{(j)}$, si les conditions suivantes sont vérifiées :

1. $f_m(x^{(i)}) \leq f_m(x^{(j)}) \quad \forall m \in \{1,...,M\}$.
2. $\exists m \in \{1,...,M\}$ tel que $f_m(x^{(i)}) < f_m(x^{(j)})$.

Si *la solution $x^{(i)}$ domine la solution $x^{(j)}$*, nous allons écrire $\mathbf{x^{(i)} \prec x^{(j)}}$.

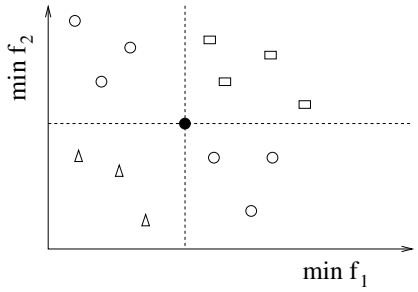


FIG. 1.2 – *Le point noir est dominé par chacun des triangles, domine chacun des rectangles et est équivalent aux anneaux au sens de la dominance*

Notons que pour toute paire de solutions $x^{(i)}$ et $x^{(j)}$, une et seulement une des affirmations suivantes est vraie (voir figure 1.2) :

– $x^{(i)}$ domine $x^{(j)}$;
– $x^{(i)}$ est dominée par $x^{(j)}$;
– $x^{(i)}$ et $x^{(j)}$ sont équivalentes au sens de la dominance.

Par la suite, les solutions équivalentes au sens de la dominance seront parfois évoquées comme *solutions équivalentes au sens de Pareto* ou, encore, comme *solutions Pareto-équivalentes*.

Propriétés de la relation de dominance

La RELATION BINAIRE DE DOMINANCE $\prec$, telle qu'elle est définie ci-dessus,

- N'EST PAS RÉFLEXIVE, car une solution ne se domine pas elle–même ;
- N'EST PAS SYMÉTRIQUE, car on n'a jamais $(x^{(i)} \prec x^{(j)})$ et $(x^{(j)} \prec x^{(i)})$;
- N'EST PAS ANTISYMÉTRIQUE, du fait de l'existence de solutions Pareto-équivalentes ;
- EST TRANSITIVE, car $(x^{(i)} \prec x^{(k)})$ & $(x^{(k)} \prec x^{(j)})$ implique $(x^{(i)} \prec x^{(j)})$.

La relation de dominance est donc une *relation d'ordre partiel strict* sur l'espace de décision [7]. Notons en particulier que $x^{(i)} \not\prec x^{(j)}$ n'implique pas $x^{(j)} \prec (x^{(i)}$.

Optimalité de Pareto

- Soit P un ensemble de solutions-candidats d'un problème d'optimisation multi-objectif. L'ensemble $P' \subseteq P$, composé de tous les éléments de P qui ne sont dominés par aucun élément de P est dit *sous-ensemble non-dominé de l'ensemble de solutions* P.

Par la suite, nous allons parfois entendre par un *ensemble* ou *sous-ensemble non-dominé*, un sous-ensemble de solutions qui sont Pareto-équivalentes entre elles même s'il existe d'autres solutions qui dominent des solutions de l'ensemble en question.

De façon analogue aux solutions optimales globale et locales dans le contexte de l'optimisation mono-objectif, les notions d'optimum local et d'optimum global au sens de Pareto peuvent être introduites :

- L'*ensemble de Pareto* **global** du problème d'optimisation multi-objectif (1.1) est l'ensemble de points tels qu'aucun autre point de l'espace faisable S ne les domine. Souvent, l'ensemble de Pareto global est évoqué simplement comme l'*ensemble de Pareto* ou encore l'*ensemble des compromis optimaux*.

- Cet ensemble est constitué par les *solutions non-dominées du problème d'optimisation* (1.1), appelées aussi *solutions de Pareto* ou *solutions Pareto-optimales* de ce problème.

- L'image de l'ensemble de Pareto dans l'espace des critères est appelée la *surface de Pareto* (ou le *front de Pareto* dans le cas de problème bi-objectif), ou également la *surface des compromis optimaux*.

- De manière similaire, un sous-ensemble P'_{loc} de l'espace de décision est appelé *ensemble de Pareto* **local** s'il existe $\epsilon > 0$ tel que pour tout élément x de P'_{loc} il n'existe pas de solutions y dans S (où ϵ est un nombre positif petit) dominant un élément de P'_{loc} et vérifiant $\|y - x\|_\infty \leq \epsilon$.

1.2.2 Procédures de recherche de l'ensemble non-dominé

La recherche de l'ensemble non-dominé P' dans un ensemble fini donné des solutions P se fait de façon similaire que la recherche du minimum dans un ensemble fini de nombres réels. Mais si chaque couple de nombres est ordonnée suivant la relation d'ordre total $<$, lors de la recherche

des éléments non-dominés, les éléments de l'ensemble P sont comparés au sens de la dominance, c'est-à-dire, à l'aide de la relation $\prec$. Dans cette section, nous allons décrire brièvement trois procédures utilisées pour identifier P'. Nous supposons dans la suite qu'il y a M objectifs et N points dans l'ensemble P.

Approche 1 : naïve et lente

Dans cette approche, chaque solution i est comparée à toutes les autres solutions jusqu'à ce qu'elle soit dominée par l'une d'elles. Si aucune ne la domine, elle est déclarée non-dominée. La complexité de cette procédure dans le pire des cas est $O(MN^2)$ [15].

Approche 2 : mise à jour continue

Cette approche est très similaire à la précédente, on lui a simplement rajouté une mémoire qui le rend en général plus efficace – mais pas dans tous les cas.

Chaque solution de P est comparée avec un sous-ensemble partiellement rempli et continûment mis à jour. Au début, l'ensemble P' contient la première solution. Ensuite, chaque solution i sera comparée aux éléments de P'. Toutes les solutions de P' dominées par i, en sont éliminées. Si aucune solution de P' ne domine i, i est ajoutée à P'. La complexité "au pire cas" de cet algorithme est la même que celle de l'approche 1, c'est-à-dire $O(MN^2)$. Mais, en moyenne, cette seconde approche est plus rapide [15].

Approche 3 : la plus efficace

Cet algorithme, proposé par Kung et al. en 1975 [66], consiste à ordonner l'ensemble P par ordre croissant du premier critère (en supposant qu'il est à minimiser) et à appeler la procédure récursive suivante:

Procédure FRONT(P)

(Notons p_i le $i^{\text{ème}}$ élément de l'ensemble P. La procédure retourne un ensemble des solutions.)

- Si $|P| = 1$, retourner P comme le résultat de la procédure FRONT(P).
- Sinon
  1. A =FRONT($\{p_1,...,p_{|P|/2}\}$) et B =FRONT($\{p_{|P|/2+1},...,p_{|P|}\}$).
  2. Pour chaque solution b_i de l'ensemble B, $i = 1,...,|B|$,
     - si b_i n'est dominée par aucune solution de A, alors $A = A \cup \{b_i\}$.
  3. Retourner A comme le résultat de la procédure FRONT(P).

La complexité de cet algorithme est $0(N(logN)^{M-2})$ pour $M > 4$, et $0(N(logN))$ pour $M = 2,3$ [66].

1.2.3 Vecteur idéal et vecteur de Nadir

Le *vecteur idéal* z^* du problème (1.1) est le vecteur de l'espace des critères dont chaque composante z_m est la solution du problème de minimisation de la fonction f_m sous les contraintes du problème (1.1) (voir figure 1.3). Généralement, ce vecteur ne correspond pas à une solution de l'espace de décision mais il est quelques fois utile en tant qu'une référence, per exemple, lors de la normalisation des valeurs des objectifs.

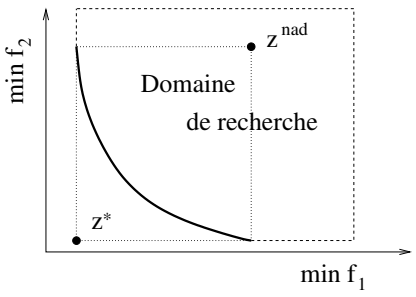


FIG. 1.3 – *Vecteur idéal z^* et vecteur de Nadir z^{nad}*

A la différence du vecteur idéal qui représente les bornes inférieures de chaque objectif dans l'espace faisable, le *vecteur de Nadir* z^{nad} correspond à leurs bornes supérieures sur la surface de Pareto et non pas dans tout l'espace faisable (voir figure 1.3). Ce vecteur est bien plus difficile de trouver que le vecteur idéal. Pour certains problèmes, la méthode de *payoff table* [76] peut être utilisée. Le vecteur de Nadir peut correspondre à une ou à aucune solution existante, en fonction du problème (notamment, de la convexité du domaine de recherche).

Pour normaliser chaque objectif, le vecteur idéal et le vecteur de Nadir sont en particulier utilisés de la façon suivante :

$$f_m^{norm} = \frac{f_m - z_m^*}{z_m^{nad} - z_m^*}.$$

1.2.4 Conditions d'optimalité

Dans cette section nous citons les conditions nécessaire et suffisante d'optimalité au sens de Pareto formulées pour le problème (1.1). Supposons que toutes les fonctions objectif f_m, $m = 1,...,M$, ainsi que les fonctions de contraintes g_j, $j = 1,...,K$, et h_j, $j = K+1,...,L$, sont continûment différentiables.

Condition nécessaire de Fritz-John
Pour que x^* soit une solution Pareto-optimale du problème (1.1), il est nécessaire qu'il existe des vecteurs $\lambda \geq 0$ et $u \geq 0$ ($\lambda \in R^M$, $u \in R^K$ et λ,$u \neq 0$) tels que les conditions suivantes soient vérifiées :

1. $\sum_{m=1}^{M} \lambda_m \nabla f_m(x^*) - \sum_{j=1}^{K} u_j \nabla g_j(x^*) = 0$,
2. $u_j g_j(x^*) = 0$ pour tout $j = 1,...,K$.

La preuve de ce théorème peut être trouvée, par exemple, dans [12]. En effet, il n'est pas difficile de noter la similarité entre ce théorème la condition nécessaire de l'optimalité de Kuhn-Tucker pour l'optimisation mono-objectif.

Condition suffisante de Karush-Kuhn-Tucker
Supposons que les fonctions f_m, $m = 1,...,M$, sont convexes, les fonctions g_j, $j = 1,...,K$ sont non-convexes et les fonctions h_j, $j = K+1,...,L$ sont linéaires (autrement dit, le problème multi-objectif (1.1) est supposé être convexe au sens de la définition donnée dans la section 1.1). Alors, si la condition nécessaire de Fritz-John est vérifiée, la solution x^* est une solution Pareto-

optimale du problème (1.1).

Pour la preuve nous référons à [76].

1.3 Equilibre de Nash

Dans cette section nous présentons une appoche de l'optimisation multi-critère dite "non-coopérative" qui, à la différence du concept introduit ci-dessus, se base sur la recherche non pas de compromis mais d'un "état d'équilibre" entre les divers objectifs.

Il s'agit de l'approche proposée par J. F. Nash en 1951 [78] initialement destinée à résoudre des problèmes d'optimisation issus de la Théorie des Jeux. Pour un problème à M objectifs, la stratégie de Nash consiste à faire en sorte que chacun des M "joueurs" optimise son objectif. La règle de ce jeu est telle que chaque joueur tente d'améliorer son critère, les $(M-1)$ objectifs restants étant "fixés" par les autres joueurs. Quand aucun joueur ne peut plus améliorer son objectif, le système est considéré comme ayant atteint son état d'équilibre, appelé *équilibre de Nash*.

Soit n la dimension de l'espace de décision X, c'est-à-dire nos M fonctions objectif dépendant de n variables $(x_1,...,x_n)$. Supposons $n \geq M$, ce qui permet de diviser l'espace de décision en M sous-espaces non-vides :

$$X = \cup_{m=1}^{M} X_m, \quad \text{t.q. } X_{m_1} \cap X_{m_2} = \emptyset \text{ pour } m_1 \neq m_2.$$

Autrement dit, toutes les variables de décision sont divisées en M groupes $(x_1,...,x_n) = (\mathbf{x_1},...,\mathbf{x_M})$. Le point $(\mathbf{x_1^\star},...,\mathbf{x_M^\star})$ est dit *équilibre de Nash* si $\forall m = 1,...,M$

$$f_m(\mathbf{x_1^\star},...,\mathbf{x_M^\star}) = \inf_{\mathbf{x_m} \in X_m} f_m(\mathbf{x_1^\star},...,\mathbf{x_{m-1}^\star},\mathbf{x_m},\mathbf{x_{m+1}^\star},...,\mathbf{x_M^\star}).$$

Les variables de décision sont donc distribuées entre les M joueurs. Chaque joueur peut améliorer le critère qui lui correspond en mettant en jeu seulement ses variables. La répartition des variables peut être arbitraire mais, en pratique, elle est souvent "suggérée" par la structure du problème.

Généralement, l'équilibre de Nash est difficile à trouver dans le cadre de l'utilisation des méthodes d'optimisation classiques. Ceci peut même être impossible, si les fonctions optimisées ne sont pas différentiables. La recherche de l'équilibre de Nash se base sur la notion d'ensemble de réaction rationnelle (*rational reaction set*) D_m de chaque joueur :

$$D_m = \{(\mathbf{x_1},...,\mathbf{x_{m-1}},\mathbf{x_m^\star},\mathbf{x_{m+1}},...,\mathbf{x_M}) \text{ t.q. } f_m(\mathbf{x_1},...,\mathbf{x_{m-1}},\mathbf{x_m^\star},\mathbf{x_{m+1}},...,\mathbf{x_M}) \leq f_m(\mathbf{x_1},...,\mathbf{x_M})\}.$$

Intuitivement, D_m est l'ensemble des meilleures solutions que le joueur m peut atteindre pour les différentes stratégies de ses adversaires. Chaque ensemble D_m peut être identifié à l'aide des vecteurs $\mathbf{x_m}$ tels que $\frac{\partial f_m(\mathbf{x_1},...,\mathbf{x_M})}{\partial \mathbf{x_m}} = 0$. L'équilibre de Nash est, en fait, le point d'intersection des ensembles D_m, $m = 1,...,M$.

Une façon élegante (et qui se marie très naturellement avec le concept de jeu compétitif) de mettre en œuvre de la méthode de Nash est décrite dans [101]. Pour trouver l'équilibre de Nash, les auteurs proposent d'utiliser les Algorithmes Evolutionnaires présentés dans le chapitre suivant.

Il y a sans doute des problèmes pour lesquels le concept de l'optimisation multi-critère basé sur l'équilibre peut être considéré comme plus approprié que celui des compromis. Cependant, les études (tant originales que faisant partie de l'état de l'art) que nous décrivons dans la suite de ce manuscrit, sont consacrées à la recherche des compromis de Pareto.

1.4 Méthodes déterministes

Dans cette section nous décrivons quelques méthodes traditionnellement utilisées pour la résolution des problèmes d'optimisation multi-objectif. Toutes ces approches passent par la transformation du problème initial en un problème d'optimisation mono-objectif.

Nous commençons par détailler les deux méthodes les plus connues et utilisées: l'optimisation par agrégation pondérée des objectifs, et l'approche par ϵ-contraintes. Quelques autres méthodes sont ensuite présentées plus brièvement.

Rappelons que l'espace des solutions faisables du problème (1.1) est noté S et est défini par:

$$S = \{x \in D: \quad g_j(x) \geq 0,\ j = 1,...,K; \quad h_j(x) = 0,\ j = K+1,...,L\}$$

1.4.1 Agrégation pondérée

Cette méthode, probablement, la plus largement utilisée dans la pratique, consiste à ramener le problème multi-critère au problème de l'optimisation d'une combinaison linéaire des objectifs initiaux. Les coefficients sont généralement choisis en fonction de l'importance relative des objectifs. Mais pour pouvoir fixer les valeurs des poids des critères de façon appropriée, il faut savoir quantifier les préférences qualitatives issues de l'application (une approche permettant de le faire de manière interactive a été proposée par Parmee et al. [82]), et mettre les fonctions objectif à l'échelle pour qu'elles aient le même ordre de magnitude.

Soit

$$F_w(x) = \sum_{m=1}^{M} w_m f_m(x),$$

où les *poids* $w_m \geq 0$ sont tels que $\sum_{m=1}^{M} w_m = 1$. La méthode "de la somme pondérée" pour la résolution du problème (1.1) consiste, alors, à résoudre le problème d'optimisation suivant:

$$\min_{x \in S} F_w(x) \tag{1.2}$$

Il a été établi un nombre de résultats théoriques clarifiant le rapport entre la solution du problème (1.2) et les solutions Pareto-optimales du problème (1.1) ([7], [76]). Notamment, les deux théorèmes suivants ont été démontrés.

Théorème 1 La solution du problème (1.2) est une solution Pareto-optimale du problème multi-objectif (1.1) si tous les poids w_m sont positifs.

Théorème 2 Si le problème d'optimisation multi-objectif (1.1) est convexe et x^* est une de ses solutions Pareto-optimales, il existe un vecteur des poids positif $(w_1,...,w_M)$ tel que x^* est la solution du problème (1.2).

Le processus de minimisation de la combinaison linéaire de deux critères est illustré dans la figure 1.4(a). En effet, l'optimisation de deux objectifs minimisant une somme pondérée correspond au "déplacement" de la droite $F = const$ (tracée dans l'espace des critères) dans le sens de la décroissance de la valeur de F tant que cette droite contient des points du domaine de recherche. (Notons que la pente de cette ligne est définie par le ratio $-w_1/w_2$.) Ce processus s'arrête alors au moment où la droite atteint la position tangente à la frontière du domaine de recherche (dans l'espace des critères). Le seul point de l'espace de recherche contenu dans la droite à ce moment-là (le point A dans la figure 1.4) représente la solution du problème (1.2), i.e. la solution de Pareto du problème (1.1) correspondante au vecteur des poids (w_1,w_2).

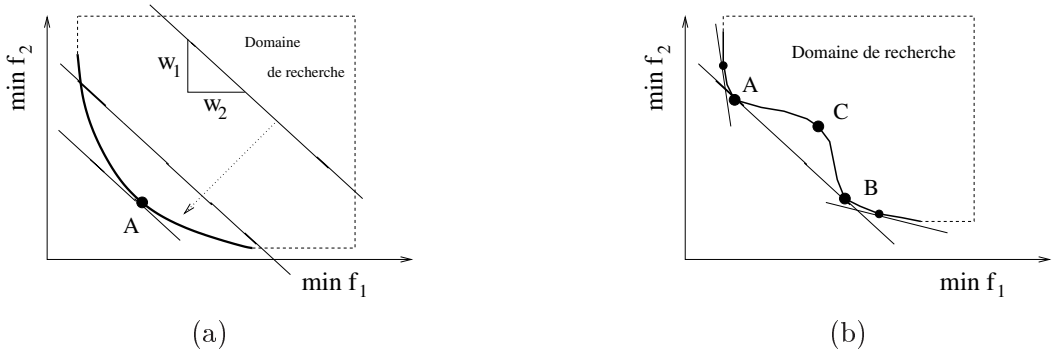


FIG. 1.4 – **(a)** *Illustration du processus de la minimisation de la somme pondérée de deux fonctions objectif. Le point A correspond à la solution du problème d'optimisation (1.2).* **(b)** *Les régions concaves de la surface de Pareto sont "inaccessibles" pour cette méthode quelques soient les valeurs w_1 et w_2.*

Discussion

Cette méthode est simple et, pour des problèmes avec la surface de Pareto convexe, elle permet théoriquement de trouver toute solution de Pareto-optimale par la minimisation répétée de sommes pondérées des critères avec des poids différents.

Cependant, pour la plupart des problèmes non-linéaires, même le fait que des valeurs de poids uniformément distribuées aient été successivement utilisées ne garantit pas du tout que les solutions de Pareto correspondantes seront, à leur tour, uniformément distribuées sur la surface des compromis. Ceci rend difficile le choix des valeurs des poids de façon qu'elles représente correctement l'importance relative des critères.

De plus, la plupart des méthodes d'optimisation mono-objectif sont basées sur le critère d'optimalité du premier ordre. Il faut alors, en principe, vérifier a posteriori si la solution retrouvée est effectivement solution du problème (1.2).

Enfin, pour des problèmes dont la surface de Pareto contient des régions non-convexes, les solutions de Pareto situées dans telles régions ne peuvent pas être retrouvées par la méthode de la somme pondérée, quelle que soit la combinaison des valeurs des poids. Ce fait est illustré dans la figure 1.4(b) : il n'y a pas de tangente au front de Pareto qui passerait par le point C sans qu'une autre droite de même pente et de valeur de F moindre ne coupe aussi le front, donnant de meilleures valeurs du critère pondéré (ceci est vrai pour tous les points entre A et B).

1.4.2 Méthode de ϵ-contraintes

En 1971, Haimes et al. [45] ont suggéré la réformulation du problème d'optimisation (1.1) sous forme suivante:

$$\begin{cases} \min_{x \in S} f_\mu(x), \\ f_m(x) \leq \epsilon_m, \qquad j = 1,...,M, \quad m \neq \mu; \end{cases} \tag{1.3}$$

où $\mu \in \{1,...,M\}$ et les paramètres ϵ_m sont à définir par l'utilisateur. En d'autres termes, une des fonctions qu'il faut optimiser est retenue comme unique objectif, tandis que les critères restants sont transformés en contraintes.

Par exemple, supposons que, lors de la résolution d'un problème de minimisation de deux objectifs, nous ayons décidé de retenir f_2 comme fonction à optimiser sous la contrainte $f_1(x) \leq \epsilon_1$. La figure 1.5 aide à illustrer cette méthode pour des scénarios correspondants à de différentes valeurs de ϵ_1. Le domaine de recherche est divisé dans l'espace des critères par la droite verticale

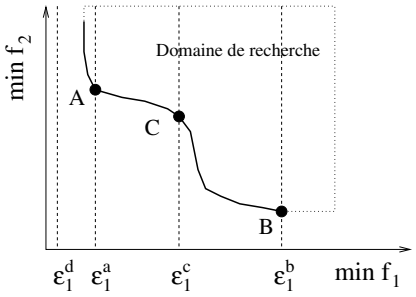


FIG. 1.5 – *Illustration de l'approche de ϵ-contraintes*

$f_1 = \epsilon_1$ en deux parties: la partie faisable ($f_1 \leq \epsilon_1$) et la partie infaisable ($f_1 \geq \epsilon_1$). Résoudre le problème (1.3) revient donc à trouver un point faisable et tel que la valeur de $f_2(x)$ soit la plus petite possible. Dans la figure 1.5, lors du scénario $\epsilon_1 = \epsilon_1^c$, c'est le point C.

En ce qui concerne l'applicabilité de la méthode d'ϵ-contraintes, le résultat suivant a été démontré:

Théorème 3 Pour tout vecteur $\epsilon = (\epsilon_1,...,\epsilon_{\mu-1},\epsilon_{\mu+1},...,\epsilon_M)$ tel que le problème (1.3) a une solution, cette solution est une solution Pareto-optimale du problème multi-objectif (1.1).

Discussion

La figure 1.5 (scénario ϵ_1^c) illustre le fait qu'à la différence de l'approche de la somme pondérée, la méthode de ϵ-contraintes est capable de retrouver les solutions de Pareto appartenantes à des régions non-convexes de la surface des compromis. En terme de paramètres demandés ces deux approches sont équivalentes.

Le choix du vecteur ϵ est très important et est lié à de certaines difficultés. Notamment, chaque valeur ϵ_m doit être choisie entre les valeurs extrémales de l'objectif correspondant f_m. Par exemple, si $\epsilon_1 = \epsilon_1^d$ (figure 1.5) a été choisi, le problème (1.3) n'a pas de solutions faisables. D'autre part, pour toute valeur de ϵ_1 supérieure à ϵ_1^b la même solution (correspondante au point B dans la figure 1.5) sera trouvée.

1.4.3 Autres approches

Comme l'approche de la somme pondérée, les trois méthodes suivantes consistent à rassembler (chacune à sa façon) les objectifs du problème (1.1) en une fonction unique, qui est ensuite optimisée.

Métriques pondérées

Notons

$$l_{p,w}(x) = \big(\sum_{m=1}^{M} w_m |f_m(x) - z_m^*|^p \big)^{1/p},$$

où $w_m \in [0,1]$ tels que $\sum_{m=1}^{M} w_m = 1$, et z^* est le vecteur idéal du problème multi-objectif initial (voir section 1.2.3). On peut également utiliser à la place du vecteur idéal n'importe quel vecteur (dit "vecteur utopique" ou *utopian vector*) dont chaque composante est plus petite (s'il s'agit de la minimisation) que la composante correspondante du vecteur idéal.

Lors de l'application de la méthode des métriques pondérées, le problème initial (1.1) est reformulé de façon suivante :

$$\min_{x \in S} l_p(x). \tag{1.4}$$

Quand $p = \infty$, il s'agit de résoudre le *problème pondéré de Tchebycheff* pour lequel la fonction objectif est la suivante :

$$l_\infty(x) = \max_{m=1}^{M} w_m |f_m(x) - z_m^*|.$$

Le degré du succès de la recherche des solutions appartenantes aux régions non-convexes de la surface de Pareto augmente avec la valeur de p. En résolvant le problème pondéré de Tchebycheff avec de différents valeurs des poids, il est théoriquement possible de trouver toutes les solutions de Pareto du problème d'optimisation initial.

Cependant, il est important de noter que quand la valeur de p croit, la fonction objectif devient non-différentiable, ce qui pose un problème lors de l'utilisation de nombreux méthodes de gradient. De plus, cette méthode demande l'optimisation préalable de chaque objectif séparément afin de trouver le vecteur idéal.

Méthodes de Benson

Comme les approches basées sur les métriques, ce type de méthodes demande de fixer un vecteur de référence z^0 dans l'espace des objectifs. Cette fois z^0 est choisi aléatoirement dans la région faisable. Ensuite, le problème d'optimisation suivant est résolu :

$$\begin{cases} \max_{x \in S} \sum_{m=1}^{M} max(0,(z_m^0 - f_m(x)), \\ f_m(x) \leq z_m^0, \qquad\qquad\qquad\qquad m = 1,...,M \end{cases} \tag{1.5}$$

A la différence de l'optimisation des métriques pondérées, les méthodes de Benson ne nécessitent pas un choix de poids w_m. D'autre part, les solutions des régions non-convexes peuvent en principe être retrouvées. Par contre, cette approche ajoute des contraintes ; et de plus, la fonction objectif du problème (1.5) n'est pas différentiable.

Optimisation de la fonction d'utilité

Une autre façon de rassembler les critères initiaux en une fonction consiste à choisir une fonction dite *d'utilité* $\mathcal{U} : \mathcal{R}^{\mathcal{M}} \to \mathcal{R}$ et de résoudre le problème

$$\begin{cases} max & \mathcal{U}(f(x)), \\ x \in S. \end{cases} \tag{1.6}$$

Notons que les méthodes de la somme pondérée et des métriques pondérées peuvent être vues comme les cas particuliers de l'application de cette approche. Des issues supplémentaires concernant l'application des fonctions d'utilité peuvent être trouvées, par exemple, dans le travail de Keeney et Raiffa [61].

1.4.4 Moralité

Toutes ces approches présentent un certain nombre d'inconvénients. Tout d'abord, elles nécessitent le choix des paramètres qui suppose la disposition a priori de certaines informations liées au problème – et ces informations ne se limitent pas à des priorités entre de différents objectifs. Afin d'obtenir des résultats qui correspondent à ces priorités, une connaissance plus profonde de la nature du problème est souvent nécessaire afin de prévoir comment tel ou tel choix des paramètres de la méthode va influencer le résultat. Ce problème est lié au fait que toutes les approches décrites dans cette section ne fournissent qu'une seule solution, qui dépend fortement des paramètres choisis au départ.

Mais même si plusieurs essais sont faits avec des paramètres différents, on ne peut être sûr d'explorer l'ensemble de l'espace de recherche efficacement. Une des raisons en est l'impossibilité de trouver les régions non-convexes de la surface des compromis lors de l'application de certaines méthodes. Mais, même quand ce n'est pas le cas, s'il les critères d'optimisation sont nombreux, de TRÈS nombreux essais sont nécessaires pour approcher la surface de Pareto multidimensionnelle.

Ceci dit, si le but de l'optimisation consiste à trouver une (ou quelques unes) solution compromis quelconque, sans forcément garantir sa conformité avec les priorités issues du problème, ou si les informations dont on dispose sont suffisamment complètes pour la garantir, il n'est nul besoin de chercher des méthodes autres que celles, présentée dans ce chapitre.

En effet, la majorité de ces méthodes garantissent la convergence vers une des solutions Pareto-optimales. De plus, elles sont simples à mettre en œuvre et relativement peu coûteuses.

1.5 Si le but justifie les moyens

Dans la pratique, lors de nombreux développements industriels, il est en général intéressant d'exploiter tous les moyens disponibles pour tirer le maximum du profit de l'étape de l'optimisation. Le champs d'application de nouvelles techniques d'optimisation multi-objectif plus performantes et plus coûteuses, telles que les Algorithmes Evolutionnaires, s'élargit au fur et à mesure que les experts de divers domaines d'applications se rendent compte de tous les avantages que ces approches peuvent leur apporter.

Notamment, quand il s'agit de résoudre un problème de conception, il est souvent préférable de prendre la décision finale à partir des informations les plus complètes possible, même si leur obtention demande un coût de calcul supplémentaire. En terme d'optimisation multi-critère, ça signifie que le choix d'une solution optimale a posteriori, c'est à dire, en disposant d'une

bonne approximation de tout l'ensemble de Pareto, est, dans tels cas, plus approprié que l'obtention des solutions des problèmes "paramétrées" a priori et sans savoir exactement comment la paramétrisation choisie va diriger la recherche.

D'autre part, la disposition d'informations plus complètes (notamment, la possibilité d'analyser l'allure de la surface de Pareto) peut quelque fois aider à mieux comprendre la nature du problème. De plus, dans des applications réelles, les fonctions objectif font généralement partie du modèle construit par des ingénieurs pour décrire le phénomène étudié. La phase de l'optimisation reboucle souvent vers la correction de ce modèle avant d'arriver à l'étape de la prise de décision finale. Dans ce cas, la connaissance de l'ensemble des compromis peut être extrêmement utile, et peut contribuer à la validation du modèle ainsi qu'aider à réaliser son applicabilité effective.

En résumé, il n'y a pas de recette générale du choix de la méthode d'optimisation multicritère, car chaque application a des exigences et des limitations particulières. Toute la suite de cette thèse est consacrée aux cas où l'on a intérêt de chercher une bonne approximation de la surface de Pareto.

Chapitre 2

Evolution Artificielle

En essayant continuellement on finit par réussir.
Donc: plus ça rate, plus on a de chances que ça marche.
Devise Shadok

Ce chapitre introduit les Algorithmes Evolutionnaires (**AE**) et présente les différentes notations et abréviations utilisées par la suite. La première section est une introduction générale aux AE, illustrant leurs origines, leur principe et leur histoire. Ensuite, les deux représentations les plus utilisées, binaire et réelle, sont introduites ainsi que les opérateurs "génétiques" correspondants. Dans la thèse, nous travaillons seulement avec la représentation réelle; la représentation binaire est décrite dans ce chapitre vue son importance dans l'histoire des AE.

2.1 Algorithmes Evolutionnaires

Les AE sont des algorithmes stochastiques d'optimisation inspirés du paradigme de l'évolution darwinienne des populations. Selon Darwin, les individus les plus aptes survivent à la sélection naturelle et se reproduisent et ceci se répète d'une génération à l'autre, menant à l'adaptation de la population des individus à l'environnement au cours de l'évolution. En termes d'optimisation, l'évolution se traduit par un processus itératif de recherche de l'optimum dans l'espace de décision. Le critère pour définir les éléments les "plus aptes" d'un sous-ensemble fini de cet espace correspond, naturellement, à l'objectif d'optimisation. L'adaptation à l'environnement est réalisée par le fait de trouver à chaque itération suivante des solutions potentielles meilleures, la performance des solutions étant évaluée à la base des valeurs correspondantes de la fonction objectif.

Les AE dont les versions différentes ont été introduites par Fogel en 1965 [32], Rechenberg en 1973 [87], Holland en 1975 [52], ont été popularisés en 1989 par Goldberg[41]. L'avantage crucial de ces méthodes devant les algorithmes d'optimisation traditionnels consiste en ce que les AE "se contentent" de connaître les valeurs de la fonction objectif non seulement sans faire appel à la dérivée de cette fonction mais même sans exiger son expression analytique.

2.1.1 Vocabulaire et principe de fonctionnement

Cette section est très importante, car elle permet d'établir le rapport entre l'optimisation et tout ce qui est écrit dans les chapitres restants de cette thèse.

- INDIVIDU - un élément de l'espace de recherche.
- PERFORMANCE - (notée F comme *fitness*) la mesure de la qualité des individus basée sur l'objectif de l'optimisation et permettant de comparer les individus entre eux afin d'en déterminer plus et moins aptes.
- EVALUATION d'un individu - le calcul de sa performance.
- POPULATION - un ensemble fini (DE TAILLE N) d'individus.
- EVOLUTION - un processus itératif de recherche d'un (ou plusieurs) individu optimal.
- GÉNÉRATION correspond à l'itération, c'est-à-dire, repère le moment de l'évolution. Mais, parfois, ce terme signifie la population en une certaine itération.
- CROISEMENT - l'opérateur de reproduction appliqué avec la probabilité p_c et correspondant à un brassage d'information entre les individus de la population. Il consiste à échanger des parties composantes (gènes) entre deux ou plusieurs individus.
- MUTATION - l'opérateur de modification d'un ou plusieurs gènes appliqué avec la probabilité p_m dans le but d'introduire une nouvelle variabilité dans la population
- SÉLECTION - processus du choix des individus pour la reproduction basé sur leur performance.
- REMPLACEMENT - processus de formation d'une nouvelle population à partir des ensembles de parents et d'enfants effectué le plus souvent sur la base de leur performance.

Cette première partie du vocabulaire suffit pour pouvoir accompagner de commentaires le schéma général du fonctionnement d'un Algorithme Evolutionnaire présenté dans la figure 2.1.

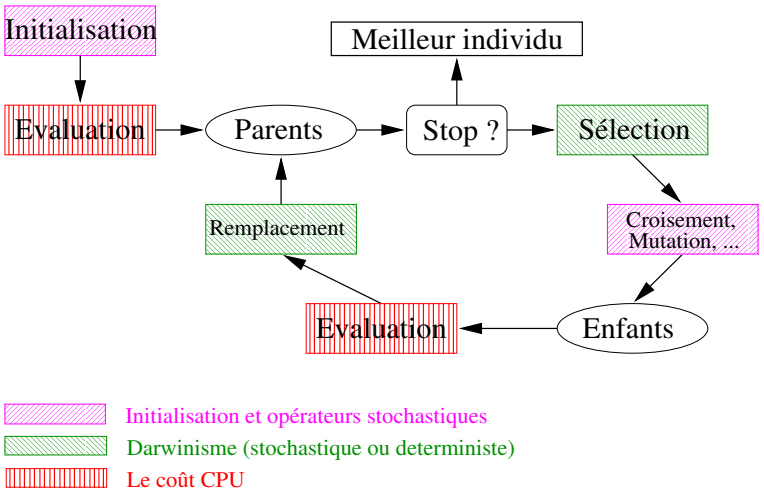


FIG. 2.1 – *Cycle d'un AE* (COPYRIGHT *Marc Schoenauer*)

Nous commençons par fixer les paramètres tels que, par exemple, la taille de la population, les probabilités du croisement et de la mutation. Nous choisissons aussi le type de sélection, de

remplacement, les opérateurs de croisement et de mutation et le critère d'arrêt. Les possibilités standard qui aident à fixer tous ces degrés de liberté seront présentées dans le sections suivantes de ce chapitre.

Le processus de l'optimisation par un AE commence par choisir aléatoirement dans l'espace de recherche un nombre fini d'individus qui vont constituer la population initiale. A l'étape de sélection, qui est (le plus souvent) stochastique et basée sur la performance, certains individus sont choisis dans la population initiale préalablement évaluée pour la reproduction. L'application des opérateurs de croisement et de mutation permet d'obtenir un nouvel ensemble d'individus, appelés "enfants", qui doivent être évalués à leur tour afin de pouvoir décider lesquels d'entre eux "méritent" de remplacer certains parents et de faire partie de la génération suivante. Notons que, à la différence de la Nature, la plupart des AE travaillent avec une population de taille fixe.

La convergence de ce processus n'est pas garantie, mais si elle a lieu, c'est grâce aux étapes "darwinistes", au cours desquelles les individus moins aptes (ayant la performance inférieure) sont éliminés avec une probabilité plus grande que ceux qui se sont mieux adaptés à l'environnement. Par exemple, la probabilité pour un individu d'être choisi pour la reproduction peut être proportionnelle à la performance relative de cet individu.

Le processus peut être arrêté au bout d'un nombre d'itérations fixé a priori, ou quand le meilleur individu de la population atteint un certain niveau de performance.

Génotype vs phénotype

L'espace de recherche des AE, appelé aussi, en termes génétiques, l'*espace des génotypes*, peut être différent de l'espace des solutions, appelé l'*espace des phénotypes* sur lequel est défini la performance. Par exemple, un génotype peut être un vecteur réel, et représenter une structure physique complexe dans l'espace des phénotypes.

La *fonction de codage* qui transforme un phénotype en un génotype doit être injective : à chaque phénotype correspond un seul génotype. La performance du génotype est celle du phénotype correspondant. L'initialisation de la population se fait, généralement, d'une façon aléatoire dans l'espace des génotypes. Le choix des opérateurs de croisement et de mutation dépend du codage des individus. Par contre, les opérateurs de sélection et de remplacement sont indépendants de la représentation génotypique, puisqu'ils utilisent uniquement la performance des individus.

Exploitation vs exploration

A chacune des étapes de l'algorithme décrit ci-dessus, il est important de réaliser un compromis entre l'*exploitation* et l'*exploration*. Avec ces deux notions, nous introduisons un dilemme aussi important que difficile à résoudre lors de l'utilisation des AE.

Le choix de l'exploitation revient à effectuer une recherche locale dans le voisinage des meilleurs individus. L'idée est d'exploiter efficacement l'information obtenue précédemment par les meilleures solutions pour spéculer sur la position de nouveaux points avec l'espoir d'améliorer la performance.

Toutefois, l'exploitation toute seule ne permet pas de préserver la *diversité génétique*, puisqu'elle guide la population vers le plus proche optimum local. La population devient alors homogène et son évolution se réduit à l'évolution de l'individu dominant. Ce phénomène est appelé *convergence prématurée*.

Le choix de l'exploration consiste à diriger la recherche de façon à préserver une population diversifiée. L'exploration de nouvelles régions de l'espace de recherche permet d'introduire dans la population des informations innovatrices. Cependant, l'excès d'exploration, clairement, empêche la convergence.

Il faut donc maintenir un équilibre entre l'exploitation de bonnes solutions rencontrées et l'exploration des zones inconnues de l'espace de recherche pour garantir l'efficacité de l'algorithme.

Typiquement, les opérations de sélection et de croisement sont des étapes d'exploitation, alors que l'initialisation et la mutation sont des étapes d'exploration. (Notons que la mutation peut devenir un opérateur d'exploitation si les perturbations générées sont très petites.)

En effet, si on sélectionne plusieurs copies des meilleurs individus, un risque de convergence prématurée se présente. Mais si la variance de la sélection est trop grande, c'est un phénomène de *dérive génétique* qui risque de se produire, permettant à certains individus de survivre au détriment d'individus meilleurs. Il faut donc que la sélection soit capable de maintenir la diversité tout en favorisant les meilleurs.

2.1.2 Historique

Les Algorithmes Evolutionnaires élaborés depuis les années soixante, se classent en 4 catégories de base que nous présentons dans cette section :

Les Algorithmes Génétiques (AG)

Les AG sont probablement les algorithmes les plus connus et utilisés dans le calcul évolutionnaire. Ils ont été développés dans les années soixante par Holland [51]. L'idée de son système était d'étudier, dans le cadre de la psychologie/biologie, le processus d'adaptation des populations à la base des données sensorielles introduites au système grâce à des détecteurs binaires [51, 52].

Les AG ont été appliqués à l'optimisation paramétrique pour la première fois par De Jong en 1975 [22], qui a posé les fondements de cette technique d'application. Cependant, le manque de puissance des ordinateurs à l'époque ne permettait pas leur application sur des problèmes réels de grande taille. Ce n'est que pendant les années quatre vingt dix, précisément, avec l'apparition de l'ouvrage de référence écrit par Goldberg [41], que les AG se sont faits connaître dans la communauté scientifique.

Les Stratégies d'Evolution (SE)

Les Stratégies d'Evolution sont dédiées à la résolution de problèmes d'optimisation dans l'espace des vecteurs de réels. Les premiers efforts pour la mise en place des SE ont eu lieu en 1964 à l'université de Berlin au cours de la résolution d'un problème aérodynamique [98]. En 1965, Rechenberg a introduit l'algorithme (1+1)-ES, qui fait évoluer un seul individu et utilise la mutation Gaussienne (section 2.2.2) pour assurer cette évolution. Il a proposé la règle 1/5 pour l'adaptation de la déviation standard de la mutation [87].

Cette stratégie a servi de base pour la transition à $(\mu + \lambda)$-ES et (μ,λ)-ES (section 2.3.2) introduits par Schwefel en 1977 [99, 100]. De même, de nouveaux opérateurs de mutation et de croisement ont été mis en place. Sont alors apparus les notions d'auto-adaptativité pour la mutation ainsi que des différents types de croisement entre vecteurs réels illustrés dans la section 2.2.2.

La Programmation Evolutionnaire (PE)

La Programmation Evolutionnaire, développée par L.J. Fogel [32], se base sur l'évolution d'une population d'automates à états finis (figure 2.2), et est utilisée pour la résolution de problèmes de prédiction. La table de transition des automates est modifiée grâce à des mutations aléatoires uniformes dans l'alphabet discret correspondant. L'évaluation de la performance des individus correspond au nombre de symboles prédits correctement. Chaque automate de la population parente génère un enfant par mutation, et les meilleures solutions entre les parents et les enfants sont sélectionnées pour survivre, ce qui correspond à la stratégie $(\mu + \mu)$ dans la terminologie des SE.

La PE a été ensuite développée et son domaine a été élargi par D.B. Fogel, afin qu'il puissent travailler dans l'espace réel [30, 31], où la sélection déterministe est remplacée par un tournoi stochastique.

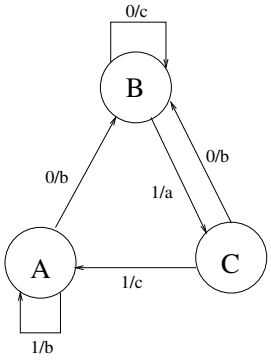


FIG. 2.2 – *Exemple d'un automate à états finis ayant trois états différents* $S = \{A,B,C\}$*, un alphabet d'entrée* $I = \{0,1\}$*, et un alphabet de sortie* $O = \{a,b,c\}$*. Chaque arête entre deux états indique une transition possible, et la fonction de transition* $\delta : S \times I \to S \times O$ *est spécifiée par les labels au niveau des arêtes ayant la forme* i/o*, signifiant que* $\delta((s_i,i)) = (s_j,o)$*.*

La Programmation Génétique (PG)

La première utilisation des structures arborescentes dans un algorithme génétique a été suggérée par Cramer en 1985 [10], dans le but de faire évoluer des sous-programmes séquentiels d'un langage algorithmique simple. L'algorithme d'évolution utilisé est le SSGA (c.f. section 2.3.3).

L'adoption de cette présentation pour définir la Programmation Génétique comme un nouvel algorithme évolutionnaire a été faite par John Koza en 1992 [63]. Son objectif initial était de faire évoluer des sous-programmes du langage LISP (figure 2.3). Grâce à l'ouvrage de Koza [63], l'utilisation de la PG s'est étendue à la résolution de nombreux types de problèmes dont les solutions peuvent être représentées par des structures arborescentes, comme les représentations fonctionnelles linéaires [79], les graphes [107, 91], les structures moléculaires etc.

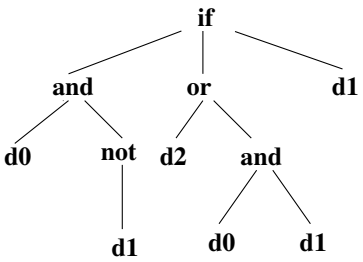


FIG. 2.3 – *Exemple d'une solution GP en LISP :* $\{d0,d1,d2\}$ *est un ensemble d'instructions constituant les terminaux, et* $\{if,and,or\}$ *sont des expressions LISP constituant les nœuds.*

2.2 Représentations et opérateurs correspondants

2.2.1 Représentation binaire

Le codage binaire est le cadre général des AG traditionnels [41]. Chaque individu I est représenté par un vecteur binaire (ou chaîne de bits), où chaque élément prend la valeur 0 ou 1 :

$$I = (a_1, \cdots, a_l) \in \{0,1\}^l,$$

où l est la taille du vecteur (nombre de bits).

Cette représentation s'adapte bien à des problèmes où les solutions potentielles ont une représentation binaire canonique, comme les problèmes booléens. Elle s'applique aussi pour des problèmes d'optimisation paramétrique continue ($f : \mathbb{R}^n \rightarrow \mathbb{R}$), mais il est alors nécessaire de définir une technique de codage adéquate de $\mathbb{R}^n$ vers $\{0,1\}^l$.

Croisement binaire

Le croisement est vu comme l'opérateur d'exploration essentiel des AG. Son rôle consiste à combiner les génotypes de deux individus pour en obtenir deux nouveaux, en échangeant un ou plusieurs fragments des deux génotypes. On distingue plusieurs croisements possibles, dont les plus utilisés sont :

- LE CROISEMENT À 1 POINT [51] :
  un seul fragment est échangé selon un point de coupure choisi aléatoirement;

- LE CROISEMENT À 2 POINTS [24, 25] :
  deux fragments sont échangés selon 2 points de coupure choisis aléatoirement;

- LE CROISEMENT UNIFORME [106] :
  On échange les bits à chaque position indépendamment avec une probabilité de 0.5. Il peut être vu comme un croisement multi-points dont le nombre de coupures est déterminé aléatoirement au cours de l'opération.

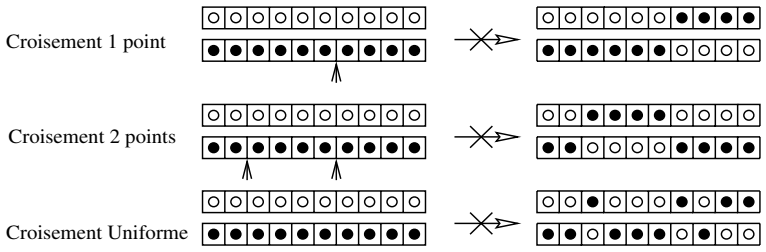


FIG. 2.4 – *Les divers croisements binaires : à 1 point, à 2 points et le croisement uniforme.*

Mutation binaire

C'est un opérateur qui inverse aléatoirement les bits du génotype avec une faible probabilité, typiquement de 0,01 à 0,001. Il est destiné à diversifier génétiquement les éléments de la population (en son absence, aucune caractéristique génétique nouvelle ne peut apparaître). Les mutations les plus utilisées sont :

- LA MUTATION STOCHASTIQUE (*bit flip*) [59, 41] :
  inverse chaque bit indépendamment avec une probabilité $\frac{c}{l}$, avec $c > 0$ et l est la taille du vecteur. C'est la mutation la plus employée dans la représentation binaire.

- LA MUTATION 1 BIT [59] :
  inverse le symbole d'un bit choisi au hasard avec une probabilité p_m.

FIG. 2.5 – La mutation 1 bit : Le symbole du 5^{eme} bit a été inversé.

2.2.2 Représentation réelle

Avec la représentation réelle, l'espace de recherche est l'espace réel $\mathbb{R}^n$ (variables non bornées) ou une partie de l'espace réel $S \subset \mathbb{R}^n$ (variables bornées).

Cette représentation a été introduite initialement pour les Stratégies d'Evolution [99], mais son utilisation s'est étendue rapidement aux autres types d'Algorithmes Evolutionnaires. Pour des problèmes d'optimisation dans l'espace réel, l'espace des génotypes s'identifie à l'espace des phénotypes : $I = \overrightarrow{x} = (x_1, \cdots, x_n) \in \mathbb{R}^n$.

Croisement réel

Dans la représentation réelle, il y a deux manières de combiner deux allèles de deux parents : choisir l'un des deux allèles (croisement discret) ou combiner linéairement les deux (croisement intermédiaire ou arithmétique). Dans le deuxième cas, plusieurs variantes ont été proposées.

- LE CROISEMENT DISCRET :

$$\forall i \in \{1, \cdots, n\},\ x'_i = x_{S,i} \text{ ou } x_{T,i},$$

où n est la taille du vecteur réel et S et T sont deux individus sélectionnés de la population parente pour l'ensemble des composantes de $\overrightarrow{x}$.

– Le croisement intermédiaire (arithmétique) :

$$\forall i \in \{1, \cdots, n\},\ x'_i = \alpha x_{S,i} + (1-\alpha) x_{T,i},$$

où α est une variable aléatoire uniforme appartenant à l'intervalle [0,1] ($\alpha = \mathcal{U}[0,1]$) et S et T sont deux individus sélectionnés pour l'ensemble des composantes de $\overrightarrow{x}$.
Une autre variante du croisement intermédiaire est le croisement BLX-ϕ ("*Blend Crossover*"), proposé par Eshelman et Schaffer [28], qui a la propriété de pouvoir élargir le domaine de définition de l'enfant, en utilisant un intervalle plus large pour α. En effet, α n'est plus compris entre 0 et 1, mais entre $(-\phi)$ et $(1+\phi)$, avec $0 < \phi < 1$ (figure 2.6). On note que le croisement BLX-0 correspond au croisement arithmétique traditionnel.

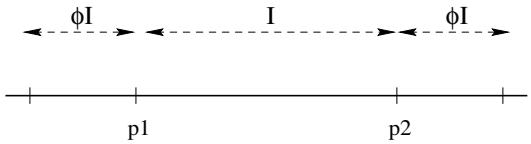


Fig. 2.6 – *BLX-ϕ.*

Dans son ouvrage sur la théorie des stratégies d'évolution [100], Schwefel a imaginé un autre type de croisement où toute la population parente peut participer à la génération d'un enfant, appelé le croisement global. Chaque parent est sélectionné indépendamment. Comme pour le croisement simple, selon la stratégie de génération des allèles de l'enfant (discrète ou intermédiaire), deux types de croisement global sont possibles : le croisement global discret et le croisement global intermédiaire.

On note que le croisement multi-parent existe aussi dans la représentation binaire [26].

– Le croisement global discret :

$$\forall i \in \{1, \cdots, n\},\ x'_i = x_{S_i,i},$$

où S_i est un individu sélectionné de la population parente pour la composante x_i. On note qu'il y a autant de parents sélectionnés que de composantes dans l'individu.

– Le croisement global intermédiaire :

$$\forall i \in \{1, \cdots, n\},\ \ x'_i = \alpha_i x_{S_i,i} + (1-\alpha) x_{T_i,i},$$

où $\alpha_i = \mathcal{U}[0,1]$ et S_i et T_i sont deux individus sélectionnés de la population parente pour la composante x_i.

Mutation réelle

– MUTATIONS GAUSSIENNES :

La principale technique utilisée pour la mutation réelle est l'ajout d'un bruit Gaussien aux différentes composantes du vecteur $\overrightarrow{x}$. La difficulté de cette approche est l'ajustement de l'écart-type σ du bruit généré. En effet, au début de l'évolution d'une population, σ doit être assez élevé pour générer de fortes perturbations et ainsi explorer rapidement tout l'espace de recherche. Une fois les pics de la fonction étudiée localisés, l'algorithme doit être capable de déterminer avec précision les solutions optimales.
Par conséquent, σ ne doit pas être constant au cours de l'évolution. Pour résoudre ce problème, plusieurs stratégies adaptatives et auto-adaptatives ont été proposées pour ajuster la déviation au cours de l'évolution. Ce sont, par exemple, la règle des 1/5 proposé par Rechenberg [87] ou la mutation Log-Normale proposée par Schwefel [100].

– AUTRES OPÉRATEURS DE MUTATION :

D'autres opérateurs de mutation de vecteurs réels ont été proposés par Michalewicz dans [72], dédiés à des problèmes contraints (variables bornées et/ou soumises à des contraintes linéaires). Ce sont des opérateurs unaires, qui à partir d'un parent $\overrightarrow{x} = (x_1, \cdots, x_k, \cdots, x_n)$, génèrent un enfant $\overrightarrow{x}' = (x_1, \cdots, x'_k, \cdots, x_n)$, où x'_k est l'élément muté et k est choisi aléatoirement dans l'intervalle [1,n]. Pour calculer x'_k, ils utilisent le domaine admissible des variables du rang k [$l(k)$,$u(k)$], défini à partir des bornes de l'espace de recherche et des contraintes du problème s'il y en a, à condition qu'elles soient linéaires.

2.3 Darwinisme artificiel

2.3.1 Sélection

Roulette (sélection proportionnelle)

La première sélection mise en place pour les AG est le tirage à la roulette, ou *Roulette Wheel Selection* (*RWS*) introduite par Goldberg [41]. C'est une méthode stochastique qui exploite la métaphore d'une roulette de casino, qui compterait autant de cases que d'individus dans la population. La largeur de la case d'un individu $\overrightarrow{x_i}$ est proportionnelle à sa performance $f(\overrightarrow{x_i})$: $\frac{f(\overrightarrow{x_i})}{\sum_j^n f(\overrightarrow{x_j})}$. Le roue étant lancée, l'individu sélectionné est désigné par l'arrêt de la roue sur sa case (figure 2.7).

L'espérance n_i de nombre de copies d'un élément $\overrightarrow{x_i}$ de la population courante est donnée par l'expression :

$$n_i = \frac{N}{\sum_j^N f(\overrightarrow{x_j})} f(\overrightarrow{x_i}),$$

où N est le nombre d'individus. L'espérance maximale $Max(n_i)$ dans l'ensemble des éléments de la population est appelée la *pression sélective*.

Cette méthode favorise les meilleurs individus, mais tous les individus ont toujours des chances d'être sélectionnés. Cependant, la méthode peut causer une perte de la diversité de la population si la pression sélective (ou n_i du meilleur) est élevée. De plus, sa variance et son coût sont élevés.

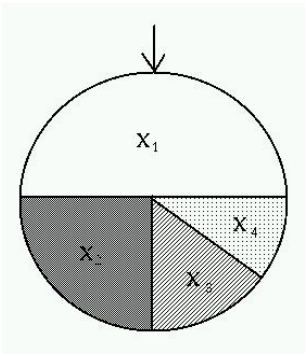


FIG. 2.7 – *Roue de loterie pour une population de 4 individus avec* $f(x_i) = \{50{,}25{,}15{,}10\}$. *Pour tirer un élément, on lance la roue, et si elle s'arrête sur la case i,* x_i *est sélectionné.*

Afin de diminuer la variance, Baker a proposé en 1987 la sélection à la roulette avec reste stochastique, ou *stochastic universal sampling* (SUS) [4]. Avec cette approche, d'une façon similaire à *RWS*, on considère une roulette partitionnée en autant de cases que d'individus, où chaque case est de taille proportionnelle à la performance. Mais cette fois, les individus sélectionnés sont désignés par un ensemble de points équidistants (figure 2.8). Le nombre effectif de copies de l'individu $\overrightarrow{x_i}$ sera la partie entière inférieure ou supérieure de son espérance n_i.

Le bruit de sélection étant plus faible, la dérive génétique se manifeste moins qu'avec la méthode *RWS*.

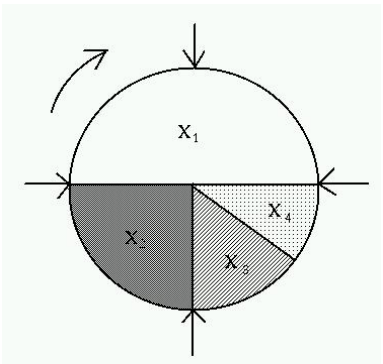


FIG. 2.8 – *Stochastic Universal Sampling Selection : pour sélectionner des individus, on lance la roue, et quand elle s'arrête, chaque individu dont la case est pointée par un marqueur est sélectionné.*

Tournois

La sélection par tournoi ne fait pas nécessairement appel aux valeurs absolues de la performance, elle se contente des performances relatives des individus qui permettent de les comparer entre eux. Le principe consiste à choisir un groupe de q individus aléatoirement dans la population, de sélectionner d'une manière déterministe le meilleur dans ce groupe, et de recommencer

l'opération jusqu'à l'obtention du nombre d'individus requis. Au cours d'une génération, il y a autant de tournois que d'individus à sélectionner. La pression de sélection est ajustée par le nombre de participants à un tournoi (q). Un q élevé conduit à une forte pression de sélection.

L'avantage de cette technique de sélection est qu'elle est paramétrable par la valeur de q, peu sensible aux erreurs sur F et n'est pas chère à mettre en œuvre et à exécuter. Par contre, sa variance est élevée [23].

Il existe plusieurs variantes de la sélection avec tournoi, dont on cite le tournoi de Boltzmann [108], qui assure que la distribution des valeurs d'adaptation d'une population soit proche d'une distribution de Boltzmann. Pour une compétition entre une solution courante $\overrightarrow{x_i}$ et une solution alternative $\overrightarrow{x_j}$, $\overrightarrow{x_i}$ gagne avec la probabilité $\frac{1}{1+e^{(f(\overrightarrow{x_i})-f(\overrightarrow{x_j}))/T}}$, où T est la température de sélection.

2.3.2 Remplacement

Remplacement déterministe

Le remplacement déterministe est utilisé typiquement dans les Stratégies d'Evolution. Son caractère purement déterministe lui donne un rôle clef dans l'évolution vu qu'il guide la recherche vers les zones des meilleurs individus. Il opère en sélectionnant les μ ($1 < \mu \leq \lambda$) meilleures solutions parmi :

- l'union de μ parents et λ enfants : schéma appelé $(\mu + \lambda)$-ES,
- l'ensemble de λ enfants : schéma appelé (μ,λ)-ES.

Le remplacement $(\mu+\lambda)$ est élitiste, et garantit une amélioration monotone de la performance de la population, mais il s'adapte mal à un éventuel changement d'environnement. Par contre, avec un remplacement (μ,λ), la meilleure performance peut décroître, mais alors l'algorithme est plus flexible avec les changements d'environnement. De plus, la régression des meilleures performances peut aider le processus de recherche à sortir des régions d'attraction des optima locaux, et à continuer l'exploration ailleurs. D'où la recommandation du schéma (μ,λ)-ES par Schwefel [100].

Remplacement générationnel

Une autre technique de remplacement est le remplacement générationnel (total), utilisé principalement dans les AG standards. Avec cette approche, la population des enfants remplace entièrement la population parente. La durée de vie d'un individu est alors d'une seule génération. Dans ses études sur la performance de cette technique, De Jong [22] a proposé de définir le paramètre G (*Generation Gap*) spécifiant le taux de remplacement à chaque génération, afin d'augmenter la durée de vie des meilleurs individus. $G = 1$ correspond au remplacement générationnel total. Une étude empirique faite par Grefenstette en 1986 [43] montre que des taux élevés de G donnent de meilleurs résultats que les taux faibles.

2.3.3 Schémas d'évolution

Un schéma d'évolution est une combinaison d'un type de sélection et d'un type de remplacement. Toute combinaison des procédures présentées plus haut est licite. Toutefois, certaines d'entre elles sont plus souvent utilisées, que ce soit pour des raisons historiques, théoriques ou

expérimentales. Pour cette raison, les noms donnés sont souvent les noms des écoles historiques qui les ont popularisées. Ces écoles travaillaient sur des espaces de recherche bien précis, cependant, les schémas sont totalement indépendants de l'espace de recherche.

- **L'algorithme génétique standard** (*Simple Genetic Algorithm, SGA*)
  Ce schéma est basé sur le remplacement générationnel. A chaque génération, N parents sont sélectionnés stochastiquement par une des méthodes décrites dans la section 2.3.1. Ces N parents donnent naissance ensuite à N enfants par application des opérateurs génétiques (avec des probabilités données). Enfin, ces N enfants remplacent purement et simplement les N parents pour la génération suivante.

- **Les stratégies d'évolution**
  Deux types de schémas sont possibles dans les Stratégies d'Evolution: le schéma "ES+" basé sur le remplacement déterministe $(\mu + \lambda)$, et le schéma "ES," basé sur le remplacement déterministe (μ,λ). A chaque génération, les λ enfants sont générés à partir des parents sélectionnés par tirage uniforme (on peut dire qu'il n'y a pas de sélection au sens darwinien).
  On note que ce schéma est souvent associé à la représentation réelle.

- **Le schéma stationnaire** (*Steady State Genetic Algorithm, SSGA*)
  Ce schéma consiste à produire, à chaque génération, seulement un ou deux enfants à partir d'un ou deux parents sélectionnés selon leurs performances. Ces enfants remplacent alors des individus de la population parente, sélectionnés soit d'une manière déterministe (les pires dans la population), soit par tournoi inversé, ou selon les âges (les plus vieux).
  Ce modèle s'applique aussi pour les Stratégies d'Evolution avec l'approche $((\mu + 1) - ES)$ où $\mu > 1$, introduite par Rechenberg en 1973 [87], où tous les parents peuvent participer à la production de l'enfant. Cependant, le modèle n'est pas très utilisé parce qu'il ne permet pas l'implantation de l'auto-adaptation pour la mutation.

- **La programmation évolutionnaire**
  Le schéma utilisé ressemble à celui des stratégies d'évolution ES+, quoique développé complètement indépendamment. Cependant, avec EP, le nombre d'enfants et celui de parents est identique: $(N + N) - ES$.

2.4 Préservation de la diversité génétique

Les techniques de maintenance de la diversité des individus dans la population sont employées afin de permettre à un AE:

- d'éviter la convergence prématurée vers un optimum local,
- d'identifier plusieurs optima ou quasi-optima quand la fonction objectif est multi-modale.

Parmi les approches connues, nous pouvons mentionner, par exemple, le "surpeuplement" (*crowding*) [22], l'éclaircissement élitiste [84] et la très populaire technique du partage (*sharing*) que nous détaillons ci-dessous parce qu'elle sera évoquée par la suite lors de la présentation de l'état de l'art dans le domaine de l'Optimisation Evolutionnaire Multi-Objectif.

2.4.1 Partage (*sharing*)

Le *partage explicite de la performance* a été introduit par Goldberg et Richardson en 1987 [42]. L'idée consiste à dégrader la valeur de la performance de l'individu en fonction de la densité de la population dans son voisinage. Le but principal est de pénaliser les individus qui sont trop proches les uns des autres. Ainsi, la performance *partagée* (*shared fitness*) F' d'un individu $\overrightarrow{x_i}$ est donnée par :

$$F'(\overrightarrow{x_i}) = \frac{F(\overrightarrow{x_i})}{\sum_{j=1}^{n} sh(d(\overrightarrow{x_i},\overrightarrow{x_j}))}$$

où sh désigne la fonction de partage (*sharing function*) qui dépend de la distance d entre deux solutions. Elle retourne '1' si les deux solutions sont identiques, '0' si la distance entre elles dépasse un certain seuil (σ_{share}) et une valeur intermédiaire pour des degrés de dissimilarité intermédiaires. La fonction de partage la plus utilisée est :

$$sh(d) = \begin{cases} 1 - (\frac{d}{\sigma_{share}})^{\alpha}, & \text{si } d < \sigma_{share} \\ 0, & \text{sinon,} \end{cases} \tag{2.1}$$

où α est une constante utilisée pour contrôler l'allure de la fonction de partage. Le calcul de la distance entre deux individus peut se faire dans l'espace génotypique (e.g. distance de Hamming) ou phénotypique (e.g. distance Euclidienne), selon le problème traité. Cependant, des études expérimentales faites par Deb et Goldberg en 1989 [19] sur les mêmes tests ont montré que le partage dans l'espace phénotypique donne des résultats légèrement meilleurs.

La procédure de partage classique a une complexité élevée ($O(N^2)$), vu qu'elle nécessite une comparaison deux à deux de tous les individus de la population. Afin de réduire cette complexité, Yin et Germay, en 1993, ont proposé de classifier la population en niches avant la procédure de partage [115]. Cette dernière est appliquée ensuite sur chaque niche, ce qui réduit la complexité à $O(Nlog(N))$.

Cette méthode a été ensuite rendue adaptative par Alliot [1], qui propose d'actualiser la distance minimale de nichage à chaque génération, de manière à maintenir un bon niveau de performance des individus appartenant aux différentes classes.

2.5 Prise en compte des contraintes

Dans cette section, nous nous plaçons dans le contexte de la résolution du problème (1.1) avec $M = 1$ et $L > 0$. Dans le cadre des AE, de nombreuses approches ont été développées pour traiter ce type de problèmes. Chacune de ces approches appartient à l'une des catégories suivantes: méthodes basées sur la pénalisation (section 2.5.1), algorithmes de recherche des solutions faisables, techniques préservant la faisabilité des solutions et méthodes hybrides (section 2.5.2). La présentation du premier de ces quatre groupes est un peu plus détaillée ici que les trois autres, car les techniques de pénalisation sont évoquées plus tard lors de la description de la prise en compte des contraintes par les AE Multi-Objectif.

2.5.1 Méthodes basées sur le concept de pénalité

Ce groupe de méthodes se base sur une idée proche de celle du partage (section 2.4.1) dans le mesure où ,là aussi, on se donne une formule selon laquelle la performance de certains individus

sera "corrigée". Dans le contexte de la prise en compte des contraintes, il s'agit de dégrader la performance des individus infaisables en fonction de leur proximité de la région faisable de l'espace de recherche.

Pour chaque élément de l'espace de recherche, sa proximité de la région faisable peut être mesurée à travers le degré de violation de chaque contrainte :

$$\delta_j(x) = \begin{cases} max(g_j(x),0), & j = 1,...,K \\ |h_j(x)|, & j = K+1,...,L \end{cases} \tag{2.2}$$

En utilisant cette mesure d'infaisabilité de l'individu x par rapport à chaque contrainte, nous pouvons introduire la *fonction de pénalité* sous forme générale suivante :

$$\mathcal{F}_{penalty}(x) = c(t) \sum_{j=1}^{L} \mathcal{P}(\delta_j(x)), \tag{2.3}$$

où $\mathcal{P}$ est un polynôme qui est dans la pratique constitué d'un seul terme - une puissance de $\delta_j(x)$ avec un coefficient. Notons que pour x vérifiant toutes les contraintes, $\mathcal{F}_{penalty}(x) = 0$. Le problème d'optimisation initial (1.1) avec $M = 1$ est alors reformulé comme un problème d'optimisation sans contraintes :

$$\begin{cases} min \quad F(x) = f(x) + \mathcal{F}_{penalty}(x), \\ x_i^l \leq x_i \leq x_i^u, & i = 1,...,n. \end{cases}$$

Les techniques qui appartiennent à cette catégorie diffèrent dans leur façon de définir la fonction de pénalité. Nous en citons ici quelques exemples :

- Pénalité mortelle [3] :

  $$\mathcal{F}_{penalty}(x) = \infty$$

  pour tout x infaisable. Ceci revient à rejeter de la population tous les individus infaisables. Notons que cette méthode ne peut fonctionner que s'il y a assez d'individus faisables dans la population initiale.

- Pénalités statiques [53] :

  $$\mathcal{F}_{penalty}(x) = \sum_{j=1}^{L} R_{ij} \delta_j^2(x),$$

  où R_{ij} sont les coefficients de pénalité définis pour chaque contrainte j. L'indice i représente une intervalle d'une famille d'intervalles de violation définie a priori par l'utilisateur.

- Pénalités dynamiques :
  - La pénalité augmentée en cours de l'évolution [58] :

    $$\mathcal{F}_{penalty}(x) = (C \times t)^{\alpha} \sum_{j=1}^{L} \delta_j^{\beta}(x),$$

    C, α et β sont les constantes à fixer par l'user ($C = 0.5$, $\alpha = \beta = 2$ chez les auteurs).

– Les pénalités analytiques [73] :

$$\mathcal{F}_{penalty}(x,\tau) = \frac{1}{2\tau}\sum_{j=1}^{L}\delta_j^2(x),$$

où le paramètre τ désigne la température de refroidissement du Recuit Simulé, qui diminue à chaque génération. τ initial ainsi que le seuil τ_f sont fixés par l'user. L'algorithme s'arrête quand τ atteigne la valeur τ_f.

– Pénalités adaptatives :

L'idée de base de ce groupe de méthodes est d'introduire dans la fonction de pénalité un composant dépendant de l'état du processus de recherche. Cela veut dire que le poids de la pénalité peut être augmenté ou diminué en fonction de la qualité des solutions dans la population courante ([44], [103]).

2.5.2 Et les autres

Recherche des solutions faisables

On compte essentiellement deux méthodes dans cette catégorie: la réparation des individus infaisables et l'échantillonnage de l'espace faisable.

La première se base sur l'idée suggérée en 1995 par Michalewicz et Nazhiyath [75]. Cette approche fait co-évaluer deux populations: une contient les individus satisfaisant toutes les contraintes linéaires (ceux-ci peuvent être préservés si les opérateurs génétiques spécifiques sont appliqués) et l'autre est constituée des individus faisables. Ces derniers sont évalués directement en utilisant la fonction objectif du problème d'optimisation; les premiers, quant à eux, sont "réparés" (rendus faisables par une procédure spéciale) avant d'être évalués.

La deuxième méthode, proposé en 1993 par Schoenauer et Xanthakis [96], s'appelle "Behavioral memory". L'algorithme consiste à échantillonner l'espace faisable en traitant les contraintes du problème une par une dans un ordre particulier. Pour chaque contrainte, il fait évaluer la population jusqu'à ce qu'un certain pourcentage des individus satisfasse la contrainte "courante" tout en continuant à être faisable par rapport aux contraintes précédentes.

Préservation de la faisabilité des solutions

Toutes les méthodes de cette catégorie utilisent des opérateurs de reproduction spécifiques (opérateurs fermés) qui permettent de générer à partir des individus faisables d'autres qui sont faisables, eux aussi. Pour plus de détails, nous renvoyons à Michalewicz et Janikow [74] (première version du système Genocop), Michalewicz et Schoenauer [93], [94] et [95] (recherche sur la frontière de la région faisable), Koziel et Michalewicz [65] (*Homomorphous mapping*).

Méthodes hybrides

La caractéristique commune des méthodes hybrides c'est qu'elles ont la tendance de séparer la fonction objectif des contraintes du problème. On trouve dans la littérature deux approches possibles pour réaliser cette séparation. La première traite les contraintes avec des procédures d'optimisation déterministes, alors que la fonction objectif est toujours optimisée par un AE

([111], [77]). Avec la deuxième approche, c'est l'AE lui-même qui réalise cette séparation soit en utilisant les techniques multi-objectif ([83], [105]), soit en adoptant le modèle de la co-évolution ([81]).

2.6 Algorithmes Evolutionnaires vs méthodes d'optimisation "classiques"

Par méthodes d'optimisation "classiques", nous entendons les algorithmes qui procèdent en mettant à jour une solution potentielle à chaque itération et qui utilisent pour ceci une règle de transition déterministe ([37, 88, 2]). Ces algorithmes commencent par choisir (aléatoirement) une solution initiale. Ensuite, en se basant sur une règle de transition fixe, et compte tenu des informations locales, ils suggèrent la direction dans laquelle la recherche sera poursuivie. A chaque itération, une nouvelle solution est trouvée et sert de point de départ au pas suivant du processus d'optimisation. Toutes les techniques d'optimisation classiques diffèrent surtout par la façon dont est choisie,à chaque itération, la direction que prendra la poursuite de la recherche.

Les approches classiques peuvent être divisées en deux catégories : les méthodes directes et les méthodes de gradient. Les premières, en général, convergent lentement, nécessitant un grand nombre d'évaluations de la fonction objectif. Les méthodes de gradient convergent assez rapidement vers l'optimum le plus proche du point initial. Leur applicabilité est soumise à certaines limitations, dont l'exigence de la régularité suffisante de la fonction objectif. La sensibilité au choix de la solution initiale et la difficulté de gérer les cas de l'espace de recherche discret, constituent les points faibles communs aux deux types d'approches classiques.

Les Algorithmes Evolutionnaires n'ont pas ces inconvénients. De plus, leur champ d'utilisation s'élargit très considérablement grâce à leur capacité de s'appliquer au cas des espaces de recherche non-traditionnels. Ils sont notamment appliqués avec beaucoup de succès à das domaines tels que l'Optimisation Topologique de Forme ([47, 89, 60]), le Traitement d'Image ([13, 38, 68]), la Classification ([49, 64, 70]) etc.

Deuxième partie

Algorithmes Evolutionnaires Multi-Objectif

Chapitre 3

Etat de l'art

Les Shadoks ne connaissaient pas plus que quatre mots.
C'étaient GA, BU, ZO et MEU.
Vous comprenez donc facilement qu'il n'y avait pas du tout suffisamment de mots pour toutes les choses qui existaient.
Les Shadoks, Num.ZO

3.1 Introduction

Comme nous l'avons vu dans le chapitre précédent, lors de l'application d'une méthode évolutionnaire, l'algorithme manipule une population des solutions au lieu d'une seule solution, comme dans le cas de l'utilisation de la plupart des autres méthodes d'optimisation. Cette différence donne aux Algorithmes Evolutionnaires un avantage crucial quand il s'agit de résoudre un problème d'optimisation multi-objectif ou, plus exactement, de trouver de multiples solutions Pareto-optimales.

Dans ce chapitre, nous allons voir quelles modifications peuvent être apportées à un AE mono-objectif afin de l'adapter à la recherche d'un ensemble de solutions de Pareto (voir la section 1.2.1). L'intérêt de ces modifications est de rendre possible l'obtention d'un tel ensemble en un seul run de l'algorithme, et sans devoir définir des paramètres supplémentaires comme les poids relatifs des objectifs, les vecteurs ϵ et des autres vus dans le chapitre 1.4.

La question principale qui se pose concerne l'évaluation de la performance des individus. Autrement dit, sur quel(s) critère(s) vont se baser les étapes darwinistes d'un algorithme évolutionnaire multi-objectif?

En effet, le but poursuivi consiste à trouver non simplement plusieurs solutions de Pareto mais une bonne approximation de l'ensemble de Pareto du problème considéré. (L'utilité de cette tâche fait l'objet de la discussion de la section 1.5.) En terme d'Algorithmes Evolutionnaires, il faut diriger la pression de sélection vers cet ensemble tout en préservant la diversité entre les individus non-dominés..

Notons que dans l'espace des critères, la direction de déplacement vers les solutions Pareto-optimales peut être indiquée par la relation de dominance de Pareto (définie dans la section 1.2.1). C'est-à-dire, si un des deux éléments de l'espace de recherche domine l'autre au sens des objectifs du problème, l'image de cet élément dans l'espace des objectifs est plus proche de la

surface de Pareto que l'image de l'élément dominé (Figure 3.1). Nous pourront remarquer par la suite (section 3.4) que l'évaluation de la "qualité" des individus dans les AEMO actuels les plus performants est effectivement basée sur la notion de dominance de Pareto.

Mais comment s'assurer contre la perte de diversité des solutions non-dominées en cours de l'évolution? Une mesure de performance supplémentaire rentre, alors, en jeu de façon à préserver la diversité génétique entre les individus Pareto-équivalents. Comme nous le verrons dans ce chapitre, de nombreuses techniques ont été développées pour maintenir un ensemble diversifié d'individus non-dominés dans la population des AEMO.

En effet, l'évaluation par dominance et la préservation de la diversité produisent des effets orthogonaux: la dominance avance la recherche en dirigeant la pression de sélection vers la surface de Pareto, tandis que la technique de préservation de la diversité préserve l'équilibre de la distribution des solutions le long de cette surface en provoquant une pression de sélection tangente (voir la figure 3.1).

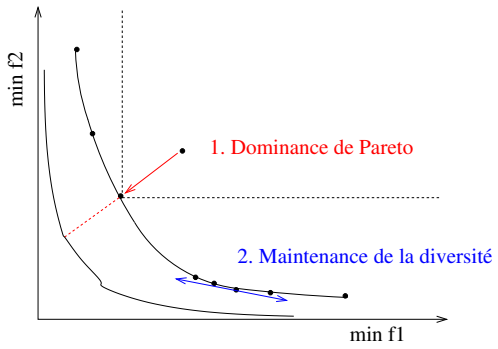


FIG. 3.1 – *Principes de l'évaluation de performance des individus dans les AEMO*

Plusieurs travaux de classification et de comparaison des AEMO ont été publiés jusqu'à aujourd'hui [34, 121, 109]. Ces études diffèrent par le principe sur lequel le groupement des algorithmes existants est basé. Par exemple, Fonseca et Fleming [34] distinguent les algorithmes selon le principe de sélection et montrent comment l'idée de "sélection multi-objectif" a évolué vers celle basée sur la dominance de Pareto. Plus récente, l'étude de comparaison par Zitzler et al. [121] souligne l'importance de l'élitisme pour améliorer la performance des AEMO.

En effet, le schéma *non-pareto* $->$ *pareto-based* $->$ *pareto-elitist* décrit l'évolution du domaine de l'Optimisation Multi-objectif Evolutionnaire. Les trois sections suivantes de ce chapitre présentent chacune un groupe des algorithmes correspondant à ce schéma, et la section 3.5 décrit quelques adaptations des AEMO à l'optimisation multi-objectif sous contraintes. En effet, la plupart des problèmes réels mettent en jeu, outre plusieurs objectifs, des contraintes souvent difficiles à satisfaire.

3.2 Premiers travaux

3.2.1 VEGA

En 1985, Schaffer a proposé le premier AEMO pour trouver les solutions non-dominées d'un problème multi-objectif [92]. Il l'a appelé VEGA - *Vector Evaluated Genetic Algorithm* - car

son algorithme utilise comme évaluation des les individus directement le vecteur des valeurs des différents objectifs au lieu d'une valeur scalaire.

Le schéma de l'évaluation est très simple : à chaque génération la population est aléatoirement divisée en autant de sous-populations (de tailles égales) qu'il y a d'objectifs, ensuite chaque sous-population est évaluée selon l'une des fonctions objectifs. L'opérateur de sélection est ensuite restreint à chaque sous-population. Cela est particulièrement important dans le cas, où les fonctions objectif sont d'ordres de magnitude très différents.

Une itération de VEGA

1. Soit le compteur des objectifs $m = 1$ et soit $q = N/M$ (M étant le nombre d'objectifs).
2. Pour toute solution de $j = 1 + (m-1) * q$ à $j = m * q$, la valeur de performance soit
$$F(x^{(j)}) = f_m(x^{(j)}).$$
3. Effectuer la sélection par "roulette" (voir section 2.3.1) pour toutes les q solutions et créer la population des parents P_m.
4. Si $m = M$, passer à l'étape 5. Sinon, incrémenter m de 1 et retourner à l'étape 2.
5. Rassembler toutes les sous-populations des parents: $P = \cup_{m=1,\ldots,M} P_m$. Appliquer les opérateurs de croisement et de mutation à l'ensemble P pour créer une nouvelle population.

Il a été observé que cet algorithme a la tendance de trouver les solutions de bonne qualité pour des objectifs individuels. Afin de rendre possible la recherche de bons compromis intermédiaires, toutes les paires d'individus de la population peuvent être croisés. Schaffer croyait que le croisement entre deux solutions bonnes chacune pour un des objectifs donnerait un bon compromis entre ces deux objectifs.

Avantages de VEGA

L'algorithme de Schaffer est basé sur une idée très simple et il est facile à mettre en œuvre. Seuls quelques changements minimaux doivent être apportés à un AE mono-objectif. De plus, cela n'implique aucun coût de calcul supplémentaire.

VEGA est particulièrement adapté aux problèmes, où on cherche surtout des bonnes solutions individuelles pour chacun des différents objectifs. Une telle application de VEGA a été décrite dans [8].

Désavantages de VEGA

La sélection de VEGA favorise les solutions qui sont bonnes pour un des objectifs. Et même si on peut espérer trouver de bons compromis à l'aide du croisement entres les "champions individuels", il est peu probable que tels compromis puissent survivre assez longtemps au cours de l'évolution.

De plus, la façon d'attribuer la performance aux individus employée dans VEGA correspond à l'évaluation des individus selon une combinaison linéaire des critères (élaborée implicitement par l'algorithme) avec des poids dépendant de la distribution de la population en chaque génération. Ceci explique la tendance de la population de VEGA de se concentrer dans les régions de la surface de Pareto, où un des critères est nettement meilleur que les autres, en cas de surface concave (Figure 3.2).

3.2.2 VOES

En 1990, Frank Kursawe a proposé la méthode VOES - *Vector-Optimized Evolution Strategy* [67]. Dans cette approche, une Stratégie d'Evolution auto-adaptative de base est modifiée afin d'être utilisable pour le résolution des problèmes multi-objectif.

Un individu est composé de deux solutions dont une est marquée comme "dominante" et l'autre comme "récessive". Ainsi, deux évaluations correspondent à chacun des individus: f_d et f_r. L'évaluation finale et la sélection sont effectuées de façon suivante.

La sélection est faite en M étapes (où M est le nombre des objectifs). À chaque étape, le vecteur des probabilités défini par l'utilisateur est utilisé pour choisir un des objectifs. Ce vecteur peut être fixé ou varier au cours de générations. Si la $m^{\text{ème}}$ fonction objectif est choisie, la performance d'un individu est calculée comme la moyenne pondérée des valeurs $(f_d)_m^{(i)}$ et $(f_r)_m^{(i)}$ en proportion 2/1:

$$F^{(i)} = \frac{2}{3}(f_d)_m^{(i)} + \frac{1}{3}(f_r)_m^{(i)}.$$

Comme dans une Stratégie d'Evolution standard, λ solutions mutées sont créées à partir de μ parents. De plus, un opérateur *swap* qui échange des variables de décision des parts "dominante" et "récessive" d'un individu est appliqué avec la probabilité 1/3 pour chacune des variables.

À chaque étape de la sélection, la population est ordonnée selon un des objectifs et $(M-1)/M$ meilleurs sont choisis en tant que parents. Cette procédure est répétée M fois et utilise à chaque fois la population qui a survécu lors de l'ordonnancement précèdent. Le rapport entre λ et μ est alors le suivant:

$$\mu = (\frac{M-1}{M})^M \lambda$$

Toutes les μ nouvelles solutions sont copiées dans un ensemble externe, dans lequel toutes les solutions non-dominées trouvées depuis le début de l'évolution sont stockées. Ensuite, l'ensemble ainsi obtenu est mis à jour par élimination de toutes les solutions dominées. Si la taille de cette archive est trop grande, un mécanisme de nichage est utilisé pour éliminer des solutions trop proches.

Notons que même si les solutions non-dominées sont préservées, elles ne sont pas utilisées dans le processus génétique ultérieur; cette approche n'appartient donc pas à la catégorie des approches élitistes.

Discussion de VOES

VOES préserve toutes les solutions non-dominées et utilise le mécanisme de nichage, c'est-à-dire il possède les deux caractéristiques essentielles d'un bon MOEA. Malheureusement, Kursawe n'a publié les résultats de simulation que pour un unique problème et il n'y a pas eu de travaux reprenant et développant cet algorithme. Il est donc difficile à prédire sa performance sur des problèmes plus complexes.

Des variantes de cet algorithme sont possibles et la nécessité de la représentation par double chromosomes est discutable. Il est probablement aussi que c'est à cause de la complexité de sa mise en oeuvre que cette méthode n'est pratiquement pas évoquée par les chercheurs travaillant dans le domaine de l'Optimisation Multi-objectif Evolutionnaire.

3.2.3 HLGA

En 1993, Hajela et Lin ont proposé la méthode HLGA [46], dont la sélection est basée sur la performance des individus calculée comme la somme pondérée des objectifs. A la différence de la méthode de la somme pondérée traditionnelle, ici, un vecteur des poids différent est associé à chaque individu.

Le point très important de cette approche est la préservation de la diversité entre les vecteurs des poids. Cela peut être fait de deux façons suivantes. Soit le vecteur des poids fait partie de l'individu et la technique de nichage est appliquée, soit des sous-populations habilement choisies sont évaluées en utilisant des vecteurs des poids différents prédéfinis.

Approche basée sur le partage

Dans le cadre de cette approche, deux options sont possibles. La première c'est qu'un vecteur de nombres réels de taille M fait partie de chaque individu. Afin de s'assurer que la somme de ses composantes soit égale à 1, chacune d'elles est normalisée par la somme de toutes les composantes:

$$w_i = \frac{w_i}{\sum_{j=1}^{M} w_j}.$$

La deuxième option est plus appropriée au cas où seulement peu de solutions de Pareto sont désirées. L'ensemble de vecteurs des poids est choisi a priori et une variable entière x_w qui désigne le numéro identifiant chaque vecteur s'ajoute à chaque individu. Le nombre de vecteurs des poids K correspond au nombre de solutions qu'on veux trouver. Dans cette section nous allons décrire plus en détails ce dernier cas.

Une solution $x^{(i)}$ est évaluée selon l'équation

$$F(x^{(i)}) = \sum_{j=1}^{M} w_j^{x_w^{(i)}} \frac{f_j(x^{(i)}) - f_j^{min}}{f_j^{max} - f_j^{min}}. \tag{3.1}$$

Ici, f_j^{min} et f_j^{max} sont les valeurs extrémales de l'objectif j trouvées parmi les solutions de la population courante.

Afin de préserver la diversité des solutions, la technique de partage (voir section 2.4.1) est utilisée. Le partage ici concerne seulement la partie de l'individu correspondante au vecteur des poids, c'est-à-dire x_w. Cela signifie que seulement les valeurs $x_w^{(i)}$ et $x_w^{(j)}$ sont prises en compte lors du calcul de la distance d_{ij} entre les solutions i et j:

$$d_{ij} = |x_w^{(i)} - x_w^{(j)}|.$$

La fonction de partage $Sh(d_{ij})$ est calculée avec le paramètre σ_{share}. Rappelons-nous que la variable x_w est entière et prend des valeurs avec un pas égale à 1. Alors, $\sigma_{share} = 1$ signifierait que seulement les solutions avec la même valeur de x_w sont partagées. $\sigma_{share} = 2$ permettrait la contribution de deux vecteurs des poids voisins mais cela risque de réduire la diversité nécessaire pour trouver toutes les K solutions Pareto-optimales différentes, chacune correspondant à une combinaison des poids.

Le "niche count" nc_i de chaque solution i est calculé comme somme des valeurs de la fonction de partage de toutes les solutions dans la population. Enfin, la performance calculée selon

l'équation 3.1 est divisée par nc_i ce qui donne la valeur de performance corrigée.

Calcul de la performance dans HLGA basé sur le partage

1. Pour toute fonction objectif j, trouver f_j^{max} et f_j^{min}
2. Pour toute solution i = 1,2,...,N calculer les distances d_{ik}, k = 1,2,...,N. Ensuite calculer la fonction de partage comme suite:

$$Sh(d_{ik}) = \begin{cases} 1 - \frac{d_{ik}}{\sigma_{share}}, & \text{si } d_{ik} \leq \sigma_{share} \\ 0, & \text{sinon.} \end{cases}$$

   Enfin, calculer le "niche count" $nc_i = \sum_{k=1}^{N} Sh(d_{ik})$.
3. Pour toute solution i = 1,2,...N, calculer la valeur de performance F_i selon l'équation 3.1 et ensuite la corriger par le partage: $F_i' = F_i / nc_i$.

Quand les valeurs de F' sont calculées pour tous les individus de la population, la sélection proportionnelle est effectuée comme dans un AG ordinaire. Les opérateurs de croisement et de mutation sont appliqués aux individus, la part x_w inclue.

Cette version de HLGA ne demande pas de changements considérables par rapport à un AE mono-objectif ordinaire. De plus, sa complexité est plus petite que celle des autres MOEA.

Les approches multi-objectif basées sur l'assemblage de critères en une somme pondérée partagent les mêmes défauts : ils sont incapables de trouver les solutions situées sur les régions concaves de la surface de Pareto. De plus, même un ensemble des vecteurs de poids uniformément distribués n'assurent pas forcément l'uniformité de la distribution des solutions correspondantes. Cela rend le choix approprié de ces vecteurs difficile.

Approche basée sur les sous-populations

Cette méthode est similaire à VEGA. D'abord, un ensemble de K différents vecteurs des poids $w^{(k)}$, k = 1,...,K, est choisi. Ensuite, chacun de ces vecteurs est utilisé pour calculer la performance pondérée normalisée pour tous les N membres de la population. Les N/M meilleurs d'entre eux sont groupés en une sous-population (qui est associée au vecteur $w^{(k)}$) pour le processus génétique postérieur. C'est-à-dire, la sélection, le croisement et la mutation sont restreints aux sous-populations. La taille de sous-population N/M pour chaque vecteur $w^{(k)}$ est maintenue, ce qui assure qu'à la fin de ces opérations la taille de la population entière est toujours N. A la génération suivante encore, chaque vecteur $w^{(k)}$ est utilisé pour trouver la sous-population correspondante des meilleurs etc.

Il y a donc K sous-populations, chacune correspondant à un vecteur de poids. Notons qu'un membre de la population peut être associé à plus d'un vecteur. En particulier, les solutions non extrémales seront incluses dans plus d'une sous-population.

Dans cet algorithme, de multiples solutions sont trouvées grâce à la préservation explicite de K différents vecteurs des poids. Il n'est alors pas nécessaire d'introduire un opérateur de nichage supplémentaire.

Calcul de la performance dans HLGA basé sur les sous-populations

1. Soit le compteur des solutions $k = 1$.

2. Calculer la performance F_j de chaque solution j utilisant l'équation 3.1 avec le vecteur $w^{(k)}$. Choisir N/K meilleures solutions à partir des valeurs F_j. Copier ces solutions dans sous-population P_k.
3. Effectuer la sélection, appliquer le croisement et la mutation aux membres de P_k pour créer une nouvelle population de taille N/K.
4. Si $k < K$, incrémenter k de 1 et retourner à l'étape 2.
5. Sinon, combiner toutes les sous-populations pour créer une population $P = \cup_{k=1}^{K} P_k$. Si $|P| < N$, ajouter des solutions aléatoires pour compléter la population.

Notons qu'à l'étape 2, N solutions sont évaluées et cette étape est répété K fois, ce qui coûte KN évaluations. Cependant, cette approche est meilleure du point de vue de la complexité par rapport à l'approche basée sur le partage grâce à l'absence du mécanisme de nichage avec ses calculs des distances.

3.2.4 Et les autres

En faisant le bilan de ces premières tentatives d'adapter les AE à la recherche de la surface de Pareto, nous pouvons remarquer qu'en général, les valeurs de performance différentes sont attribuées aux individus non-dominés de la population. Ceci rend probable l'apparition du phénomène appelé "speciation"(Schaffer). Il s'agit de la tendance de la population de se concentrer dans des régions particulières de la surface des compromis.

Typiquement, ceci peut se traduire par une difficulté (voir impossibilité) à trouver les solutions appartenant aux régions concaves de la surface de Pareto, comme nous l'avons noté dans les cas de VEGA et de HLGA (Figure 3.2).

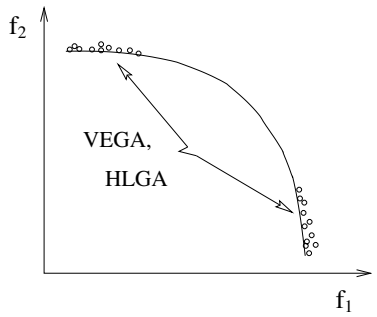


FIG. 3.2 – *Défauts de l'attribution de la performance dans de premiers AEMO*

En ce qui concerne VOES, bien que sa sélection soit quelque part similaire de celle de VEGA, Fonseca et Fleming [34] supposent que cette méthode a tendance à éliminer des individus sur les extrémités de la surface des compromis (si un objectif selon lequel ces individus sont "mauvais" est choisi pour décider de leur sort).

Les algorithmes VEGA et HLGA sont habituellement cités dans les travaux consacrés à la comparaison de la performance de différents AEMO (voir, par exemple, [121]). Comparés aux approches basées sur la dominance (voir sections 3.3 et 3.4), ces algorithmes (surtout VEGA) ne peuvent pas être considérés comme inférieurs aux méthodes décrites dans la section suivante, tant qu'il s'agit des problèmes pour lesquels le front de Pareto est convexe. Compte tenu de

son faible coût de calcul et de la simplicité de sa mise en œuvre, VEGA peut être parfois plus approprié à l'usage que certains algorithmes basés sur le principe de dominance.

3.3 Algorithmes basés sur la dominance de Pareto

La convexité de la surface de Pareto dépend de la façon dont les objectifs sont mis à l'échelle. Une transformation non-linéaire de l'échelle peut convertir une surface concave en une surface convexe et vice versa. Par exemple, la figure 3.3 montre trois possibilités de mise à l'échelle des critères qui correspondent à différentes formulations du même problème d'optimisation.

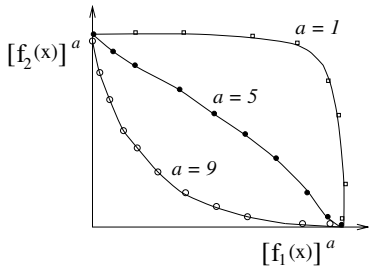


FIG. 3.3 – *La sélection par le rang basé sur la dominance est aveugle à la convexité de la surface de Pareto*

Vu que l'ordre est préservé par les transformations monotones, la sélection par le rang (ou *ranking*) basé sur la dominance est aveugle à la convexité de la surface de Pareto. Ceci ne signifie pas que l'attribution de la performance basée sur le ranking par dominance prévient la spéciation; ce phénomène peut toujours avoir lieu si certaines régions de la surface des compromis sont plus faciles à trouver que les autres. Mais le ranking élimine effectivement la sensibilité à la non-convexité de la surface de Pareto.

3.3.1 MOGA

Fonseca et Fleming [33] ont été les premiers à proposer l'algorithme qui utilise la notion de dominance directement pour évaluer la performance des individus.

Dans leur algorithme, pour chaque solution i, le nombre n_i de solutions la dominant est calculé, et le rang $r_i = (1 + n_i)$ lui est associé (Figure 3.5 à gauche).

Le rang de chaque solution non-dominée est donc égal à 1 et le rang maximal ne peut pas être plus grand que la taille de la population N. La performance F_i attribuée à chaque individu est basée sur son rang. La fonction qui transforme le rang en une valeur de performance est telle que tous les individus du même rang ont la même performance et cette performance est à maximiser.

Bien que l'idée de convertir les valeurs de la "performance vectorielle" en une métrique utilisant le *rang de dominance* représente un avancement considérable dans l'optimisation multi-objectif, la préservation de la diversité des solutions ne peut pas être assurée à l'aide de cette simple procédure.

Fonseca et Fleming ont donc introduit le nichage entre les solutions du même rang. D'abord, les distances normalisées entre chaque paire de solutions du même rang sont calculées dans

l'espace de critères:

$$d_{ij} = \sqrt{\sum_{k=1}^{M} \left(\frac{f_k^{(i)} - f_k^{(j)}}{f_k^{max} - f_k^{min}} \right)^2},$$

où f_k^{min} et f_k^{max} sont les valeurs minimale et maximale de la fonction objectif k atteintes sur la population courante. La fonction de partage $Sh(d_{ij})$ est calculée selon l'équation 2.1 avec $\alpha = 1$ et "niche count" donné par la formule suivante:

$$nc_i = \sum_{j=1}^{\mu(r_i)} Sh(d_{ij}),$$

où $\mu(r_i)$ est le nombre de solutions de rang r_i. Enfin, la valeur de performance F_i attribuée à la base du rang est corrigée comme nous l'avons vu dans l'algorithme HLGA:

$$F_i' = F_i / nc_i.$$

Les valeurs de la performance "partagée" F' ne sont plus les mêmes pour les individus du même rang, les solutions situées dans des régions plus éparses ayant meilleure performance. Cela se traduit par la pression de sélection plus haute pour telles solutions.

Mise à jour de σ_{share}

Dans le contexte de l'optimisation multi-objectif, le paramètre de partage σ_{share} doit être choisi de façon à assurer la distribution uniforme de la population de taille N sur la surface de Pareto. Fonseca et Fleming [33] ont suggéré une procédure simple pour calculer σ_{share} à chaque génération.

Tout d'abord, les valeurs extrémales f_j^{min} et f_j^{max} de chaque objectif j sont calculées à partir des solutions de la population courante. Les images de toutes ces solutions dans l'espace de critères se trouvent dans l'hypervolume $V = \prod_{i=1}^{M} (f_j^{max} - f_j^{min})$.

Par définition, une surface de Pareto ne peut pas croiser une droite parallèle à une des axes en plus d'un point. Supposons que l'on a tracé une telle surface passant par toutes les solutions non-dominées de la population courante. L'aire de cette surface est inférieure à la somme de ses projections le long de chacune des axes.

Enfin, vu que l'aire maximale de chaque projection n'est pas supérieure à l'aire de la face correspondante de l'hyperparallélogramme défini par les points (f_1^{min},...,f_M^{min}) et (f_1^{max},...,f_M^{max}) (voir la figure 3.4), l'aire de notre surface est inférieure à la somme de ces faces, c'est-à-dire à la semi-aire de l'hyperparallélogramme:

$$A = \sum_{i=1}^{M} \prod_{j=1, j\neq i}^{M} (f_j^{max} - f_j^{min}).$$

Supposons que les objectifs sont normalisés, ce qui permet de choisir le même σ_{share} pour tous les objectifs. Alors, le nombre de points destinés à représenter la surface A sans interférer entre eux, peut être calculé comme le nombre d'hypercubes de volume σ_{share}^M D'autre part, nous avons à notre disposition N individus de la population et notre but initial consiste à choisir σ_{share} de façon qu'ils soient uniformément distribués le long la surface de Pareto.

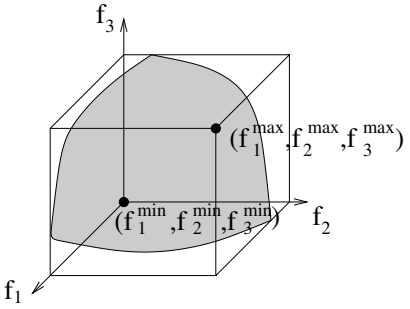


FIG. 3.4 – *Surface de Pareto, $M = 3$*

Autrement dit, étant donné un nombre de points N, il est possible d'estimer σ_{share} en résolvant l'équation suivante:

$$N = \frac{\prod_{j=1}^{M}(f_j^{max} - f_j^{min} + \sigma_{share}) - \prod_{j=1}^{M}(f_j^{max} - f_j^{min})}{\sigma_{share}^{M}}$$

pour $\sigma_{share} > 0$, ce qui revient à trouver les racines positives d'un polynôme du degré $M - 1$.

CALCUL DE LA PERFORMANCE DANS MOGA

1. Choisir une valeur de σ_{share}.
   Initialiser $\mu(r_i) = 0$ pour tous les rangs possibles j = 1,2,...,N. Soit $i = 1$ le compteur des solutions.
2. Pour la solution i, calculer n_i, le nombre de solutions la dominant, puis son rang $r_i = 1+n_i$. Incrémenter le compteur de solutions de rang r_i: $\mu(r_i) \leftarrow \mu(r_i) + 1$.
3. Si $i < N$, $i \leftarrow i + 1$ et retourner à l'étape 2. Sinon, passer à l'étape 4.
4. Identifier le rang maximal r^* c'est-à-dire r_i maximal tel que $\mu(r_i) > 0$. Pour toutes les solutions i = 1,2,...,N, calculer leur valeurs de performance comme suite:

$$F_i = N - \sum_{k=1}^{r_i - 1} \mu(k) - 0.5(\mu(r_i) - 1).$$

   Soit $r = 1$ le compteur des rangs.
5. Pour toute solution i de rang r, calculer le "niche count" nc_i avec les autres solution du même rang et, ensuite, la performance "partagée" F_i'. Afin de préserver la même performance moyenne, mettre F_i' à l'échelle de façon suivante:

$$F_i' \leftarrow \frac{F_i \mu(r)}{\sum_{k=1}^{\mu(r)} F_k'} F_i'.$$

6. Si $r < r^*$, $r \leftarrow (r + 1)$ et retourner à l'étape 5. Sinon, la procédure est terminée.

Complexité de MOGA

Le étapes 2 et 3 sont répétés N fois et à chaque fois $(N-1)$ solutions sont comparées au sens de dominance. Cette procédure exige $O(MN^2)$ comparaisons. Comme il peut y avoir au maximum N rangs, les étapes 4, 5 et l'opération de mise à l'échelle de la performance "partagée" ne coûteront pas plus que $O(N^2)$. Alors, la complexité totale de MOGA est de l'ordre $O(MN^2)$.

Discussion de MOGA

La façon de calculer la performance est simple dans MOGA. Mais, d'autre part, bien que le concept de dominance soit utilisé pour calculer le rang d'une solution, différentes solutions appartenant au même front non-dominé (à part le premier) n'ont pas nécessairement la même performance. Ceci peut impliquer quelques biais indésirables vers certaines solutions dans l'espace de recherche. En particulier, l'algorithme peut être sensible à l'allure de la frontière de Pareto ou à la densité des solutions sur cette frontière.

La procédure du partage n'assure pas que les solutions d'un rang plus "pauvre" auront une moins bonne valeur de performance "partagée" F' que les solutions d'un rang meilleur. Cela peut se produire en particulier s'il y a un nuage très dense de solutions d'un meilleur rang. Dans les cas pareils, il y a le risque d'une pression de sélection insuffisante pour ces solutions ce que mène à la convergence ralentie ou incapacité à trouver les solutions bien reparties.

3.3.2 NSGA

L'idée de Goldberg d'utiliser le concept de *ranking par dominance* dans les AG [41] a été plus directement mise en œuvre par Srinvas et Deb en 1994 dans leur méthode NSGA - *Nondominated Sorting Genetic Algorithm* [104]. Encore une fois, le double objectif *convergence-diversité* est atteint, d'une part, par l'utilisation d'un schéma du calcul de la performance qui préfère les solutions non-dominées et, d'autre part, par l'application de la technique du partage entre les solutions du même front non-dominé.

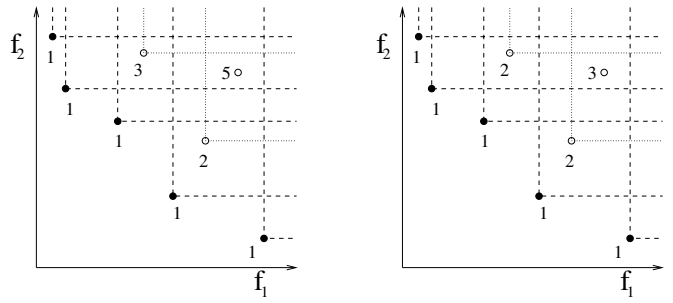
Première étape de NSGA consiste à trier la population P selon le principe de dominance. Cette procédure divise la population en un nombre de classes distinctes P_j de façon suivante: tous le individus non-dominés de P appartiennent à l'ensemble P_1; ensuite, tous les éléments non-dominés de $P \backslash P_1$ sont placés dans l'ensemble P_2 etc. La procédure est terminée quand toute la population est triée:

$$P = \cup_{j=1}^{r} P_j.$$

De divers algorithmes pour identifier le sous-ensemble non-dominé d'un ensemble donné ont été présentés dans la section 1.2.2.

Notons que entre deux solutions de la même classe, aucune ne peut être considérée meilleure de l'autre compte tenu de tous les objectifs du problème. Le nombre total de classes, noté r dans l'équation ci-dessus, dépend de la population P. La figure 3.5(b) présente un exemple de telle classification (les numéros associés à chaque point correspondent au numéro de la classe à laquelle il appartient) comparée avec le ranking de MOGA.

Quand toute la population est triée, l'ensemble P_1 contient toutes les solutions non-dominées de P. Ces solutions sont les meilleures au sens de leur proximité de la surface de Pareto du problème. La valeur de performance la plus grande sera donc attribuée à ces individus; la performance va progressivement diminuer en passant d'un front à l'autre. Toute solution i de l'ensemble P_1 reçoit la valeur de performance $F_i = N$.



(a) MOGA (b) NSGA

FIG. 3.5 – *Résultats de ranking: MOGA vs NSGA*

Afin de préserver la diversité entre les solutions de même qualité (c'est-à-dire, celles qui appartiennent au même ensemble non-dominé), la valeur de performance F_i sera dégradée en fonction de nombre de solutions "voisines" de la solution i.

Dans la version originale de NSGA, la procédure du partage est basée sur les distances calculées dans l'espace de décision. C'est-à-dire, la distance entre deux solutions i et j du même front P_l est donnée par l'équation

$$d_{ij} = \sqrt{\sum_{k=1}^{|P_l|} \left(\frac{x_k^{(i)} - x_k^{(j)}}{x_k^{max} - x_k^{min}} \right)^2}.$$

Ces distances sont utilisées pour calculer la fonction du partage $Sh(d_{ij})$ comme dans l'équation 2.1 avec $\alpha = 2$. Le "niche count" est calculé comme d'habitude et, enfin, la performance initiale F_i est dégradée de façon à augmenter la pression de sélection dans les régions plus éparses: $F_i' = F_i / nc_i$.

CALCUL DE LA PERFORMANCE DANS NSGA

1. Choisir le paramètre σ_{share} et un nombre petit positif ϵ et initialiser $F_{min} = N + \epsilon$. Soit le compteur de fronts $j = 1$.
2. Trier la population P selon non-dominance:

$$(P_1, P_2, ..., P_r) = Sort(P, \preceq)$$

3. Pour tout $q \in P_j$
   - Lui associer la performance $F_j^{(q)} = F_{min} - \epsilon$.
   - Calculer nc_q comme la somme des fonctions de partage des individus de P_j seulement.
   - Calculer la performance "partagée" $F_j'^{(q)} = F_j^{(q)} / nc_q$.
4. $F_{min} = min(F_j'^{(q)} : q \in P_j)$ et $j \leftarrow (j + 1)$.
5. Si $j \leq r$, retourner à l'étape 3. Sinon, le processus est terminé.

Cette procédure peut être insérée dans un AG mono-objectif. Ainsi, les auteurs de NSGA ont initialement employé la sélection par roulette (voir la section 2.3.1).

Complexité de NSGA

Le coût de calcul le la procédure ci-dessus est principalement déterminé par l'opération de ranking de la population et par la mise en oeuvre du partage. Le ranking demande au plus $O(MN^2)$ opérations. Le partage contient les comparaisons de chaque solution avec toutes les autres appartenant au même front ce qui se traduit par $\sum_{j=1}^{r} |P_j|^2$ calculs de distances. Chaque calcul de distance comprend n différences entre les paramètres de décision. Dans le pire cas (quand toutes les solutions appartiennent à P_1), ce calcul est de complexité $O(nN^2)$. Alors, le coût total de cet algorithme est $max(O(nN^2), \quad O(MN^2))$.

Discussion de NSGA

L'avantage principal de NSGA c'est que les valeurs de performance sont attribuées aux individus à la base de l'ensemble non-dominé auquel ils appartiennent. Ceci permet à l'algorithme d'assurer l'avancement de la population vers la surface Pareto-optimale le long de tout le front. De plus, le partage dans l'espace de décision assure la diversité génotypique des solutions. La technique de partage peut être également appliquée dans l'espace des objectifs.

Notons cpendant que l'utilisation du partage demande de fixer le paramètre σ_{share} qui a une influence très importante à la performance de l'algorithme [104].

3.3.3 NPGA

En même 1994, Horn et al. [55] ont proposé leur méthode, elle aussi basé sur le concept de dominance. NPGA - *Niched-Pareto Genetic Algorithm* - diffère des algorithmes décrits ci-dessus à la stade de sélection. Dans NPGA, la sélection par tournoi est utilisée au lieu de la sélection proportionnelle. Ce choix est basé sur l'étude théorique [40] consacrée aux opérateurs de sélection utilisés dans l'optimisation évolutionnaire mono-objectif.

Un nichage mis à jour dynamiquement [80] est utilisé dans NPGA. Deux solutions i et j sont choisies de la population P de façon aléatoire. Ensuite, toujours aléatoirement, une sous-population T de taille t_{dom} ($\ll N$) est choisie. Chacune des solutions i et j est comparée avec toutes les solutions de l'ensemble T pour la dominance. Les scénarios suivants sont considérés:

- Si une des solution i ou j domine toutes les solutions de T tant que l'autre est dominée par au moins une solution, alors celle première est choisie.
- Si les deux solutions i et j sont soit dominées par au moins une solution de T, soit dominent tout l'ensemble T, alors elles sont mises toutes les deux dans la population des enfants Q (partiellement remplie) et leur "niche count" dans Q est calculé. La solution, qui a le "niche count" plus petit gagne le tournoi.

Au début, quand la population des enfants est vide, il peut se produire une légère déviation due à la procédure décrite ci-dessus. Quand le second scénario est suivi pour les deux premières solutions, une des deux solutions est choisie aléatoirement en tant que parent potentiel. Ensuite deux parents sont croisés et mutés pour créer deux enfants. Dès la troisième reprise de ce scénario, la procédure originale est adoptée.

Notons qu'à la différence des méthodes présentés jusqu'à maintenant, NPGA ne suggère pas de calcul de performance scalaire pour chaque individu. La pression de sélection vers les solutions Pareto-optimales ainsi que la préservation de la diversité des solutions sont assurées par le principe du tournoi.

Une itération de NPGA

1. Réordonner aléatoirement la population P. Soient $i = 1$ et $Q = \emptyset$.
2. Effectuer la sélection par tournoi comme décrit ci-dessus,
   trouver le premier parent $p_1 = NPGA - tournament(i,i+1,Q)$.
3. $i \leftarrow (i+2)$ et trouver le second parent $p_2 = NPGA - tournament(i,i+1,Q)$.
4. Croiser p_1 et p_2 en créant les enfants c_1 et c_2 et appliquer l'opérateur de mutation à chacun d'eux.
5. Mettre à jour la population des enfants $Q \leftarrow Q \cup \{c_1,c_2\}$.
6. $i \leftarrow (i+1)$. Si $i < N$, retourner à l'étape 2. Sinon, si $|Q| = N/2$, réordonner aléatoirement P, mettre i égal à 1 et retourner à l'étape 2. Sinon, le processus est terminé.

Complexité de NPGA

Pour chaque tournoi, le premier scénario exige la comparaison de deux solutions avec l'ensemble de t_{dom} solutions pour la dominance ce qui se traduit par $2Mt_{dom}$ comparaisons et donc par $2Mt_{dom}N$ comparaisons pour toute la population. Dans le second scénario, le calcul de "niche count" demande $2|Q|$ calculs des distances pour chaque tournoi. Comme la taille de la population des enfants varie de 2 à $(N-2)$, au total $O(N^2)$ calculs de distances sont nécessaires. Alors, le coût total de l'algorithme est $O(N^2)$.

Avantages de NPGA

Si le paramètre t_{dom} est beaucoup plus petit que N, alors le coût de calcul ne dépend presque pas du nombre d'objectifs, ce qui rend NPGA particulièrement intéressant pour la résolution de problèmes mettant en jeu un grand nombre d'objectifs.

Désavantages de NPGA

NPGA demande la définition de deux paramètres importants σ_{share} et t_{dom}. Or, l'influence de σ_{share} est plus grande encore dans NPGA que dans NSGA. En effet, nc_i est calculé comme le nombre de membres de la population qui sont situés à la distance inférieure à σ_{share} de la solution i. La distance même (qui peut être proche de 0 ou de σ_{share}), n'est pas du tout prise en compte.

La valeur de t_{dom} est également d'une grande importance. Si elle est trop petite, la sélection sera trop bruitée ce qui est dangereux pour la convergence. Si, par contre, elle est trop grande, le coût de calcul va considérablement augmenter.

3.4 Elitisme au sens de la dominance

Les méthodes présentées jusqu'à maintenant peuvent être utilisées pour la recherche des solutions non-dominées multiples bien distribuées et plus ou moins proches de l'ensemble de solutions Pareto-optimales, et ceci avec un degré de succès indépendant de l'allure de la surface des compromis de chaque problème.

Cependant, par analogie avec l'optimisation evolutionnaire mono-objectif, l'introduction de l'élitisme était inévitable aussi dans le domaine des AEMO. Mais avant de parler des "meilleurs",

il fallait établir un système de préférances. En effet, les AEMO élitistes sont apparus et ont proliféré après que certains principes de l'attribution de la performance aux individus (tels que la dominance et la préservation de la diversité à l'intérieur de l'ensemble non-dominé) aient été largement reconnus comme particulièrement appropriées à la recherche de la surface de Pareto.

Les algorithmes décrits dans cette section diffèrent surtout par la façon dont les "meilleurs" individus se distinguent de la population restante, et interagissent avec elle.

3.4.1 SPEA

En 1998, Zitzler et Thiele ont proposé une méthode évolutionnaire élitiste pour l'optimisation multi-critère qu'ils ont appelée *Strength Pareto Evoutionary Algorithm* [120]. L'élitisme est introduit par le maintien explicite d'une population externe P' appelée aussi *archive*. Cette population est destinée à contenir un nombre limité de solutions non-dominées trouvées depuis le début de l'évolution. A chaque itération, de nouveaux individus non-dominés sont comparés aux membres de P' au sens de dominance et seules les solutions non-dominés restent ensuite dans P'.

Il est important de noter que SPEA non seulement préserve l'élite mais aussi la fait participer aux opérations génétiques.

Cet algorithme démarre avec une population P_0 de taille N créée aléatoirement et avec une population externe (ou "archive") P'_0 vide. À chaque génération t, les solutions non-dominées de P_t sont copiées dans la population externe P'_t et ensuite, toutes les solutions dominées (qui peuvent apparaître du fait de l'ajout de nouveaux éléments à P'_t) sont éliminées.

Quand la taille de P'_t atteint la limite N', un critère supplémentaire pour accepter une solution dans l'élite rentre en jeu. Ce critère, issu de la préoccupation de la préservation de la diversité entre les membres de P'_t, est basé sur le principe de *clustering*. L'algorithme de mise à jour de la population externe sera présenté en détail un peu plus loin dans cette section.

Après la mise à jour de l'ensemble P'_t, l'algorithme retourne vers la population courante. Le premier pas consiste à associer une valeur de performance à chaque solution. Notons que dans SPEA cette valeur sera à minimiser. Comme nous l'avons déjà dit, SPEA utilise l'archive P'_t dans le processus génétique. Une valeur scalaire de performance doit donc être attribuée non seulement aux solutions de la population courante mais aussi aux éléments de l'ensemble P'_t. La performance d'un individu de l'archive i est désignée par S_i (S comme *strength*). La valeur de S_i est proportionnelle au nombre (n_i) des individus de la population courante qui sont dominés par la solution externe i:

$$S_i = \frac{n_i}{N+1}. \tag{3.2}$$

La division par $(N+1)$ garantit que les valeurs de S_i sont inférieures à 1. De plus, une solution externe dominant moins de solutions de la population courante reçoit une valeur de performance plus petite (i.e. meilleure).

Ensuite, la valeur de performance d'un membre j de la population courante est égale à la somme des valeurs de "strength" de toutes les solutions externes la dominantes:

$$F_j = 1 + \sum_{i \in P'_t \wedge i \preceq j} S_i. \tag{3.3}$$

L'ajout de 1 garantit que la performance de toute solution de P'_i sera meilleure que la performance de chacune des solutions de la population courante P_i.

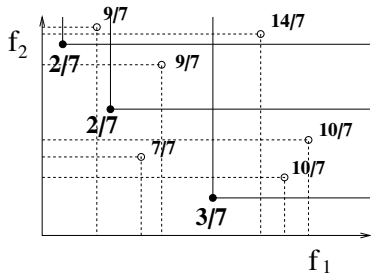


FIG. 3.6 – *Calcul de la performance dans SPEA*

La sélection par tournoi de la taille 2 effectuée sur l'ensemble $P'_i \cup P_i$ se base donc sur les valeurs S_i et F_j de façon à préférer les performance inférieures.

UNE ITÉRATION DE SPEA

1. Trouver l'ensemble des individus non-dominés $\mathcal{F}_1(P_t)$ de la population P_t.
   $P'_t \leftarrow P'_t \cup \mathcal{F}_1(P_t)$.
2. Trouver toutes les solutions non-dominées $\mathcal{F}_1(P'_t)$ de la population externe modifiée P'_t et éliminer toutes les solutions dominées: $P'_t \leftarrow \mathcal{F}_1(P'_t)$.
3. Si $|P'_t| > N'$, utiliser la technique de *clustering* pour réduire la taille de l'archive jusqu'à N'. La population résultante est la population externe de la génération suivante P'_{t+1}.
4. Calculer la performance des solutions de P'_{t+1} utilisant l'équation 3.2 et des solutions de P_{t+1} selon l'équation 3.3.
5. Appliquer le tournoi de la taille 2, le croisement et la mutation aux individus de l'ensemble $P'_{t+1} \cup P_t$ pour créer une nouvelle population P_{t+1} de taille N.

Procédure de clustering

Afin de réduire la taille de la population externe P'_t de N'_{trop} à N', la procédure suivante est utilisée.

D'abord, tout élément de P'_t est considéré être situé dans son propre cluster. Initialement, nous avons donc N'_{trop} clasteurs. Ensuite, les distances entre chaque paire de clusters sont calculées comme suite

$$d_{kl} = \frac{1}{|C_k||C_l|} \sum_{i \in C_k, j \in C_l} d(i,j).$$

Les distances $d(i,j)$ entre les solutions i et j peuvent être calculées dans l'espace de décision ou dans l'espace de critères, comme le font les auteurs de SPEA. Après ce calcul, les clusters k' et l' tels que $d_{k'l'} = \min_{k,l \in P'_t} d_{kl}$ sont réunis pour former un cluster plus grand. Toutes les distances d_{kl} sont alors recalculées et les deux clusters les plus proches sont de nouveau fusionnés en un seul. Le processus est répété jusqu'à ce que le nombre de clusters atteint N'. Dans chaque cluster, une solution est retenue. C'est la solution dont la distance moyenne des autres solutions du même cluster est minimale.

Algorithme de clustering

1. Toute solution i appartient à son cluster: $C_i = \{i\}$. Soit $C = \{C_1,...,C_{N'_{trop}}\}$ l'ensemble de tous les clusters.
2. Si $|C| \leq N'$, passer à l'étape 5. Sinon, passer à l'étape 3.
3. Pour tout pair de clusters, calculer les distances utilisant l'équation ci-dessus. Trouver le pair (i_1,i_2), qui correspond à la distance minimale.
4. Fusionner les clusters C_{i_1} et C_{i_2}. Retourner à l'étape 2.
5. Dans chaque cluster, trouver la solution dont la distance moyenne des autres solutions du même cluster est minimale et éliminer toutes les autres solutions du cluster.

L'algorithme présenté ici facilite la compréhension du mécanisme de clustering mais coûte $O(M(N'_{trop})^3)$. En effet, le coût du calcul des distances entre toutes le solutions est de l'ordre $O(MN^2)$. Ensuite, chaque passage entre de l'étape 2 à 4 coûte $O(N^2)$ et il est répété $(N'_{trop} - N')$ fois. Mais en fait, il est possible de réduire le coût de chaque itération de cette boucle jusqu'à $O(N)$ en utilisant la procédure assez complexe qui consiste à stocker les informations sur les clusters voisins et les mettre à jour après chaque fusion. L'étape 5 ayant un coût linéaire par rapport à la taille de l'archive, il est possible de mettre en œuvre l'algorithme de clustering de façon que sa complexité soit $O(MN^2)$.

Complexité de SPEA

L'étape 1 qui consiste à trouver le meilleur front non-dominé de P_t, exige $O(MN^2)$ opérations. Si N et N' sont du même ordre, l'étape 2 a le même coût de calcul que l'étape 1. Le clustering coûte $O(MN^2)$. La complexité d'autres étapes de l'algorithme ne dépasse pas cette valeur. Le coût total de SPEA est donc de l'ordre $O(MN^2)$.

Avantages de SPEA

Il est clair que si SPEA trouve une solution Pareto-optimale, cette solution est immédiatement placée dans la population externe. Elle peut être éliminée de l'ensemble P' seulement si une autre solution Pareto-optimale la rejoint. L'autre avantage de SPEA est que le principe du calcul de la performance scalaire associée à chaque individu est suffisamment simple.

Désavantages de SPEA

L'équilibre entre la tailles des populations externe et courante influence beaucoup la performance de l'algorithme. Si l'archive est trop grande, il y a le risque de non-convergence. D'autre part, si N' est trop petit, l'effet de l'élitisme peut être perdu. De plus, dans le dernier cas de nombreuses solutions de la population courante ne seront dominées par aucune des solutions externes et leur valeurs de performance seront donc les mêmes.

Les solutions externes dominant plus de solutions de la population courante sont évaluées comme moins bonnes que les solutions externes qui en dominent moins. Un tel principe est justifié seulement quand toutes les solutions dominées se trouvent très proche de la solution dominante. Ceci risque de fausser la pression de sélection pour des solutions non-dominées, surtout aux premiers stades de l'évolution.

Et enfin, le mécanisme de clustering a tendance à ne pas préserver les individus qui se trouvent sur les extrémités de l'archive, ce qui peut être très important pour la bonne représentation de la surface de Pareto par l'élite finale.

3.4.2 SPEA2

Nous allons présenter ici la version de SPEA corrigée par les mêmes auteurs [118]. SPEA2 diffère de son prédécesseur en plusieurs aspects.

Tout d'abord, la taille de l'archive P' dans SPEA2 est fixe, c'est-à-dire s'il n y a pas suffisamment d'individus non-dominés, l'archive est complété par ceux dominés. Notons que, dans la pratique, une telle situation n'a des chances de se produire que durant le début de l'évolution, et durant une période relativement courte.

Le calcul de la performance dans SPEA2 est plus "raffiné" que celui de SPEA, au sens qu'il tient plus compte de la densité des solutions. D'abord, la valeur de *strength* S_i est calculée pour toute solution $i \in P_t \cup P'_t$:

$$S_i = |\{j | j \in P_t \cup P'_t \wedge i \succ j\}|,$$

où $\succ$ correspond à la relation de dominance. Ensuite, la performance préliminaire R_i (*raw fitness*) est définie comme la somme des valeurs de *strength* des solutions qui dominent i:

$$R_i = \sum_{j \in P_t \cup P'_t, j \succ i} S_j.$$

Bien que cette façon d'attribuer une performance scalaire aux individus aide à préserver la population diversifiée dans l'espace de critères, elle n'est pas très efficace pour ce but quand la plupart des individus appartiennent au même ensemble non-dominé. L'information supplémentaire sur la densité des solutions est alors incorporée dans le calcul de leur performance.

SPEA2 utilise une adaptation de la technique du "$k^{\text{ième}}$ plus proche voisin" [102], où la densité en chaque point est une fonction décroissante de la distance entre ce point et son $k^{\text{ème}}$ point voisin. Dans SPEA2, l'inverse de la distance à la $k^{\text{ème}}$ solution voisine est pris comme la mesure de densité. Plus exactement, les distances entre chaque individu i et tous les autres individus $j \in P_t \cup P'_t$ sont calculées dans l'espace des objectifs et sont stockées dans une liste. Cette liste est ensuite ordonnée dans l'ordre croissant son $k^{\text{ème}}$ élément correspondant à la distance recherchée $\sigma_i^{(k)}$. Habituellement, la valeur de k est choisie comme la racine carrée de la taille de l'ensemble dont la densité est mesurée, i.e. ici $k = \sqrt{N + N'}$. La densité des solution autour de chaque solution i est caractérisée alors par la quantité suivante:

$$D(i) = \frac{1}{\sigma_i^{(k)} + 2}.$$

La valeur de performance définitive dans SPEA2 est calculée comme suite:

$$F(i) = R(i) + D(i).$$

Ces valeurs sont utilisées lors de l'étape de mise à jour de l'archive P'_t. D'abord, tous les individus non-dominés, c'est-à-dire ceux qui ont les valeurs de F inférieures à 1, sont copiés de l'archive et de la population courante dans l'archive de la génération suivante:

$$P'_{t+1} = \{i | i \in P_t \cup P'_t \wedge F(i) < 1\}.$$

Si $|P'_{t+1}| = N'$, la procédure de la mise à jour est terminée. Si l'archive est trop petit, $(N' - |P'_{t+1}|)$ individus dominés meilleurs au sens de F sont aussi copiés dans P'_{t+1}.

De fait, les changements qui ont été apportés à SPEA répondent à certaines remarques citées plus haut sous titre "Désavantages de SPEA".

Ainsi, la procédure du clustering qui contrôlait la taille de l'archive dans SPEA est remplacée dans SPEA2 par une méthode de troncation qui, contrairement au clustering, préserve les individus situés aux extrémités du front de Pareto. Alors, si la nouvelle archive est trop grande, l'individu qui se trouve à la plus petite distance d'un autre individu de P'_{t+1} est rejeté de l'archive. Ceci est répété jusqu'à ce que $|P'_{t+1}| = N'$.

Une autre différence importante entre ces deux approches est que dans SPEA2 seuls les membres de l'archive participent au processus de reproduction.

3.4.3 NSGA-II

En 2000, Deb et al. [18] ont proposé un AEMO élitiste qu'ils ont appelé NSGA-II pour indiquer les origines de la nouvelle approche. En fait, ce n'est que le principe de base du calcul de la performance que NSGA-II a hérité de son prédécesseur.

Dans NSGA-II, la population des enfants Q_t est d'abord créée à partir de la population des parents P_t. Ensuite, elles sont réunies en ensemble $R_t = P_t \cup Q_t$, qui est trié selon le principe de dominance. La population suivante est remplie par les solutions des fronts non-dominés de R_t l'un après l'autre en commençant, évidement, par le premier front. Vu que la taille de R_t est $2N$, tous les fronts ne peuvent pas rentrer dans la nouvelle population dont la taille doit être égale à N. Pour choisir les solutions qui vont survivre du front dont seulement une partie peut être placée dans la population suivante, une mesure de la densité des solutions dans l'espace de critères est utilisée.

Il est clair qu'au début de l'évolution ces solutions peuvent être choisies aléatoirement sans influencer le processus de l'optimisation. Mais à partir d'un certain moment t, le premier front de R_t commence à contenir plus de N individus. Dans ce cas la procédure de préservation de la diversité est appliquée à tous les individus non-dominés, ce qui a une grande importance pour la bonne représentation finale de l'ensemble de Pareto.

La population initiale de NSGA-II, P_0, est remplie aléatoirement, et ensuite triée selon le principe de dominance. A l'étape de sélection des parents, NSGA-II utilise un tournoi qui est basé tout d'abord sur le rang non-dominé, et utilise ensuite une mesure de densité dite *crowding distance* pour comparer deux solutions du même rang (cette mesure sera détaillée plus loin). La population Q_0 de taille N est obtenue par l'application des opérateurs de croisement et de mutation aux parents sélectionnés lors du tournoi.

Une itération de NSGA-II

Après initialisation aléatoire de la population initialie P_0, une itération de NSGA-II se déroule comme suit :

1. Créer Q_t à partir de P_t en utilisant le tournoi et en appliquant des opérateurs de variation génétique au individus gagnants.
2. Réunir les populations des parents et des enfants $R_t = P_t \cup Q_t$. Trier l'ensemble résultant R_t en fronts $\mathcal{F}_i$.
3. Soit une nouvelle population $P_{t+1} = \emptyset$. Soit le compteur des fronts $i = 1$.

4. Tant que $|P_{t+1}| + |\mathcal{F}_i| < N$, $\quad P_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i$ et $i \leftarrow (i+1)$.
5. Ordonner l'ensemble $\mathcal{F}_i$ selon les "distances de surpeuplement" (*crowding distances*; la procédure de leur calcul est présentée ci-dessous) et inclure $N - |P_{t+1}|$ solutions ayant les valeurs de distance les plus grandes dans la population P_{t+1}.

Il est important de noter que le tri de R_t en fronts non-dominés fait en stade 1 et le remplissage de la population P_{t+1} peuvent être effectuer simultanément. Chaque fois qu'un nouveau front est trouvé, on vérifie s'il peut rentrer dans P_{t+1} entierèment. Si ce n'est pas le cas, le processus du ranking s'arrête.

Calcul des distances de surpeuplement

Pour estimer la densité des solutions voisines d'une solution i dans un ensemble non-dominé $\mathcal{F}$, la quantité d_i, appelée ici "la distance de surpeuplement", est calculée de façon suivante.

Calcul des distances de surpeuplement dans un ensemble non-dominé $\mathcal{F}$

1. Soit $l = |\mathcal{F}|$. Soit d'abord $d_i = 0$ pour toute solution i de $\mathcal{F}$. Soit le compteur d'objectifs $m = 1$.
2. Pour l'objectif m, réordonner l'ensemble $\mathcal{F}$ de façon que les valeurs de f_m sur ses éléments diminues. Soit $I^m = sort_{[f_m,>]}(\mathcal{F})$ le vecteur des indices, c'est-à-dire I_i^m dénote l'indice de la solution i dans la liste ordonnée selon l'objectif m.
3. Pour chaque solution i t.q. $2 \leq I_i^m \leq (l-1)$, mettre à jour la valeur de d_i comme suite

$$d_i \leftarrow d_i + \frac{f_m^{I_i^m+1} - f_m^{I_i^m-1}}{f_m^{max} - f_m^{min}}.$$

   Et associer les valeurs de distance très grandes aux solutions sur les extrémités de $\mathcal{F}$, c'est-à-dire si $I_i^m = 1$ ou $I_i^m = l$, $d_i = \inf$.
4. Si $m = M$, la procédure est terminée, sinon incrémenter le compteur d'objectifs $m \leftarrow (m+1)$ et retourner à l'étape 2.

La quantité d_i correspond au semi-périmètre du cuboide dont les vertexes sont les voisins les plus proches de i (voir la figure 3.7).

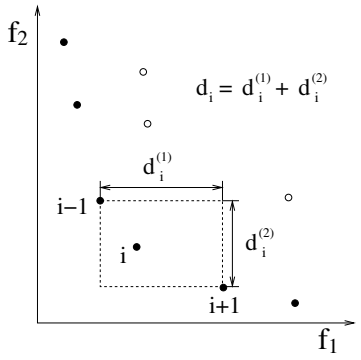


Fig. 3.7 – *Crowding distance - aide à la compréhention*

Le calcul des distances de surpeuplement exige d'ordonner chacun des fronts considérés M fois, chaque ordonnancement ayant le coût $O(NlogN)$. Le coût de l'étape 3 est $O(N)$. Le coût du calcul des distances de surpeuplement est alors de l'ordre $O(MNlogN)$.

Complexité de NSGA-II

L'étape 1 de NSGA-II consiste à trier la population de taille $2N$ en fronts non-dominés, ce qui exige $O(MN^2)$ opérations. L'étape 4 utilise les valeurs de distance de surpeuplement de toutes les N solutions de la population P_t, dont le calcul coûte $O(MNlogN)$. La complexité totale de NSGA-II est alors $O(MN^2)$.

Avantages de NSGA-II

A la différence de toutes les stratégies de préservation de la diversité vues jusqu'à présent, celle employée dans NSGA-II n'exige aucun paramètre à fixer. Notons que rien n'empêche de calculer les distances de surpeuplement dans l'espace de décision si cela est considéré plus adapté au problème.

Désavantages de NSGA-II

Comme nous l'avons déjà dit, le mécanisme d'évolution de NSGA-II est tel qu'à partir d'une certaine génération toute la population est contenue dans le premier front non-dominé. A ce stade, des solutions Pareto-optimales situées dans une région très "peuplée", peuvent être éliminées en laissant la place à des solution non-dominées dans la population courante mais qui ne sont pas Pareto-optimales.

NSGA-II exige le ranking d'une population de taille $2N$ au lieu de N comme dans la majorité des algorithmes.

3.4.4 PAES & PESA

PAES

Cette même année 2000, Knowles et Corne ont suggéré un AEMO, appelé PAES - *Pareto-Archived Evolution Strategy* [62] - qui est quasiment une (1+1)-ES (voir la section 2.1.2). La motivation pour utiliser une Stratégie d'Evolution vient de l'expérience des auteurs en résolution de problèmes de conception de réseaux dans le domaine de télécommunications. En étudiant un problème d'optimisation mono-objectif, ils ont observé que les stratégies basées sur des voisinanges, telles que *hill climber*, *tabu* ou *simulated annealing*, donnent de meilleurs résultats que les approches qui travaillent avec une population de solutions. Les auteurs de PAES ont donc essayé d'employer une stratégie similaire dans le cas multi-objectif.

D'abord, une solution de départ x_0 est choisie aléatoirement. Soit c_0 un enfant résultant de la mutation du parent x_0.

À chaque génération t, en plus d'un parent p_t et d'un enfant c_t, PAES garde en archive les meilleures solutions trouvées depuis le début de l'évolution. La taille limite de l'archive doit être choisie a priori. Les individus p_t et c_t sont d'abord comparés entre eux. Si un domine l'autre, l'individu dominant est retenu comme parent de la génération suivante et sa copie est gardée dans l'archive. Si aucune des solutions p_t ou c_t ne domine l'autre, c_t est comparée avec les solutions de l'archive courante. Si elle est dominée par un membre de l'archive, c_t est éliminée et

le parent p_t sera muté encore une fois à la génération suivante. Si, par contre, c_t domine quelques solutions de l'archive, ces solutions sont éliminées et c_t est acceptée dans l'archive. Notons que dans ce cas l'archive ne grandit pas car le nouveau membre élimine au moins un ancien. Si l'individu c_t appartient au front non-dominé formé par les membres de l'archive (c'est-à-dire ne domine personne et personne le domine), il est accepté dans l'archive à la condition que la taille limite de l'archive n'est pas encore atteinte. Il est clair que dans ce dernier cas, le parent p_t fait partie de l'archive, lui aussi. Pour décider qui sera le parent dans cette situation, les densités des solutions dans les voisinages de p_t et de c_t sont comparées.

La mesure de la densité des solutions dans PAES est différente de celles que nous avons rencontrées dans d'autres méthodes jusqu'à maintenant. Tout l'espace de recherche est divisé en $(2^d)^M$ hypercubes M-dimensionnels, où d est un paramètre défini par l'utilisateur. Le nombre de solutions dans chaque hypercube est calculé. La mesure de diversité d'une solution est définie comme le nombre de solutions qui sont situées dans le même hypercube (voir la figure 3.8). L'utilisation des puissances de 2 permet la division récursive de l'espace de recherche et rend plus efficace la localisation et le placement des solutions dans la grille.

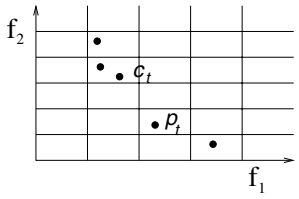


FIG. 3.8 – *Préservation de la diversité dans PAES*

En cas de dépassement de la taille limite, la mise à jour de l'archive est faite comme suit. Tout d'abord, l'hypercube contenant le nombre maximal de solutions est identifié. Si la solution en question (candidate pour rentrer dans l'archive) n'appartient pas à cet hypercube, elle rejoint l'archive tandis qu'une des solutions de l'hypercube le plus "peuplé", choisie aléatoirement, en est éliminée.

Complexité de PAES

En une génération de PAES, $O(MN)$ comparaisons sont nécessaires pour voir si l'enfant domine quelques solutions de l'archive (de taille N).

Lors de l'analyse de la diversité, l'hypercube de chaque membre de l'archive est d'abord identifié, ce qui coûte $O(NMd)$ comparaisons pour l'archive de N éléments. Tel est le coût d'une génération de PAES. Mais pour comparer la complexité de cette méthode avec celle des autres AEMO, il faut tout d'abord remarquer que N générations de PAES sont équivalentes à une génération de NSGA-II ou SPEA en terme de nombre de nouvelles solutions trouvées et de nombre d'évaluations.

PESA

Après PAES, les mêmes auteurs ont proposé une autre méthode appelée PESA -*Pareto Envelope-based Selection Algorithm* [9]. Dans PESA, le principe de l'archivage des solutions

non-dominées et la technique de préservation de la diversité sont les mêmes que dans l'algorithme précedant. Dans PESA, comme dans SPEA2, seules les solutions de l'archive participent au processus de variations génétique.

Algorithme PESA

1. Générer et évaluer une population "interne" (IP) de P_I individus. Une population "externe" (EP) est au début un ensemble vide.
2. Mettre à jour l'archive (EP).
3. Si le critère d'arrêt est satisfait, l'ensemble (EP) est retourné comme résultat de l'optimisation. Sinon, vider l'ensemble (IP) et répéter la procédure suivante jusqu'à ce que P_I nouveaux individus soient générés:
   - Avec la probabilité p_C, sélectionner deux parents de la population (EP), obtenir un seul enfant en les croisant et muter cet enfant. Avec la probabilité $(1-p_C)$, sélectionner un parent de (EP) et le muter.
4. Retourner à l'étape 2.

La procédure de la mise à jour de l'archive (EP) est identique à celle de PAES. Elle compare toutes les solutions non-dominées de (IP), une par une, aux membres de (EP).

La sélection des parents à l'étape 3 est basée sur la mesure de diversité qui est appelée par les auteurs "squeeze factor" et qui est calculée, comme dans à PAES, comme le nombre de solutions dans le même hypercube que la solution en question. Si la sélection par tournoi est employée, deux solutions de l'archive sont choisies aléatoirement et celle, dont la valeur de "squeeze factor" est plus petite gagne.

Discussion

Chacune des deux méthodes, PAES et PESA, exige la définition de deux paramètres importants: la taille de l'archive N et le paramètre de diversité d. Le nombre d'hypercubes change exponentiellement avec le changement de d, ce qui rend plutôt difficile le contrôle arbitraire de l'uniformité de la répartition des solution. De plus, pour trouver la longueur de chaque côté d'un hypercube, il faut connaître ou estimer le rang des valeurs de chaque fonction objectif, ce qui est difficile pour la plupart des problèmes réels.

Compte tenu de l'analyse de la complexité de PAES, une itération de PESA coûte $O(MdN^2)$.

3.4.5 Faites à votre guise

Les travaux comparant les AEMO ne laissent pas de doute [117] sur le fait que l'introduction de l'élitisme a mené à des performances nettement supérieures. Par contre, les comparaisons de dernières méthodes élitistes entre elles ne permettent pas de faire un choix sans hésitation. Il est difficile de dire que la performance d'un des algorithmes SPEA2, NSGA-II et PESA est meilleure de celles de deux autres. Zitzler et al. [118] ont établi qu'aux premières générations, PESA s'approche plus vite de la surface des compromis, mais la performance finale de SPEA2 et de NSGA-II semble être meilleure sur certains problèmes tests au sens de la métrique utilisée dans leur étude. Ce comportement de PESA est probablement dû, au début de l'évolution, à une plus grande intensité de l'élitisme, et, vers la fin de l'évolution, à la difficulté du choix approprié du paramètre d, en même temps qu'à la forte influence de ce paramètre sur la performance.

Il ne faut pas oublier que toute analyse comparative des AEMO est limitée non seulement par le choix des tests mais aussi des métriques de performance (ce problème fera l'objet de la discussion de la section 4.2). La figure 3.9 résume quelques études de comparaison des Algorithmes Evolutionnaires Multi-Objectif, dont la plupart sont faites par les auteurs de nouveaux algorithmes dans le cadre de leur validation.

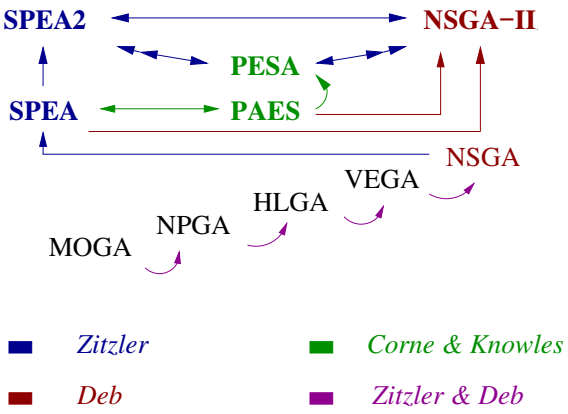


FIG. 3.9 – *Comparaison des performances de différents AEMO; les flèches pointent vers les meilleurs*

Dans la pratique, le choix de l'utilisateur parmi ces méthodes est plutôt dicté par des particularités de leur mise en œuvre, et par la nécessité du choix a priori de certains paramètres importants. Par exemple, pour les applications présentées dans la partie 2 de cette thèse, c'est la méthode NSGA-II qu'a été utilisée. L'avantage de NSGA-II consiste en absence de paramètres liés à la mesure de la diversité (comme d dans PESA), et à la taille de l'archive.

3.5 Optimisation Multi-Objectif Evolutionnaire sous contraintes

Les méthodes décrites dans les sections précédentes de ce chapitre sont destinées à résoudre le problème d'optimisation (1.1) avec $K = L = 0$, c'est-à-dire sans contraintes. Cependant, dans la plupart des applications réelles, l'optimisation est soumise à un certain nombre de contraintes qui sont souvent difficiles à satisfaire. Cette section présente quelques AEMO utilisables dans ce cas.

Disons dès le début qu'aucune étude comparative n'a été faite pour ces algorithmes. Des résultats numériques isolés sont présentés pour chacun d'eux dans les travaux des auteurs respectifs.

3.5.1 Pénalisation et AEMO

La prise en compte des contraintes par la majorité des AE mono-objectifs se base sur l'utilisation d'une fonction de pénalisation (voir la section 2.5.1). Cette stratégie s'adapte facilement aux AEMO.

En effet, il suffit d'appliquer un des AEMO au problème de l'optimisation multi-objectif

suivant

$$\begin{cases} min \quad F_m(x), & m = 1,...,M; \\ x_i^l \leq x_i \leq x_i^u, & i = 1,...,n \end{cases}$$

avec

$$F_m(x) = f_m(x) + \mathcal{F}_{penalty}^{(m)}(x).$$

Ici, la fonction de pénalité $\mathcal{F}_{penalty}$ est définie sous forme générale par l'égalité

$$\mathcal{F}_{penalty}^{(m)}(x) = c_m(t) \sum_{j=1}^{L} \mathcal{P}(\delta_j(x))$$

et $\delta_j(x)$ dénote la mesure du degré de violation de la $j^{\text{ème}}$ contrainte (voir l'équation 2.2). A la différence de l'optimisation mono-objectif, il est important ici de mettre à l'échelle les valeurs de δ_j pour tous les $j = 1,...,L$ avant de calculer $\mathcal{F}_{penalty}^{(m)}$. Les coefficients $c_m(t)$ sont différents pour différents objectifs afin de rendre $\mathcal{F}_{penalty}^{(m)}$ du même ordre de magnitude que f_m.

Discussion

L'inconvenient de cette approche est toujours le même (voir la section 2.5.1), c'est-à-dire, la nécessité de définir $\mathcal{P}$ et $c_m(t)$. Il est renforcé par le fait que la sélection dans les AEMO les plus performants n'est pas basée directement sur les valeurs des objectifs mais seulement à travers le concepte de dominance. Cette particularité rend le choix des paramètres de pénalisation encore plus difficile en l'absence d'études consacrées à ce sujet. Un exemple de l'utilisation des fonctions de pénalité dans le cas de l'optimisation multi-objectif est cité dans l'étude de Deb [16], où $\mathcal{P}(\delta_j(x)) = \delta_j(x)$ et $c_m(t) = c_m$.

3.5.2 Tournois "indirects"

Jiménezet al. [57] ont proposé une approche pour l'optimisation multi-objectif sous contraintes qui utilise le tournoi binaire basé sur les notions de faisabilité et de dominance. De plus, une mesure de la densité des solutions sert du critère secondaire pour choisir un gagnant.

Pour deux individus participant au tournoi x_1 et x_2, il y a trois possibilités: soit les deux sont faisables, soit les deux sont infaisables, soit, enfin, un des deux respecte toutes les contraintes et l'autre non. Dans ce dernier cas, l'individu faisable est declaré gagnant contre l'infaisable. Par contre, quand x_1 et x_2 sont équivalents au sens de faisabilité, un des deux scénarios suivants (similaires de ceux employés dans NPGA - section 3.3.3) est utilisé:

Tous les deux sont faisables
Un certaine nombre d'individus faisables est choisi aléatoirement pour jouer le rôle d'ensemble de comparaison. Chacun des x_1 et x_2 est comparé à cet ensemble au sens de dominance. Si un des deux domine tout l'ensemble et l'autre non, le premier est declaré gagnant. Si tous les deux sont équivalents de ce point de vue, la technique de partage est utilisée pour declarer comme gagnant l'individu situé dans une région de moindre densité.

Tous les deux sont infaisables
Dans ce cas, l'ensemble de comparaison est choisi parmi les individus infaisables. x_1 et x_2 sont

comparés cette fois seulement au meilleur représentant de cet ensemble. Le critère de comparaison approprié à la situation peut être, par exemple, le degré de l'infaisibilité des individus en question. Pour décider entre les individus équivalents, le partage rentre en jeu comme dans le scénario précedent.

Notons que dans les deux scénarios, la fonction de partage pour chacun des individus comparés est calculée selon l'équation 2.1 avec $\alpha = 2$ et elle nécessite les distances entre ces individus et tous les autres membres da la population. La complexité de cette méthode est $O(N^2)$.

Discussion

La raison pour laquelle deux participants du tournoi ne sont jamais comparés entre eux au sens de dominance ou selon leur degré d'infaisabilité n'est pas très claire. Pour rendre la vérification des relations de dominance dans la population moins stochastique, la taille de l'ensemble de comparaison doit être suffisamment importante.

Comme NPGA, cette méthode exige la définition appropriée de deux paramètres dont la taille de l'ensemble de comparaison et le rayon du partage.

Notons que lors de l'approche de Jiménez et al. (à la différence des méthodes basées sur la pénalisation), les individus faisables et infaisables sont comparés séparement, les individus vérifiant toutes les contraintes étant considérés a priori meilleurs de ceux qui en violent au moins une.

3.5.3 Mesures d'infaisabilité

Les méthodes que nous allons présenter dans cette section diffèrent par des détails (plus ou moins significatifs) mais se basent sur le même système de préférences, qui permet d'ordonner la population en mettant en tête les meilleurs et finissant par les pires individus. Cet ordonnancement est effectué en utilisant le ranking par dominance qui tient compte non seulement des objectifs mais aussi (et, en fait, tout d'abord) des degrés de violation des contraintes.

Soit $\mathcal{C}(x)$ une mesure (scalaire ou vectorielle) de l'infaisabilité d'un individu x. La préferance accordée aux individus respectant toutes les contraintes fait partie du "système de valeurs" adopté par les approches décrites ci-dessous. Ceci dit, la regle de base est que tout individu x tel que $\mathcal{C}(x) = 0$ est meilleur que tout individu y pour lequel $\mathcal{C}(y) > 0$ (s'il s'agit d'une mesure $\mathcal{C}$ vectorielle, il suffit que la dernière inégalité soit vérifiée au moins pour une composante du vecteur $\mathcal{C}(y)$).

La partie faisable de la population est classée par le principe de dominance comme dans NSGA ou MOGA, et une technique de préservation de la diversité est appliquée.

Il reste à choisir le critère qui indiquerait le pire des pires, c'est-à-dire, à définir la mesure $\mathcal{C}$.

Rangs d'infaisabilité

Dans la méthode de Fonseca et Fleming [36],

$$\mathcal{C}(x) = (\delta_1(x),...,\delta_L(x)),$$

où $\delta_j(x)$ est le degré de violation de la $j^{\text{ème}}$ contrainte défini par l'équation 2.2. L'opérateur de comparaison $\mathcal{C}(x) < \mathcal{C}(y)$ est défini par la relation de dominance entre ces vecteurs en vue de

minimisation de toutes leurs composantes. L'égalité $\mathcal{C}(x) = \mathcal{C}(y)$ correspond au fait qu'aucun des deux vecteurs ne domine l'autre.

Dans ce cas, tous les individus infaisables peuvent être ordonnés selon les principes de comparaison suivants:

- Si $\mathcal{C}(x) < \mathcal{C}(y)$, x est meilleur que y;
- Si $\mathcal{C}(x) = \mathcal{C}(y)$, alors x et y sont comparés au sens de dominance dans l'espace des objectifs du problème, f_m pour $m = 1,...,M$.
- Si $\mathcal{C}(x) = \mathcal{C}(y)$ et x et y sont équivalents aussi au sens de dominance dans l'espace des objectifs, alors la technique de partage entre les individus du même front non-dominé dans l'espace des objectif est appliquée.

Une mesure scalaire d'infaisabilité

En 1997, un an avant la parution de l'approche de Fonseca et Fleming, Binh et Korn ont publié un travail [6], dans lequel ils ont proposé d'utiliser la mesure d'infaisabilité suivante:

$$\mathcal{C}(x) = \big(\sum_{j=1}^{L} \delta_j^p(x)\big)^{1/p}.$$

Les individus pour lesqueles cette quantité est plus petite, sont considérés meilleurs.

Binh et Korn ont également suggéré de préserver un certain pourcentage des individus infaisables dans la population sous le prétexte qu'il y a des situations, où tels individus peuvent produire des enfants meilleurs que les individus faisables . Cependant, ceci s'est traduit par l'introduction de deux paramètres supplémentaires, dont un est lié à la proximité des individus infaisables préservés de la région faisable et l'autre est le pourcentage des individus infaisables maintenus dans la population.

Une mesure scalaire d'infaisabilité II

En 2001, Wright et Loosemore ont suggéré une méthode de prise en compte des contraintes dans les AEMO qu'ils ont appelée *Infeasibility Objective Approach* [114]. Elle consiste à utiliser la somme des violations des contraintes mises à l'échelle en tant que la mesure d'infaisabilité des individus:

$$\mathcal{C}(x) = \sum_{j=1}^{L} \delta_j(x).$$

Les auteurs ont comparé leur approche seulement à celle de Fonseca et Fleming en disant que l'utilisation d'une mesure d'infaisabilité scalaire est plus économique car ceci ne nécessite pas le ranking des individus infaisables.

Discussion

En effet, la méthode de Fonseca et Fleming est plus coûteuse à cause du ranking de la partie infaisable de la population dans l'espace des contraintes. De plus, à la différence de leur algorithme, aucune des deux dernières approches n'exige le calcul des objectifs pour les individus infaisables, ce qui peut être très important dans les applications où le calcul des fonctions objectif est cher, voir, peut ne même pas être possible.

Par contre, lors de l'utilisation des algorithmes basés sur une mesure d'infaisabilité scalaire, il faut faire très attention à la mise à l'échelle des valeurs de violation des contraintes.

Notons que ni dans le travail de Binh et Korn ni dans celui de Wright et Loosemore aucun argument pour justifier le choix de la métrique $\mathcal{C}(x)$ n'était présenté. D'ailleurs, il est facil de remplacer les métriques proposées par une de plusieurs autres mesures possibles, en s'inspirant de nombreuses études des fonctions de pénalité (voir section 2.5.1). En effet, dans les deux dernières méthodes il s'agit, en quelque sorte, de considérer la fonction de pénalité sans coefficients comme un objectif prioritaire dans l'optimisation multi-objectif.

Chapitre 4

Problèmes clés

S'il n'y a pas de solution,
c'est qu'il n'y a pas de problème.
Devise Shadok

Ce chapitre peut être considéré comme la critique constructive de l'état de l'art dans le domaine de l'Optimisation Multi-objectif Évolutionnaire. Il révèle certains problèmes qui se posent lors de l'utilisation des méthodes décrites dans le chapitre précédent et résume les solutions possibles trouvées jusqu'à présent. L'analyse de ces problèmes permet d'indiquer des directions éventuelles pour la recherche future dans ce domaine.

4.1 Représenter l'ensemble des solutions non-dominées

Il ne faut pas sous-estimer l'importance et la difficulté de cette tache, surtout quand le nombre de critères à optimiser est supérieur à trois. Et même pour des problèmes bi-objectif, il y a parfois le besoin de trouver une représentation particulière, qui permettrait de voir à quoi correspondent de différentes régions du front non-dominé dans l'espace de décision. Cette question se pose typiquement lors de la résolution des problèmes de conception.

Les quatre premières méthodes de représentation décrites dans la suite seront toutes illustrées sur le même exemple d'un problème imaginaire à trois objectifs. Une couleur est associée à chacune des cinq solutions non-dominées de ce problème pour permettre leur idéntification lors des représentations différentes.

Méthode de "projections" (*Scatter-plot Matrix Method*)

Il s'agit de la méthode de visualisation des dépendances entre plusieurs variables qui consiste à tracer leurs valeurs sur les plans correspondants à chaque paire. Dans notre cas, l'utilisation de *Scatter-plot Matrix Method* revient à tracer les projections des solutions de Pareto sur les plans correspondants à chaque paire d'objectifs. Tous les graphes sont rassemblés en une matrice triangulaire supérieure ou inférieure (voir figure 4.1). Différentes solutions peuvent être distinguées grâce à une couleur et à un symbole particuliers, qui lui sont associés. $M(M-1)$ graphes suffisent (mais sont nécessaires !) pour représenter les solutions d'un problème à M objectifs.

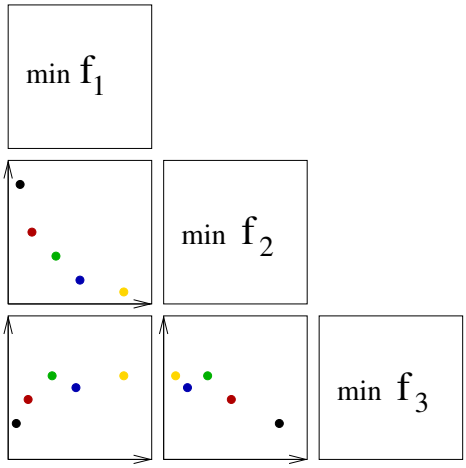


FIG. 4.1 – *Un exemple de l'utilisation de la méthode de projections*

Représentation par "le chemin de valeurs" (*Path Value Method*)

Cette méthode, proposée par Geoffrion et al. en 1972 [39], est souvent employée pour visualiser les solutions non-dominées des problèmes à plus que 2 objectifs (voir figure 4.2). Les valeurs entières sur l'axe horizontal désignent les numéros identifiant les objectifs du problème. L'axe verticale correspond aux valeurs normalisées des fonctions objectifs. Chaque ligne reliant toutes les "planches des objectifs" représente une solution de l'ensemble non-dominé trouvé par l'algorithme. Les points par lesquels la ligne passe correspondent aux valeurs des objectifs de la solution représentée par cette ligne.

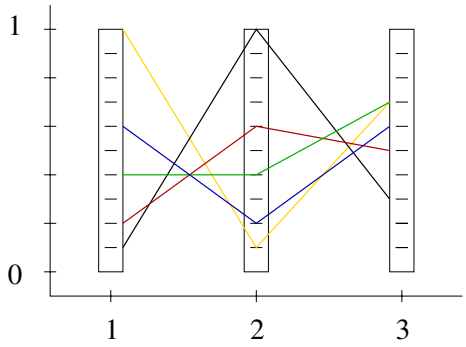


FIG. 4.2 – *Un exemple de la représentation par "le chemin des valeurs"*

Visualisation par diagramme (*Bar Chart Method*)

Une autre façon possible de visualiser les différentes solutions non-dominées dans l'espace des

objectifs est de tracer un diagramme comme dans la figure 4.3. D'abord, toutes les fonctions objectif sont mises à l'échelle et toutes les solutions sont ordonnées de façon quelconque. Ensuite, les barres correspondantes aux valeurs de chacun des objectifs pour toutes les solutions non-dominées sont tracées suivant cet ordre. Une couleur est associée à chaque solution pour distinguer les barres correspondantes aux solutions différentes (figure 4.3).

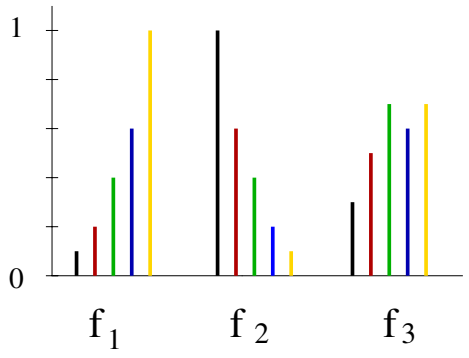


FIG. 4.3 – *Bar Chart Method*

Notons que dans la pratique, lors de l'application des AEMO, il est rare que l'ensemble non-dominé final soit suffisamment petit pour pouvoir le représenter en utilisant cette méthode.

Univers M-dimensionnel (*Star Coordinate System*)

En 1982, Manas a suggéré la représentation des solutions non-dominées multiples à l'aide d'un "système de coordonnées en étoile" - *Star Coordinate System* [69]. Un cercle divisé en M (nombre de critères) secteurs égaux est destiné à représenter une solution dans l'espace des objectifs. Chaque ligne de séparation des secteurs, qui relie le contour du cercle avec son centre, correspond à l'axe d'un des critères d'optimisation.

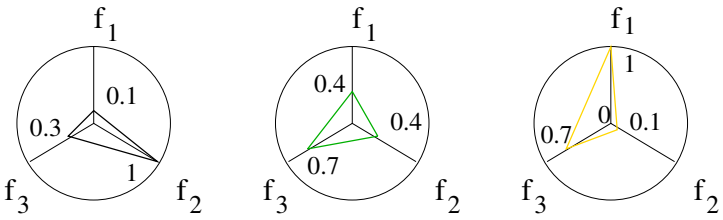


FIG. 4.4 – *Star Coordinate System*

A la différence de deux méthodes précédentes, la mise à l'échelle des fonctions objectif n'est pas exigée dans ce cas mais elle est souhaitable car permet d'éviter de marquer les valeurs minimale et maximale de ces fonctions sur les extrémités des lignes de séparation correspondantes. (quand le nombre d'objectifs est grand, les légendes diminuent alors la lisibilité de la figure). Chaque solution est visualisée par un polygone dont les côtés sont les segments reliant les points

sur les lignes radiales correspondants aux valeurs des objectifs de la solution en question (voir figure 4.4).

Objectifs hiérarchisés

Cette approche à la visualisation de l'ensemble des solutions non-dominées est largement utilisée lors de la résolution des problèmes de conception. Les critères sont arbitrairement ordonnés, et les solutions sont présentées l'une après l'autre dans l'ordre total induit par l'ordre hiérarchique sur les critères.

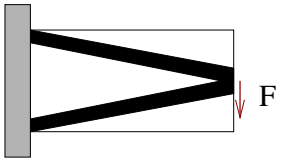


FIG. 4.5 – *Problème de la plaque cantilever standard*

Par exemple, cette approche est très appropriée aux problèmes d'Optimisation de Forme. La figure 4.6 représente un ensemble de solutions non-dominées de l'optimisation topologique de forme d'une structure contenue dans un domaine rectangulaire 2×1 [48]. Le problème consiste à minimiser simultanément le poids et le déplacement maximal d'une plaque console "cantilever" soumise à une force ponctuelle appliquée au milieu de sa frontière verticale droite, la plaque étant encastrée à son côté gauche (cf. figure 4.5). Dans ce cas, les structures sont ordonnées par poids décroissant.

Pour des problèmes bi-objectif, il est possible de visualiser le front de Pareto et un certain nombre de structures sur la même image (cf. figure 4.7). De telles illustrations permettent de mieux voir ce que représentent de différentes régions du front non-dominé dans l'espace de décision.

Enfin, si le nombre des objectifs est supérieur à deux mais n'est pas très grand, la combinaison de cette représentation avec la méthode des projections (*Scatter-plot Matrix*) peut également être utile lors du travail de conception.

La visualisation des paramètres de décision et des valeurs correspondantes des fonctions objectif dans le même graphe constitue un avantage important de cette méthode. Une telle représentation des résultats de l'optimisation multi-objectif est très efficace non seulement lors de l'analyse du problème mais particulièrement à l'étape de la prise de décision sur la conception finale.

Discussion

Les méthodes de visualisation décrites dans cette section représentent de façon suffisamment adéquate la distribution et l'étendue de l'ensemble non-dominé dans l'espace des objectifs.

Malgré ça, l'applicabilité de ces techniques est assez limitée. En effet, aucune des approches présentées, à part la méthode de projections, n'est prévue pour visualiser un grand nombre de

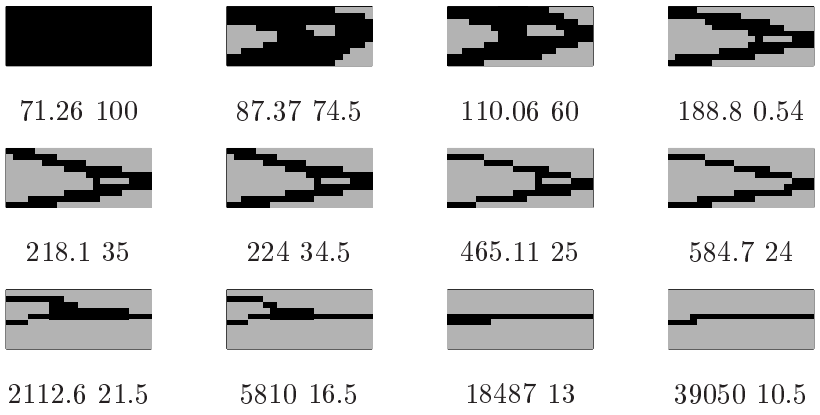


FIG. 4.6 – *Solutions non-dominées pour le problème de la plaque console 2x1. Le poids (en %) et le déplacement maximal sont indiqués sous chaque structure.*

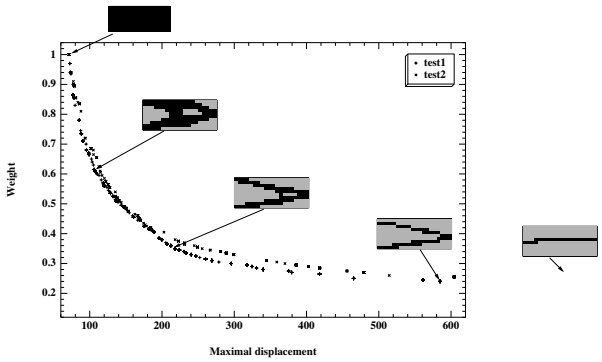


FIG. 4.7 – *Les fronts de Pareto pour le problème de la plaque console; certains points sont accompagnés par les structures correspondantes.*

solutions alternatives. C'est un désavantage important compte tenu du fait que, lors de l'augmentation du nombre d'objectifs, les AEMO sont supposés de fournir des solutions suffisamment nombreuses pour bien représenter la surface de Pareto multidimensionnelles.

En ce qui concerne *Scatter-plot Matrix* , son inconvénient est tel que l'ensemble de projections ne facilite pas la vision générale de la situation, ce qui serait pourtant souhaitable lors de l'analyse des résultats.

Le choix entre les méthodes présentées peut être dicté, d'une part, par des particularités du problème étudié et, d'autre part, par l'habitude de l'utilisateur qui peut être plus à l'aise en analyse des diagrammes qu'en comparaison de la longueur des rayons des étoiles ou vice versa.

4.2 Évaluer la performance d'un AEMO

La notion de la performance d'une méthode d'optimisation comprend deux aspects: la qualité des solutions qui peuvent être trouvées lors de l'application de cette méthode, et le coût de calcul nécessaire pour atteindre un certain niveau de qualité. Quelquefois, il est possible de garantir que l'optimum sera trouvé, ou au moins approché avec une précision fixée si le problème satisfait certaines conditions. Si telle garantie ne peut pas être donnée, la performance de la méthode doit être évaluée expérimentalement sur un ensemble de problèmes tests. En fonction de l'algorithme et des tests choisis, il peut être ensuite possible d'extrapoler les résultats de telle évaluation à d'autres problèmes similaires.

Pour les méthodes d'optimisation stochastiques, aucune garantie de qualité des résultats ne peut en général être donnée a priori. Les premiers travaux théoriques donnant une complexité moyenne de l'algorithme viennent d'être publiés et ne traitent que de fonctions d'école très simples. D'autre part, la notion même de performance expérimentale de ces approches doit tenir compte de leur nature stochastique, qui oblige de se baser sur une analyse statistique de séries d'expériences indépendantes.

Un ensemble de tests doit donc être soigneusement choisi de façon à mettre à l'épreuve l'efficacité des méthodes étudiées dans diverses situations difficiles, chaque test correspondant à une difficulté de nature particulière.

Mais avant de fixer les tests, encore faut-il formaliser des critères de qualité des solutions. Depuis quelques années, de nombreuses mesures quantitatives ont été proposées pour évaluer la qualité de l'ensemble des compromis optimaux (voir section 4.2.1).

Par contre, la question concernant la méthodologie statistique nécessaire pour l'analyse appropriée des résultats obtenus lors de plusieurs runs d'un AEMO, pour le moment, ne semble pas faire l'objet d'un large intérêt dans la communauté de l'Optimisation Multi-objectif Évolutionnaire.

4.2.1 Mesures de qualité pour la surface de Pareto approchée

Dans le chapitre précédent, nous étions souvent amené à évoquer deux buts principaux des AEMO qui sont la convergence vers la surface de Pareto et la distribution uniforme des solutions le long de cette surface. Ces deux aspects doivent donc être pris en compte par les métriques destinées à mesurer la qualité des résultats obtenus par un AEMO. De fait, les métriques décrites dans la suite peuvent être divisées en métriques de convergence, métriques de diversité, et métriques qui implicitement tiennent compte des deux aspects.

D'autre part, les métriques se divisent en celles, qui se basent sur la connaissance de la surface de Pareto exacte et celles qui n'exigent pas cette information. Notons que seules ces dernières sont utilisables pour évaluer l'applicabilité de différents algorithmes à un problème réel.

Dans cette section, Q sera l'ensemble de solutions non-dominées trouvé par l'algorithme et P^* l'ensemble de Pareto exact (lorsqu'il est connu).

Métriques I

Sous ce titre nous avons réuni des métriques intuitivement liées soit à la notion de dominance soit à celle de diversité entre les solutions non-dominées, mais nécessitant la connaissance de P^*. Elles peuvent servir, par exemple, lors de l'apparition d'une nouvelle méthode, qui est comparée aux approches existantes sur un ensemble de problèmes tests afin de justifier son existence. La

nécessité de connaître l'ensemble P^* les rend inutilisables dans les cas, où un AEMO est testé sur un problème réel.

– *Error Ratio* :
En 1999, Veldhuizen a proposé quelques métriques (dont deux sont présentées dans la suite) mesurant la "proximité" des solutions non-dominées trouvées par un algorithme de la surface de Pareto exacte [110]. Il est supposé que l'ensemble P^* est connu. La première de ces distances et le taux d'erreur (*Error ratio*) défini par :

$$ER(Q) = \frac{\sum_{i=1}^{|Q|} e_i}{|Q|},$$

où $e_i = 0$ si $i \in P^*$ et $e_i = 1$ sinon. Cette métrique calcule donc le nombre d'individus de Q qui n'appartiennent pas à P^*. Plus petite est la valeur de $ER(Q)$, meilleur est l'ensemble Q. L'inconvénient de cette métrique est que elle ne distingue pas la proximité relative entre P^* et les solutions de Q n'appartenant pas à P^*. Cet effet de seuil rend la métrique ER très peu utilisée.
Si e_i est redéfini comme

$$e_i = \begin{cases} 0, & \text{si } d(i,P^*) < \delta \\ 1, & \text{sinon} \end{cases}$$

$(1 - ER(Q))$ représente la proportion de solutions de Q, qui sont proches (au sens de δ) de P^*, ce qui est plus utile en pratique, mais garde un seuil lié au choix de δ.

– *Generational Distance* :
La deuxième distance proposée par Veldhuizen [110] est la distance générationnelle (Generational Distance), définie par :

$$GD = \frac{(\sum_{i=1}^{|Q|} d_i^p)^{1/p}}{|Q|},$$

où d_i est la distance entre une solution $i \in Q$ et le membre de P^* le plus proche dans l'espace des critères. Cependant si les fluctuations entre les différentes valeurs de d_i sont importantes, cette mesure ne représente pas correctement la distance entre les ensembles Q et P^*. Notons que les fonctions objectif doivent être normalisées avant de calculer les valeurs d_i.
Cette métrique a été utilisée par d'autres chercheurs en donnant à p la valeur 1 ([116, 17], ce qui revient à prendre la moyenne des d_i. Dans ces études, la déviation standard entre plusieurs runs a également été utilisée, pour réduire l'influence du biais dû aux fluctuations des distances.

– *Set Coverage Metric* :
Cette première métrique suggérée par Zitzler dans sa thèse [116] est destinée à être utilisée pour comparer deux ensembles non-dominés au sens de dominance. Elle définit une semi-distance entre deux ensembles de solutions A et B par :

$$\mathcal{C}(A,B) = \frac{|\{b \in B \quad | \quad \exists a \in A : a \preceq b\}|}{|B|}.$$

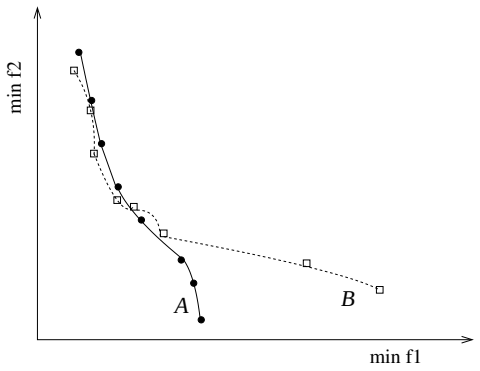


FIG. 4.8 – *Cas, où la métrique de recouvrement des ensembles (Set Coverage Metric) est inconsistante:* $\mathcal{C}(A,B) = 2/8, \quad \mathcal{C}(B,A) = 4/8$, *ce qui évalue* B *considérablement mieux que* A

Cette quantité correspond à la proportion des solutions de l'ensemble B, qui sont faiblement dominées par au moins une solution de A. Notons que $\mathcal{C}(B,A)$ n'est pas nécessairement égale à $1 - \mathcal{C}(A,B)$. Toutes les deux valeurs $\mathcal{C}(A,B)$ et $\mathcal{C}(B,A)$ doivent donc être calculées pour comparer les ensembles A et B. Il faut remarquer que le nombre de solutions dans les ensembles comparés est supposé le même.
Zitzler lui-même a utilisé cette métrique pour comparer plus de deux méthodes en présentant les résultats à l'aide de *Scatter-plot Matrix* (la première des méthodes présentées dans la section 4.1). Bien qu'une telle approche semble permettre l'évaluation de la performance relative de divers algorithmes pour une application réelle (vu que la connaissance de P^* n'est pas demandée), en réalité, ce n'est pas la proportion des dominants/dominés qui est du plus grand intérêt. En effet, dans la pratique, il est souvent plus important de savoir dans quelles régions un algorithme converge mieux que l'autre (s'il y a une tendance particulière de ce type). De plus, il est toujours utile de pouvoir évaluer le "degré de dominance" qui correspond à la distance entre deux individus dont un domine l'autre (voir par exemple la figure 4.8).

– *Maximum Spread*:
Ceci est la seconde métrique suggérée par Zitzler dans sa thèse [116]. Elle mesure l'étendue de la surface non-dominée:

$$D = \sqrt{\frac{1}{M} \sum_{m=1} M \Big(\frac{max_{i=1}^{|Q|} f_i^m - min_{i=1}^{|Q|} f_i^m}{f_m^{max} - f_m^{min}} \Big)^2}$$

Cette quantité correspond à la longueur normalisée de la diagonale de l'hyperparallelogramme formé par les valeurs extrêmes trouvées parmi les solutions de l'ensemble Q. Dans le cas de l'optimisation bi-objectif D représente la distance euclidienne normalisée entre les solutions extrémales dans l'espace des objectifs (voir figure 4.9). Le principal défaut de cette mesure est qu'elle ne tient pas du tout compte de l'uniformité de la distribution des solutions.

– *Spacing*:
La métrique suivante, suggérée par Schott en 1995 [97], évalue l'uniformité de la distribu-

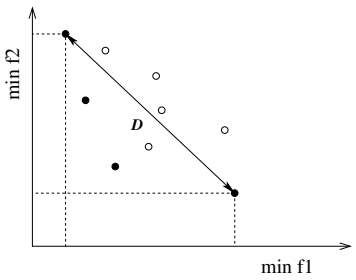


FIG. 4.9 – *Maximum Spread*

tion des solutions non-dominées :

$$S = \sqrt{\frac{1}{|Q|}\sum_{i=1}^{|Q|}(d_i - \bar{d})^2},$$

où $d_i = min_{k \in Q \wedge k \neq i} \sum_{m=1}^{M} |f_m^i - f_m^k|$, et $\bar{d}$ est la moyenne parmi les distances d_i pour i = 1,...,$|Q|$. S représente donc la déviation standard des valeurs de distance entre deux solutions consécutives de l'ensemble Q. Plus S est petit, mieux distribuées sont les solutions.

Le coût du calcul de S est de l'ordre $O(|Q|^2)$ à cause de la recherche de la somme minimale d_i pour chaque solution i. Afin de réduire cette complexité, il est envisageable de calculer d_i comme les distances de surpeuplement (cf. section 3.4.3) ce qui demanderait seulement $O(|Q|log|Q|)$ opérations.

– *Hypervolume* :
Présentons enfin une métrique très simple, qui tient compte des deux aspects, à savoir la convergence vers P^* et la distribution des solutions [119, 110] :

$$HV = volume(\cup_{i=1}^{|Q|} \mathcal{C}_i),$$

où $\mathcal{C}_i$ est l'hypercube dont la diagonale relie la solution i et le point de référence w qui correspond aux pires valeurs des objectifs (les valeurs exactes si elles sont connues, ou celles trouvées au cours de l'exécution de l'algorithme). La figure 4.10 montre le volume calculé dans le cas de deux critères.

R-métriques de Jaszkiewicz et Hansen

En 1998, Hansen et Jaszkiewicz ont proposé quelques métriques qui sont basées sur des "fonctions d'utilité" (*utility functions*) et qui déterminent le nombre attendu d'occasions où les solutions d'un ensemble A seront meilleures que celles de un autre ensemble B [50].

Une fonction d'utilité est un modèle qui associe une "valeur d'utilité" à chaque point de l'espace des objectifs compte tenu des préférences du preneur de décision. Autrement dit, entre deux ensembles de solutions, les métriques présentées ci-dessous "essayent de préférer" celui qui serait choisi par le preneur de décision.

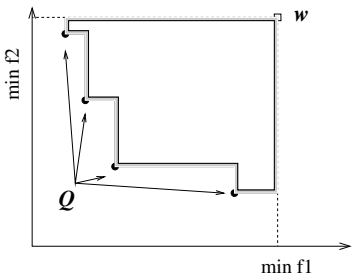


FIG. 4.10 – *Hypervolume*

Soit

$$R1(A,B,U,p) = \int_{u \in U} C(A,B,u)p(u)du,$$

où U est l'ensemble des fonctions d'utilité considérées, $p(u)$ est la fréquence d'occurrences de la fonction u, et la fonction C est définie comme suite:

$$C(A,B,u) = \begin{cases} 1, & \text{si } u^*(A) > u^*(B), \\ 1/2, & \text{si } u^*(A) = u^*(B), \\ 0, & \text{si } u^*(A) < u^*(B). \end{cases}$$

Ici, $u^*(A)$ est la valeur maximale de la fonction d'utilité u sur l'ensemble A, i.e. $u^*(A) = max_{z \in A} u(z)$. La métrique $R1$ mesure alors la proportion attendue des occasions, où l'ensemble A sera meilleur que l'ensemble B. Les auteurs ont suggéré de déclarer A comme meilleur si $R1(A,B,U,p) > 1/2$, B comme meilleur si $R1(A,B,U,p) < 1/2$ et si $R1(A,B,U,p) = 1/2$, aucune conclusion ne peut en être tirée.

La difficulté de l'utilisation de cette métrique est qu'elle demande de définir l'ensemble U et les probabilités correspondantes $p(u)$. De plus, afin de pouvoir l'utiliser pour comparer plus de deux ensembles (voir algorithmes), il faut introduire un ensemble de référence R, qui permettra de calculer $R1_R(A,U,p) = R1(A,R,U,p)$ pour chaque ensemble de solutions A, qui doit être évalué. Afin de réduire le biais dû au choix d'un tel ensemble R, les auteurs proposent d'utiliser plusieurs ensembles de référence.

Pour quantifier la supériorité d'un ensemble sur un autre, deux métriques suivantes ont ainsi été suggérées :

$$R2(A,B,U,p) = \int_{u \in U} (u^*(A) - u^*(B))p(u)du,$$

$$R3(A,B,U,p) = \int_{u \in U} \frac{u^*(A) - u^*(B)}{u^*(A)} p(u)du.$$

$R2$ mesure le degré de supériorité et $R3$ la proportion de supériorité attendus. Ces deux métriques peuvent également être utilisées avec un ensemble de référence, en définissant $R2_R(A,U,p) = R2(R,A,U,p)$ et $R3_R(A,U,p) = R3(R,A,U,p)$.

Signalons qu'avant d'introduire ces métriques, Hansen et Jaszkiewicz ont défini la relation de "performance supérieure" (*outperformance*) entre deux ensembles de vecteurs basée sur la relation de dominance. Ils donnent ensuite la définition de la *compatibilité d'une fonction avec*

une relation de performance supérieure. Il se trouve que chacune des métriques $R1$, $R2$ et $R3$ est compatible avec la relation de performance supérieure pour tout ensemble $U \subseteq U_c$, où U_c est l'ensemble des fonctions d'utilité compatibles avec la relation de dominance. C'est-à-dire, si $\preceq$ dénote la relation de dominance, alors

$$(u \in U_c) \Leftrightarrow [\quad (\forall x,y \in \mathcal{R}^n) \quad (x \succ y \Rightarrow u(x) \geq u(y)) \quad]$$

En effet, le choix de l'ensemble des fonctions d'utilité U influence beaucoup les résultats de l'évaluation par les R-métriques. Hansen et Jaszkiewicz proposent d'employer les fonctions d'utilité paramétriques définies à partir des normes L_p, $p \in \{1,2,...,\} + \{\infty\}$:

$$u_k(x) = -\Big[\sum_{i=1}^{n} \lambda_k (x_j^* - x_j)^p\Big]^{1/p},$$

où x^* est le point idéal (cf. section 1.2.3) du problème considéré, λ_k sont les poids non négatifs.

Dans leur travail, Hansen et Jaszkiewicz discutent également quelques points concernant la recherche du point idéal et le choix de la fonction d'intensité $p(u)$.

Notons qu'à la différence des métriques présentées sous le titre "Métriques I" ci-dessus, $R2$ et $R3$ permettent de quantifier la supériorité d'un ensemble non-dominé sur un autre, ce qui peut être très important en pratique comme nous l'avons remarqué sur l'exemple de *Set Coverage Metric* (cf. figure 4.8).

4.2.2 Performance statistique par Fonseca et Fleming

Le travail de Fonseca et Fleming paru en 1996 [35], semble rester le plus cité quand il s'agit de l'approche statistique pour l'analyse des résultats de plusieurs runs d'un AEMO. En effet, ce travail est, en quelque sorte, une exception au sens que les auteurs ne se limitent pas à l'introduction d'une métrique mesurant la qualité d'un ensemble non-dominé, mais proposent une stratégie statistique complète pour comparer la performance de deux méthodes stochastiques d'optimisation multi-objectif.

A la base de cette stratégie, il y a la notion de *attainment surface*, que nous traduirons dans la suite par *frontière de dominance* : en effet, il s'agit de la surface qui divise tout l'espace des objectifs en deux parties: celle qui contient tous les points dominés par au moins un des points de l'ensemble Q (ensemble non-dominé trouvé par l'algorithme), et la partie restante (voir figure 4.11).

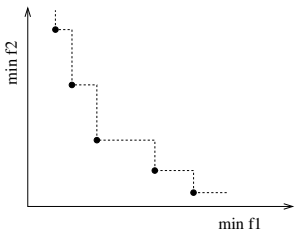


FIG. 4.11 – *Frontière de dominance (attainment surface)*

La frontière de dominance combine l'information sur la proximité de Q à la surface de Pareto exacte P^* (même si celle-là est inconnu) et sur la distribution des points de l'ensemble Q. En

effet, plus les éléments de Q sont proches de la surface P^*, plus la frontière de dominance sera proche de P^*. D'autre part, s'il y a un "trou" dans la distribution des solutions de Q, la région correspondante de la frontière de dominance sera plus loin de la surface P^* qu'en l'absence de trou.

Lors de plusieurs runs de l'algorithme d'optimisation, les frontières de dominance correspondantes aux ensembles non-dominés résultants peuvent être tracées dans la même graphe (cf. figure 4.12). Ces surfaces divisent l'espace des critères en trois parties. La première partie

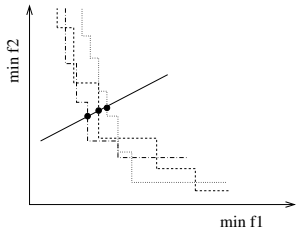


FIG. 4.12 – *Frontières de dominance de trois runs différents de l'algorithme*

située en bas à gauche des frontières est composée des points qui n'ont jamais été atteints. Les points dans la partie en haut à droite ont été dominés lors tous les runs effectués. Enfin, la partie restante contient les points qui ont été atteints lors d'un certain nombre de runs de l'algorithme évalué. Cette dernière partie de l'espace peut être encore divisée en sous-parties en fonction du pourcentage de runs lors desquels les points appartenant à cette partie ont été atteints.

Une telle interprétation de la superposition des frontières de dominance donne une base pour construire une mesure quantitative de la performance d'un algorithme stochastique multi-objectif. Par exemple, on peut essayer d'identifier une famille de points qui sont ont une grande probabilité d'être atteints lors de 50% de runs exactement (ils vont former la *frontière de dominance* 50% Pour ce faire, considérons les points d'intersection de l'ensemble des frontières de dominance avec une droite arbitraire tracée comme diagonale aux axes en direction de l'amélioration de tous les critères (cf. figure 4.12). Ces points définissent une distribution unidimensionnelle, qui, donc, peut être ordonnée. Le résultat désiré (qui est indépendant de la pente de la droite) est obtenu par l'estimation successive de la moyenne de tous les échantillons possibles définis de cette façon. Les frontières de dominance 25% et 75% peuvent être identifiées de la même manière.

La comparaison des frontières de dominance 50% représente une des façons possibles d'évaluer la performance relative de deux algorithmes. D'autres tests statistiques utilisant la distribution des points d'intersection peuvent également servir de base pour faires ces comparaisons.

4.2.3 Performance en dynamique (*Running metrics*)

En 2002, Deb et Jain ont suggéré d'évaluer la qualité de la sous-population non-dominée d'un AEMO à chaque génération afin d'estimer quelles difficultés peuvent être rencontrées à différentes étapes de l'évolution [20].

Selon Deb et Jain, les métriques utilisées pour cela doivent vérifier les conditions suivantes:

– De telles métriques doivent prendre des valeurs entre 0 et 1.

- La valeur "désirée" de la métrique (calculée sur un ensemble parfaitement convergé et diversifié) doit être connue.
- La métrique doit posséder la propriété d'une certaine continuité par rapport à la "qualité de la population" en chaque génération.
- De telles métriques ne doivent pas être très coûteuses à calculer.

Les auteurs suggèrent l'utilisation de deux métriques, une pour la convergence et l'autre pour la diversité, dont chacune ne correspond qu'à un de ces deux critères. C'est pour cela que, par exemple, la métrique de l'hypervolume S n'est pas un bon critère en tant que "métrique dynamique".

C'est la métrique GD (*Generational Distance* de Veldhuizen) qui a été retenue dans [20] en tant que marqueur privilégié de la convergence. La mise à l'échelle est faite comme suite:

$$\overline{GD}(Q_t) = GD(Q_t)/GD(Q_0),$$

où Q_t est l'ensemble des individus non-dominés trouvés lors de la génération t.

Pour mesurer la diversité, les auteurs utilisent leur propre technique, finalement assez proche de celle proposée récemment par Farhang-Mehr et Azarm [29]. Cette dernière consiste à associer une fonction d'entropie normalement distribuée à chaque solution projetée sur un hyperplan de dimension $(M-1)$. Toutes ces fonctions sont ensuite sommées en une fonction d'entropie normalisée. Si les projections des solutions sont bien distribuées sur l'hyperplan et si la déviation standard est bien choisie, la fonction d'entropie résultante sera plate, et l'entropie de Shannon calculée en utilisant cette fonction sera grande. D'autre part, si la fonction d'entropie résultante présente des pics, cela signifie que les points sont accumulés dans quelques régions de l'hyperplan.

4.2.4 Performance de l'algorithme vs qualité des résultats

Comme nous l'avons déjà noté, l'étude de Fonseca et Fleming où ils proposent une stratégie statistique pour la comparaison des AEMO [35] est un cas plutôt isolé par rapport à de nombreux travaux, qui ne font qu'introduire des mesures quantitatives pour évaluer la qualité de l'ensemble non-dominé approché.

Jusqu'à présent, la nature stochastique des AEMO n'a pas été suffisamment prise en compte lors de l'analyse comparative de ces méthodes. Par exemple, dans les études [121] et [15], les auteurs se limitent à calculer des métriques de type "Métriques I" (décrites ci-dessus) pour l'ensemble des solutions non-dominées obtenues lors de 5 runs de chaque algorithme participant à la compétition. Il faut reconnaître que les travaux (surtout purement théoriques), qui portent sur la comparaison des AEMO, seraient plus complets et, même, plus corrects si la richesse des techniques statistiques était exploitée de façon appropriée.

Rappelons deux aspects, dont chacun est important lors de l'analyse de l'efficacité d'un algorithme d'optimisation:

1. La qualité des solutions obtenues en un temps fixé;
2. Le temps nécessaire pour arriver aux solutions d'un certain niveau de qualité.

Le problème des AEMO c'est qu'en l'absence d'un critère d'arrêt autre qu'un nombre fixé de générations, l'évaluation du second aspect n'est pas possible. Or, le concept récent de "métriques dynamiques" [20] (présenté ci-dessus) semble tout à fait approprié pour cette tache, bien que les

auteurs ne mentionnent pas cette possible utilisation. Bien sûr, il ne pourrait s'agir dans ce cas que d'études comparatives théoriques utilisant des problèmes tests dont l'ensemble de Pareto exact est connu.

4.2.5 Construction des problèmes tests

Plusieurs problèmes tests ont été proposés lors des études des AEMO. Dans les premiers travaux, chaque apparition d'une nouvelle approche était l'occasion pour les auteurs d'introduire leur propres problèmes tests pour la validation de leur technique. Parmi les plus complexes, nous pouvons noter, par exemple, le test bi-objectif de Kursawe [67] dont le front de Pareto est concave et discontinu ou, encore, celui de Poloni [85].

En 1999, Deb a proposé un générateur de tests bi-objectif permettant de modéliser de diverses situations typiques dans lesquelles les AEMO rencontrent des difficultés soit à converger vers le front de Pareto, soit à assurer une bonne distribution des solutions obtenues [14]. Notamment, si une ou plusieurs fonctions objectif sont multi-modales, il y aura un certain nombre de surfaces de Pareto locales, pouvant gêner l'algorithme à converger vers la surface de Pareto "globale". Le même problème se présente, si la densité des points dans l'espace des objectifs est très petite à proximité de la surface de Pareto, et augmente rapidement lorsqu'on s'en éloigne. En cas de surface de Pareto optimale concave ou discontinue, ou si les solutions non-dominées ne sont pas uniformément distribuées le long de cette surface, il y a un risque accru de dérive génétique qui peut empêcher certains AEMO de retrouver toutes les régions de la surface optimale.

En 2001, Deb et al. ont adapté ce générateur au cas de M objectifs [21]. Ces auteurs ont indiqué un certain nombre de propriétés qu'un ensemble de tests doit posséder pour permettre l'analyse adéquate d'un AEMO. Selon cette étude, un "bon" test doit être tel que

1. il représente un danger particulier pour la convergence ou pour la diversité;
2. le nombre de variables de décision soit facile à changer;
3. le nombre d'objectifs soit facile à changer;
4. la forme et la position de la surface de Pareto soient connues et les valeurs des variables de décision correspondantes soient faciles à trouver.

Ensuite, dans le même travail, trois approches pour la construction des problèmes tests sont décrites. La première (la plus intuitive), consiste à prendre comme critères différentes fonctions n'ayant pas le même optimum. Dans ce cas, l'ensemble de Pareto est constitué de plus d'une solution, puisqu'il contient au moins chacun des optima individuels. L'inconvénient de cette approche est qu'il est en général difficile de connaître l'allure de la surface optimale.

Une autre méthode possible de la construction des tests pour les AEMO, est la *Constraint Surface Approach*: M fonctions objectif non contradictoires f_i sont choisies, et l'espace de recherche est limité en limitant les plages de valeurs autorisées pour ces fonctions: $f_i^L \leq f_i \leq f_i^U$, pour $i = 1,...,M$. Il est clair que la solution optimale correspond aux valeurs f_i^L. Ensuite, un certain nombre de contraintes sont introduites. Résoudre un tel problème revient donc à trouver la partie non-dominée de la frontière de la région faisable. Les difficultés liées à la diversité peuvent être introduites dans le cadre de cette approche, mais pas les difficultés de la convergence (multi-modalité ou optimum isolé). D'autre part, pour résoudre un problème d'optimisation sous contraintes avec au moins une contrainte active, il faut coupler l'AEMO avec une bonne stratégie de prise en compte des contraintes, et le choix d'une telle stratégie "bruite" considérablement l'évaluation de l'AEMO.

Nous allons décrire un peu plus en détail la méthode dite "de paramétrisation" (*Bottom-Up Approach*) car elle a été utilisée par Deb et al. pour créer leur générateur des tests.

Paramétrisation de surface (*Bottom-Up Approach*)

Dans cette approche, il faut d'abord définir analytiquement la surface de Pareto dans l'espace des objectifs. Ainsi, si par exemple nous voulons que la surface optimale de notre problème test soit concave, il suffit de choisir pour décrire la surface de Pareto une fonction simple et bien connue, telle qu'une fonction sphérique.

$$\begin{cases} f_1(\theta,\gamma) &= \cos(\theta)\cos(\gamma+\frac{\pi}{4}), \\ f_2(\theta,\gamma) &= \cos(\theta)\sin(\gamma+\frac{\pi}{4}), \\ f_3(\theta,\gamma) &= \sin(\theta), \\ \text{où} & 0 \le \theta \le \frac{\pi}{2}, \\ & -\frac{\pi}{4} \le \gamma \le \frac{\pi}{4}. \end{cases}$$

Pour que la surface décrite par cette fonction puisse représenter un ensemble de points non-dominés, nous avons limité les valeurs des paramètres θ et γ de façon à prendre la partie de la sphère qui se trouve dans le premier quadrant.

Ensuite le domaine de recherche dans l'espace des objectifs est construit "en dessus" de la surface optimale. Dans cet exemple, ceci peut être fait en construisant les surfaces parallèles à notre surface optimale, c'est-à-dire, en prenant les sphères de rayon plus grand que 1. Nous ajoutons donc un troisième paramètre r.

Le problème d'optimisation qui résulte de cette procédure devient ainsi :

$$\min\left(f_m(\theta,\gamma,r)\right),\ m = 1,2,3, \quad \text{avec } f_m(\theta,\gamma,r) = (1+g(r))f_m(\theta,\gamma),$$

où $g(r)$ est une fonction non négative quelconque. Les solutions optimales sont telles que

$$0 \le \theta^* \le \frac{\pi}{2}, \quad -\frac{\pi}{4} \le \gamma^* \le \frac{\pi}{4}, \quad g(r^*) = 0.$$

Bien que les fonctions objectif ne dépendent ici que de trois variables indépendantes, il n'est pas difficile d'augmenter la dimension de l'espace de décision en considérant les paramètres θ, γ et r comme *méta-variables* dont chacune dépend de n variables de décision :

$$\begin{aligned} \theta &= \theta(x_1,\ldots,x_n), \\ \gamma &= \gamma(x_1,\ldots,x_n), \\ r &= r(x_1,\ldots,x_n). \end{aligned}$$

Les fonctions doivent être choisies de façon à ne pas dépasser les bornes imposées pour θ, γ et $g(r)$.

De plus, ces dépendances peuvent être l'occasion d'introduire certaines difficultés typiques supplémentaires mentionnées plus haut.

Notons ainsi que la surface optimale correspond au minimum de la fonction g, i.e. $g^* = 0$. Par conséquent, si g est multi-modale, le problème multi-objectif final aura plusieurs fronts de Pareto locaux, correspondant aux minima locaux de la fonction g. La densité biaisée des points dans l'espace des critères peut également être introduite, par exemple en faisant dépendre g de r exponentiellement.

De la même manière, en définissant θ et γ d'une façon particulière, il est possible de biaiser la distribution des solutions le long de la surface optimale.

Il est clair qu'à l'aide de la fonction décrivant une sphère M-dimensionnelle, nous pourront construire une série des tests avec un nombre d'objectif quelconque. Et, bien entendu, d'autres fonctions f_m peuvent être utilisées. Le générateur de tests, présenté ci-dessous est un autre exemple d'application de l'approche *Bottom-Up*.

Générateur des tests de Deb

En 2001, Deb et al. [21] ont proposé de construire des tests à M objectifs, pour $M \geq 3$, de la façon suivante: tout d'abord, le vecteur des variables de décision est partitionné en M groupes $x \equiv (\mathbf{x}_1,...,\mathbf{x}_M)$. Ensuite, à partir de $M-1$ fonctions $f_1,\dots f_{M-1}$, d'une fonction g non-négative et d'une fonction h à M variables, on construit la fonction f_M par:

$$f_M(\mathbf{x}) = g(\mathbf{x}_M)h(f_1(\mathbf{x}_1),...,f_{M-1}(\mathbf{x}_{M-1}),g(\mathbf{x}_M)), \quad \mathbf{x}_m \in R^{|\mathbf{x}_m|}, \text{ pour } m = 1,...,M-1. \quad (4.1)$$

Le problème d'optimisation est ensuite simplement

$$\min \quad f_m(\mathbf{x}_m)_{m=1,...,M-1}, f_M(\mathbf{x})$$

La surface optimale correspond ici aux solutions sur lesquelles la fonction g atteint son minimum et elle est donc décrite comme

$$f_M = g^* h(f_1,...,f_{M-1},g^*).$$

La forme de la surface de Pareto sera alors définie par la fonction h qui dépend de M variables. Si h est non-convexe, la surface optimale sera non-convexe; si h est multi-modale, la surface sera discontinue. La fonction h permet donc de modéliser des difficultés liées à la diversité tandis que la fonction g est "responsable" pour empêcher la convergence car c'est g, qui est utilisée pour construire l'espace de recherche. Notons que le choix de la fonction f_m ($m = 1,...,M-1$), non-linéaire, permet d'introduire une densité variable de solutions le long de cet objectif.

Les cas tests bi-objectif qui ont été le plus largement utilisés dans les études portant sur les AEMO publiées dans les 2-3 dernières années sont présentés dans l'annexe A. Ces problèmes test sont utilisés dans deux chapitres suivants, pour la validation et l'illustration de l'applicabilité du critère d'arrêt introduit dans le chapitre 5, et de l'effet de l'opérateur de croisement proposé dans le chapitre 6.

Chapitre 5

Critère d'arrêt "par désespoir"

Il vaut mieux pomper même s'il ne se passe rien
que risquer qu'il se passe quelque chose en ne pompant pas.
Devise Shadok

La performance d'un algorithme d'optimisation est une caractéristique propre à l'algorithme, même si son évaluation est subjective au choix des tests et des métriques (ce qui est le cas des Algorithmes Evolutionnaires Multi-Objectif, cf. section 4.2). L'amélioration de la performance d'une méthode d'optimisation comprend généralement l'obtention de meilleurs résultats en un temps de calcul le plus réduit possible et sur une classe de problèmes la plus large possible.

Le choix du critère d'arrêt ne peut pas influencer la performance mais est important pour *l'efficacité de l'application* d'un AE. Ce terme n'a un sens que dans le contexte d'un problème ou d'une classe de problèmes concrets. Autrement dit, l'efficacité de l'utilisation d'une méthode est évaluée en fonction de la spécification donnée par des experts travaillant dans un domaine d'application. Telle spécification contient des contraintes et des préférences qui doivent être prises en compte, tout d'abord, lors du choix de la méthode à utiliser, mais ensuite, également, lors de la fixation des degrés de liberté de l'approche choisie, y compris le choix du critère d'arrêt.

Pour les AE mono-objectif, trois types de critère d'arrêt de base sont connus:

1. L'arrêt après un nombre fixe d'itérations (ou d'évaluations).
2. L'arrêt dès que la valeur de la performance du meilleur individu a atteint un certain seuil.
3. L'arrêt après qu'un certain nombre d'itérations ont été faites sans amélioration.

Dans la pratique une combinaison des critères énumérés est généralement utilisée. Vu que le temps de calcul est limité, c'est le premier des trois critères qui est souvent combiné avec un des deux (ou tous les deux) autres. Par exemple, la condition du second critère est d'abord testée mais, si au bout d'un certain nombre de générations la valeur attendue de la performance n'a toujours pas été atteinte, l'algorithme s'arrête.

En ce qui concerne les AEMO, seul l'arrêt après un nombre fixe de générations est employé. Notons que le second des trois critères énumérés ci-dessus utilise les valeurs absolues de la performance des individus. En cas des AEMO basés sur la dominance de Pareto, la notion de qualité de chaque individu est relative aux autres membres de la population. En l'absence de préférences particulières, la performance absolue d'un ensemble non-dominé peut être évaluée seulement en faisant référence à la surface de Pareto exacte. Clairement, un tel concept ne peut pas être utilisée dans le cadre d'une application pratique.

Par l'analogie avec la troisième condition d'arrêt de celles mentionnées plut haut, dans ce chapitre, nous proposons un critère selon lequel le processus de l'évolution d'AEMO est interrompu si l'amélioration de la performance de la population observée lors d'un certain nombre de générations successives a été trop faible. Cette étude se base sur la méthode NSGA-II (section 3.4.3).

5.1 Interruption de l'évolution "inefficace"

La maximisation du *ratio d'efficacité* {qualité des solutions}/{temps de calcul} se situe, généralement, en tête de la liste des priorités lors de la résolution d'un problème réel. Quand c'est le cas, le critère d'arrêt doit être choisi de façon à rendre ce ratio le plus grand possible dans les limites de la performance de l'algorithme utilisé.

En effet, si l'amélioration de la performance des individus d'un AE observée pendant plusieurs générations est très faible, la valeur du ratio qualité/coût diminue. Si cette situation persiste trop longtemps, l'évolution est "coincée". Dans ce cas, dans la perspective de l'optimisation du ratio d'efficité, il est préférable d'arrêter les itérations. Une telle situation peut survenir pour diverses raisons, et ne correspond pas nécessairement à la convergence. Néanmoins, nous considérons important de pouvoir repérer le moment à partir duquel l'espoir de gagner en efficacité (qualité/coût) est faible, pour éviter les itérations inutiles et, éventuellement, essayer de "réanimer" le processus d'optimisation en prenant des mesures supplémentaires telles que, par exemple, l'application de la recherche locale.

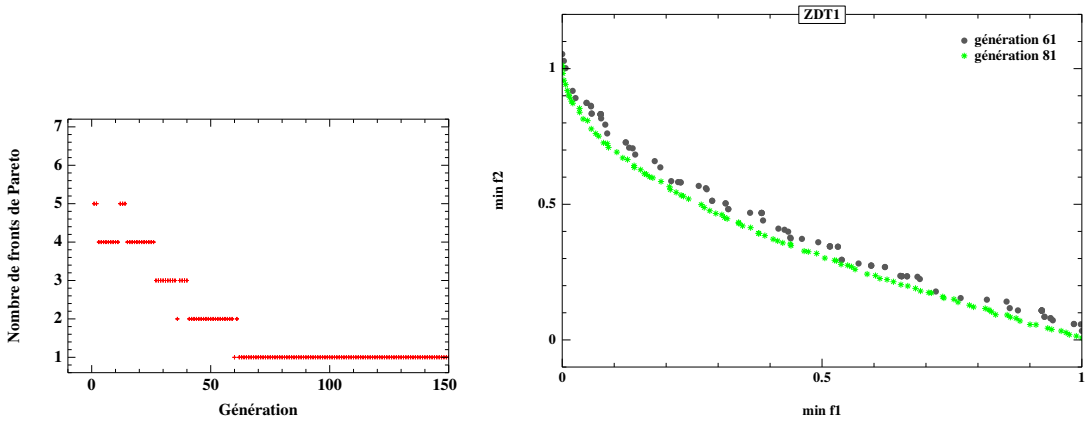
5.1.1 Situation révolutionnaire

L'observation du comportement des populations de NSGA-II au cours de l'évolution permet de remarquer la tendance générale suivante: la progression de la population dans le sens de l'amélioration des critères d'optimisation ainsi que l'uniformisation de la distribution des solutions non-dominées ralentissent au fur et à mesure que les populations s'approchent de la surface Pareto-optimale. L'idée de critère d'arrêt proposé dans ce chapitre consiste à dire que le moment où l'amélioration de la performance au sens "dominance + diversité" ralentit trop (voir, se stabilise), est un moment approprié pour interrompre l'évolution. Cette étude est basée sur les résultats obtenus lors de l'application de NSGA-II à quatre premiers problèmes tests formulés dans la section 4.2.5, qui sont tous bi-objectifs. Dans ce contexte, par *stabilisation* de la performance, nous allons entendre le moment, où il devient difficile de distinguer différentes générations visuellement sous les deux points de vue de la dominance et de la diversité des solutions dans l'espace des objectifs.

Absence de dominance et difficulté de la progression

Il a été remarqué [15] qu'en un certain stade de l'évolution de NSGA-II, les individus dominés sont éliminés, c'est-à-dire, toute la population commence à appartenir au même ensemble non-dominé. Effectivement, ce phénomène a été constaté lors de tous les essais (voir, par exemple, la figure 5.1(a)). Il est dû, en partie, au principe de la sélection et au remplacement déterministe de NSGA-II basé sur la relation de dominance. Mais il est important de noter que, tant que la population a encore "beaucoup de place" pour progresser, la population ne devient pas "monofrontale". Et même si, selon Deb [15], le fait que tous les individus soient non-dominés est

souvent la raison du ralentissement du processus d'optimisation, on dirait plutôt que les deux phénomènes (la concentration de la population dans un ensemble non-dominé et le ralentissement de sa progression) s'expliquent par la proximité de la frontière de Pareto globale ou locale qui empêche la progression de la recherche stochastique.



(a) (b)

FIG. 5.1 – *La population devient "mono-frontale" à la 61ème génération et le reste ensuite (a), mais une progression considérable a encore lieu pendant les 20 générations suivantes (b)*

Notons que la stabilisation de la performance, au sens qu'il est difficile de distinguer de différentes générations visuellement dans l'espace des critères, n'a jamais lieu avant que toute la population ne soit "rentrée" dans un même ensemble non-dominé. Au contraire, il a été observé que quand tous les individus sont déjà non-dominés, une progression considérable et relativement rapide peut encore avoir lieu (cf. figure 5.1(b)). Il semble donc clair que le moment de la stabilisation de la performance que nous cherchons à repérer, ne correspond pas forcément au moment à partir duquel toute la population est contenue dans un même ensemble non-dominé.

Problème de dégradation

La progression vers la surface de Pareto pourrait être mesurée à chaque génération en terme du nombre de nouveaux individus dominant au moins un individu non-dominé de la génération précédente. Cependant, en observant la dynamique du nombre de nouveaux individus dominant, il est difficile de parler d'une tendance particulière (en effet, on pourrait s'attendre que cette quantité tende peu à peu vers zéro) : ce nombre varie de zéro jusqu'à, à peu près, 20% de la taille de la population et ceci quelque soit le nombre de générations (c'est-à-dire, tant que nous laissons l'algorithme tourner).

Ce phénomène n'est pas aussi étrange qu'il peut paraître et est loin d'être le signe de l'amélioration infinie. En effet, le mécanisme de remplacement de NSGA-II n'empêche pas la dégradation. Par exemple, comme montre la figure 5.2, un individu non-dominé peut disparaître grâce à la maintenance de la diversité, et, quelques générations plus tard, un individu dominé par celui disparu peut apparaître dans l'ensemble non-dominé courant. Le phénomène de dégradation se produit de plus en plus souvent au fur et à mesure que la population s'approche de l'attrac-

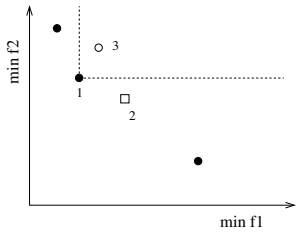


FIG. 5.2 – *Mécanisme de dégradation dans NSGA-II : en Nème génération, le point 1 disparaît en laissant la place au point 2; en (N+1)ème génération, le point 3 peut être retenu comme non-dominé*

teur, car dans ces conditions, une méthode stochastique (telle qu'est NSGA-II) trouve de "vraies" meilleures solutions de plus en plus difficilement.

Limite de l'uniformisation

L'efficacité du processus d'uniformisation de la distribution des points non-dominés dans l'espace des objectifs peut également être limitée. L'analyse de la dynamique des distances de surpeuplement au cours de l'évolution permet de remarquer leur tendance à se stabiliser.
Lors de l'application de NSGA-II aux problèmes test, les observations suivantes (illustrées dans la figure 5.3) ont été faites :

- La distance moyenne se stabilise au moment où toute la population rentre dans un même ensemble non-dominé (70ème génération dans la figure 5.3), ce qui, de fait, n'est pas surprenant.
- Il est plus difficile de parler de la stabilisation de la distance minimale car, parfois, elle oscille, assez peu mais constamment, du début à la fin de l'évolution. Mais dans la plupart des cas, il est possible de distinguer le moment, où l'amplitude de son oscillation diminue et se stabilise. Ce moment coïncide (grossièrement) avec celui de la stabilisation de la distance moyenne.
- Après la stabilisation des distances minimale et moyenne il peut y avoir encore des "trous" dans la distribution des solutions non-dominées et une certaine marge de progression au sens de la dominance (cf. figure 5.3(c)). On note expérimentalement sur tous les essais que la distance maximale se stabilise toujours après les distances moyenne et minimale (génération 100 dans la figure 5.3) et que **la poursuite des itérations après la stabilisation de la distance maximale n'améliore plus l'uniformité de la distribution des solutions non-dominées**.
- De même, lors de tous les essais, il a été noté que **la poursuite de l'évolution de NSGA-II après la stabilisation de la distance maximale n'améliore pas (ou seulement très peu) la performance des solutions au sens de Pareto non plus**. Ceci peut être dû à la probabilité de dégradation qui devient rapidement assez importante à cette étape de l'évolution.

La figure 5.3 présente une illustration de ces observations sur l'exemple de l'application de l'algorithme NSGA-II au problème ZDT3 (cf. annexe A). La 70ème génération correspond au moment, où toute la population a commencé à appartenir au même ensemble non-dominé et après la 100ème, la distance maximale peut être considérée comme stabilisée. 250 itérations

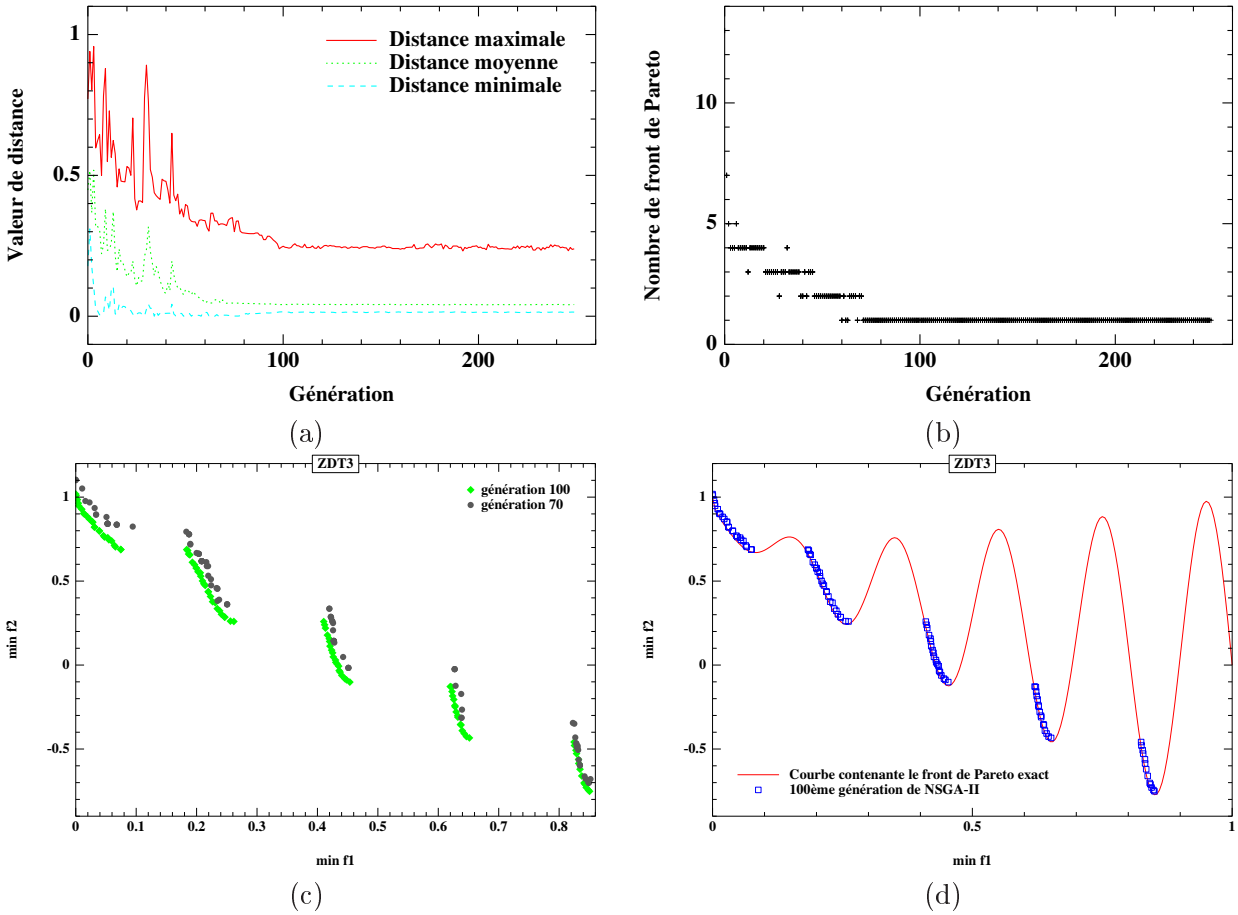


FIG. 5.3 – *Variation des distances de surpeuplement et de nombre de fronts de Pareto au cours de l'évolution lors d'un essai de NSGA-II sur le problème ZDT3*

ont été effectuées avec une taille de population égale à 100 (ces valeurs sont tirées du livre de Deb [15]) mais il est très difficile de distinguer visuellement les populations obtenues à la 100ème et à la 250ème générations. D'ailleurs, la figure 5.3 à droite permet de voir que le front de Pareto de ce problème a été presque retrouvé à la 100ème génération de NSGA-II.

Compte tenu des observations énumérées ci-dessus, la stabilisation de la distance maximale semble être un bon repère pour déterminer à quel moment interrompre l'évolution. Il reste alors de choisir une mesure appropriée de la variation de la distance maximale, ce qui permettra de formaliser la notion de sa stabilisation.

De plus, la notion de stabilisation implique forcement la nécessité de deux paramètres : δ_{lim}, le seuil de la variation de la quantité observée (distance maximale, en l'occurrence), et L, la longueur de la période sur laquelle la variation de cette quantité est mesurée. Cette variation doit être inférieure à δ_{lim} pour qu'on puisse constater la stabilisation de la quantité en question.

5.1.2 Moment de désespoir

Mesure de désespoir

Une mesure très simple de la variation d'une quantité d_l sur l'ensemble discret $l = 1,...,L$ est la différence entre les valeurs extrémales de d_l :

$$\delta_L = max_{l=1}^{L} d_l - min_{l=1}^{L} d_l.$$

Ici, d_l correspond à la distance de surpeuplement maximale dans la population, calculée à la génération l de NSGA-II, et δ_L est donc sa variation mesurée durant les L dernières générations. Dans ce contexte, la condition d'arrêt proposée s'écrit en fonction d'un paramètre δ_{lim} à définir par l'utilisateur de la façon suivante :

$$\delta_L(d_l) < \delta_{lim}. \tag{5.1}$$

Mais telle définition de la variation de d_l a au moins un inconvénient. En effet, il a été observé qu'après le moment de stabilisation, la valeur de la distance maximale peut avoir de petits sauts aléatoires qui empêchent le choix approprié de δ_{lim}. Si δ_{lim} est trop petit, la condition 5.1 est difficilement vérifiée même si la quantité d_l s'est déjà stabilisée. Et si δ_{lim} est suffisamment grande pour prendre en compte les sauts possibles, le risque d'arrêter l'algorithme trop tôt devient important. Il est d'ailleurs très difficile de prévoir l'amplitude de ces sauts car elle dépend du problème.

Afin d'aplanir l'effet de ces sauts, il a été décidé d'utiliser la déviation standard de la distance maximale :

$$\sigma_L \overset{\text{def.}}{\equiv} \sqrt{\frac{1}{L}\sum_{l=1}^{L}(d_l - \overline{d}_L)^2} < \delta_{lim},$$

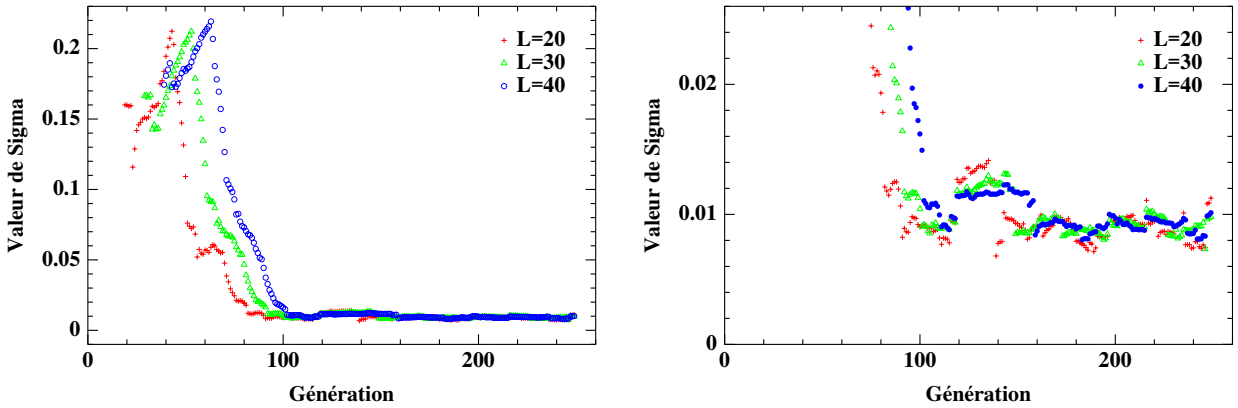
où $\overline{d}_L$ est la moyenne des valeurs d_l sur les L dernières générations.

La valeur de σ_L ne dépend pas de l'échelle des fonctions objectif du problème grâce au fait que les distances normalisées sont utilisées lors de la maintenance de la diversité (cf. section 3.4.3). Plus L est grand, moins σ_L est bruitée par des sauts singuliers de d_l qui peuvent avoir lieu après sa stabilisation.

Le moment critique

Afin de pouvoir donner quelques suggestions concernant le choix des paramètres δ_{lim} et L, le comportement dynamique de σ_L au cour de l'évolution a été analysé pour de différentes valeurs de L. Les remarques suivantes, qui résultent de cette analyse, sont utiles surtout pour restreindre la possibilité de mauvais choix de L et de δ_{lim} :

1. σ_L se stabilise toujours (voir par exemple figure 5.4(a)), conséquence naturelle de la stabilisation de la distance maximale au cours de l'évolution.
2. σ_L se stabilise au même niveau pour de différentes valeurs de L (Figure 5.4(a) : $\sigma_{20} \approx \sigma_{30} \approx \sigma_{40} \approx 0.01$ après sa stabilisation).
3. σ_L se stabilise avec un décalage temporel d'environ L générations par rapport à la stabilisation de la distance maximale.



(a) Vue générale (b) Zoom autour de la valeur de stabiliation

FIG. 5.4 – *Dynamique de la valeur de σ_L au cours de l'évolution de NSGA-II appliqué à ZDT1 avec une taille de la population égale à 100*

4. σ_{L_1} se stabilise $(L_2 - L_1)$ générations plus tôt que σ_{L_2} pour $L_1 < L_2$ (Figure 5.4(a) : σ_{40}, σ_{30} et σ_{20} se stabilisent l'un après l'autre avec un décalage d'environ 10 générations) et σ_{L_1} n'oscille quasiment plus après sa stabilisation (Figure 5.4(b)).

Comme nous l'avons remarqué plus haut, le moment d'arrêt approprié, généralement, suit de très près ou, même, coïncide avec la stabilisation de la distance maximale. Ce fait, en combinaison avec la troisième des observations citées ci-dessus, signifie qu'il est très probable que le moment d'arrêt doit précéder la stabilisation de σ_L, quelle que soit la valeur de L.

Cependant, il est dangereux de fixer L trop petite pour la même raison qui empêche d'utiliser δ_L comme mesure de stabilité. C'est-à-dire, quand L est petit, les oscillations importantes de σ_L rendent difficile un choix approprié de δ_{lim}.

D'autre part, la valeur de δ_{lim} doit être en dessus du niveau de la stabilisation de σ_L car sinon, il y a le risque de "rater" le moment d'arrêt lors de certains essais. De plus, elle ne doit pas être trop grande à cause du danger de s'arrêter beaucoup trop tôt.

Il est important de réaliser qu'il n'est pas possible de fixer les paramètres δ_{lim} et L indépendement de la taille de la population. Notons $\sigma_{L,stab}$ le niveau auquel σ_L se stabilise lors d'un essai de NSGA-II avec une taille de la population égale à N (par exemple, pour l'essai illustré dans la figure 5.4, $\sigma_{L,stab} \approx 0.01$). Un rapport suffisamment net entre les grandeurs σ_L^N et N pour diverses valeurs de N a été observé lors des essais avec les populations de taille 20, 100 et 200 pour tous les problèmes tests, et la relation suivante semble être toujours à peu près vérifiée :

$$\frac{\sigma_{L,stab}^{(N1)}}{\sigma_{L,stab}^{(N2)}} = \frac{N1}{N2}.$$

Par exemple, $\sigma_{L,stab}^{(20)} \approx 0.05$, $\sigma_{L,stab}^{(100)} \approx 0.01$ et $\sigma_{L,stab}^{(200)} \approx 0.005$ et ceci pour les tests avec ZDT1, ZDT2 et ZDT4. Pour le problème ZDT3, les valeurs absolues de $\sigma_{L,stab}^{(N)}$ sont un peu plus petites mais le rapport entre elles est le même.

Taille de population	Nombre de générations sur 21 essai	ZDT1	ZDT2	ZDT3
20	minimal	100	108	97
	moyen	105	146	122
	maximal	132	232	164
100	minimal	94	106	97
	moyen	98	116	110
	maximal	103	132	133
200	minimal	86	107	87
	moyen	88	110	95
	maximal	9	115	110

TAB. 5.1 – *Nombre de générations de NSGA-II nécessaires pour l'obtention des résultats de la figure 5.5.*

5.2 Le critère d'arrêt "par désespoir"

Compte tenu des remarques faites précédemment, les valeurs suivantes ont été retenues :

$$\delta_{lim}^{(20)} = 0.06, \quad L_{20} = 60,$$

$$\delta_{lim}^{(100)} = 0.02, \quad L_{100} = 40,$$

$$\delta_{lim}^{(200)} = 0.01, \quad L_{200} = 20.$$

Notons que chaque valeur a été fixées dans l'idée de pouvoir l'utiliser pour les quatre tests ZDT1-ZDT4 (cf. annexe A), ce qui, on l'espère, augmente les chances qu'elles soient également utilisables lors de l'application de NSGA-II à d'autres problèmes. Comme toute tentative de généralisation, cela implique la perte probable de la précision. Le compromis accepté ici est tel qu'il est préférable de s'arrêter un peu trop tard qu'un peu trop tôt, ou que beaucoup trop tard.

Les résultats obtenus après 21 essai de NSGA-II pour trois premiers tests ZDT1-3 en utilisant ces valeurs de δ_{lim} et L sont présentés dans la figure 5.5 Nous n'avons pas inclus les graphes correspondant aux essais faits avec la taille de population égale à 200 car il est difficile voir la différence entre ces solutions et celles trouvées avec la taille de population égale à 100.

Le tableau 5.2 montre en combien de générations de NSGA-II les résultats illustrés (tailles de population $N = 20$ et 100), ainsi que ceux concernant une taille de population de $N = 200$, ont été obtenus.

L'allure de la progression des populations au cours de l'évolution est différente pour de différents problèmes, ce qui explique le fait que la variation du nombre de générations d'un essai à l'autre peut être relativement grande pour un problème et petite pour un autre.

Au premier regard, il peut sembler étrange que, d'une part, le nombre moyen de générations nécessaires dans le cas $N = 20$ ne diffère pas trop de celui du cas $N = 100$ selon le critère d'arrêt utilisé et, d'autre part, que les populations de taille 20 n'atteignent pas le même niveau

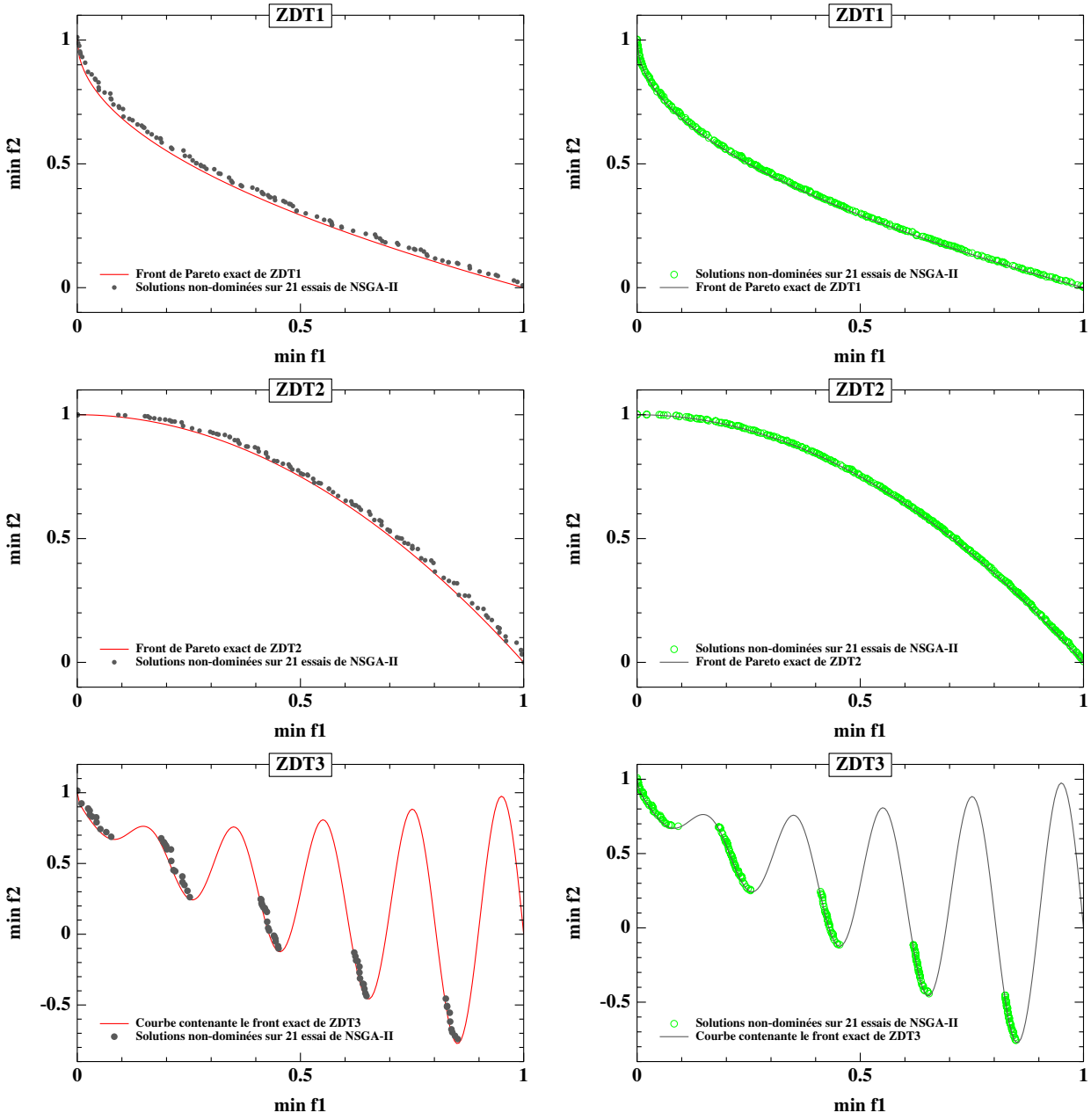


FIG. 5.5 – *Solutions non-dominées trouvées lors de 21 essais de NSGA-II avec une taille de population égale à* **20** *(à gauche) et* **100** *(à droite) comparées aux fronts de Pareto exacts pour les problèmes test ZDT1-ZDT3 présentés dans l'annexe A (de haut en bas).*

de qualité. A première vue, on pourrait penser que si l'algorithme avec $N = 20$ pouvait tourner plus longtemps, de meilleurs résultats seraient obtenus. Mais, après vérification, pour chacun des tests, il est impossible de distinguer visuellement l'ensemble des solutions non-dominées obtenues lors de 21 essais pour $N = 20$ avec le nombre de générations fixé à 500 de celui présenté dans la

figure correspondante ci-dessus (à gauche), quand les deux sont tracés sur la même graphe.

Le fait que les populations de taille 20 n'arrivent pas aussi proche du front de Pareto exact que celles de taille 100, (bien que, pour les problèmes considérés ici, la différence est assez petite) a donc une autre explication : quand une population relativement petite s'approche de l'ensemble optimal, le processus d'optimisation stochastique de NSGA-II commence à avoir plus de "liberté" pour explorer l'espace de recherche dans la direction perpendiculaire à celle de la dominance de Pareto que pour avancer vers l'optimum. C'est-à-dire, grâce au fait que les distances entre les membres d'une petite population sont relativement grandes, à proximité de l'optimum, NSGA-II retrouve de nouveaux individus appartenant au même ensemble non-dominé que la population courante avec plus de probabilité que de nouveaux individus dominants. Il en résulte un important phénomène de dégradation (détaillé en section 5.1.1), qui prend le pas sur la progression vers le front optimal.

Les résultats obtenus pour le problème ZDT4 sont présentés séparément, car, à la différence de trois autres tests, l'algorithme a convergé vers des fronts locaux lors de tous les essais qui ont été faits. La figure 5.6 présente les résultats de sept essais de NSGA-II effectués avec $N = 20$ et $N = 100$.

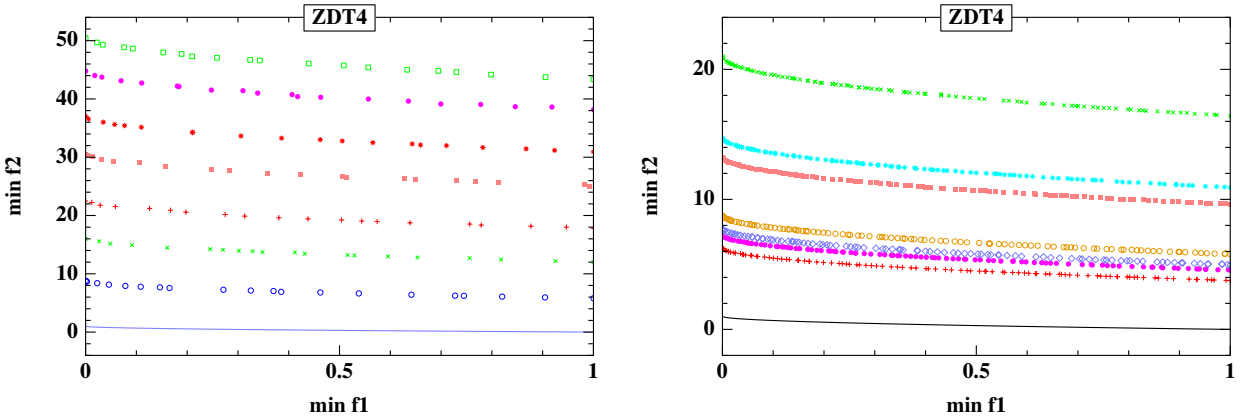


FIG. 5.6 – *Sept essais de NSGA-II avec la taille de population égale à 20 (à gauche) et 100 (à droite) comparées au front de Pareto exact du problème ZDT4*

Les ensembles de solutions les plus proches du front exact ont été trouvées en 249 itérations pour $N = 20$ et en 345 pour $N = 100$. Comme pour les autres tests, les essais avec le nombre de générations égal à 500 ont été faits. Les graphes correspondant à ces résultats ne sont pas inclus ici car ils présentent les mêmes "arcs en ciel" que dans la figure A.5. La figure 5.7 montre le comportement typique des populations au cours de l'évolution de NSGA-II lors de la résolution du problème ZDT4. Ce graphe correspond à un essai avec le nombre de générations fixé à 500 et $N = 100$. Si le critère d'arrêt était utilisé avec $\delta_{lim}^{(100)} = 0.02$ et $L_{100} = 40$, l'évolution serait arrêtée en 214ème génération (cf. figure 5.8).

Ces illustrations permettent de voir que la population, une fois convergée vers un front local, n'arrive plus à en sortir et que le critère d'arrêt proposé identifie le moment à partir duquel la poursuite des itérations n'apporte plus rien à la qualité des solutions, ni du point de vue de la progression vers le front exact, ni en ce qui concerne la diversité.

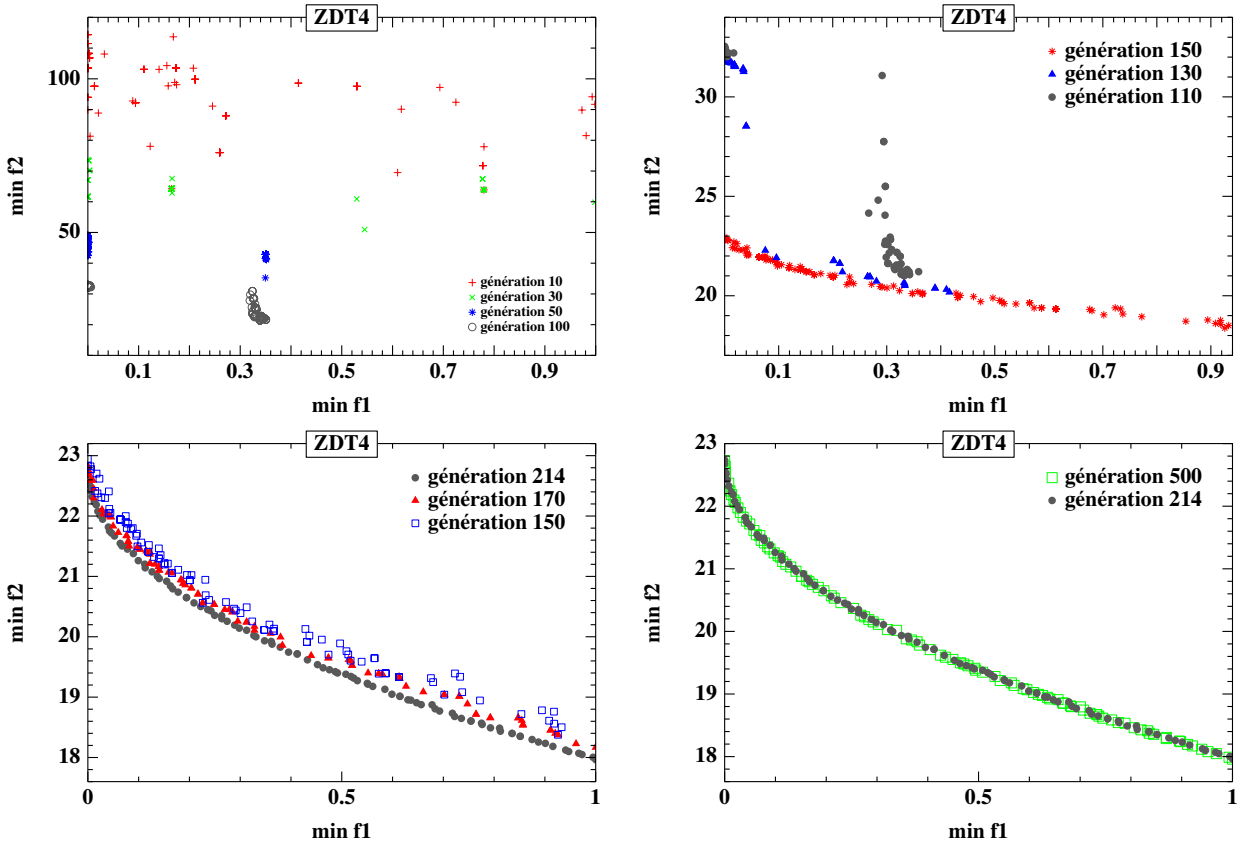


FIG. 5.7 – *Dynamique des populations de NSGA-II au cours de la résolution du problème ZDT4 avec* $N = 100$ *(Attention aux échelles en* f_2*)*

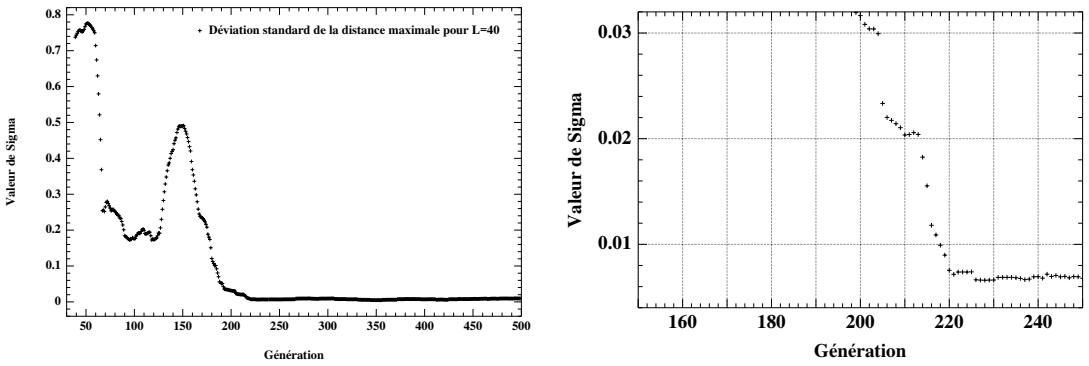


FIG. 5.8 – *Dynamique de la déviation standard de la distance maximale pour l'essai de la figure 5.7 (zoom à droite)*

5.3 Leçons et perspectives

La condition d'arrêt formulée dans ce chapitre pour l'algorithme NSGA-II se base sur la stabilisation de la mesure de diversité des solutions au cours des générations et, éventuellement,

peut être utilisée avec d'autres AEMO. Il serait, notamment, intéressant de le vérifier pour SPEA2. Il s'agirait dans ce cas de la stabilisation des distances euclidiennes qui servent lors de la mise à jour de l'archive.

Dans les cas, où l'algorithme converge vers le front global sans rencontrer de difficultés particulières, cette condition est un bon indicateur de la fin du processus d'optimisation.

Si au contraire, l'évolution se retrouve coincée près d'un optimum local, comme, par exemple, lors de la résolution du problème ZDT4, le critère d'arrêt proposé ici permet de repérer le moment, où il n'y a plus de raison de continuer les itérations car ceci n'améliore pas la qualité des solutions.

En absence de tel attracteur, si la progression de la recherche ralentit beaucoup trop à proximité de l'ensemble optimal sans l'atteindre (ce qui peut arriver à cause de la possibilité de dégradation), l'utilisation du critère introduit dans ce chapitre est aussi appropriée. En effet, dans la plupart des applications pratiques, il n'y a pas vraiment d'intérêt à faire des centaines d'itérations supplémentaires pour améliorer de façon négligeable les solutions.

La question du choix des paramètres nécessités par le nouveau critère a été étudiée sur la base des observations des résultats de l'application de NSGA-II aux problèmes test standards. En particulier, les valeurs suggérées ne dépendent pas de la mise à l'échelle des fonctions objectif.

Notons que dans certaines applications réelles, l'attente du moment, où il devient difficile de distinguer les générations visuellement, correspond même à une précision exagérée. Notamment, quand le calcul de fonctions objectif approchées est utilisé dans le processus d'optimisation, tenter de distinguer deux solutions dont la distance est inférieure à la précision du calcul des objectifs n'a aucun sens.

Cette étude est restreinte en cas des problèmes bi-objectifs. A priori, la même idée semble pouvoir s'appliquer également pour des problèmes à 3-4 objectifs – mais celà reste à vérifier. Cependant, une difference importante lors du passage à l'espace des critères d'une dimension plus élevée est le fait qu'il ne sera plus possible de se servir de l'analyse visuelle pour déterminer à quoi correspond le moment de stabilisation en terme de solutions retrouvées en différentes itérations.

Une autre priorité est d'étudier l'effet de l'utilisation du même critère d'arrêt lors de la résolution des problèmes sous contraintes. Dans ce cas, il faudrait baser le critère sur la stabilisation des distances entre les individus non-dominées faisables.

Chapitre 6

Croisement basé sur la dominance de Pareto

Malgrès la croissance rapide du domaine des AEMO, il n'y a pas eu beaucoup de tentatives d'adapter des opérateurs génétiques au but particulier de la recherche de l'ensemble non-dominé.

Hajela et Lin [46] et Fonseca et Fleming [33] ont utilisé dans leur algorithmes la restriction du choix des partenaires pour le croisement (*restricted mating*, ou croisement restreint) suggérée par Goldberg [41] pour des AE mono-objectif. L'idée consiste à interdire le croisement de deux individus dont la distance (mesurée dans l'espace de décision ou dans l'espace des critères) excède une valeur $\sigma_{maiting}$ définie par l'utilisateur. Horn et al. [54], au contraire, présentent un argument pour le choix des parents dissimilaires: l'information fournie par de très différents compromis peut être combinée de façon à permettre l'obtention d'autres types de bons compromis. La nécessité du croisement restreint dans les AEMO a été soulignée aussi par Jaszkiewicz [56]. Or, Zitzler et Thiele [119] ont constaté l'absence d'amélioration de leur résultats en cas de choix de partenaires proches; Schaffer est arrivé à la même conclusion en croisant des individus éloignés l'un de l'autre. En résumé, divers témoignages basés sur des évidences empiriques donnent des arguments pour et contre les mêmes types de la restriction du choix des parents dans les AEMO.

Wildman et Parks [112], dans le cadre de l'application d'un AEMO à un problème de conception concret, avec une représentation assez particulière, ont essayé de différentes stratégies de restriction du choix des individus pour la reproduction. Leurs stratégies correspondent à la minimisation ou à la maximisation de diverses mesures de dissimilarité, dont la dissimilarité au sens du rang de dominance. Cette étude utilise un AEMO dont l'élitisme se base sur la maintenance de l'archive des individus non-dominés. L'efficacité du croisement des individus de l'archive avec ceux de la population courante a été constatée.

Dans ce chapitre, nous proposons un opérateur de croisement basé sur l'idée née indépendamment du travail Wildman et Parks et qui utilise la notion de dominance de Pareto plus directement pour choisir les partenaires du croisement.

6.1 Croisement biaisé vers le parent dominant

Dans les travaux cités ci-dessus, la motivation principale de l'application du croisement restreint était de rendre les AEMO capable de retrouver des solutions de meilleure qualité. L'évaluation de chaque principe de restriction du choix des partenaires pour le croisement se

basait donc sur la comparaison des solutions finales trouvées avec et sans restriction.

L'opérateur proposé dans ce chapitre a été initialement pensé comme un moyen d'accélérer l'avancement de la population vers la surface de Pareto. L'idée générale consiste à croiser les individus non-dominés avec ceux ayant le rang de Pareto supérieur à 1.

Les tests numériques ont été faits en utilisant l'algorithme NSGA-II. Comme il a été remarqué dans la section 5.1.1, toute la population de NSGA-II "rentre" dans un même ensemble non-dominé en une certaine étape de l'évolution et reste "mono-frontale" jusqu'à ce que les itérations soient arrêtées. Le but que nous poursuivons en introduisant le croisement entre les individus dominés et le reste de la population est d'accélérer la progression vers les optima de Pareto pendant les générations où au moins deux ensembles non-dominés différents persistent dans la population.

6.1.1 Les parents

Parents de rangs différents

Lors des premiers tests basés sur cette idée simple – choisir pour partenaire des individus du premier front des individus n'y appartenant pas – son implantation telle quelle ne semblait pas se justifier. En effet, aucune accélération particulière par rapport au choix aléatoire des partenaires n'a pas été constatée lors de l'utilisation du croisement entre les individus non-dominés et le reste de la population.

Notons que le choix aléatoire de deux partenaires appartenant à deux ensembles non-dominés différents ne revient pas du tout à croiser des individus dont l'un domine l'autre. En faite, la probabilité qu'un individu du front non-dominé domine un individu donné d'un autre front décroît très rapidement avec le temps et elle est élevée seulement en toutes premières générations (voir figure 6.1). Alors, même si le choix de deux parents, dont un est non-dominé et l'autre ne l'est pas, est effectivement restrictif, cette restriction ne peut pas vraiment être dite basée sur la dominance de Pareto. (Imaginons, par exemple, le croisement entre les individus $x1$ et $x2$ dans la figure 6.1).

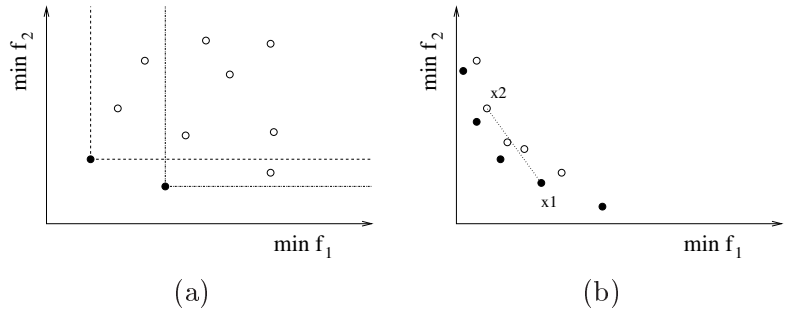


FIG. 6.1 – *Situation typique: (a) au tout début de l'évolution, il y a peu d'individus non-dominés mais chacun d'eux domine beaucoup d'individus; (b) au bout d'un nombre de générations, la population est constituée de moins de fronts et chaque non-dominé domine très peu d'individus*

Parents dont un domine l'autre

La seconde tentative de trouver une façon efficace de choisir des partenaires pour le croisement a mené à des résultats qui témoignent d'une certaine réussite. Ces résultats sont présentés dans la section 6.2. Le **choix du partenaire pour chaque individu non-dominé** est fait **parmi les individus qu'il domine**, s'il y en a dans la population. S'il y en a pas ou si la population est "mono-frontale", le partenaire est choisi aléatoirement.

La "logique" de l'utilisation de la notion de dominance lors du choix des partenaires du croisement dans le cadre de la recherche des optima de Pareto est assez intuitive: en croisant l'individu dominant avec un individu qu'il domine, nous espérons d'explorer la direction de l'amélioration de tous les critères. Il est clair, alors, que la réussite de cette idée dépend des fonctions objectif de chaque problème concret et de la région de l'espace de décision, où se trouvent les individus croisés.

6.1.2 Tel père, tel fils

Un prolongement naturel de la restriction du choix d'un partenaire à l'ensemble des individus domniés de la population est de biaiser également la distribution donnant lieu au choix de l'enfant vers parent dominant, avec toujours l'idée d'explorer préférentiellement la direction de l'amélioration de tous les critères. Cette idée de restriction du lieu d'apparition des enfants fait appel à l'analogie (même si très grossière) avec la méthode de gradient qui, à chaque pas, fait avancer la recherche dans le sens de l'amélioration de l'objectif (notre objectif étant la recherche de l'ensemble non-dominé).

Une forme particulière du croisement BLX (voir section 2.2.2), avec $\phi = 0.5$, a ainsi été utilisée dans le cas oú il y a eu restriction du choix du partenaire (i.e. losrque le premier parent, choisi suivant la relation de dominance, domine d'autres individus de la population) : ce croisement BLX "biaisé" assure que l'enfant sera plus proche du parent dominant que du dominé, un peu à la manière du croisement *heuristique* proposé par Wright [113] qui générait un enfant plus proche du meilleur parent. La Figure 6.2 illustre l'application de cet opérateur, qui s'écrit formellement, en supposant que $(x_i)_{i\in[1,n]}$ est le parent dominant et $(y_i)_{i\in[1,n]}$ le parent dominé :

$$\left((x_i)_{i\in[1,n]},(y_i)_{i\in[1,n]}\right) \longrightarrow \left(\alpha_i x_i + (1-\alpha_i)y_i\right)_{i\in[1,n]},\ \alpha_i = U(0.5,1.5)$$

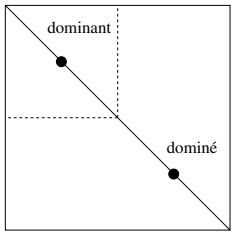


FIG. 6.2 – *L'enfant produit lors de l'application du BLX peut être partout dans le grand carrée, tandis que le résultat du BLX biaisé se trouverait seulement dans le petit carrée séparé par la ligne pointillée*

Type de croisement	Nombre de générations sur 21 essais	ZDT1	ZDT2	ZDT3
aléatoire BLX	minimal	94	106	97
	moyen	**98**	**116**	**110**
	maximal	103	132	133
élitiste BLX biaisé	minimal	83	97	82
	moyen	**90**	**106**	**101**
	maximal	99	120	131

TAB. 6.1 – *Réduction du coût de calcul environ de 8% est constaté lors de l'application du croisement élitiste*

6.1.3 Le croisement élitiste

Dans tout ce qui suit, *croisement élitiste* fera référence au croisement dans lequel un individu non-dominé est systématiquement croisé avec un des individus qu'il domine, ce dernier est choisi uniformément de la liste de tous les dominés par le premier (voir section 6.1.1), et dans lequel un seul enfant est généré, plus proche du parent dominant que du parent dominé, comme introduit section 6.1.2.

6.2 Résultats expérimentaux

Le croisement élitiste défini dans la section précédente a été d'abord testé sur les problèmes ZDT1-ZDT3. La comparaison de la vitesse d'avancement des générations lors des essais avec et sans utilisation du croisement élitiste faits pour la même population initiale montre clairement l'avantage que ce croisement peut apporter (voir, par exemple, les figures 6.3, 6.4, 6.5).

Cet avantage est accumulé pendant que le croisement élitiste se produit relativement souvent. Ensuite, notamment pendant que la population est déjà mono-frontale, cet avantage est préservé. En effet, à cette étape finale (dans chacune des trois figures, 6.3, 6.4, 6.5, elle correspond au passage d'avant-dernier au dernier graphe), il n'y a pas de différence dans le comportement des populations obtenues avec et sans la restriction du choix de partenaires. Alors, le processus de l'uniformisation de la distribution de la population le long du front (éventuellement, accompagné d'un avancement relativement "petit") finalisant l'évolution *se termine* aussi plus tôt, si le croisement élitiste a été précédemment appliqué. Le moment de la *terminaison* de ce stade final peut être défini notamment à travers la "mesure du désespoir" introduite dans la section 5.1.2.

Le fait que l'avantage accumulé lors de l'application du croisement élitiste est préservé à la fin de l'évolution nous donne de bonnes raisons de penser à combiner dans la pratique l'utilisation du critère d'arrêt décrit dans le chapitre précédent et de ce croisement. Mais nous tenons à souligner que ceci ne revient pas à employer la mesure du désespoir pour évaluer l'accélération apportée par le croisement élitiste, car telle évaluation risquerait d'être bruitée par le choix des paramètres du critère d'arrêt. Le tableau 6.2 sert plutôt de l'illustration du fait que le coût de calcul nécessaire pour l'obtention des solutions d'une certaine qualité peut quelque fois être

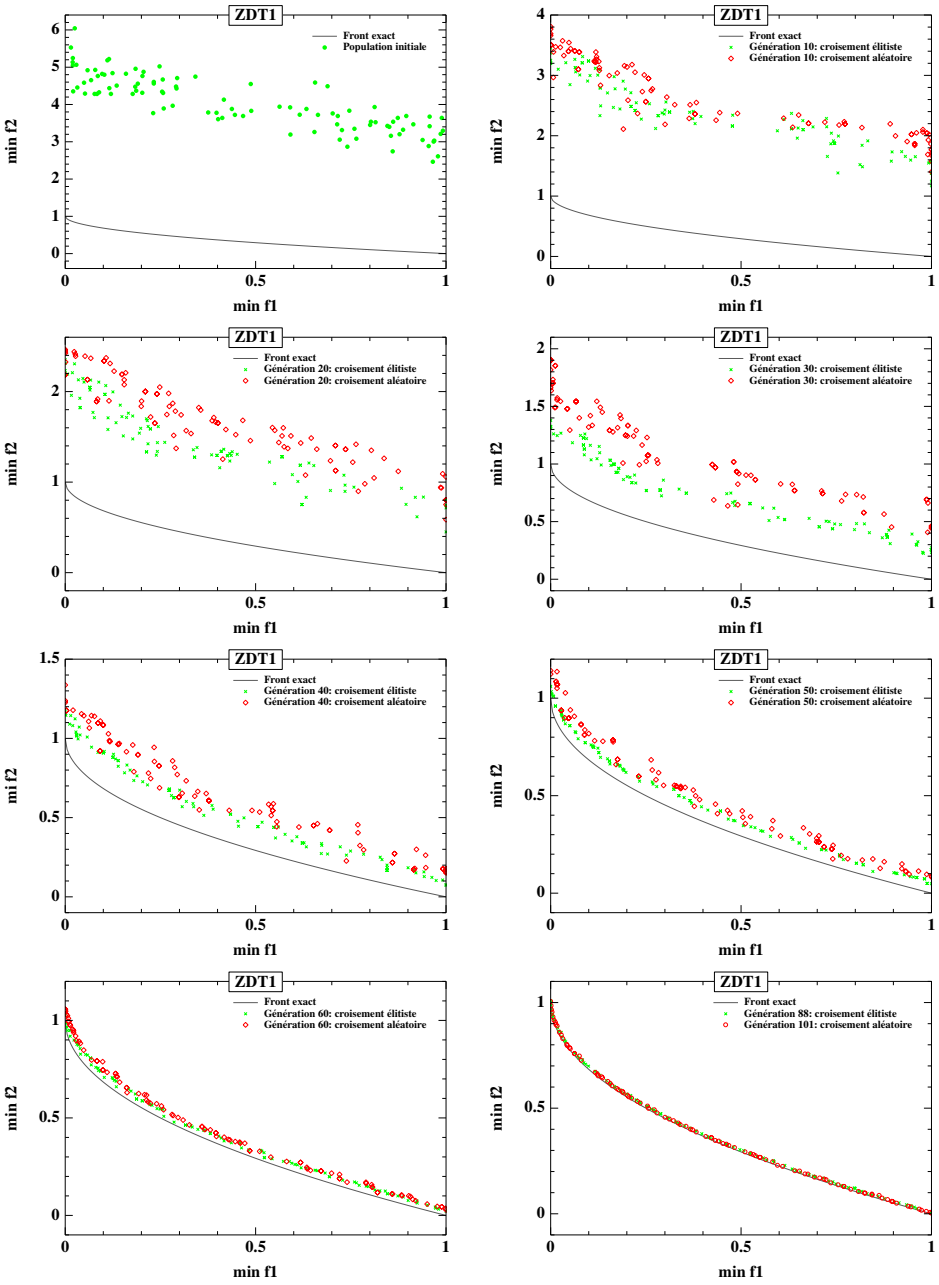


FIG. 6.3 – *Dynamique des populations de NSGA-II en cours de résolution du problème ZDT1 avec l'utilisation du croisement élitiste (population verte) et sans (population rouge). 88 itérations "vertes" et 101 itérations "rouges" ont été effectuées selon le critère d'arrêt introduit dans le chapitre précédent*

réduit lors de la combinaison du croisement élitiste avec le critère d'arrêt "par désespoir".

Ce tableau représente les statistiques de base calculées pour le nombre d'itérations faites lors de 21 essai de NSGA-II sur les mêmes populations initiales pour chacun des trois problèmes test. Nous pouvons constater le gain d'environ 8% pour chaque test. Il est clair que pour ces tests, où

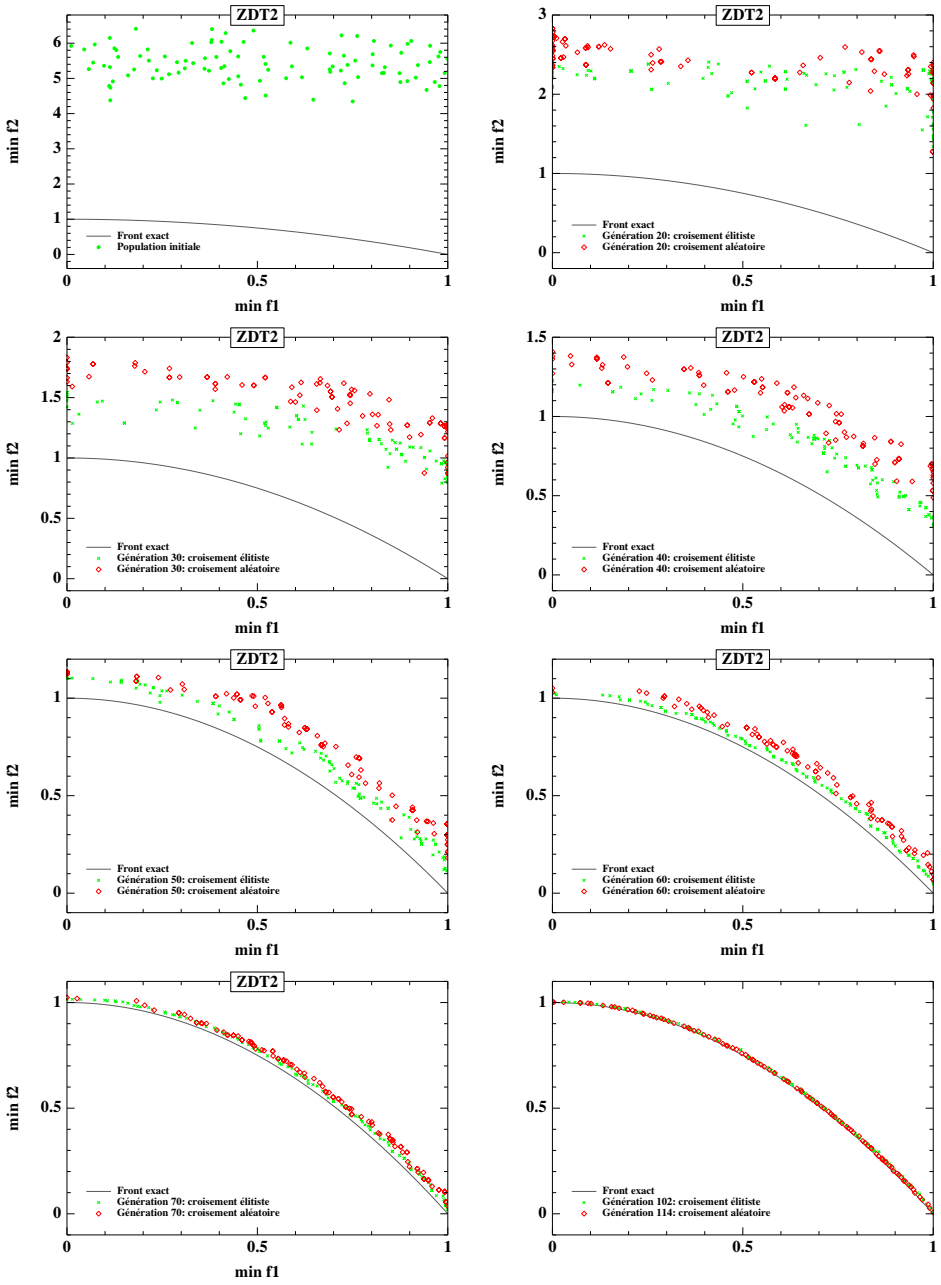


FIG. 6.4 – *Dynamique des populations de NSGA-II en cours de résolution du problème ZDT2 avec l'utilisation du croisement élitiste (population verte) et sans (population rouge). 102 itérations "vertes" et 114 itérations "rouges" ont été effectuées selon le critère d'arrêt introduit dans le chapitre précédent.*

le coût du calcul des fonctions objectif est très petit, ce gain est pratiquement négligeable. Mais si les résultats similaires étaient obtenus pour un problème réel, où le coût de chaque évaluation est relativement élevé, ce gain serait considérable compte tenu que, en plus, plusieurs essais sont toujours faits lors de l'utilisation d'un algorithme stochastique.

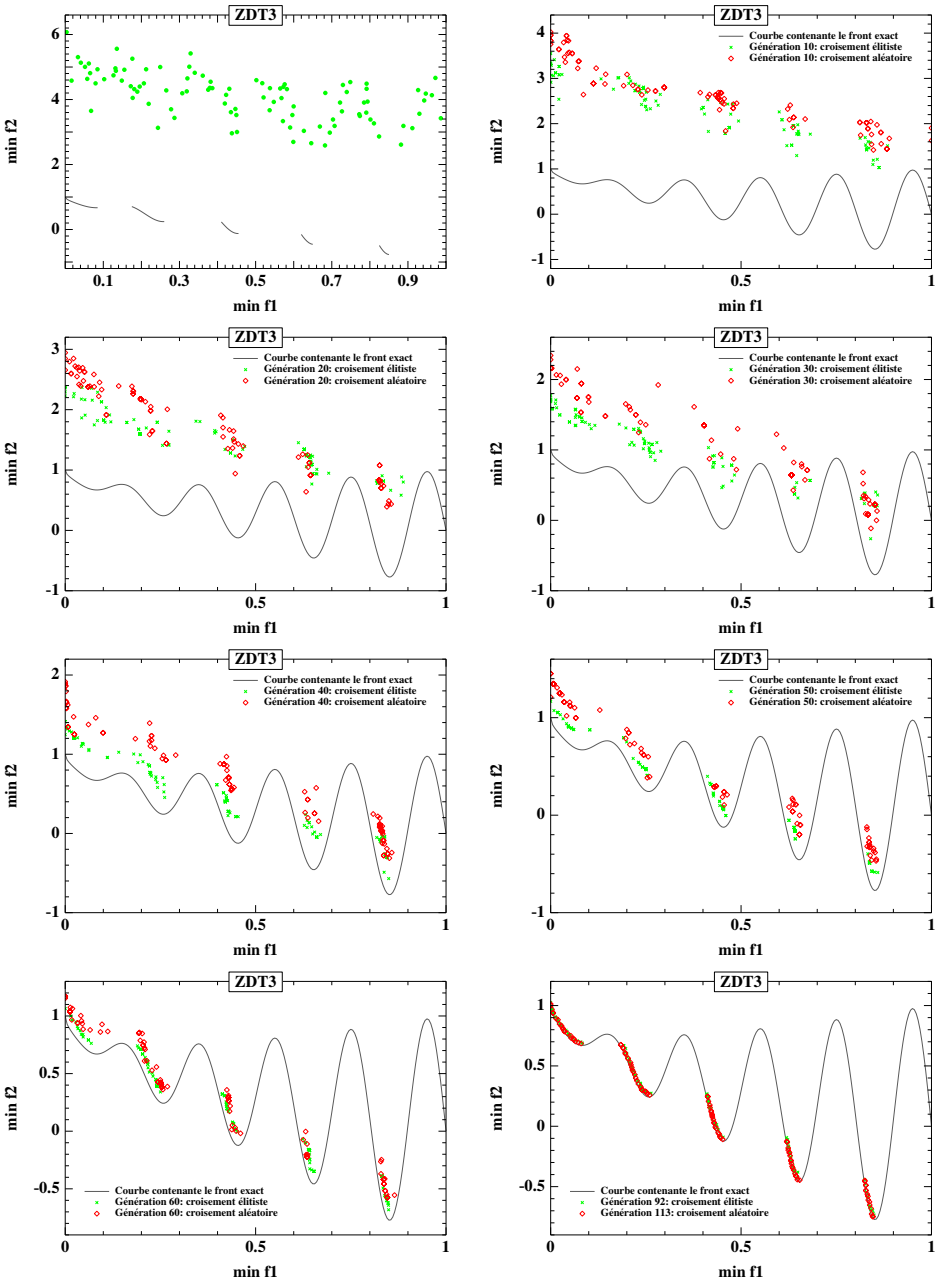


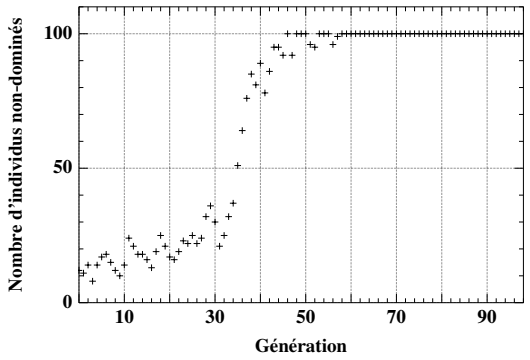
FIG. 6.5 – *Dynamique des populations de NSGA-II en cours de résolution du problème ZDT3 avec l'utilisation du croisement élitiste (population verte) et sans (population rouge). 92 itérations "vertes" et 113 itérations "rouges" ont été effectuées selon le critère d'arrêt introduit dans le chapitre précédent*

6.3 Discussion et directions de recherche

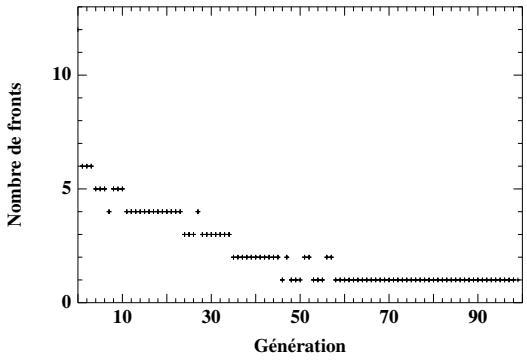
L'étude présentée dans ce chapitre permet de constater l'utilité du croisement élitiste – restriction du choix des partenaires pour le croisement de façon que chaque individu non-dominé

soit croisé avec un des individus qu'il domine (tant qu'il y en a dans la population), et choix de l'enfant biaisé vers le parent dominant. En effet, lors de la comparaison des générations correspondantes en cours de l'évolution, la tendance de la population "pratiquant" le croisement élitiste de "surpasser" la population se limitant par les "partenariats aléatoires" a été expérimentalement mise en évidence, même si des tests plus intensifs seraient nécessaires pour évaluer le champ d'application de ce croisement (voir aussi les résultats sur l'application présentés dans la section 7.4.3).

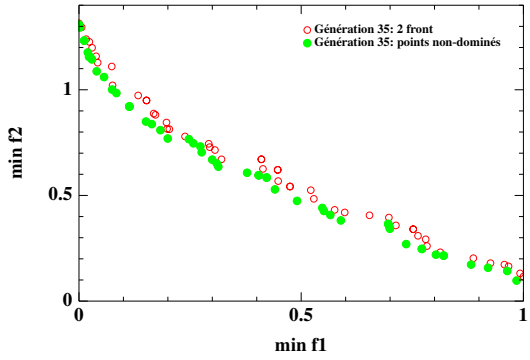
Il est cependant important de se rendre compte que l'application "excessive" du croisement proposé peut provoquer la réduction trop rapide de la diversité des solutions dans le sens indiqué par relation de dominance. Ceci crée le danger supplémentaire de "rater" des régions de la surface de Pareto plus difficilement retrouvables, notamment, ces extrémités. De plus, il est intuitive que l'efficacité du croisement élitiste doit diminuer avec le temps tout simplement par ce qu'il reste de moins en moins de place pour avancer. Ces deux remarques font naturellement penser à implémenter un contrôle adaptatif de la probabilité de l'application du croisement élitiste.



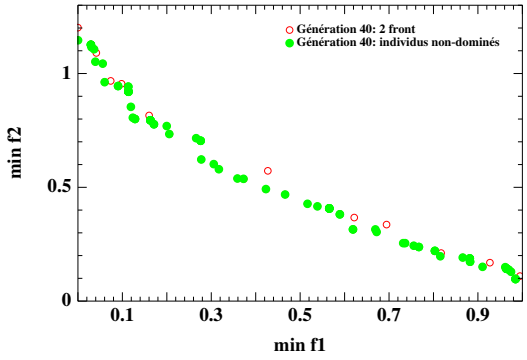
(a) Nombre de non-dominés en cours de l'évolution



(b) Nombre de fronts en cours de l'évolution



(c) 51 non-dominés en 35ème génération



(d) 90 non-dominés en 40ème génération

FIG. 6.6 – *La moitié de la population rejoint l'ensemble non-dominé environ au même moment où elle devient "bi-frontale"; la taille de population est égale à 100*

Mais, d'autre part, NSGA-II préserve l'uniformité de la distribution entre les individus de chaque rang. Par suite, au moment, où le nombre d'individus non-dominés excède la moitié de

la taille de la population, il est très probable que chacun d'eux n'a pas d'individus dominés (cf. figure 6.6(b)). En effet, lors de tous les tests il a été remarqué que le moment, où la moitié de la population rejoint l'ensemble non-dominé, correspond au moment, où elle devient "bi-frontale" (une illustration de ce fait est donnée dans la figure 6.6(a), (b)). De plus, il a été constaté qu'après ce moment la population devient assez vite mono-frontale (cf. figure 6.6(a)).

En même temps, grâce au même mécanisme de la maintenance de la diversité, avant que la moitié de la population devient non-dominée, le pourcentage des croisements élitiste est pratiquement défini par le pourcentage d'individus non-dominés. Ceci dit, d'un côté, le contrôle et la réduction rapide de la probabilité du croisement élitiste lorsqu'il menace la diversité sont "prévus" par le mécanisme de NSGA-II. D'autre côté, cette probabilité peut être trop basse au début de l'évolution quand il y a encore peu d'individus non-dominés. Une des possibilités d'augmenter l'efficacité du croisement lors de la recherche des optima de Pareto consiste probablement à croiser chaque individu non-dominé avec plusieurs dominés en première étape de l'évolution.

Notons que l'utilisation des métriques dynamiques (cf. section 4.2.3) permettrait de rendre l'analyse de l'efficacité de l'opérateur proposé plus stricte. Le choix et la mise en œuvre de telles métriques est prioritaire sur la liste des prolongements du travail présenté dans cette section.

L'utilisation de l'opérateur proposé semble particulièrement appropriée lors des applications industrielles, où le temps de calcul est fortement limité et l'arrêt de l'algorithme est possible tant que la population a encore une réserve de l'avancement.

Troisième partie

Application des AEMO à deux problèmes de l'industrie automobile

Introduction

Dans les dernières décénies les développeurs d'automobiles se sont retrouvés confrontés à une forte augmentation des volumes de données à analyser et de la quantité d'informations à prendre en compte, ainsi qu'à l'accroissement du nombre de paramètres de décision. Par exemple, les moteurs modernes contiennent à présent un grand nombre de composantes contrôlées électroniquement, ainsi donc que des unités de contrôle électroniques puissantes. De tels systèmes assurent la conformité avec des règles variées imposées par une législation stricte et permettent de satisfaire les attentes des consommateurs. Au niveau du développement, cependant, cela implique l'utilisation de ressources de plus en plus importantes pour déterminer une combinaison optimale des paramètres de conception. Cette tendance a très vite rendu les approches traditionnelles au travail de conception (essais-erreurs avec expertise ad hoc) insuffisantes pour explorer l'espace de recherche efficacement. Et aujourd'hui, il est impossible d'imaginer un processus de développement dans l'industrie automobile sans l'utilisation de méthodes statistiques, qui permettent de réduire fortement le coût et le temps nécessaires pour concevoir un nouveau produit.

La résolution d'un problème de conception dans le développement complexe moderne peut être décomposé en trois étapes : planification d'expériences, modélisation statistique des indicateurs de qualité qui dépendent des paramètres de décision et optimisation dont l'objectif est d'améliorer la qualité du produit développé. C'est l'étape de l'optimisation lors de la résolution de deux différents problèmes de conception de l'industrie automobile qui fait l'objet des deux derniers chapitres de cette thèse.

Chapitre 7

L'optimisation d'un pare-choc de voiture

Cette application a été étudiée dans le cadre du projet européen INGENET (NETworked INdustrial Design and Control Applications using GEnetic Algorithms and Evolution Strategies, `http://ingenet.ulpgc.es/`), dont l'objectif était de rassembler les industriels et les chercheurs afin d'évaluer les performances des techniques évolutionnaires récentes sur des problèmes test industriels.

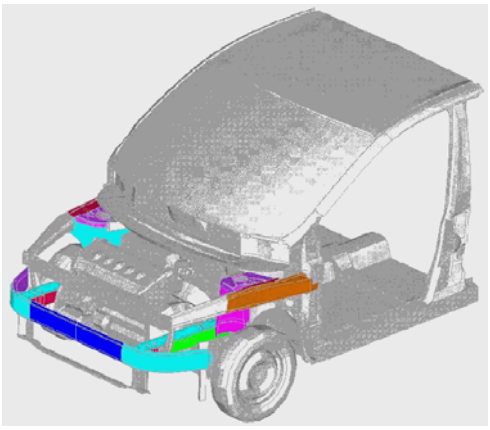


FIG. 7.1 – *Pare-choc avant de Fiat Multipla*

Le problème proposé par le Centre de Recherche de Fiat consiste à trouver la géométrie optimale d'un pare-choc de voiture sur l'exemple du modèle "Multipla" (figure 7.1). Les objectifs principaux sont la minimisation du poids et la maximisation de l'énergie interne absorbée lors d'un crash; les autres critères d'optimisation sont liés au comportement acoustique et statique de la voiture.

L'application des Algorithmes d'Évolution à la résolution de ce problème serait impossible sans utiliser des fonctions d'approximation pour calculer les réponses du système mécanique étudié. Ces fonctions, dites "émulateurs" permettent un calcul de la réponse du pare-choc des milliers de fois plus rapide que l'utilisation directe des simulations basées sur la Méthode des

Éléments Finis. Les émulateurs ont été construits dans le cadre du projet européen $(CE)^2$: Computer Experiments for Concurrent Engineering (`http://www.warwick.ac.uk/ wrsar/ce2.html`).

7.1 Modélisation du pare-choc et critères de l'optimisation

La paramétrisation du pare-choc considérée dans ce travail est définie par 13 variables de décision présentées dans la figure 7.2.

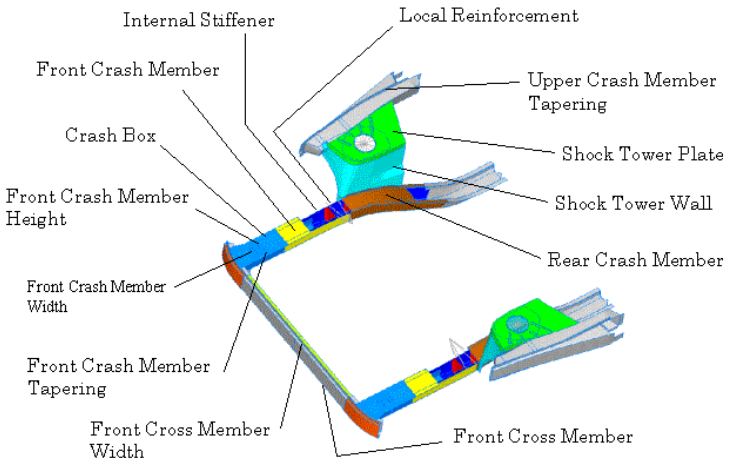


FIG. 7.2 – *Paramétrisation du pare-choc*

Les paramètres peuvent être divisés en deux catégories : variables d'épaisseur et variables de forme. Les bornes des paramètres du premier groupe sont définies par la technologie du feuilletage mécanique. Les bornes pour les variables de forme ont été choisies de façon à atteindre une variation significative dans la section principale.

Le problème étudié implique neuf fonctions objectif, dont chacune est calculée par une des trois simulations mécaniques différentes, plus le critère de masse. Plus précisément, les objectifs de l'optimisation et les domaines mécaniques correspondants sont les suivants :

- **masse :** calculée directement en fonction des valeurs de 13 paramètres utilisant les données mécaniques des matériaux utilisés.
- **crash :** l'objectif correspondant est la maximisation de l'énergie interne absorbée par le pare-choc lors de l'impact. Notons que cette simulation est de très loin la plus coûteuse.
- **acoustique :** les rigidités dynamiques moyenne et minimale dans la tour de choc et dans les points d'attache du moteur sont deux objectifs de plus à maximiser.
- **statique :** la simulation standard de deux différentes réponses de la résistance statique et quatre différentes réponses de la rigidité statique complètent la liste des objectifs.

Du point de vue de l'ingénieur, tous les objectifs n'ont pas la même importance. En particulier, dans cette application, le poids et l'énergie interne sont les critères nettement plus significatifs.

Par ailleurs, certains critères dépendent seulement d'une partie de variables de décision. La figure 7.3 montre ces relations de dépendance. La masse et les réponses statiques dépendent de 13 variables, alors que le crash et la simulation acoustique ne dépendent chacun que de 10 variables, mais pas les mêmes.

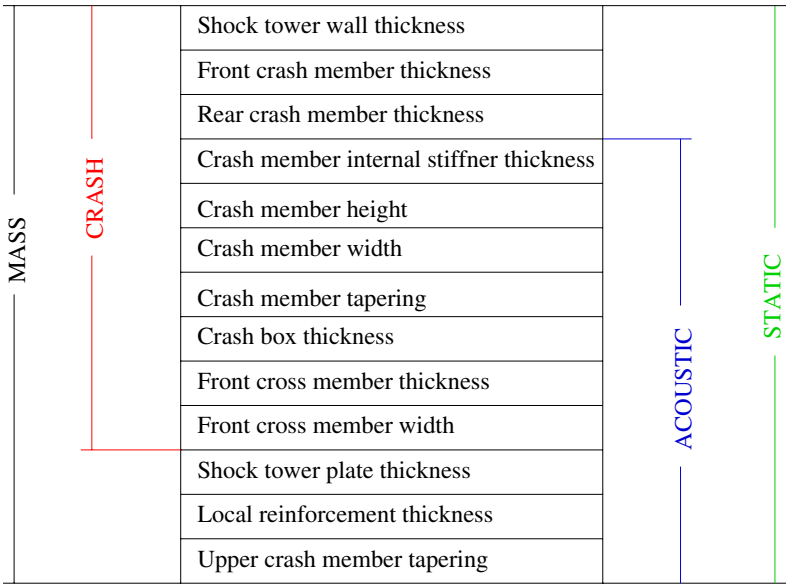


FIG. 7.3 – *Dépendances des domaines mécaniques considérés par rapport aux paramètres de conception*

7.2 Évaluation approchée

Les codes de simulation basés sur l'analyse par la Méthode des Éléments Finis avec la description de la géométrie de la coquille du pare-choc ont été préparés au CRF afin d'explorer son comportement mécanique pour chacun des trois domaines mécaniques considérés. Mais le temps CPU nécessaire pour calculer la valeur d'une réponse est de 6 heures pour le crash, 1.5 heures pour l'acoustique et 0.5 heures pour le comportement statique. Par conséquent, il n'est pas envisageable d'utiliser directement les calculs EF au sein de l'optimisation évolutionnaire.

Une stratégie pour réduire le coût de l'évaluation, largement utilisée dans le cadre de l'optimisation par Algorithmes Évolutionnaires, mais également couplée avec des méthodes plus classiques, consiste à utiliser simultanément des approximations des fonctions objectif avec leur calcul exact (voir par exemple [27]). Concrètement, alors que la plupart des individus sont évalués à l'aide d'approximations des vraies fonctions, l'évaluation exacte d'une (petite) partie de la population (en général les meilleurs individus) est faite régulièrement – et les approximations sont remises à jour à partir de ces nouveaux points exacts. Cela permet d'éviter un biais trop large dû à l'erreur d'approximation.

Cependant, une telle technique est inapplicable quand le coût du calcul exact est beaucoup trop élevé, ce qui est le cas de l'application étudiée ici, et nous sommes donc contraints d'utiliser les approximations des critères pendant tout le processus d'optimisation.

7.2.1 Fonctions d'approximation

La méthode utilisée par les ingénieurs de Fiat pour construire les "émulateurs" des simulateurs est basée sur le *krigeage* des statistiques spatiales [11].

Nous supposons que toute réponse y du système se représente sous forme suivante

$$y \quad = \quad m(\varphi,x) \quad + \quad V(\rho,\theta,\xi),$$

où $x = (x_1,...,x_n)$ est l'ensemble de paramètres de conception desquels dépend y, m est un modèle de régression avec des paramètres φ, et V est un processus stochastique de covariance inconnue et qui dépend des paramètres ρ, θ et ξ. Les paramètres ρ etθ sont estimés à l'aide du principe du Maximum de Vraisemblance. Le paramètre ξ est à ajuster en fonction du niveau de bruit supposé des points "expérimentaux". Dans le cas présent, oú les expériences sont des simulations numériques, et donc sans erreur de mesure, le paramètre ξ est fixé à 0, et le modèle donne alors une interpolation exacte des données expérimentales.

Pour trouver les paramètres des émulateurs, les valeurs de y doivent être "mesurées" (calculées en utilisant les simulateurs) pour un ensemble des points (configurations) x. Le choix de cet ensemble est une étape importante, car un protocole mal conçu peut ruiner toute tentative d'interprétation des données.

7.2.2 Plan d'expérience

Diverses techniques de planification d'expériences ont été adoptées dans le domaine de l'automobile. Le choix de l'une d'entre elles dans chaque étude particulière dépend du type de données qui devront être tirées de l'expérience, du type du modèle qui sera nécessaire pour la meilleure représentation de ces données, du degré de précision exigé et de nombre de tests réalisables.

Dans le cadre de cette application, la méthode de McKay et al. [71] de la planification d'expérience pour les codes déterministes a été employée. Ces auteurs ont introduit l'*échantillonnage d'hypercube Latin* (Latin Hypercube Sampling, LHS), une extension de l'échantillonnage stratifié qui assure que chacune des variables d'entrée (facteurs contrôlés) aura toutes les portions de son rang présentes dans le dispositif. Les hypercubes Latins ne sont pas coûteux à générer et leur utilisation est appropriée dans les cas, où le nombre des variables d'entrée est grand.

En normalisant les facteurs contrôlés de façon que toutes leurs valeurs soient dans l'intervalle $[-0.5,0.5]$, toutes les combinaisons possibles de d facteurs appartiendront à l'hypercube $[-0.5,0.5]^d$. Pour une expérience de n essais , un LHS est obtenu en divisant l'intervalle $[-0.5,0.5]$ en n parties égales pour chacun de d facteur et en "randomisant" les points de leurs séparations. Soit $z = 0,1,...,n-1$. Alors,

$$s_j = \frac{\pi_j(z) - 1/2}{n}, \quad j = 1,...,d$$

est la $j$ème colonne de l'échantillon S, où $\pi_1,...,\pi_d$ sont des permutations aléatoires indépendantes de z. Cet algorithme place les points de mesure dans le centre des cellules du maillage choisies de façon aléatoire.

7.3 Le problème d'optimisation

7.3.1 Solution de référence

L'étude du problème d'optimisation décrit dans la section 7.1 peut être vu comme la continuation du travail commencé dans le cadre du projet européen $(CE)^2$ qui portait principalement sur la construction des émulateurs. Un des buts ultimes de $(CE)^2$ était de permettre l'application des techniques d'optimisation multi-objectif performantes, comme les Algorithmes Évolutionnaires, qui exigent un coût de calcul des fonctions objectif suffisamment réduit.

Le travail du projet $(CE)^2$ consistait à trouver un modèle statistique approprié pour construire des fonctions d'approximation permettant le calcul rapide des réponses, et à tester cette approximation en terme de précision. L'optimisation, n'étant pas l'objet principal de cette étude, n'a pas été faite de manière approfondie, et n'a servi qu'à valider l'ensemble du travail réalisé durant le projet, se contentant d'obtenir une configuration relativement performante utilisant les $(CE)^2$-émulateurs [5]. Un algorithme génétique multi-objectif non-élitiste a été utilisé pour résoudre le problème d'optimisation de 10 objectifs. Un ensemble de Pareto approché a été trouvé, dont une solution a été choisie pour servir du point de départ pour une méthode locale. Notons que, dans [5], il n'est pas précisé comment la solution qui a servi du point initial à la méthode locale avait été choisi parmi l'ensemble de solutions non-dominées 10-D. La méthode locale qui a ensuite été appliquée était une méthode mono-objectif, basée sur une agrégation pondérée des 10 objectifs, et le choix des poids accordait une très forte priorité à l'objectif de masse. Finalement, une solution a été retenue et présentée comme très performante. C'était, avant le début de ce travail, la seule solution de référence pour ce problème. Elle sera désignée dans la suite comme la $(CE)^2$-solution.

A la différence du $(CE)^2$, les études réalisées dans le cadre d'INGENET sont consacrées à la recherche de méthodes d'optimisation efficaces. En particulier, dans le cas de l'optimisation multi-objectif, le but est de trouver une bonne approximation de l'ensemble de Pareto en un temps de calcul acceptable dans le contexte du travail de développement industriel.

De ce point de vue, l'algorithme d'optimisation utilisé par $(CE)^2$ avait au moins deux défauts. Tout d'abord, lors des études récentes sur les AEMO, il a été établi l'importance de l'élitisme pour parvenir à trouver une bonne approximation du front de Pareto (voir la section 3.4). Cela sera donc notre premier critère pour le choix de l'algorithme.

Deuxièmement, notons que la solution de référence a été trouvée en appliquant une méthode d'optimisation globale au problème à dix fonctions objectif. Ceci appelle deux remarques. D'une part, pour espérer d'obtenir un ensemble des points qui approcherait un front de Pareto dans un espace de dimension aussi grande de façon satisfaisante, nous devrions travailler avec une population très grande, ce qui se traduirait par un temps de convergence beaucoup trop élevé. D'autre part, la question aussi importante que difficile, qui se poserait alors, serait celle de la représentation des résultats. La prise de décision à partir d'une surface de Pareto 10-D apparaît sinon irréaliste, du moins inutilement complexe.

7.3.2 Optimisation sous contraintes

Par conséquent, compte tenu du fait que, lors de la conception du pare-choc avant, les objectifs de masse et de crash sont industriellement beaucoup plus importants de ceux liés au comportement acoustique et statique de la voiture, il a été décidé de transformer le problème

Objectif	Réponse		Valeur de contrainte		Solution optimale par $(CE)^2$
max	*Mean dynamic stiffness*	$>$	*35000*	$>$	33735.3 N/mm
max	*Min dynamic stiffness*	$>$	*4100*	$>$	4001.2 N/mm
min	*Max equiv. stress for bump*	$<$	*360*	$<$	367.4 Mpa
min	*Shock towers displacements for bump*	$<$	*2.3*	$<$	2.353 mm
min	*Torsion for turn*	$<$	*0.02*	$=$	0.02 degrees
min	*Max equiv. stress for rebound-max brake turn*	$<$	*203*	$<$	206.5 Mpa
max	*Shock towers stiffness*	$>$	*255*	$>$	249.6 Kg/mm
max	*Torsional stiffness*	$>$	*250000*	$>$	243204.4 Kgm/rad

TAB. 7.1 – *Contraintes sur 8 objectifs secondaires*

initial en un problème d'optimisation à deux objectifs sous contraintes. Autrement dit, les huit objectifs secondaires seront vus comme des contraintes pour l'optimisation.

Le choix des valeurs limites de ces contraintes s'est basé sur la $(CE)^2$–solution : pour chaque contrainte, la valeur retenue a été un peu plus contrainte que la valeur correspondante trouvée lors du $(CE)^2$ (cf. tableau 7.1) – ainsi si une solution est trouvée respectant ces contraintes, nous serons assurés qu'elle est plus performante sur ces objectifs-là que la $(CE)^2$–solution.

Une telle reformulation du problème initial facilite beaucoup l'interprétation des résultats de l'optimisation, et réduit l'effort de la prise de décision sur la configuration finale des paramètres de conception.

En plus des contraintes de l'optimisation, la région d'intérêt dans l'espace des objectifs est limitée par une borne supérieure de 36 kg pour la masse et de 30000 Joules pour l'énergie interne.

7.4 Méthode de résolution et résultats

7.4.1 Prise en compte des contraintes

L'application de *Infeasibility Objective Approach* (IOA), (présenté à la fin de la section 3.5.3) nous a paru parfaitement appropriée dans ce cas. En effet, dans le cadre de cette étude, les contraintes sont toutes équivalentes, en ce sens qu'il n'y en a pas une, dont l'influence sur l'optimisation nous intéresserait plus que toutes les autres. D'autre part, les contraintes sont suffisamment nombreuses et donc l'approche de Fonseca et Fleming (brièvement présentée dans la section 3.5.3), par exemple, impliquerait un coût de ranking dans l'espace infaisable qui n'est pas justifié dans ce cas. Une autre mesure scalaire d'infaisabilité aurait pu également être employée. La mesure suggérée par Wright et Loosemore [114] a été choisie pour sa simplicité. Rappelons qu'il s'agit de la somme des violations des contraintes mises à l'échelle :

$$\mathcal{C}(x) = \sum_{j=1}^{L} \delta_j(x).$$

Nous nous sommes, alors, retrouvés confrontés au **problème d'optimisation à trois objectifs suivant :**

- **minimiser la masse**,
- **maximiser l'énergie interne absorbée**,
- **minimiser l'objectif "d'infaisabilité"**.

L'algorithme NSGA-II (section 3.4.3), implanté pour la représentation réelle, a été utilisé pour le résoudre, et les premiers résultats ont été publiés dans [90].

Il est très important cependant de noter la différence entre le principe du ranking de la partie infaisable de la population employé ici (pour obtenir les résultats présentés dans la section suivante 7.4.4) et celui utilisé dans [90]. L'approche utilisée précédemment (qui semble maintenant assez naïve), en plus de l'objectif d'infaisabilité, prenait en compte les objectifs du problème lors des comparaisons entre individus infaisables. C'est-à-dire que le ranking habituel de NSGA-II, appliqué au problème de l'optimisation de trois objectifs cités ci-dessus, était utilisé aussi pour attribuer la performance aux individus infaisables. La seule (mais importante) nuance est que tous les individus faisables avaient la priorité par rapport aux infaisables afin de "retenir" la population dans la région faisable, une fois cette région est retrouvée.

Dans le présent travail, les individus infaisables sont ordonnés uniquement selon la mesure d'infaisabilité, ce qui permet d'attirer plus rapidement la population vers la région faisable. Dans le cadre du problème du pare-choc (dont les contraintes ne sont pas très dures à satisfaire), ceci revient à réduire considérablement le coût de l'obtention des résultats d'une certaine qualité. (Notons que la taille de population 700 et le nombre de générations 400 [90] ont pu être remplacé ici par la population de 100 individus évoluant pendant 150-200 générations, comme nous le verrons dans la section suivante.) Mais si nous étions confrontés à un problème pour lequel la région faisable soit difficile à trouver, le ranking employé précédemment pourrait tout simplement échouer à trouver le moindre individu faisable. Enfin, un autre avantage important des comparaisons basées uniquement sur le degré de violation des contraintes est qu'il n'est plus alors nécessaire de calculer les valeurs des objectifs pour les individus non faisables.

7.4.2 Paramètres de l'algorithme évolutionnaire

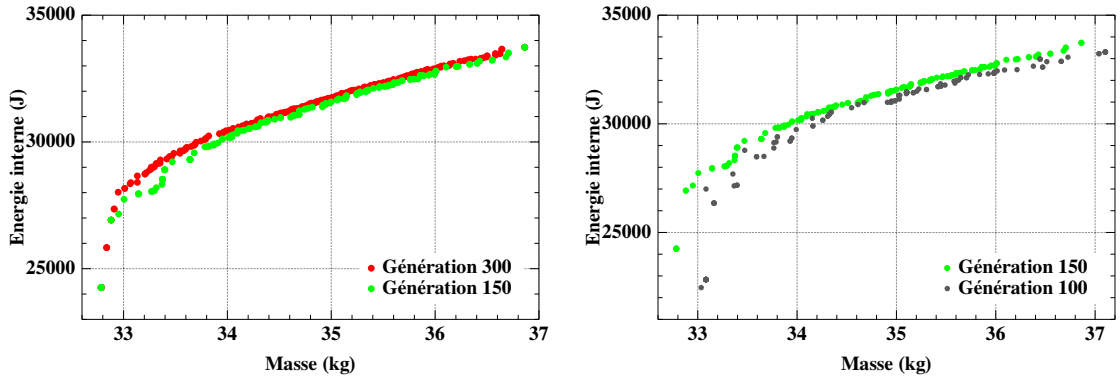
Chaque individu de la population de notre algorithme d'évolution est un vecteur de 13 composantes dont chacune correspond à un des paramètres de conception énumérés dans la figure 7.2.

Le croisement intermédiaire BLX (section 2.2.2) a été appliqué avec la probabilité 0.9 et la mutation uniforme (section 2.2.2) avec la probabilité 0.05. A l'étape de sélection, nous avons utilisé les tournois de taille 2 (section 2.3.1).

Nombre de générations vs taille de population

Nous avons commencé par faire 11 essais avec des populations de 100 individus évoluant pendant 300 générations. Notons tout d'abord que, lors de tous les essais, la population devient faisable aux alentour de la génération 60-70. L'analyse des résultats obtenus lors des générations intermédiaires a montré que, dans la pratique, avec cette taille de population, 150 générations suffisent. C'est-à-dire, l'amélioration observée lors de 250 dernières générations est négligeable et ne justifie pas le coût de calcul supplémentaire (voir la figure 7.4(a)).

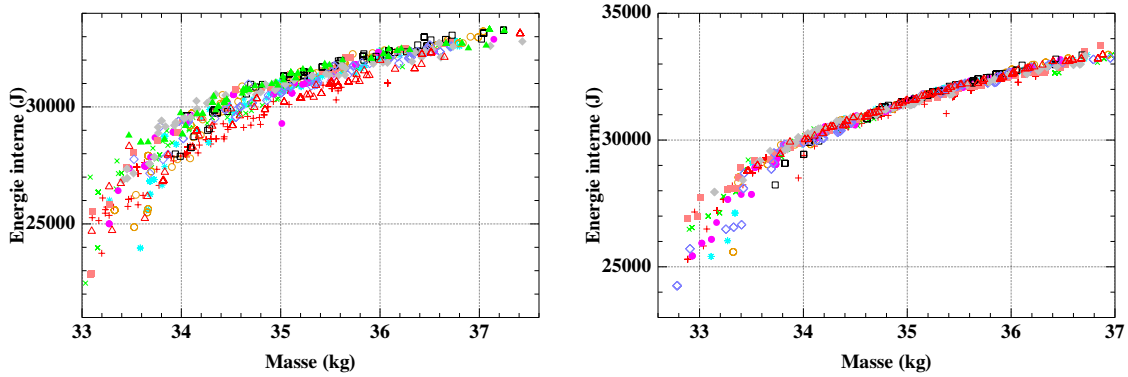
En effet, compte tenu de l'erreur de l'approximation des émulateurs (6.73% pour l'énergie interne [5]!), la différence entre les ensembles non-dominés obtenus lors de 11 essais après 150



(a) (b)

FIG. 7.4 – *Illustration du choix du nombre de génération :* **(a)** *les ensembles non-dominés obtenus lors de 11 essais après 150 et 300 générations (la taille de population est fixée à 100) sont pratiquement confondus dans la région* $\{$*la masse* $< 36kg$ *et l'énergie interne* $> 30000J\}$*;* **(b)** *la différence non négligeable entre les générations 100 et 150 est observée*

et après 300 générations est trop petite pour être significative. De plus, dans la pratique, un raffinement des résultats de l'optimisation globale par l'application ultérieure d'une méthode locale est souvent employé. De ce point de vue, la poursuite des itérations après la 150ème n'ajoute aucune information consistante qui pourrait influencer le choix de quelques (ou d'une seule) solutions destinées à servir des points initiaux pour la méthode locale.



(a) 11 essais après 100 générations (b) 11 essais après 150 générations

FIG. 7.5 – *La qualité des solutions à la 150ème génération varie beaucoup moins d'un essai à l'autre que la qualité des solutions à la 100ème génération*

De plus, même si du point de vue de la précision des émulateurs et compte tenu de la recherche locale ultérieure, nous pourrions encore tenter de diminuer le nombre de générations jusqu'à 100 (figure 7.4(b)), l'instabilité des résultats obtenus après 100 générations d'un essai à l'autre nous en dissuaderait (voir figure 7.5).

Les essais effectués avec la taille de population égale à 200 ne permettent pas de constater une amélioration des résultats qui justifierait le fait de doubler la taille de population.

Compte tenu de ces observations, les résultats d'optimisation obtenus avec la taille de population 100 et le nombre de générations 150 sont présentés dans deux sections suivantes et servent à illustrer l'efficacité de l'algorithme utilisé lors de la résolution du problème de l'optimisation du pare-choc.

7.4.3 Applicabilité du croisement élitiste

La nouvelle étude du problème de l'optimisation du pare-choc, motivée, tout d'abord, par la mise à l'épreuve de l'efficacité d'une comparaison des individus infaisables a priori plus performante (section 7.4.1), nous a en plus donné une opportunité parfaite d'essayer d'adapter à l'optimisation sous contraintes l'opérateur du *croisement élitiste*, introduit dans le chapitre 6.

Rien à changer

Rappelons-nous qu'en absence de contraintes, le croisement élitiste consiste à restreindre le choix des partenaires pour les individus non-dominés de la population au stade de la reproduction. Chaque individu non-dominé ne peut être croisé qu'avec un des individus qu'il domine – tant qu'il y en a dans la population – et de façon que l'enfant résultant est plus proche du parant dominant dans l'espace des objectifs.

D'autre part, selon le principe du ranking employé ici, les individus sont d'abord comparés au sens de leurs valeurs de l'objectif "d'infaisabilité" et, ensuite, les individus faisables (ayant tous cette valeur égale à zéro) sont triés sur la base du principe de dominance dans l'espace des objectifs d'optimisation.

Pour appliquer le croisement élitiste aux individus faisables, il n'y a donc aucune modification à faire. En ce qui concerne la partie infaisable de la population, ce croisement a été appliqué suivant exactement le même principe que pour la partie faisable. C'est-à-dire, il s'est basé sur la dominance au sens de la valeur de l'objectif d'infaisabilité.

La recherche de la région faisable

Pour comparer les résultats obtenus avec et sans application de croisement élitiste, 11 essais ont été faits pour chacun des deux cas utilisant les mêmes 11 populations initiales.

Il a été observé, tout d'abord, une légère mais robuste (constatée sur tous les essais) accélération à l'étape de la recherche de la région faisable. L'algorithme sans restriction du choix des partenaires retrouvait le premier faisable entre la 36ème et la 52ème générations (minimum et maximum parmi 11 essais) tandis que la méthode utilisant le croisement élitiste le retrouvait entre la 30ème et la 43ème générations. Notons que pendant que toute la population est infaisable, très peu de croisements élitistes se produisent car, dans la majorité des cas, il n'y a qu'un seul individu (probablement, présent dans la population en plusieurs exemplaires) qui constitue "l'ensemble non-dominé" résultant du ranking par le degré d'infaisabilité. Donc, même si l'accélération observée n'est pas très grande, le fait qu'elle est constatée lors de chacun de 11 essais effectués représente un début très prometteur.

Ensuite, l'attraction de toute la population vers la région faisable (compris entre le moment, où le premier faisable est trouvé et le moment, où toute la population est constituée d'individus faisables) était régulièrement rendue plus rapide avec l'application du croisement élitiste : de 16 à 20 générations ont été nécessaires avec ce croisement, contre 18 à 30 générations avec le croisement standard.

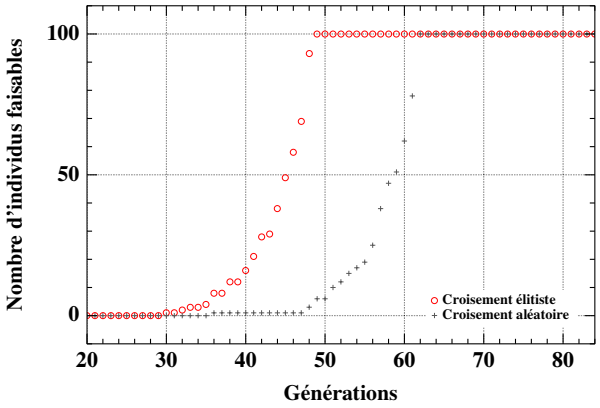


FIG. 7.6 – *La variation du nombre d'individus faisables lors d'un essai effectué avec la taille de la population égale à 100*

La figure 7.6 illustre ces observations: la population pratiquant le croisement élitiste retrouve le premier faisable à la 30ème génération et à la 49ème génération elle appartient toute à la région faisable. Pour la population se contentant des partenariats aléatoires, ces deux moments correspondent à la 36ème et à la 62ème générations. générations.

Sans appliquer le croisement élitiste, la population rentre entièrement dans la région faisable au plus tard (parmi 11 essais) à la 76ème génération tandis que, avec ce croisement, la population devient faisable à la 61ème génération au plus tard.

L'optimisation des objectifs principaux

L'effet produit par l'application du croisement élitiste lors de l'optimisation de deux objectifs principaux est illustré dans la figure 7.7. Ces graphes présentent les solutions non-dominées parmi 11 essais. Ils permettent de constater l'accélération de la convergence quand le croisement élitiste est appliqué. Le graphe dans la figure 7.8 sert également d'illustration de l'avantage apporté par le croisement élitiste. Les résultats obtenus avec ce croisement après la **120**ème génération sont comparés aux solutions trouvées par l'algorithme utilisant seulement le croisement aléatoire après la **150**ème génération. En effet, on constate le même niveau de performance de ces deux ensembles de solutions dans la région $\{Masse < 36 \text{ kg}, Energie\ Interne > 30000 \text{ J}\}$ et meilleure performance de la méthode utilisant le croisement élitiste en dehors de cette région.

Nous pouvons noter (voir, par exemple, le dernier graphe dans la figure 7.7) que ce croisement collabore de façon efficace à la recherche des extrémités du front de Pareto là, où cela pose un problème à l'algorithme utilisant les croisements aléatoires. Et même si dans le cadre de l'application donnée c'est la région "centrale" du front qui est plus intéressante du point de vue des ingénieurs, c'est une observation très importante pour l'évaluation de l'opérateur du croisement élitiste.

7.4.4 Résultats finaux

Les solutions obtenues en utilisant la méthode décrite dans la section 7.4.1 sont présentées dans la figure 7.9.

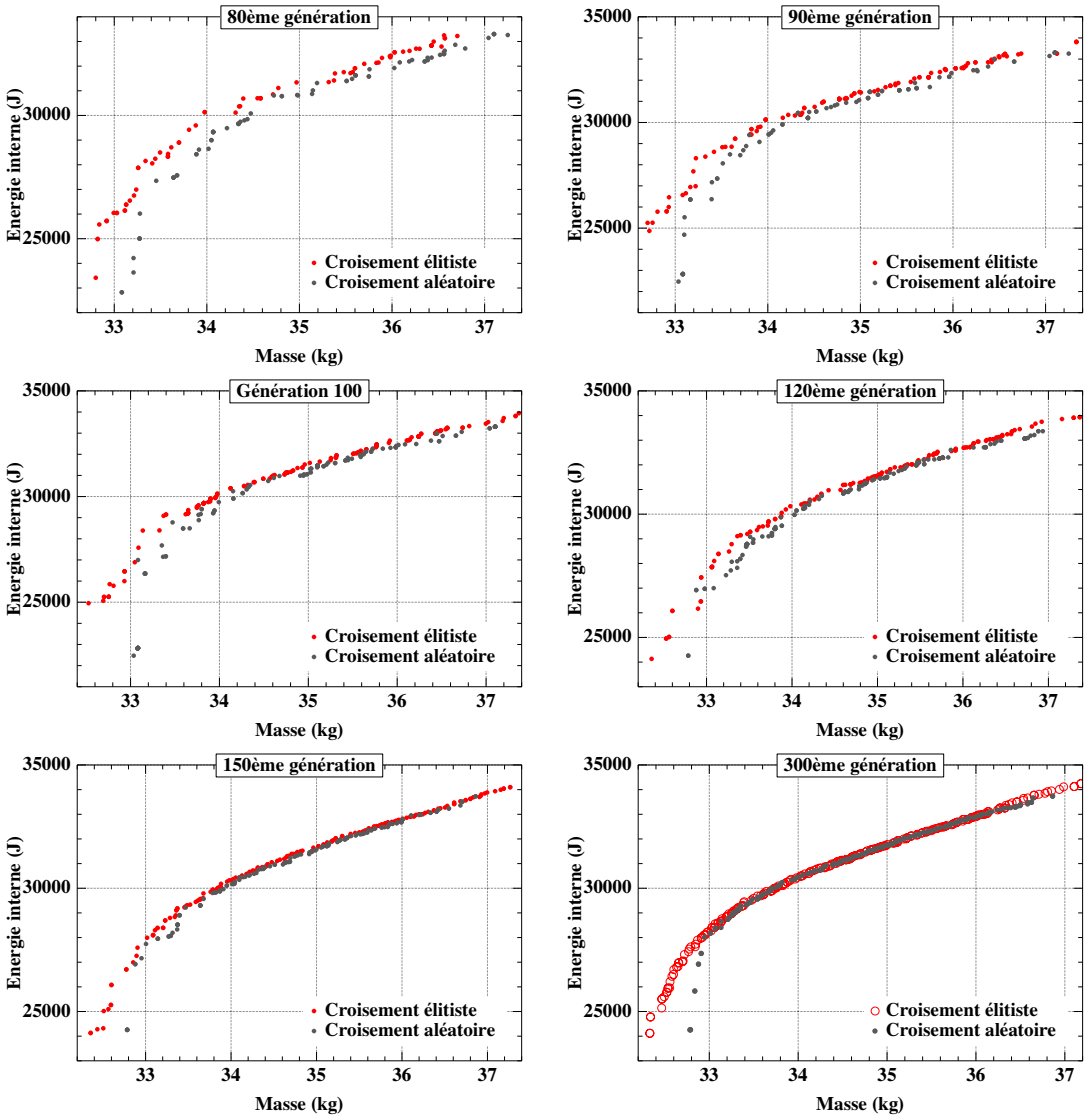


FIG. 7.7 – *Dynamique des populations en cours de résolution du problème du pare-choc avec l'utilisation du croisement élitiste (population rouge) et sans (population noire)*

Ce sont les solutions non-dominées trouvées après 150 générations lors de 11 essais de l'algorithme utilisant le croisement élitiste. Elles sont toutes faisables et, comme nous pouvons remarquer, la majorité d'entre elles domine la solution de référence sur le plan ($Masse \times Energie\ Interne$) de façon assez significative. Vu le choix des valeurs de contraintes, ces solutions dominent la solution du $(CE)^2$ relativement à tous les 10 objectifs du problème d'optimisation initial. Notons aussi qu'une partie assez grande des solutions appartient à la région $\{Masse < 36kg, Energie\ Interne > 30000J\}$.

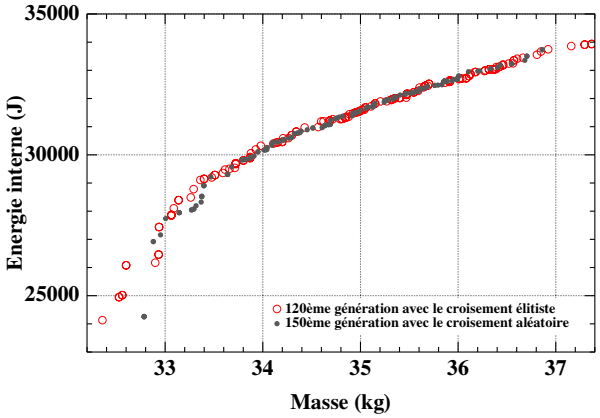


FIG. 7.8 – *Le croisement élitiste permet d'atteindre le même niveau de qualité des solutions en un nombre d'itérations plus petit : 120 générations avec le croisement élitiste contre 150+ avec le croisement standard.*

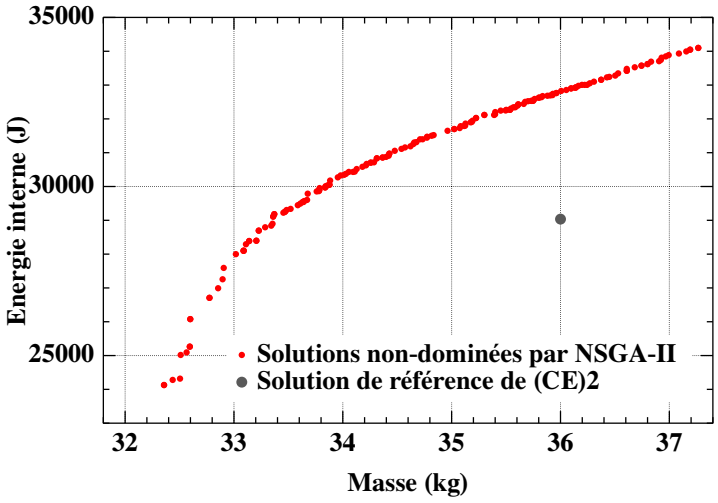


FIG. 7.9 – *La performance des solutions non-dominées trouvées en combinant NSGA-II et Infeasibility Objective Approach dépasse de loin la performance de la solution de référence obtenue précédemment*

7.5 Conclusions et perspectives

Le problème de l'optimisation du pare-choc étudié dans ce chapitre est un problème de conception. La qualité des solutions qu'il est possible de trouver lors de l'optimisation apparaît donc (jusqu'à une certaine limite, bien sûr) plus importante que la rapidité de leur obtention. D'autre part, le développement industriel reste un processus comprenant des essais répétés qui sont nécessaires pour valider le modèle en variant ses diverses composantes. Le choix des paramètres de l'algorithme d'optimisation qui définissent le temps de calcul a été fait compte tenu ces deux remarques.

Nous avons réussi à obtenir une grande variété de solutions pour l'optimisation du pare-choc de Fiat Multipla de qualité jusqu'à présent hors d'atteinte.

L'idée de transformer le problème initial à dix objectifs en un problème à deux objectifs sous contraintes se justifie autant pour la réduction du coup de calcul que du point de vue de la qualité des résultats et surtout de la facilité de leurs représentation.

Les valeurs des contraintes ont été choisies à partir de la solution de référence ce qui est tout à fait dans l'esprit du développement industriel. Leur ajustement a été faite en évitant de sacrifier trop la qualité des objectifs principaux. Par contre, la question intéressante qui n'a pas été étudiée dans ce travail est comment rendre le processus de leur ajustement plus rigoureux.

La performance du *croisement élitiste* (voir chapitre 6) a été testée dans le cadre de l'optimisation du pare-choc formulé ici comme un problème bi-objectif sous contraintes. La comparaison de résultats obtenus avec et sans utilisation de cet opérateur de reproduction témoigne de l'accélération de la convergence lors de l'application du croisement élitiste au stade de la recherche de la région faisable ainsi qu'à l'étape de l'optimisation des objectifs principaux du problème. Il a été, de plus, noté que, dans ce cas, le croisement élitiste permet de retrouver les extrémités du front de Pareto qui restent hors d'atteinte quand le croisement standard est appliqué.

Chapitre 8

L'optimisation de l'injection de carburant pour un moteur diesel Common Rail

Après la clôture du projet INGENET, la collaboration avec les chercheurs du Centre de Recherche de Fiat (CRF) a été poursuivie. Dans ce chapitre nous présentons l'étude d'un problème d'optimisation lié à la calibration du moteur diesel des poids lourds posé par Iveco.

8.1 Introduction "technique"

Un des challenges majeurs pour les producteurs d'automobiles est de faire face à des exigences contradictoires comme l'économie de carburant et la performance du véhicule, tout en respectant des niveaux maximum d'émissions polluantes. Le Standard Européen d'Emissions (*Euro Emission Standard*) a été introduit à fin de réduire progressivement les émissions des polluants classiques locaux (NOx, monoxyde de carbone, particules,...) et globaux GES (gaz à effet de serre, notamment CO2). Les normes établies pour les moteurs diesel des poids lourds sont citées dans le tableau sur la figure 8.1.

EU Emission Standards For Heavy Duty Diesel Engines, g/kWh (smoke in m^{-1})

Tier	Date	CO	HC	NOx	PM	Smoke	Remarks
Euro1	1992	4.5	1.10	8.0	0.61		<85kW
Euro1	1992	4.5	1.10	8.0	0.36		>85kWS
Euro1	1996	4.0	1.10	7.0	0.015		
Euro2	2000	1.5	0.25	2.0	0.02	0.15	EEVs
Euro3	2000	2.1	0.66	5.0	0.10	0.8	
Euro4	2005	1.5	0.46	3.5	0.02	0.5	
Euro5	2008	1.5	0.46	2.0	0.02	0.5	

FIG. 8.1 – *Euro Emission Standard pour les moteurs diesel des poids lourds*

8.1.1 Moteurs diesel

Si le marché européen des véhicules diesel connaît une telle croissance, il le doit notamment aux systèmes d'injection modernes. En effet, de la qualité de l'injection dépendent le couple, la consommation de carburant, les émissions polluantes et le comportement sonore du moteur.

On distingue deux types de moteurs diesel : moteur à injection directe et indirecte. La technique de l'injection directe effectue l'injection du gazole directement dans le cylindre et non pas dans une chambre de turbulence aménagée dans la culasse juste avant le cylindre (injection indirecte). La technique de l'injection directe a été adoptée car elle permet de baisser la consommation de carburant, assure un meilleur rendement et collabore à la réduction de la pollution.

Il est vrai que depuis quelques années le diesel n'a pas bonne publicité entre le bruit qu'il émet et sa mauvaise réputation en terme de pollution. Pour résoudre ces problèmes, les ingénieurs ont réussi à conserver les avantages de l'injection directe tout en réduisant fortement les inconvénients. Ils sont unanimes pour dire que la haute pression est la solution d'avenir pour le diesel. C'est de là qu'est née l'injection directe haute pression appelée Common Rail - la rampe d'injection commune.

Deux facteurs déterminent essentiellement la qualité de l'injection : d'une part, la pression avec laquelle le carburant est injecté dans le cylindre ; d'autre part, la modulation d'injection. Le Common Rail est le seul système où la mise sous pression est indépendante de l'injection : le carburant est d'abord stocké à haute pression dans un réservoir central, puis "appelé" par les différents injecteurs en fonction de leurs besoins. De ce fait, la pression élevée (elle peut aller jusqu'à 1350 bars) est disponible en continu, même à bas régime, ce qui assure une atomisation très fine du carburant, pour une consommation optimisée et plus propre. Le conducteur d'automobile en ressent lui-même les effets, au niveau des reprises du moteur comme des économies de carburant.

8.1.2 Fonctionnement du système Common Rail

Sur le système Common Rail, c'est une pompe à haute pression qui comprime le carburant et le refoule vers la rampe d'injection appelée "rail". Cette rampe est reliée avec les injecteurs pilotés par des électrovalves. Une forte pression du carburant est appliquée en permanence aux injecteurs. **Le carburant est injecté** dans le cylindre **à des instants particuliers et en quantité précises**.

La pompe haute pression (fig. 8.2 - 1) est spécifique au système. Elle alimente la rampe commune. **Le débit est**, bien sûr, **lié à la vitesse du moteur**. Toute la haute pression sera dirigée vers la rampe commune, elle sera ensuite réglée par un régulateur de pression intégré en bout de rampe.

La rampe commune (fig. 8.2 - 6) est un tube de forte épaisseur pouvant résister à de très hautes pressions, portant à ses extrémités le capteur de pression (fig. 8.2 - 2) et le régulateur de pression (fig. 8.2 - 4). Ce dernier reçoit et stocke le gazole haute pression en provenance de la pompe, il alimente en permanence les injecteurs (fig. 8.2 - 5) toujours sous pression. Cette pression est réglée par le régulateur qui est commandé par le boîtier électronique selon une cartographie en mémoire. Une cartographie est un ensemble de données mis en mémoire dans le calculateur qui sert de référence a celui-ci pour **commander le régulateur de façon optimale**, c'est le "logiciel" du calculateur.

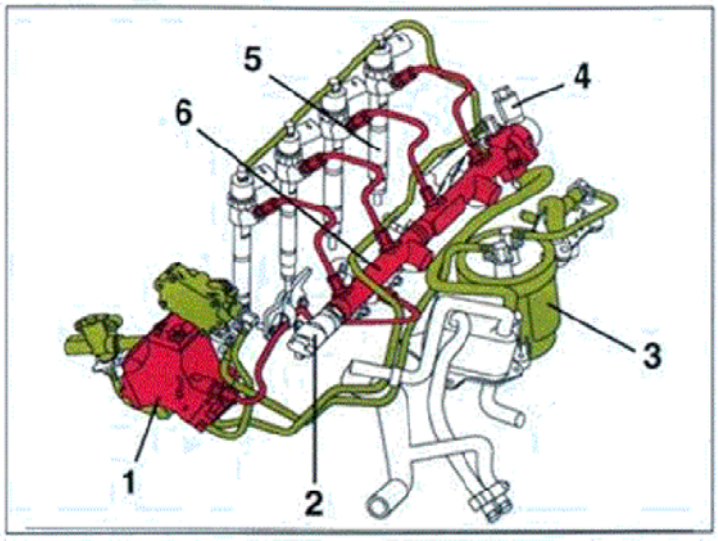


FIG. 8.2 – *Schéma de circulation du carburant. En rouge - haute pression, en vert - basse pression. 1 : Pompe haute pression ; 2 : Capteur de pression ; 3 : Refroidissement de carburant ; 4 : Régulateur de pression ; 5 : Injecteur ; 6 : Rampe commune ou Common Rail.*

Le calculateur

La technologie Common Rail apporte à la motorisation Diesel la maîtrise du temps d'injection. Le calculateur, en fonction des multiples informations envoyées par les différents capteurs (voir figure 8.3) reçoit les signaux d'entrée et gère les sorties **en fonction des critères de fonctionnement du moteur**. Pour adapter la quantité de gazole, **le calculateur agit soit sur la pression du carburant** dans la rampe (valve du régulateur de pression) **soit sur la durée d'activation des électrovannes des injecteurs**, en synchronisant, bien sûr, ces actions. La quantité injectée dépend de la commande des électrovannes, de la vitesse d'ouverture et de la fermeture de l'aiguille de l'injecteur, de la pression du carburant dans le rail, de la quantité passée par l'injecteur et de la levée de l'aiguille. Le schéma dans la figure 8.3 représente les principales informations d'entrée et de sortie au calculateur.

Injections multiples

La consommation est plus régulière lorsque l'arrivée de carburant se fait en plusieurs fois dans le même cycle de travail. Une pré-injection de très faible volume de carburant est suivie par l'injection principale, complétée, au besoin, par une post-injection relativement faible.

La pré-injection sert surtout à assurer une hausse de pression régulière dans la chambre de combustion et donc à réduire le bruit lié à la combustion ; la post-injection agit quant à elle au niveau du retraitement des gaz d'échappement.

Grâce à ses injections rapides et précises le système Common Rail présente des atouts déterminants au regard des prescriptions futures en matière d'émissions de gaz d'échappement (voir figure 8.1).

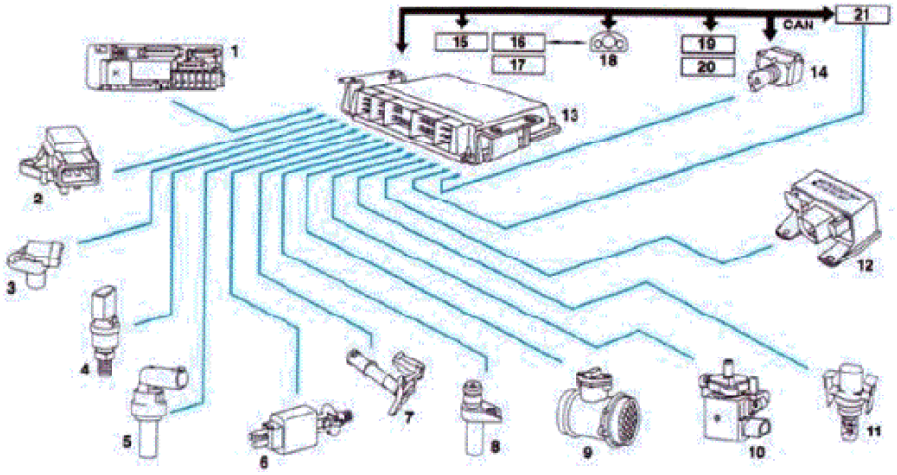


FIG. 8.3 – *1 : module fusible et relais ; 2 : capteur de pression 3 : info régime ; 4 : capteur de pression sur rampe commune ; 5 : sonde température d'eau ; 6 : contacteur pédale de frein ; 7 : capteur de niveau d'huile ; 8 : capteur position vilebrequin ; 9 : débitmètre d'air massique a film chaud ; 10 : capteur position pédale d'accélérateur ; 11 : sonde température d'air ; 12 : préchauffage ;* **13 : calculateur** *; 14, 15, 16, 17, 18, 19, 20, 21: informations concernant les signaux ABS ,EGS, ANTIVOL, CLIM, ASR.*

8.2 Problème de l'optimisation de l'injection de carburant

8.2.1 Critères d'optimisation et paramètres de contrôle

En reprenant ce qui a été dit au début de la section précédente, par *bonne qualité de l'injection* nous allons entendre consommation du carburant réduite, bas niveau de bruit de combustion et satisfaction des contraintes imposées par le *European Emission Standard*.

Vu que ce sont essentiellement la modulation de l'injection et la pression qui influencent le niveau de satisfaction de ces critères, nous nous plaçons dans le cadre, où le processus de l'injection est défini par les paramètres suivants : le timing et la pression de l'injection principale et le timing et la durée de chacune des pré- et post-injections (voir figure 8.4).

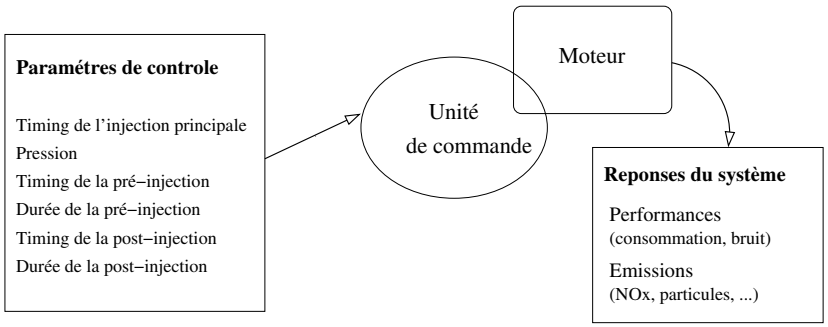


FIG. 8.4 – *Paramètres de décision et critères d'optimisation*

Carte des régimes de moteur

L'ensemble des points du plan (*vitesse du moteur* × *couple*) où les émissions polluantes sont mesurées est défini par les normes législatives et il varie en fonction du type de moteur. Tous les points n'ayant pas la même importance, un poids "législatif" est associé à chacun d'eux, généralement, en fonction du pourcentage statistique de temps durant lequel le moteur travaille avec le ratio vitesse/couple déterminé par le point en question. C'est-à-dire, lors du test d'un moteur au sens de pollution, la quantité d'émission d'un polluant correspondant à chacun de ces "régimes contrôlés" est mesurée, et la somme pondérée des quantités mesurées est comparée à la limite autorisée.

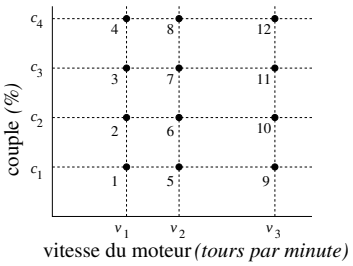


FIG. 8.5 – *Un exemple de carte des régimes du moteur (engine map) avec 12 points du contrôle*

Les systèmes de timing mécaniques sont capables d'adapter le timing d'injection à la vitesse du moteur. Il existe également des outils pour les conditions *full-load* et *low-idle*. Mais avec de tels systèmes il est plutôt difficile de satisfaire même les contraintes Euro3 (figure 8.1). Dans les systèmes du contrôle des moteurs électroniques, les paramètres comme le timing et la pression dans la rampe ainsi que les caractéristiques de l'injecteur sont stoqués dans une carte 3-D. Cela permet le contrôle individuel de la combustion en différents points sur le plan (*vitesse du moteur* × *couple*) (voir figure 8.5).

Cette flexibilité du processus d'injection de carburant traduite en terme d'optimisation, signifie que nous allons chercher un ensemble de valeurs optimales des paramètres de contrôle énumérés dans la figure 8.4 pour chacun des points du contrôle de la "carte des régimes du moteur" (*engine map*) (voir figure 8.5).

8.2.2 Position du problème d'optimisation

Notons x_k vecteur des variables de décision correspondant au $k$ème point du contrôle sur la carte des régimes de moteur. Par la suite nous allons utiliser les notations suivantes :

f_{jk} - la consommation ($j = 1$) ou le bruit de combustion ($j = 2$) mesurés au point k;
g_{jk} - l'émission du polluant j mesurée au point k;
c_j - la valeur de contrainte pour l'émission du polluant j, imposée par *European Emission Standard*;
w_k - le poids législatif correspondant au point k.

Soit enfin $x = (x_1,...,x_K)$, où K est le nombre de points du contrôle.

Le problème d'optimisation étudié dans ce chapitre consiste à trouver un vecteur $x^* = (x_1^*,...,x_K^*)$ tel que

$$f_{jk}(x_k^*) \quad = \quad \min_{x_k} \quad f_{jk}(x_k), \qquad j = 1,2, \quad k = 1,...,K, \tag{8.1}$$

et

$$g_j(x^*) = \sum_{k=1}^{K} w_k g_{jk}(x_k^*) < c_j \qquad \text{pour tout polluant } j. \tag{8.2}$$

Dans le cadre du développement complexe (fait par Iveco), nous sommes intervenus au stade de la résolution d'un problème simplifié avec seulement deux contraintes (sur cinq énumérées dans la figure 8.1), NOx et *particules* (PM).

Le modèle du moteur considéré dans cette étude correspond à l'injection à deux étapes : pré-injection et l'injection principale, c'est-à-dire la longueur de chaque vecteur x_k est égale à 4 (voir figure 8.4).

Le nombre des points du contrôle pour le type de moteur en question est égale à 12 (comme dans la figure 8.5), ce qui signifie $|x| = |(x_1,...,x_K)| = 48$.

8.3 Approches "globale" et "locale"

8.3.1 Résolution du problème "global"

Soit

$$f_j(x) \quad = \quad \sum_{k=1}^{K} w_{jk} f_{jk}(x_k), \qquad j = 1,2. \tag{8.3}$$

L'étude du problème posé dans la section 8.2.2 a été commencé par la résolution du problème d'optimisation suivant :

$$min f_j(x), \qquad j = 1,2, \tag{8.4}$$

sous contraintes (8.2), où les fonctions f_j sont définies par les égalités (8.3). Dans tout ce qui suit, ce problème est évoqué comme *problème global*, car chacune des fonctions objectif de la consommation du carburant et du bruit de la combustion est écrit sous forme des *sommes pondérées sur tous les points du contrôle*.

Une telle écriture du problème (8.1), (8.2) réduit la dimension de l'espace des objectifs de 24 à 2, ce qui enlève la question de la représentation des résultats et, de plus, réduit le coût du ranking lors de l'application de l'Algorithme Evolutionnaire Multi-Objectif.

Une remarque très importante consiste à dire que, dans ce cas, les fonctions objectifs f_{jk} pour différents k (variant de 1 à 12) ne dépendent d'aucune variable commune et donc, a priori, ne sont pas "en conflit" lors de leur optimisation simultanée. Ainsi, le poids associé à chaque fonction objectif *locale* f_{jk} décrit effectivement l'importance du point du contrôle correspondant relativement aux autres points de contrôle (rappelons que le poids w_k correspond au pourcentage du temps durant lequel le moteur travaille dans le régime k).

La figure 8.6 présente l'ensemble des solutions non-dominées trouvées lors de 11 essais de NSGA-II (section 3.4.3) appliqué au problème (8.4), (8.2). Lors de chaque essai, la population de taille 100 a évolué pendant 300 générations. Les autres paramètres de l'algorithme évolutionnaire sont les mêmes que ceux utilisés pour l'optimisation du pare-choc (voir section 7.4.2).

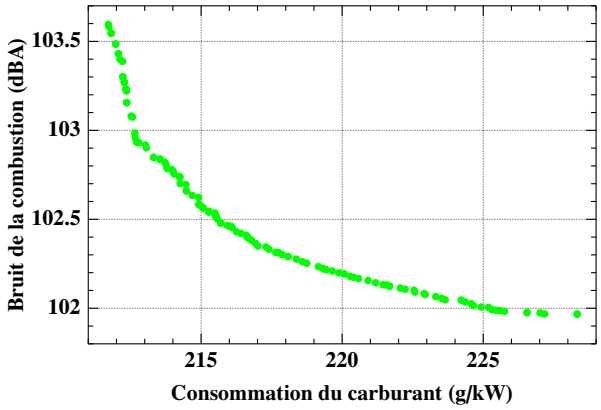


FIG. 8.6 – *Solutions non-dominées du problème "global" trouvées lors de 11 essais de NSGA-II combiné avec Infeasibility Objective Approach*

Dans cette étude, les contraintes des émissions NOx et PM correspondent au normatif $Euro3$ (figure 8.1). En fait, pour les valeurs c_j de $Euro4$, non seulement aucune solution faisable du problème (8.4), (8.2) n'a pu être trouvée, mais même les solutions du problème de minimisation des $g_j(x)$, $j = 1{,}2$ n'a pas atteint le niveau de qualité suffisant pour satisfaire les inégalités (8.2) (voir la figure 8.7).

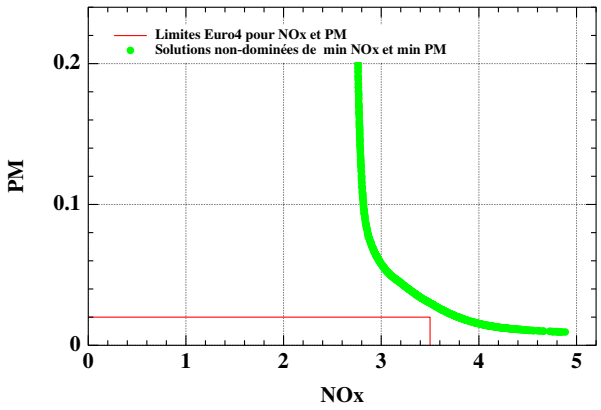


FIG. 8.7 – *Les solutions Pareto-optimales du problème de la minimisation de NOx et de PM n'arrivent pas à rentrer dans la région $NOx < 3.5$ g/kWh $PM < 0.02$ g/kWh correspondante à l'exigence de Euro4*

Mais ceci est plutôt normal que décevant dans notre cas, car le modèle étudié ne prévoit pas la post-injection, "responsable", comme nous l'avons déjà dit plus haut, pour la réduction du niveau des émissions.

Pour la prise en compte des contraintes, *Infeasibility Objective Approach* a été employé ici de la même façon que lors de la résolution du problème de l'optimisation du pare-choc (voir section 7.4.1).

8.3.2 Considération "point par point"

Le fait que les fonctions objectif f_{jk}, k = 1,...,12 ne dépendent pas des mêmes variables de décision, fait penser à la possibilité de traiter indépendamment 12 problèmes d'optimisation *locaux*, chacun correspondant à un point du contrôle. D'une part, ceci reviendrait à réduire la dimension de l'espace de recherche de 48 que nous avions pour le problème global (8.4), (8.2) à 4 pour chaque problème local.

La question qui se pose concerne la détermination des valeurs des contraintes pour chacun des problèmes locaux de façon que leur satisfaction implique la satisfaction des contraintes du problème initial (8.2).

Une solution naïve consiste à prendre la même valeur des contraintes pour tous les problèmes locaux, notamment les valeurs c_j. Grâce au fait que la somme des poids w_k est égale à 1, si les inégalités

$$g_{jk}(x_k) < c_j, \quad k = 1,...,12$$

sont vérifiées pour tout polluant j, alors chacune des contraintes (8.2) sera également vérifiée.

Les inconvénients de ce choix des valeurs des contraintes locales sont les suivants. Tout d'abord, il ne tient pas du tout compte des importances relatives des points du contrôle. De plus, les tests montrent qu'avec de tel choix de contraintes, pour certains problèmes locaux, l'algorithme n'arrive pas à trouver de solutions faisables.

Une autre approche aux choix des contraintes locales a été testée. Elle s'appuie sur les résultats obtenus lors de la résolution du problème global. On calcule, d'abord, les valeurs de $g_{jk}(x_k)$, k = 1,...,12, pour toute solution $x = (x_1,...,x_{12})$ du problème (8.1), (8.2). Notons, $\overline{g}_{jk}$ la moyenne des $g_{jk}(x_k)$ parmi toutes les solutions x. La valeur de la contrainte c_{jk} pour chaque problème local est choisie de façon à ce qu'elle soit proportionnelle à l'impact de $\overline{g}_{jk}$ dans la moyenne globale $\overline{g}_j$, c'est-à-dire $c_{jk} = c_j(\overline{g}_{jk}/\overline{g}_j)$.

La "logique" de ce choix des valeurs c_{jk} est telle que le ratio $\overline{g}_{jk}/\overline{g}_j$, "élaboré" au cours de l'optimisation globale (qui tient compte des "importances législatives" de différents points du contrôle) correspond à la meilleure performance au sens des objectifs de la consommation et du bruit optimisés sous les contraintes de *European Emission Standard*.

La figure 8.8 montre les solutions des problèmes locaux comparées aux solutions du problème global dans les espaces $< f_{1k}, f_{2k} >$, k = 1,...,12. Les graphes dans cette figure sont placés en conformité avec les positions des points du contrôle correspondants sur la carte des régimes du moteur décrits figure 8.5.

Il est important de noter que toutes les solutions du problème local k vérifient les contraintes locales c_{jk}, tandis que les projections des solutions du problème global sur le sous-espace $< x_{k,1}, x_{k,2}, x_{k,3}, x_{k,4} >$ peuvent violer les contraintes locales. C'est-à-dire, on peut combiner n'importe quelle solution d'un problème local avec n'importe quelle solution d'un autre problème local en gardant les contraintes globales satisfaites. Ceci constitue un avantage important de la considération "point par point".

8.3.3 Optimisation "point par point" vs optimisation "globale"

Pour comparer les résultats de l'optimisation locale avec les solutions du problème global dans l'espace $< f_1, f_2 >$, un nombre suffisamment petit de solutions locales a été sélectionné pour chaque k (points bleus sur la graphes dans la figure 8.8). Ensuite, chacune des solutions

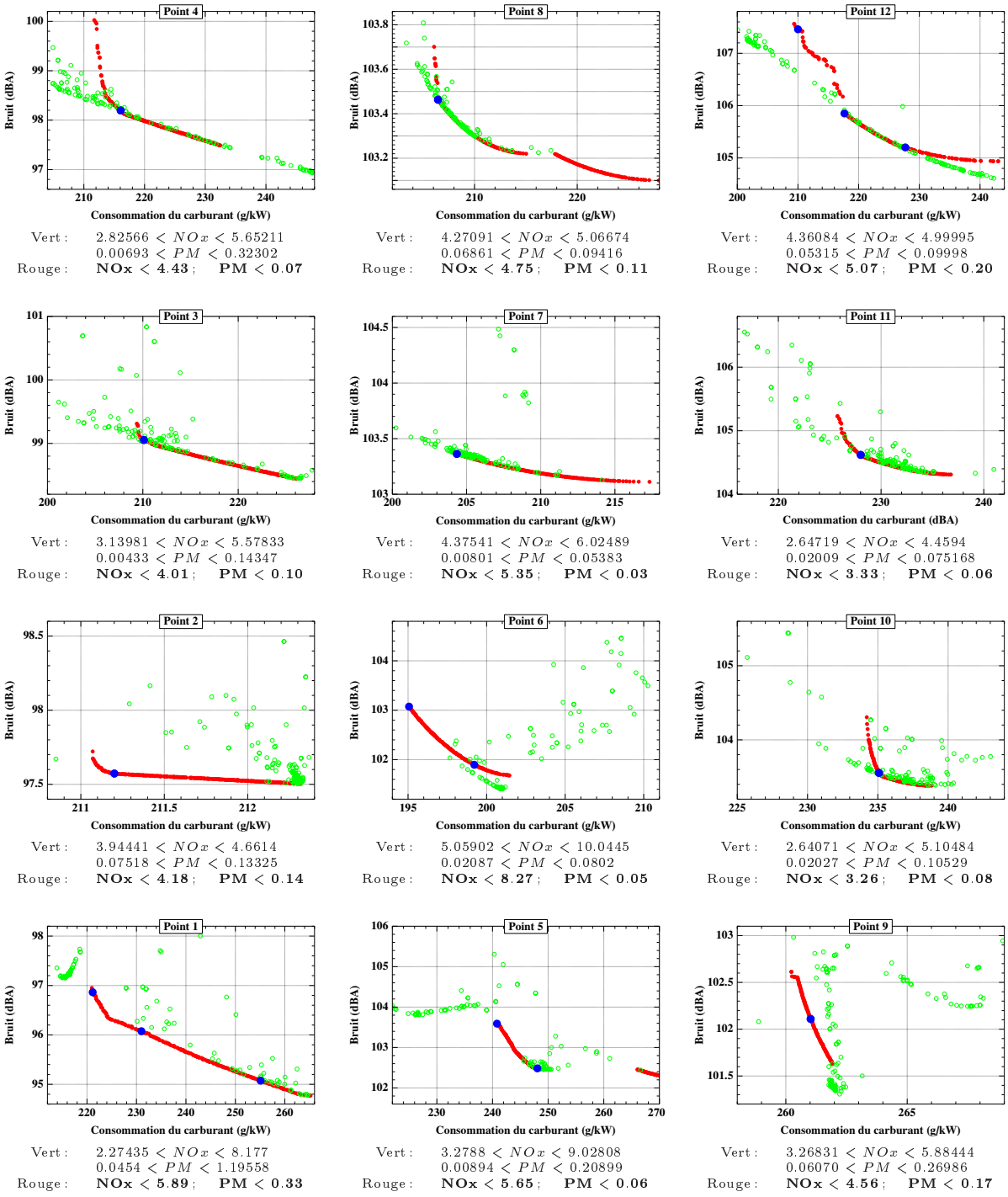


FIG. 8.8 – *Solutions "locales" (points verts) vs solutions non-dominées du problème "global" (points rouges)*

choisies (qui est un vecteur de $\mathbb{R}^4$) a été combinée avec les solutions de tous les autres problèmes locaux de façon à obtenir les vecteurs de $\mathbb{R}^{48}$ sur lesquels les valeurs de f_1 et f_2 ont été calculées.

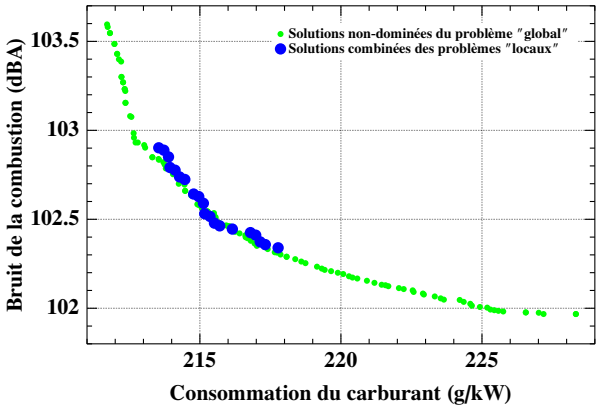


FIG. 8.9 – *Les solutions du problème "global" comparées aux points obtenues en combinant les solutions sélectionnées des problèmes "locaux"*

Notons que quel que soit le choix des contraintes locales, prises toutes ensemble, elle sont plus restrictives que les contraintes initiales (8.2). Malgré cela, les résultats d'optimisation, illustrés dans la figure 8.9 montrent lors de la résolution des problèmes locaux, la même qualité d'optimisation (au sens des sommes pondérées f_1 et f_2) est atteinte.

8.3.4 Discussion

La motivation initiale de diviser le problème global (8.1), (8.2) en sous-problèmes correspondants à différents points du contrôle était d'essayer de réduire la dimension de l'espace de recherche, ce qui permet, généralement, de l'explorer plus efficacement avec un coût de calcul réduit.

Cependant, cela ne peut pas être le cas ici (au moins, en ce qui concerne la réduction de coût), tant que nous utilisons les solutions du problème global pour fixer les valeurs des contraintes locales.

L'idée de la division du problème pondéré a été, néanmoins, reconnue non seulement originale mais très appropriée à l'application étudiée et ceci principalement pour les deux raisons suivantes.

D'une part, à l'étape de l'élaboration des modèles statistiques pour le calcul des réponses du système étudié (en l'occurrence, ce sont la consommation, le bruit et les émissions) l'analyse des résultats obtenus pour les problèmes locaux permet quelquefois de détecter des modèles incorrects pour certains de ces points.

Mais la raison principale pour laquelle la considération "point par point" a été reconnue appopriée lors de l'étude de ce problème, c'est que même si tous les cinq polluants du normatif européen (énumérés dans le tableau de la figure 8.1) sont pris en compte, il y a, en réalité, d'autres contraintes (autant dites contraintes opérationnelles) qui ne font pas partie du modèle considéré dans cette étude, mais auxquelles les solutions de notre problème d'optimisation doivent être adaptées ultérieurement. Ces contraintes imposent notamment des limites sur la différence entre les valeurs des paramètres optimisés (timing, durée et la pression de l'injection) dans des points de contrôle voisins . La possibilité de combiner explicitement les solutions locales en un vec-

teur $\mathbb{R}^{48}$ (qui va correspondre à une configuration des paramètres de contrôle pour toute la carte des régimes du moteur) représente un avantage considérable dans ce contexte applicatif.

Conclusions et perspectives

Une tendance remarquable des Algorithmes Evolutionnaires est que ces méthodes "pénètrent" aujourd'hui dans de nombreux nouveaux domaines d'applications malgré l'absence de bases théoriques (notamment, de preuves de convergence) aussi solides que celles qu'on peut trouver pour des approches alternatives. Inspirée par cette observation, la motivation principale de ce travail était de contribuer au développement des Algorithmes Evolutionnaires Multi-Objectif (AEMO) de façon à rendre leur application aux problèmes réels la plus efficace possible.

Ainsi, le critère d'arrêt "par désespoir", proposé dans le chapitre 5, est destiné à optimiser le rapport entre la qualité des solutions et le coût de calcul : dans la pratique, c'est ce compromis qui est le plus souvent recherché.

De même, l'opérateur de croisement, introduit dans le chapitre 6, a été évalué dans l'idée de mettre en valeur l'accélération de la progression vers la surface de Pareto qu'il apporte et non pas, par exemple, le pourcentage des solutions de Pareto exactes qu'il permet de retrouver lors de la résolution des problèmes test (la mesure qui a été utilisée lors de certaines études proposant de nouveaux opérateurs de variation).

L'étude de deux applications réelles issues du domaine de l'Industrie Automobile, peut être estimée comme une réussite dans les deux cas. Pour le problème de l'optimisation du pare-choc de voiture, la transformation des objectifs secondaires en contraintes de l'optimisation a permis d'obtenir une variété suffisamment grande de solutions faisables qui dominent de loin la solution de référence, et ceci en temps de calcul plus que raisonnable compte tenu le fait qu'il s'agit de la résolution d'un problème de conception.

Pour la seconde application, en absence de toute référence, la réussite consiste tout d'abord à avoir trouvé des solutions satisfaisantes en applicant un AEMO performant, combiné avec une approche de la prise en compte des contraintes basée sur le tri par degré de la violation. Mais c'est surtout l'idée de la division du problème d'optimisation initialement posé dans un espace de décision de dimension 48 en 12 sous-problèmes dont chacun est résolu dans un espace de décision de dimension 4, qui a été appréciée par nos collaborateurs du Centre de Recherche de Fiat. L'avantage de cette division en possibilité d'exploiter les solutions des sous-problèmes à d'autres stades de ce développement.

Toutes les études effectuées dans le cadre de cette thèse ouvrent des perspectives de recherche future. Que ce soit l'utilisation des métriques dynamiques pour rendre plus rigoureuse l'évaluation du critère d'arrêt "par désespoir" et du croisement élitiste, ou la combinaison de la restriction du choix des partenaires basée sur la dominance avec d'autres types du croisement que BLX biaisé, ou encore l'exploration des idées liées à la façon plus systématique de fixer des valeurs de contraintes pour le problème du pare-choc.

D'autres directions possibles pour la recherche future ont également été indiquées au fil du

texte, et à la fin des chapitres correspondants. Certaines de ces idées seront sans doutes mises en œuvre dans le futur le plus proche.

Mais la chose sur laquelle il faut insister ici, c'est que ce manuscrit, bien que présentant le bilan du travail effectué jusqu'à présent, ne le conclut pas. Il faudrait plutôt le comparer à un instantanné (*snapshot*) qui fixe l'état de l'évolution à un moment précis. Mais, pendant qu'on l'analyse, cet état, généralement, change et continue à évoluer. La seule véritable conclusion consiste à dire que la condition d'arrêt "par désespoir" dans ce cas est encore loin d'être satisfaite, car il y a encore un potentiel de progression suffisant pour de nombreuses générations.

Annexe A

Tests bi-objectif de Zitzler, Deb et Thiele

Dans cet annexe, nous allons présenter les tests bi-objectifs [117], qui ont servi dernièrement de la base commune pour la comparaison des AEMO existants et pour l'évaluation de nouvelles techniques.

Tous les problèmes se posent sous forme suivante (correspondant aux équations 4.1):

$$\begin{array}{ll} min & f_1(x_1), \\ min & f_2(x) = g(\mathbf{x_2})h(f_1(x_1),g(\mathbf{x_2})). \end{array} \tag{A.1}$$

où $x = (x_1,\mathbf{x_2})$ et $\mathbf{x_2} = (x_2,...,x_n)$.

Notons par ailleurs que tous ces tests ont une particularité commune qui n'est pas mentionnée par les auteurs mais qui peut causer une grande difficulté aux AEMO même pour s'approcher de l'ensemble optimal. Il s'agit du fait que toutes les solutions optimales sont regroupées dans un petit "coin" de l'espace de recherche, car, dans tous le cinq tests, le front de Pareto correspond à $x_i^* = 0$ pour $i = 2,...,n$ avec $n = 30$ ou $n = 10$.

Nous n'avons pas cité ici le problème ZDT5, qui est formulé pour les variables binaires.

ZDT1

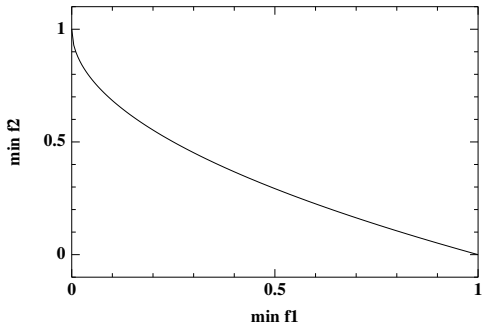


FIG. A.1 – *Solution de ZDT1*

Le premier de cet ensemble de tests est le plus simple, le front de Pareto correspondant étant continu, convexe et avec la distribution uniforme des solutions le long du front. Ce problème peut être considéré comme l'étape de qualification lors du test d'une méthode: si une bonne approximation de l'ensemble des optimaux n'a pas été trouvé, ça ne vaut pas la peine d'essayer d'appliquer l'approche étudiée aux tests suivants.

$$ZDT1: \quad \begin{cases} f_1(x) &= x_1, \\ g(\mathbf{x_2}) &= 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i, \\ h(x) &= 1 - \sqrt{f_1/g}, \end{cases} \tag{A.2}$$

où $x_i \in [0,1]$ pour tout $i = 1,...,n$. Dans [117], $n = 30$. La solutions de ce problème sont telles que $0 \leq x_1^* \leq 1$ et $x_i^* = 0$ pour $i = 2,...,n$. Le front de Pareto du problème ZDT1 est présenté dans la figure A.1.

ZDT2

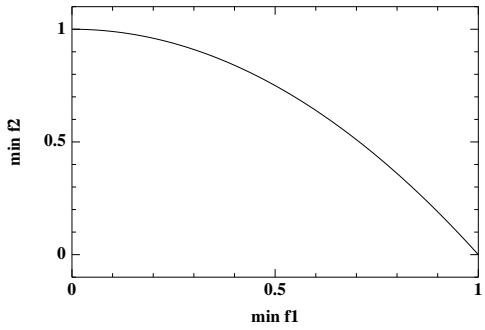


FIG. A.2 – *Solution de ZDT2*

La seule difficulté de ce problème consiste en non-convexité du front de Pareto.

$$ZDT2: \quad \begin{cases} f_1(x) &= x_1, \\ g(\mathbf{x_2}) &= 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i, \\ h(x) &= 1 - (f_1/g)^2, \end{cases} \tag{A.3}$$

où $x_i \in [0,1]$ pour tout $i = 1,...,n$. Dans [117], $n = 30$. La solutions de ce problème sont telles que $0 \leq x_1^* \leq 1$ et $x_i^* = 0$ pour $i = 2,...,n$. Le front de Pareto du problème ZDT2 est présenté dans la figure A.2.

ZDT3

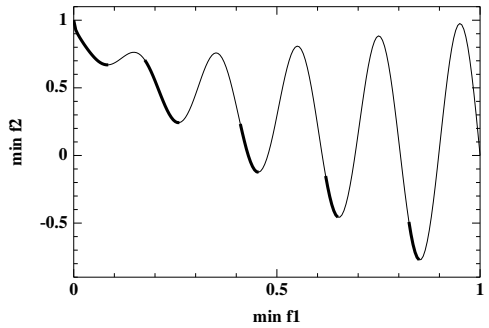


FIG. A.3 – *Solution de ZDT3*

La difficulté de ce problème c'est que le front de Pareto est discontinu.

$$ZDT3: \quad \begin{cases} f_1(x) &= x_1, \\ g(\mathbf{x_2}) &= 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i, \\ h(x) &= 1 - \sqrt{f_1/g} - (f_1/g)sin(10\pi f_1), \end{cases} \tag{A.4}$$

où $x_i \in [0,1]$ pour tout $i = 1,...,n$. Dans [117], $n = 30$. La solutions de ce problème sont telles que $x_i^* = 0$ pour $i = 2,...,n$ mais non pas toutes les solutions avec $0 \leq x_1 \leq 1$ sont situé sur le front de Pareto (voir figure A.3).

ZDT4

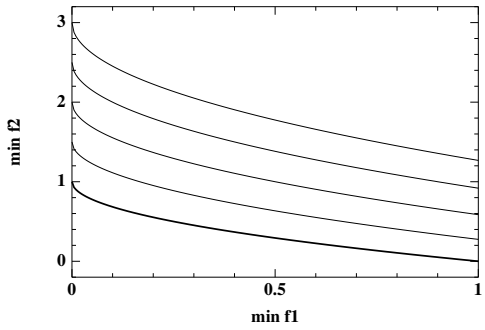


FIG. A.4 – *Solution de ZDT4*

Ce test modélise la difficulté de la convergence vers le front de Pareto globale à la présence de plusieurs fronts locaux à cause du fait que la fonction g est multi-modale.

$$ZDT4: \quad \begin{cases} f_1(x) &= x_1, \\ g(\mathbf{x_2}) &= 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10cos(4\pi x_i)), \\ h(x) &= 1 - \sqrt{f_1/g}, \end{cases} \tag{A.5}$$

où $x_1 \in [0,1]$, $x_i \in [-5,5]$ pour tout $i = 2,...,n$. Dans [117], $n = 10$. Les optima globaux sont tels que $0 \leq x_1^* \leq 1$ et $x_i^* = 0$ pour $i = 2,...,n$. Les fronts locaux correspondent à $0 \leq x_1 \leq 1$ et $x_i = 0.5j$, où j est un entier appartenant au segment $[-10,10]$, $i = 2,...,n$. Quelques fronts locaux et le front global sont présentés dans la figure A.4.

ZDT6

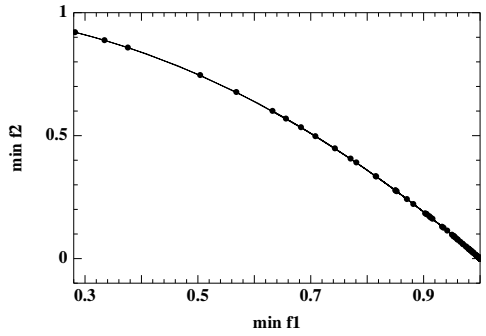


FIG. A.5 – *Solution de ZDT6*

La particularité de ce problème c'est que les solutions optimales ne sont pas uniformément distribuée le long du front de Pareto. Cette effet est du à la non-linéarité de la fonction f_1.

$$ZDT6: \quad \begin{cases} f_1(x) &= 1 - exp(-4x_1)sin^6(-\pi x_1), \\ g(\mathbf{x_2}) &= 1 + 9\Big((\sum_{i=2}^{n} x_i)/9\Big)^{1/4}, \\ h(x) &= 1 - (f_1/g)^2, \end{cases} \tag{A.6}$$

où $x_i \in [0,1]$ pour tout $i = 1,...,n$. Dans [117], $n = 10$. La solutions de ce problème sont telles que $0 \leq x_1^* \leq 1$ et $x_i^* = 0$ pour $i = 2,...,n$. Le front de Pareto du problème ZDT6 est présenté dans la figure A.5 ainsi que les solutions correspondantes à 100 valeurs de x_1 uniformément distribuées sur le segment [0,1].

Bibliographie

[1] J. M Alliot. *Techniques d'optimisation stochastique appliquées aux problemes du trafic aérien*. PhD thesis, ENA, 1996.

[2] J. S. Arora. *Introduction to Optimum Design*. McGraw-Hill, New-York, 1989.

[3] T. Bäck, F. Hoffmeister, and H. P. Schwefel. A survey of evolution strategies. In R. K. Belew and L. B. Booker, editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 2–9. Morgan Kaufmann, 1991.

[4] James E. Baker. Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette, editor, *Proceedings of the* 2^{nd} *International Conference on Genetic Algorithms*, pages 14–21. Laurence Erlbaum Associates, 1987.

[5] R. Bates, T. Bosio, and R. Fontana. Computer experiments for concurrent engineering. In *European Congress on Computational Methods in Applied Sciences and Engineering*, 2000.

[6] T. T. Binh and U. Korn. Mobes: A multiobjective evolution strategy for constrained optimization problems. In Th. Bäeck, editor, *Proceedings of the* 7^{th} *International Conference on Genetic Algorithms*, pages 176–182. Morgan Kaufmann, 1997.

[7] V. Chankong and Y.Y. Haimes. *Multiobjective Decision Making Theory and Methodology*. North-Holland, New-York, 1983.

[8] C. A. C. Coello and G. Toscano. A micro-genetic algorithm for multi-objective optimization. Technical report, Laboratoria Nacional de Informatica Avanzada, Xalara, Veracruz, Mexico, 2000.

[9] D. W. Corne, J. D. Knowles, and M.L. Oates. The pareto envelope-based selection algorithm for multiobjective optimization. pages 839–848, 2000.

[10] N. L. Cramer. A representation for the adaptive generation of simple sequential programs. In J. J. Grefenstette, editor, *Proceedings of the* 1^{st} *International Conference on Genetic Algorithms*, pages 183–187. Laurence Erlbaum Associates, 1985.

[11] N. Cressie. *Statistics for Spatial Data*. Wiley, NY, 1993.

[12] N. O. D. Cunha and E. Polak. Constrained minimization under vector-evaluated criteria in finit dimansional spaces. *Jornal of Mathematical Analysis and Applications*, 19(1):103–124, 1967.

[13] J. M. Daida, J. D. Hommes, S. J. Ross, and J. F. Vesecky. Extracting curvilinear features from SAR images of arctic ice: Algorithm discovery using the genetic programming paradigm. In T. Stein, editor, *Proceedings of IEEE International Geoscience and Remote Sensing*, pages 673–675, Florence, Italy, 1995. IEEE Press.

[14] K. Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation Journal*, 7(3):311–338, 1999.

[15] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. New York: John Wiley, 2001.

[16] K. Deb and S. Agrawal. A niched-penalty approach for constraint handling in genetic algorithmd. In *Proceedings of the 1999 Genetic and International Conference on Artificial Neural Networks and Genetic Algorithms*, pages 235–243, 1999.

[17] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: Nsga-ii. Technical Report 200001, Indian Institute of Technology, Kanpur.

[18] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. pages 849–858, 2000.

[19] K. Deb and D. E. Goldberg. An investigation of niche and species formation in genetic function optimization. In J. D. Schaffer, editor, *Proceedings of the* 3^{rd} *International Conference on Genetic Algorithms*, pages 42–50. Morgan Kaufman, 1989.

[20] K. Deb and S. Jain. Running performance metrics for evolutionary multi-objective optimization. Technical Report 2002004, Indian Institute of Technology, Kanpur, May 2002.

[21] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multi-objective optimization. Technical Report 2001001, Indian Institute of Technology, Kanpur, 2001.

[22] K. A. DeJong. *The Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Harbor, 1975. *Dissertation Abstract International*, 36(10), 5140B. (University Microfilms No 76-9381).

[23] K. A. DeJong and J. Sarma. On decentralizing selection algorithms. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 17–23. Morgan Kaufmann, 1995.

[24] K .A. DeJong and W. M. Spears. An analysis of multi-point crossover. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 301–315, San Mateo, 1991. Morgan Kaufmann.

[25] K. A. DeJong and W. M. Spears. A formal analysis of the role of multi-point crossover in genetic algorithms. *Artificial Intelligence*, 5:1–26, 1992.

[26] A. E. Eiben, C. H. M. Van Kemenade, and J. N. Kok. Orgy in the computer: multi-parent reproduction in genetic algorithms. In *European Conference on Artificial Life*, Granada, 1995.

[27] M.A. El-Beltagy, P.B. Nair, and A.J. Keane. Metamodeling techniques for evolutionary optimization of computationally expensive problems: Promises and limitations. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 196–203. Morgan Kaufmann, 1999.

[28] L. Eshelman and J. D. Schaffer. Real-coded genetic algorithms and interval-schemata. In L. D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 187–202, Los Altos, CA, 1993. Morgan Kaufmann.

[29] A. Farhang-Mehr and S. Azarm. Diversity assesment of pareto-optimal solution sets: An entropy approach. In *World Congress on Computational Intelligence*, pages 723–728, 2002.

[30] D. B. Fogel. An analysis of evolutionary programming. In D. B. Fogel and W. Atmar, editors, *Proceedings of the* 1^{st} *Annual Conference on Evolutionary Programming*, pages 43–51, La Jolla, CA, 1992. Evolutionary Programming Society.

[31] D. B. Fogel. *Evolving artificial intelligence*. PhD thesis, University of California, San Diego, CA, 1992.

[32] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. New York: John Wiley, 1966.

[33] Carlos M. Fonseca and Peter J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Genetic Algorithms: Proceedings of the Fifth International Conference*, pages 416–423. Morgan Kaufmann, 1993.

[34] Carlos M. Fonseca and Peter J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[35] Carlos M. Fonseca and Peter J. Fleming. On the Performance Assessment and Comparison of Stochastic Multiobjective Optimizers. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN IV*, Lecture Notes in Computer Science, pages 584–593, Berlin, Germany, September 1996. Springer-Verlag.

[36] Carlos M. Fonseca and Peter J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms—Part I: A unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 28(1):26–37, 1998.

[37] R. L. Fox. *Optimization Methods for Ingineering Design*. Addison-Wesley, MA, 1971.

[38] A. Fukunaga and A. Stechert. Evolving nonlinear predictive models for lossless image compression with genetic programming. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Goldberg, Hitoshi Iba, and Rick Riolo, editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 95–102, University of Wisconsin, Madison, Wisconsin, USA, 22-25 July 1998. Morgan Kaufmann.

[39] A. M. Geofrion, J. S. Dyer, and A. Feinberg. An interactive approach for multi-criterion optimization with an application to the operation of an academic department. *Management Science*, 19(4):357–368, 1972.

[40] D. Goldberg and K. Deb. A comparison of selection schemes used in genetic algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 69–93, San Mateo, 1991. Morgan Kaufmann.

[41] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison Wesley, 1989.

[42] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multi-modal function optimization. In J. J. Grefenstette, editor, *Proceedings of the* 2^{nd} *International Conference on Genetic Algorithms*, pages 41–49. Lawrence Erlbaum Associates, 1987.

[43] J. J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Trans. on Systems, Man and Cybernetics*, SMC-16:122–128, 1986.

[44] A. B. Hadj-Alouane and J. C. Bean. A genetic algorithm for the multiple-choice integer program. Technical Report TR 92-50, Department of Industrial and Operations Engineering, The University of Michigan, 1992.

[45] Y. Y. Haimes, L.S. Lasdon, and D. A. Wismer. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Trans. on Systems, Man and Cybernetics*, SMC-16:122–128, 1986.

[46] P. Hajela and C. Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.

[47] H. Hamda, F. Jouve, E. Lutton, M. Schoenauer, and M. Sebag. Unstructured representations in evolutionary topological optimum design. *IJAI, The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies, Special Issue on Creative Evolutionary Systems*, 2000.

[48] H. Hamda, O. Rudenko, and M. Schoenauer. Multi-objective evolutionary topological optimum design. In I. Parmee et al., editor, *Adaptive Computing in Design and Manufacture V*, pages 121–132, 2002.

[49] Simon Handley. Classifying nucleic acid sub-sequences as introns or exons using genetic programming. In Christopher Rawlins, Dominic Clark, Russ Altman, Lawrence Hunter, Thomas Lengauer, and Shoshana Wodak, editors, *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology (ISMB-95)*, pages 162–169, Cambridge, UK, 1995. AAAI Press.

[50] Michael Pilegaard Hansen and Andrzej Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Institute of Mathematical Modeling, Technical University of Denmark, 1998.

[51] J. H. Holland. Outline for a logical theory of adaptive systems. *Journal of the association of computing machinery*, 3, 1962.

[52] J. H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, 1975.

[53] A. Homaifar, S. H.-Y. Lai, and X. Qi. Constrained optimization via genetic algorithms. *Simulation*, 62(4):242–254, 1994.

[54] J. Horn, S. N. Nafpliotis, and D. E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, editors, *Proceedings of the First IEEE International Conference on Evolutionary Computation*, volume 1, pages 82–87. IEEE Press, 1994.

[55] J. Horn, N. Nafploitis, and D. Goldberg. A nichad pareto genetic algorithm for multiobjective optimization. In Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, editors, *Proceedings of the First IEEE International Conference on Evolutionary Computation*, pages 82–88. IEEE Press, 1994.

[56] A. Jaszkiewicz. Genetic local search for multi-objective combinatorial optimization. *European Jornal of Operational Research*, 137:50–71, 2002.

[57] F. Jimenez, J. L. Verdegey, and A. F. Gomez-Skarmeta. Evolutionary techniques for constrained multiobjective optimization problems. In GECCO99editors, editor, *Proceedings of the Workshop on multi-Criterion Optimization Using Evolutionary Methods held at GECCO-1999.* Morgan Kaufmann, 1999.

[58] J.A. Joines and C.R. Houck. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs. In Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, editors, *Proceedings of the First IEEE International Conference on Evolutionary Computation*, pages 579–584. IEEE Press, 1994.

[59] L. Kallel. *Convergence des algorithmes génétiques: aspects spatiaux et temporels.* PhD thesis, Ecole Polytechnique, Février 1999.

[60] C. Kane. *Algorithmes génétiques et Optimisation topologique.* PhD thesis, Université de Paris VI, July 1996.

[61] R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. New York: John Wiley, 1976.

[62] J. Knowles and D. Corne. The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization. In *Proceedings of the 1999 Congress On Evolutionary Computation*, pages 98–105. IEEE Service Center, 1999.

[63] J. R. Koza. *Genetic Programming: On the Programming of Computers by means of Natural Evolution*. MIT Press, Massachussetts, 1992.

[64] John R. Koza. Evolution of a computer program for classifying protein segments as transmembrane domains using genetic programming. In Russ Altman, Douglas Brutlag, Peter Karp, Richard Lathrop, and David Searls, editors, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, pages 244–252. AAAI Press, 1994.

[65] S. Koziel and Z. Michalewicz. Evolutionary algorithms, homomorphous mapping and constrained parameter optimization. *Evolutionary Computation*, 7(1):19–44, 1999.

[66] H. T. Kung, F. Luccio, and F. P. Preparata. On finding the maxima of a set of vectors. *Jornal of the Assiciationfor Computing Machinery*, 22(4):469–476, 1975.

[67] F. Kursawe. A variant of evolution strategies for vector optimization. In Hans-Paul Schwefel and Reinhard Männer, editors, *Proceedings of the* 1^{st} *Parallel Problem Solving from Nature*, pages 193–197. Springer Verlag, 1991.

[68] Evelyne Lutton. *Genetic Algorithms and Fractals - Algorithmes Génétiques et Fractales*. 1992. Dossier d'Habilitation à diriger des recherches.

[69] M. S. Manas. Graphical methods of multi-criterion oprimization. *Zeitschrift für Angewandte Mathematik und Mechanik*, 62(5):375–377, 1982.

[70] Brij Masand. Optimising confidence of text classification by evolution of symbolic expressions. In Kenneth E. Kinnear, Jr., editor, *Advances in Genetic Programming*, chapter 21, pages 445–458. MIT Press, 1994.

[71] M. D. McKay, W. J. Conover, , and R. J. Beckman. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21:239–245, 1979.

[72] Z. Michalewicz. *Genetic Algorithms+Data Structures=Evolution Programs*. Springer Verlag, New-York, 1996. 3rd edition.

[73] Z. Michalewicz and N. Attia. Evolutionary optimization of constrained problems. In A.V. Sebald and L.J. Fogel, editors, *Proceedings of the* 3^{rd} *Annual Conference on Evolutionary Programming*, pages 98–108. World Scientific, 1994.

[74] Z. Michalewicz and C. Z. Janikow. Handling constraints in genetic algorithms. In R. K. Belew and L. B. Booker, editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 151–157. Morgan Kaufmann, 1991.

[75] Z. Michalewicz and G. Nazhiyath. Genocop III: A co-evolutionary algorithm for numerical optimization problems with nonlinear constraints. In D. B. Fogel, editor, *Proceedings of the Second IEEE International Conference on Evolutionary Computation*, pages 647–651. IEEE Press, 1995.

[76] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer, Boston, 1999.

[77] H. Myung, J.-H. Kim, and D.B. Fogel. Preliminary investigation into a two-stage method of evolutionary optimization on constrained problems. In J. R. McDonnell, R. G. Rey-

nolds, and D. B. Fogel, editors, *Proceedings of the 4^{th} Annual Conference on Evolutionary Programming*, pages 449–463. MIT Press, 1995.

[78] J. F. Nash. Noncooperative games. *Annals of Mathematics*, 54:289, 1951.

[79] Peter Nordin. A compiling genetic programming system that directly manipulates the machine code. In Kenneth E. Kinnear Jr., editor, *Advances in Genetic Programming*, pages 311–332. MIT Press, 1994.

[80] C. K. Oei, D. Goldberg, and S.-J. Chang. Tournament selection, niching, and the preservation of diversity. Technical report, University of Illinois, Urbana-Champaign, 1991.

[81] J. Paredis. Coevolutionary constraint satisfaction. In Y. Davidor, H.-P. Schwefel, and R. Manner, editors, *Proceedings of the 3^{rd} Conference on Parallel Problems Solving from Nature*, pages 46–55. Springer-Verlag, LNCS 866, 1994.

[82] I. C. Parmee, D. Cevtkovic, A. W. Watson, and C. R. Bonham. Multiobjective satisfaction within an interactive evolutionary design environment. *Evolutionary Computation*, 8(2):197–222, 2000.

[83] I.C. Parmee and G. Purchase. The development of directed genetic search technique for heavily constrained design spaces. In *Proceedings of the Conference on Adaptive Computing in Engineering Design and Control*, pages 97–102. University of Plymouth, 1994.

[84] A. Petrowski. A clearing procedure as a niching method for genetic algorithms. In T. Fukuda, editor, *Proceedings of the Third IEEE International Conference on Evolutionary Computation*, pages 798–803. IEEE press, 1996.

[85] C. Poloni, A. Giurgevich, L. Onesti, and V. Pediroda. Hybridisation of a multiobjective genetic algorithm, a neural network and a classical optimizer for complex design problem in fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*, 186(2-4):403–420, 2000.

[86] S. S. Rao. *Optimization: Theory and Applications.* Wiley, New-York, 1984.

[87] I. Rechenberg. *Evolution Strategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution.* Fromman-Hozlboog Verlag, Stuttgart, 1973.

[88] G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell. *Ingenering Optimization Methods and Applications.* Wiley, New-York, 1983.

[89] Gerald P. Roston. *A Genetic Methodology for Configuration Design.* PhD thesis, Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213-3891, USA, December 1994.

[90] O. Roudenko, M. Schoenauer, T. Bosio, and R. Fontana. A multiobjective evolutionary algorithm for car front end design. In P. Collet C. Fonlupt J-K. Hao E. Lutton and M. Schoenauer, editors, *Artificial Evolution'2001*, pages 205–216, Creusot, France, October 2001. Springer Verlag.

[91] Conor Ryan, J. J. Collins, and Michael O Neill. Grammatical evolution: Evolving programs for an arbitrary language. In Wolfgang Banzhaf, Riccardo Poli, Marc Schoenauer, and Terence C. Fogarty, editors, *Proceedings of the First European Workshop on Genetic Programming*, volume 1391 of *LNCS*, pages 83–95, Paris, 14-15 April 1998. Springer-Verlag.

[92] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In J. J. Grefenstette, editor, *Proceedings of the 1^{st} International Conference on Genetic Algorithms*, pages 93–100, 1985.

[93] M. Schoenauer and Z. Michalewicz. Evolutionary computation at the edge of feasibility. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problem Solving from Nature*, pages 245–254. Springer-Verlag, LNCS 1141, 1996.

[94] M. Schoenauer and Z. Michalewicz. Boundary operators for constrained parameter optimization problems. In Th. Bäeck, editor, *Proceedings of the* 7^{th} *International Conference on Genetic Algorithms*, pages 322–329. Morgan Kaufmann, 1997.

[95] M. Schoenauer and Z. Michalewicz. Sphere operators and their applicability for constrained parameter optimization problems. In B. Porto, editor, *Proceedings of the* 7^{th} *Annual Conference on Evolutionary Programming*, LNCS. Springer Verlag, 1998.

[96] M. Schoenauer and S. Xanthakis. Constrained GA optimization. In S. Forrest, editor, *Proceedings of the* 5^{th} *International Conference on Genetic Algorithms*, pages 573–580. Morgan Kaufmann, 1993.

[97] J. R. Schott. Fault tolerant design using single and multi-criteria genetic algorithms. Master's thesis, Department of Aeronautics and Astronautics, MIT, Boston, MA, 1995.

[98] H.-P. Schwefel. *Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik*. Technical University of Berlin, 1965. Dipl.-Ing. Thesis.

[99] H.-P. Schwefel. Numerische optimierung von computer-modellen mittels der evolutionsstrategie. *Interdisiplinary systems research*, 26, 1977.

[100] H.-P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, New-York, 1981. 1995 – 2^{nd} edition.

[101] M. Sefrioui and J. Periaux. Nash genetic algorithms: Examples and applications. In *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*, pages 509–516, La Jolla Marriott Hotel La Jolla, California, USA, 6-9 2000. IEEE Press.

[102] B. W. Silverman. *Density estimation for statistics and data analysis*. Chapman and Hall, London, 1986.

[103] A. Smith and D. Tate. Genetic optimization using a penalty function. In S. Forrest, editor, *Proceedings of the* 5^{th} *International Conference on Genetic Algorithms*, pages 499–503. Morgan Kaufmann, 1993.

[104] N. Srinvas and K. Deb. Multi-objective function optimizationusing non-dominated sorting genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[105] P.D. Surry, N.J. Radcliffe, and I.D. Boyd. A multi-objective approach to constrained optimization of gas supply networks. In T. Fogarty, editor, *Proceedings of the AISB-95 Workshop on Evolutionary Computing*, volume 993, pages 166–180. Springer Verlag, 1995.

[106] G. Syswerda. Uniform crossover in genetic algorithms. In J. D. Schaffer, editor, *Proceedings of the* 3^{rd} *International Conference on Genetic Algorithms*, pages 2–9. Morgan Kaufmann, 1989.

[107] Astro Teller and Manuela Veloso. PADO: A new learning architecture for object recognition. In Katsushi Ikeuchi and Manuela Veloso, editors, *Symbolic Visual Learning*, pages 81–116. Oxford University Press, 1996.

[108] Bruce Tidor. An analysis of selection procedures with particular attention paid to proportional and boltzmann selection. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo, CA, 1993. Morgan Kaufman.

[109] D. A. Van Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithms: Analyzing the state of the art. *Evolutionary Computation*, 8(2):125–147, 2000.

[110] D. V. Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analysis, and Innovations*. PhD thesis, Air Force Instituteof Technology, Dayton, OH, 1999.

[111] D. Waagen, P. Diercks, and J. McDonnell. The stochastic direction set algorithm: A hybrid technique for finding function extrema. In D. B. Fogel and W. Atmar, editors, *Proceedings of the 1^{st} Annual Conference on Evolutionary Programming*, pages 35–42. Evolutionary Programming Society, 1992.

[112] A. Wildman and G. Parks. A comparative study of selective breeding strategies in a multiobjective genetic algorithm. pages 418–432, 2003.

[113] A. Wright. Genetic algorithms for real parameter optimization. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 205–218. Morgan Kaufmann, 1991.

[114] J. Wright and H. Loosemore. An infeasibility objective for use in constrained pareto optimization. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *Proceedings of the 1^{st} International Conference on Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 256–268, Zurich, Switzerland, March 2001. Springer-Verlag.

[115] X. Yin and N. Germay. A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In *Artificial Neural Nets and Genetic Algorithms*, pages 450–457, Wien, 1993. Springer Verlag.

[116] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zürih, Switzerland, 1999.

[117] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, 8(2):125–148, 2000.

[118] E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratory, ETH, Zurich, Switzerland, 2001.

[119] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms - a comparative case study. pages 292–301, 1998.

[120] E. Zitzler and L. Thiele. Tournament selection, niching, and the preservation of diversity. Technical Report 43, Computer Engineering and Networks Laboratory, ETH, Zurich, Switzerland, 1998.

[121] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms on Test Functions of Different Difficulty. In Annie S. Wu, editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 121–122, Orlando, Florida, 1999.