HAL
open science

Optimisation évolutionnaire multi-objectif parallèle : application à la combustion Diesel

Mouadh Yagoubi

To cite this version:

Mouadh Yagoubi. Optimisation évolutionnaire multi-objectif parallèle : application à la combustion Diesel. Autre [cs.OH]. Université Paris Sud - Paris XI, 2012. Français. NNT : 2012PA112111 . tel-00734108

HAL Id: tel-00734108

https://theses.hal.science/tel-00734108

Submitted on 20 Sep 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ PARIS SUD XI
ECOLE DOCTORALE D'INFORMATIQUE
DE PARIS-SUD

T H È S E

pour obtenir le titre de

Docteur en Sciences

de l'Université de Paris Sud XI
Mention : Informatique

Mouadh Yagoubi

Optimisation évolutionnaire multi-objectif parallèle : application à la combustion Diesel

Thèse dirigée par Marc Schoenauer

préparée à l'INRIA Saclay, équipe TAO et PSA PEUGEOT CITROEN

Jury :

Rapporteurs :	Mohamed Masmoudi	-	MIP-Université de Toulouse
	Frédéric Saubion	-	LERIA-Université d'Angers
Directeur :	Marc Schoenauer	-	TAO-INRIA-Saclay
Encadrant :	Ludovic Thobois	-	Anciennement PSA
Examinateurs :	Laurent Duchamps Delageneste	-	PSA PEUGEOT CITROEN
	Laurent Dumas	-	UVSQ
	Nouredine Melab	-	DOLPHINE-INRIA Lille
	Abdel Lisser	-	LRI

Résumé : Avec la sévérisation des réglementations environnementales sur les émissions polluantes (normes Euro) des moteurs d'automobiles, la nécessité de maitriser les phénomènes de combustion a motivé le développement de la simulation numérique comme outil d'aide à la conception. Tenant compte de la complexité des phénomènes à modéliser, et de l'antagonisme des objectifs à optimiser, l'optimisation évolutionnaire multi-objectif semble être la mieux adaptée pour résoudre ce type de problèmes. Cependant, l'inconvénient principal de cette approche reste le coût très élevé en termes de nombre d'évaluations qui peut devenir très contraignant dans le contexte des optimisations réelles caractérisées par des évaluations très coûteuses.
L'objectif principal de ce travail de thèse est de réduire le coût global des optimisations du monde réel, en explorant la parallélisation des algorithmes évolutionnaires multi-objectifs, et en utilisant les techniques de réduction du nombre d'évaluations (méta-modèles).
Motivés par le phénomène d'hétérogénéité des coûts des évaluations, nous nous proposons d'étudier les schémas d'évolution stationnaires asynchrones dans une configuration parallèle de type « maître-esclave ». Ces schémas permettent une utilisation plus efficace des processeurs sur la grille de calcul, et par conséquent de réduire le coût global de l'optimisation.
Ce problème a été attaqué dans un premier temps d'un point de vue algorithmique, à travers une adaptation artificielle des algorithmes évolutionnaires multi-objectifs au contexte des optimisations réelles caractérisées par un coût d'évaluation hétérogène.
Dans un deuxième temps, les approches développées et validées dans la première partie sur des problèmes analytiques, ont été appliquées sur la problématique de la combustion Diesel qui représente le contexte industriel de cette thèse. Dans ce cadre, deux types de modélisations ont été utilisés : la modélisation phénoménologique 0D et la modélisation multidimensionnelle 3D.
La modélisation 0D a permis par son temps de retour raisonnable (quelques heures par évaluation) de comparer l'approche stationnaire asynchrone avec celle de l'état de l'art en réalisant deux optimisations distinctes. Un gain de l'ordre de 42 % a été réalisé avec l'approche stationnaire asynchrone.
Compte tenu du temps de retour très coûteux de la modélisation complète 3D (quelques jours par évaluation), l'approche asynchrone stationnaire déjà validée a été directement appliquée. L'analyse physique des résultats a permis de dégager un concept intéressant de bol de combustion permettant de réaliser un gain en termes d'émissions polluantes.

Table des matières

1 Introduction **1**

I Algorithmes évolutionnaires multi-objectif parallèles **5**

2 Contexte et démarche **7**

2.1 Introduction 8
2.2 Notions préliminaires 8
2.2.1 Problème d'optimisation multi-Objectif 8
2.2.2 Principe de dominance 9
2.2.3 Optimalité de Pareto 10
2.3 Méthodes classiques 10
2.3.1 Méthode d'agrégation 10
2.3.2 Méthode de ε-contraintes 11
2.3.3 Mesure de Chebyshev 11
2.4 Algorithmes évolutionnaires 11
2.4.1 Vocabulaire et Terminologie 12
2.4.2 Principe de fonctionnement 12
2.4.3 Le dilemme exploration/exploitation 13
2.4.4 Les procédures de sélection 14
2.5 Les moteurs d'évolution 14
2.5.1 Algorithmes générationnels 15
2.5.2 Algorithmes stationnaires (*"steady-state"*) 15
2.6 Optimisation évolutionnaire multi-objectif 15
2.6.1 Indicateurs de Performance 15
2.6.2 Procédures de comparaison 17
2.7 Etat de l'art des algorithmes évolutionnaires multi-objectifs 19
2.7.1 NSGA-II 19
2.7.2 SPEA2 20
2.7.3 IBEA 21
2.7.4 ε-MOEA 22
2.7.5 MO-CMA-ES 23
2.8 Fonctions tests 24
2.9 Comparaisons des algorithmes de l'état de l'art 26
2.9.1 Réglages 27
2.9.2 Résultats 27
2.9.3 Discussions 28
2.10 Algorithmes évolutionnaires et modèles parallèles 29
2.10.1 Modèle Maître-esclave 29
2.10.2 Modèle en îlots 31

2.10.3 Modèle totalement distribué 31
2.11 Hétérogénéité des coûts d'évaluation et asynchronicité des AEMO . . 31
2.12 Démarche 32
2.13 Conclusion 34

3 AEMO asynchrones avec coûts d'évaluation hétérogènes 35
3.1 Introduction 35
3.2 Parallélisation des AEMO avec le modèle Maître-esclave 37
3.2.1 NSGA-II Stationnaire 37
3.2.2 MO-CMA-ES Stationnaire 37
3.2.3 AEMO Asynchrones Stationnaires 37
3.3 Implémentation des modèles de test artificiels 38
3.3.1 L'hétérogénéité aléatoire "Rand-VC" 40
3.3.2 L'hétérogénéité proportionnelle aux valeurs des objectifs "Eps-VC" 40
3.3.3 L'hétérogénéité liée à une région "Region-VC" 41
3.4 Conditions expérimentales générales 41
3.4.1 Réglages algorithmiques 41
3.4.2 Représentation graphique des résultats 42
3.5 Accélération de convergence asynchrone 42
3.6 Contexte hétérogène 46
3.6.1 Le modèle Eps-VC 46
3.6.2 Le modèle Région-VC 48
3.7 Conclusion 53

4 AEMO asynchrones et méta-modèles 55
4.1 Introduction 55
4.2 Modèles d'approximation : État de l'art 56
4.2.1 Approximation quadratique des moindres carrés 57
4.2.2 Modèles de krigeage 57
4.2.3 SVM (*Support Vector Machines*) 58
4.3 Méta-modèles & AEMO générationnels 61
4.4 Méta-modèles & asynchronicité 62
4.5 Réglages expérimentaux 63
4.5.1 Réglages du méta-modèle 63
4.5.2 Réglages de l'AEMO 65
4.6 Résultats 65
4.6.1 Réglage de N_{iter} et N_{eval} 65
4.6.2 Comparaison des algorithmes 66
4.7 Conclusion 67

II Optimisation de la combustion Diesel **71**

5 Généralités **73**
5.1 Introduction . . . 73
5.2 Cycle d'un moteur Diesel . . . 73
5.3 Mécanisme d'auto-inflammation . . . 74
5.4 Formation des polluants . . . 74
5.5 Modélisation de la combustion . . . 75

6 Modélisation phénoménologique (0D) **77**
6.1 Introduction . . . 77
6.2 Présentation de l'outil PDF0D . . . 78
6.2.1 Equilibre thermodynamique . . . 79
6.2.2 Mélange des particules . . . 79
6.2.3 Pertes Thermiques aux Parois . . . 80
6.2.4 Injection du carburant . . . 80
6.2.5 Combustion . . . 81
6.3 Définition du problème d'optimisation . . . 81
6.3.1 Paramètres de l'optimisation . . . 81
6.3.2 Objectifs de l'optimisation . . . 83
6.3.3 Contrainte de l'optimisation . . . 83
6.4 Réglages . . . 84
6.5 Résultats . . . 85
6.5.1 Variabilité du temps de calcul . . . 86
6.5.2 Coût global de l'optimisation . . . 87
6.5.3 Comparaison des fronts de Pareto . . . 88
6.5.4 Analyse physique . . . 88
6.6 Conclusion . . . 90

7 Modélisation multidimensionnelle (3D) **93**
7.1 Introduction . . . 93
7.2 Travaux connexes . . . 94
7.3 Spécificités de la modélisation Diesel . . . 95
7.3.1 Symétrie de la chambre de combustion . . . 95
7.3.2 Injection à soupapes fermées . . . 96
7.4 Présentation de la chaine de modélisation 3D . . . 96
7.5 Définition du problème d'optimisation . . . 97
7.5.1 Les paramètres de l'optimisation . . . 97
7.5.2 Les contraintes de l'optimisation . . . 100
7.6 La chaine d'optimisation . . . 103
7.7 Implémentation de la chaine d'optimisation . . . 104
7.7.1 La plateforme CINES . . . 104
7.7.2 Le serveur PSA . . . 105
7.7.3 Discussion . . . 105

7.8 Réglages 106
7.9 Résultats 106
7.9.1 Front de Pareto 106
7.9.2 Analyse physique 107
7.9.3 Méta-modèle : résultats préliminaires 111
7.10 conclusion 112

8 Conclusion **115**

Bibliographie **119**

A Courbes des expérimentations réalisées avec le modèle artificiel "Eps-VC" **127**

B Courbes des expérimentations sur l'accélération avec différentes tailles de grille de calcul **133**
B.1 accélération en fonctions de la taille de la grille 133
B.2 Courbes des expérimentations réalisées avec le modèle séquentiel . . 138

Table des figures

2.1 Principe de fonctionnement d'un AE [Schoenauer 2003] 13
2.2 Indicateur de l'hypervolume 17
2.3 Gauche : front de Pareto trouvé par les AEMO pour ZDT1 après 50000 évaluations. Droite : phénomène de convergence prématurée de la fonction IHR3 avec l'algorithme NSGA-II 28
2.4 Evolution de l'indicateur de l'hypervolume moyen pour les différents AEMO avec les benchmark ZDT et IHR 30
2.5 Durées des évaluations des individus de la première génération aléatoire 33

3.1 Évolution de l'indicateur de l'hypervolume médian pour AS-NSGA-II sur Rand-VC-ZDT3 pour différentes tailles de queue. Gauche : en termes de nombre d'évaluations. Droite : en termes de temps écoulé. 43
3.2 Évolution de l'indicateur de l'hypervolume médian pour AS-MO-CMA-ES sur Rand-VC-ZDT3 pour différentes tailles de queue. Gauche : en termes de nombre d'évaluations. Droite : en termes de temps écoulé. 43
3.3 Evolution de l'indicateur de l'hypervolume moyen pour les variantes de NSGA-II sur ZDT3. gauche : toutes les 4 variantes vs nombre d'évaluation. droite : variantes générationnelle et asynchrone stationnaire avec le modèle Eps-VC. 50
3.4 Evolution de l'indicateur de l'hypervolume moyen pour les variantes de NSGA-II sur IHR1. gauche : toutes les 4 variantes vs nombre d'évaluation. droite : variantes générationnelle et asynchrone stationnaire avec le modèle Eps-VC. 50
3.5 Effets néfastes de l'asynchronicité sur Region-VC-ZDT3 (région coûteuse : $f_1 \in [0.3, 0.5]$). gauche : captures après 5000, 25000 et 50000 évaluations avec AS-NSGA-II. droite : tous les individus visités lors d'un essai avec AS-MOCMA-ES. 51
3.6 Sélection basée sur la durée :résultats de DBS-AS-NSGA-II sur Region-VC-ZDT3 (région coûteuse : $f_1 \in [0.3, 0.5]$. gauche : population après 5000, 25000 and 50000 évaluations pour $P_s = 0.5$. droite : évolution de l'indicateur de l'hypervolume moyen pour différentes valeurs de P_s 52
3.7 évolution de l'indicateur de l'hypervolume moyen pour DBS-AS-NSGA-II avec différentes valeurs de P_s sur Rand-VC-ZDT3. 52
3.8 Résultats des variantes de DBS-AS-NSGA-II avec des valeurs différentes de P_s sur Region-VC-ZDT6 avec la région coûteuse $f_1 \in [0.8, 1]$. gauche : évolution de l'indicateur de l'hypervolume moyen. droite : population après 5000, 15000 and 50000 évaluations des essais typiques. 53

4.1 réglage de la fonction perte dans le cas d'un SVM linéaire [Scholkopf 2002] . . . 59
4.2 Evolution de l'indicateur de l'hypervolume moyen pour différents couplets de valeurs de N_{iter} et N_{eval} . . . 68
4.3 Evolution de l'indicateur de l'hypervolume moyen pour les deux variantes utilisant la régression SVM et la version standard avec les benchmarks ZDT et IHR . . . 69

5.1 Phases d'un cycle Diesel . . . 73
5.2 Diagramme de Pischinger [Pischinger 1988] . . . 75

6.1 Variation de la PMI en fonction de la Masse de carburant injectée sur plusieurs points de calcul . . . 84
6.2 Durées des évaluations des individus de la première génération aléatoire (en minutes) . . . 86
6.3 Comparaison des durées d'optimisation à différents nombres d'évaluations . . . 87
6.4 Plan suies-NOx du front de Pareto . . . 88
6.5 Plan conso-suies du front de Pareto . . . 89
6.6 Plan conso-NOx du front de Pareto . . . 89
6.7 Evolution des paramètres : Pression d'admission (g), Température d'admission (h), EGR (c) et début d'injection (d) au cours de l'optimisation . . . 92

7.1 Illustration des paramètres géométriques sur la géométrie du cas de référence . . . 98
7.2 Le mouvement de swirl . . . 99
7.3 Comparaison des valeurs estimées et des valeurs réelles du paramètre profondeur de bol . . . 101
7.4 Illustration du risque de chevauchement entre les injections main et split avec la définition classique du paramètre d'injection Dwell-main-split . . . 102
7.5 Illustration de la nouvelle définition du paramètre d'injection Dwell-main-split . . . 103
7.6 Durées des évaluations des individus de la première génération aléatoire (heures) . . . 107
7.7 Plan suies-NOx du front de Pareto . . . 108
7.8 Plan consommation-NOx du front de Pareto . . . 108
7.9 Plan suies-consommation du front de Pareto . . . 109
7.10 Forme géométrique du cas de référence . . . 110
7.11 Forme géométrique de l'individu optimal . . . 110
7.12 Comparaison des grandeurs physiques de l'individu de référence et de l'individu optimal . . . 112
7.13 Effet du swirl sur l'injection . . . 113

7.14 Fraction massique des suies (plan vertical) avec une iso-surface du taux d'oxygène à 17 %- 380 degrés vilebrequin 113
7.15 Fraction massique des suies (plan vertical) avec une iso-surface du taux d'oxygène à 17 %- 420 degrés vilebrequin 114
7.16 Indicateur de l'hypervolume calculé pour les deux algorithmes AS-NSGA-II et Reg-SVM-AS-NSGA-II pour un budget de 40 évaluations 114

A.1 Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de NSGA-II avec le benchmark Eps-VC-ZDT 128
A.2 Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de NSGA-II avec le benchmark Eps-VC-IHR 129
A.3 Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de MO-CMA-ES avec le benchmark Eps-VC-ZDT 130
A.4 Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de MO-CMA-ES avec le benchmark Eps-VC-IHR 131

B.1 Evolution de l'indicateur de l'hypervolume moyen pour AS-NSGA-II avec le benchmark Rand-VC-ZDT pour différentes tailles de queue d'attente 134
B.2 Evolution de l'indicateur de l'hypervolume moyen pour AS-NSGA-II avec le benchmark Rand-VC-IHR pour différentes tailles de queue d'attente 135
B.3 Evolution de l'indicateur de l'hypervolume moyen pour AS-MO-CMA-ES avec le benchmark Rand-VC-ZDT pour différentes tailles de queue d'attente 136
B.4 Evolution de l'indicateur de l'hypervolume moyen pour AS-MO-CMA-ES avec le benchmark Rand-VC-IHR pour différentes tailles de queue d'attente 137
B.5 Évolution de l'indicateur de l'hypervolume moyen pour ZDT1 et ZDT2 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500 . . . 139
B.6 Évolution de l'indicateur de l'hypervolume moyen pour ZDT3 et ZDT6 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500 . . . 140
B.7 Évolution de l'indicateur de l'hypervolume moyen pour IHR1 et IHR2 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500 141
B.8 Évolution de l'indicateur de l'hypervolume moyen pour IHR3 et IHR6 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500 142

Liste des tableaux

2.1 Les fonctions du benchmark ZDT 26
2.2 Les fonctions du benchmark IHR 26

3.1 accélérations de la convergence pour un niveau d'hypervolume donnée sur les fonctions Rand-VC-ZDT (ratio des temps écoulés des algorithmes asynchrones entre $nProc$ et 1 processeur). 44
3.2 accélérations de la convergence pour un niveau d'hypervolume donnée sur les fonctions Rand-VC-IHR (ratio des temps écoulés des algorithmes asynchrones entre $nProc$ et 1 processeur). 45
3.3 Résultats moyens de l'indicateur de l'hypervolume obtenus sur 30 runs après 50000 évaluations avec les variantes de NSGA-II. Les exposants indiquent la signification statistique de la différence avec la colonne correspondante en utilisant le test de Kruskal-Wallis avec $\alpha = 0.01$. 49
3.4 Résultats moyens de l'indicateur de l'hypervolume obtenus sur 30 runs après 50000 évaluations avec les variantes de MO-CMA-ES. Les exposants indiquent la signification statistique de la différence avec la colonne correspondante en utilisant le test de Kruskal-Wallis avec $\alpha = 0.01$. 49

6.1 Tableau récapitulatif des paramètres de l'optimisation 0D 83

7.1 Tableau récapitulatif des paramètres de l'optimisation 3D 100
7.2 Comparaison des paramètres du cas de référence et du cas optimal . 109

Chapitre 1

Introduction

Depuis quelques années, la réduction de notre empreinte environnementale est une des priorités majeures de l'humanité. Cette priorité se retrouve dans l'élaboration de normes sur les émissions polluantes des moteurs d'automobiles de plus en plus sévères (normes Euro) et sur les multiples incitations envers les consommateurs à se tourner vers des véhicules consommant moins et émettant moins de CO2 (plan climat, bonus/malus automobile, conférences internationales de Kyoto et de Copenhague). Pour répondre à ces exigences, c'est-à-dire réduire la consommation et les émissions polluantes des moteurs, les constructeurs automobiles essayent de maîtriser de plus en plus finement les phénomènes de combustion dans les moteurs à piston grâce notamment à la simulation numérique. Cette dernière offre la possibilité d'explorer de nouvelles technologies des moteurs avec un coût relativement faible comparativement aux essais réalisés sur les bancs moteurs. L'optimisation numérique reste l'une des méthodes les plus efficaces permettant de réaliser cette exploration en améliorant les valeurs des objectifs à optimiser, en l'occurrence les émissions polluantes et la consommation du carburant dans le cas de la combustion Diesel. Néanmoins, la grande complexité des phénomènes de combustion (turbulence, mélange, chimie...etc) mis en jeu rend quasiment impossible l'utilisation des méthodes d'optimisation classiques. D'autre part, le problème d'optimisation de la combustion Diesel met en jeu des objectifs qui sont naturellement en conflit : pour les émissions polluantes, les deux principales grandeurs à optimiser sont les émissions des oxydes de carbone (NOx) et les suies, or la réduction des suies générés dans un moteur Diesel s'accompagne souvent d'une augmentation des NOx formés et vice versa. Pour résoudre ce type de problèmes caractérisés par une forte non-linéarité et plusieurs objectifs antagonistes, les méthodes globales stochastiques comme les algorithmes évolutionnaires multi-objectifs (AEMO) [Deb 2001, Coello 2002] semblent représenter une bonne approche grâce à leur robustesse et à leur flexibilité. De plus, ces méthodes ne requièrent pas des hypothèses très fortes sur la fonction objectif contrairement aux méthodes classiques (gradient...etc). Néanmoins, le principal inconvénient des AEMO réside dans le coût très élevé en termes de nombre d'évaluations de la fonction objectif requis pour atteindre des solutions satisfaisantes. Pour les problèmes réels très coûteux où l'évaluation de la fonction objectif est réalisée à travers des simulations numériques qui peuvent durer des heures, voire des jours, cet inconvénient peut devenir très contraignant voire dans certains cas, rendre l'optimisation irréalisable.
Parallèlement, le monde du matériel informatique est confronté à ce qui est souvent appelé "la fin de la loi de Moore", qui prévoyait une augmentation exponentielle de

la capacité de calcul que peut avoir un processeur. Face à cette problématique critique, le calcul parallèle semble offrir une bonne alternative pour réaliser les calculs couteux sur les grilles de calcul qui comportent plusieurs nœuds avec plusieurs processeurs. Pour les AEMO, le calcul parallèle représente une piste très intéressante puisque le processus de parallélisation à gros grains de ces derniers est immédiat et ne requiert pas de repenser la structure de l'algorithme. Plusieurs modèles de parallélisation des AEMO ont été proposés dans la littérature parmi lesquels le modèle "maître-esclave" reste le plus utilisé grâce à sa mise en œuvre plus simple que celle des autres modèles tels que le modèle en îlots ou le modèle totalement distribué. Dans ce modèle, le processus maître distribue l'évaluation des objectifs aux esclaves. L'application simple et directe du modèle "maître-esclave" utilise la version générationnelle des AEMO. Dans cette version, le passage d'une génération à l'autre est effectué une fois que tous les individus de la génération actuelle ont été évalués. Cependant, dans le cas des applications réelles en général, et du problème d'optimisation de la combustion Diesel en particulier, le coût des évaluations des individus est rarement homogène. Cette hétérogénéité peut être due à la différence de capacité de calcul des serveurs ou à la non-linéarité des simulations. Dans un tel contexte d'hétérogénéité, utiliser un algorithme générationnel revient à laisser plusieurs processeurs inoccupés en attendant que l'individu le plus lent finisse son évaluation et ceci contribue à augmenter d'avantage le coût de l'optimisation déjà caractérisée par une fonction objectif très coûteuse. Une solution à ce problème est d'utiliser les schémas d'évolution stationnaires dans lesquels un seul individu est généré à chaque génération. Dans le contexte d'une application parallèle caractérisée par une hétérogénéité du coût d'évaluation des individus, l'algorithme stationnaire devient asynchrone : les individus ne regagnent pas la population dans le même ordre que celui dans lequel ils ont été créés. Avec cette configuration, la grille de calcul est utilisée plus efficacement car aucun processeur ne reste inoccupé. L'objectif principal de ce travail de thèse est de réduire le coût global des optimisation du monde réel, pour faire face au nombre important d'évaluations devant être réalisées d'une part, et au coût élevé des évaluations réelles d'autre part, en explorant la parallélisation des AEMO stationnaires.

Nous tenterons de répondre à cette problématique tout d'abord d'un point de vue algorithmique à travers l'adaptation des AEMO parallèles au contexte des problématiques réelles caractérisées par un coût d'évaluation hétérogène. Par la suite, nous appliquerons les approches algorithmiques développées sur deux problématiques de la combustion Diesel qui représentent le contexte industriel de cette thèse.

Pour la partie algorithmique, la première étape sera consacrée à l'étude des AEMO parallèles asynchrones sur des benchmarks de fonctions test de l'état de l'art (chapitre 3), en simulant artificiellement l'hétérogénéité du coût d'évaluation sur la grille de calcul. Dans la deuxième étape nous nous intéressons aux méta-modèles qui permettent de réduire le nombre global d'évaluations, en les adaptant toujours au contexte de l'hétérogénéité du coût d'évaluations, toujours en utilisant des benchmarks de fonction test (chapitre 4).

Après avoir validé les approches algorithmiques proposées dans un contexte de

benchmarks de fonctions analytiques, nous les appliquerons sur deux types de modélisation d'un problème réel lié à la combustion Diesel.
Après une brève introduction au domaine de la combustion dans les moteurs Diesel (chapitre 5), nous étudierons dans la première partie la problématique de l'optimisation de la combustion Diesel en utilisant la modélisation phénoménologique, appelée modélisation 0D (chapitre 6). Bien que ce type de modélisation ne permette pas de prendre en compte certains aspects importants de la combustion Diesel telles que la géométrie et l'aérodynamique de la chambre, le temps de retour "raisonnable" qui est de l'ordre de quelques heures par évaluation, nous offre l'opportunité de comparer l'approche asynchrone stationnaire avec celle de l'état de l'art et de valider le gain en termes de coût global de l'optimisation en réalisant deux optimisation distinctes.
Le deuxième type de modélisation que nous proposons d'étudier concerne la modélisation multidimensionnelle ou 3D (chapitre 7). Avec ce modèle, le coût de calcul d'une évaluation est de l'ordre de quelques jours, ce qui donne lieu à un budget de quelques mois pour l'optimisation complète. De ce fait, une seule optimisation utilisant l'approche asynchrone, déjà validée sur les fonctions analytiques et sur la problématique 0D, sera réalisée.
Nous commencerons dans le prochain chapitre par dresser l'état de l'art de l'optimisation évolutionnaire multi-objectif en décrivant les principales méthodes ainsi que les différents modèles de parallélisation qui permettent de les utiliser dans le contexte des problématiques réelles. Après comparaison de ces méthodes, nous détaillerons nos motivations concernant l'hétérogénéité du temps d'évaluation ainsi que la démarche que nous proposons pour attaquer cette problématique.

Première partie

Algorithmes évolutionnaires multi-objectif parallèles

Chapitre 2

Contexte et démarche

Sommaire

2.1 Introduction **8**
2.2 Notions préliminaires **8**
2.2.1 Problème d'optimisation multi-Objectif 8
2.2.2 Principe de dominance 9
2.2.3 Optimalité de Pareto 10
2.3 Méthodes classiques **10**
2.3.1 Méthode d'agrégation 10
2.3.2 Méthode de ε-contraintes 11
2.3.3 Mesure de Chebyshev 11
2.4 Algorithmes évolutionnaires **11**
2.4.1 Vocabulaire et Terminologie 12
2.4.2 Principe de fonctionnement 12
2.4.3 Le dilemme exploration/exploitation 13
2.4.4 Les procédures de sélection 14
2.5 Les moteurs d'évolution **14**
2.5.1 Algorithmes générationnels 15
2.5.2 Algorithmes stationnaires (*"steady-state"*) 15
2.6 Optimisation évolutionnaire multi-objectif **15**
2.6.1 Indicateurs de Performance 15
2.6.2 Procédures de comparaison 17
2.7 Etat de l'art des algorithmes évolutionnaires multi-objectifs **19**
2.7.1 NSGA-II 19
2.7.2 SPEA2 20
2.7.3 IBEA 21
2.7.4 ε-MOEA 22
2.7.5 MO-CMA-ES 23
2.8 Fonctions tests **24**
2.9 Comparaisons des algorithmes de l'état de l'art **26**
2.9.1 Réglages 27
2.9.2 Résultats 27
2.9.3 Discussions 28
2.10 Algorithmes évolutionnaires et modèles parallèles **29**
2.10.1 Modèle Maître-esclave 29
2.10.2 Modèle en îlots 31
2.10.3 Modèle totalement distribué 31

2.11 Hétérogénéité des coûts d'évaluation et asynchronicité des AEMO 31
2.12 Démarche 32
2.13 Conclusion 34

2.1 Introduction

Dans la plus part des problèmes pratiques d'optimisation, plusieurs critères sont à prendre en considération afin d'obtenir une solution satisfaisante. Comme son nom l'indique, l'optimisation multi-objectif a pour but d'optimiser plusieurs objectifs simultanément. Ces objectifs sont dans le cas général en conflit : l'amélioration d'un objectif provoque la détérioration d'un autre objectif. Par conséquent, le résultat final de l'optimisation n'est plus donné par une solution unique mais plutôt par un ensemble de solutions qui représentent chacune un compromis entre les différents objectifs à optimiser.
Dans ce chapitre, nous présentons l'état de l'art de l'optimisation évolutionnaire multi-objectif. Dans un premier temps, nous rappelons quelques notions élémentaires de l'optimisation multi-objectif, tels que les notions de dominance et de l'optimalité de Pareto. Par la suite, après avoir présenté le vocabulaire et le principe général de fonctionnement des algorithmes évolutionnaires, nous nous focalisons sur l'adaptation de ces derniers au cas multi-objectif et nous passerons en revue les principaux travaux de l'état de l'art dans ce domaine. Nous finissons ce chapitre d'état de l'art en décrivant les différents modèles de parallélisation des algorithmes évolutionnaires qui permettent de les utiliser dans le contexte des problématiques du monde réel.

2.2 Notions préliminaires

Nous présentons dans cette section quelques notions élémentaires de l'optimisation multi-objectif, en donnant quelques définitions et en décrivant le vocabulaire du domaine.

2.2.1 Problème d'optimisation multi-Objectif

Un problème d'optimisation multi-objectif est un problème qui possède plusieurs fonctions objectif qui sont à minimiser ou à maximiser et un certain nombre de contraintes à satisfaire. La forme générale d'un problème d'optimisation multi-objectif est donnée par le système d'équations suivant :

$$\begin{cases} Minimiser/Maximiser f_m(X) & m = 1, 2, ..., M; \\ g_j(x) \geq 0, & j = 1, 2, ..., J; \\ h_k(x) = 0, & k = 1, 2, ..., K; \\ x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1, 2, ..., n \end{cases}$$

Le vecteur x est un vecteur de n variables de décision $x = (x_1, x_2, ..., x_n)^T$. $x_i^{(L)}$ et $x_i^{(U)}$ sont les bornes inférieure et supérieure de la variable x_i respectivement. Ces variables définissent *l'espace de décision* ou *l'espace de recherche* D. Généralement, un élément de l'espace de recherche est appelé *solution possible ou potentielle.*
Les termes $g_j(x)$ et $h_k(x)$ sont les fonctions contraintes. Les contraintes d'inégalité sont traitées en tant que contraintes de type "supérieur ou égal" étant donné que les contraintes de type "inférieur ou égal" peuvent être traitées par dualité. Une solution x qui ne satisfait pas la *totalité* des $(J + K)$ contraintes est dite *solution infaisable.* L'ensemble des solutions faisables constitue la *région faisable* S. Le vecteur $f(x) = (f_1(x), f_2(x), ..., f_M(x))^T$ est le vecteur objectif. Chacune des M fonctions objectif est soit à maximiser ou à minimiser selon le problème traité. En utilisant le principe de dualité [Deb 1995], un problème de maximisation peut être ramené à un problème de minimisation en multipliant la fonction objective par -1. Dans le reste de ce document, nous supposons que toutes les fonctions objectif sont à minimiser.

2.2.2 Principe de dominance

Le principe de dominance est utilisé par la plus part des algorithmes d'optimisation multi-objectifs pour comparer deux solutions.

Définition (*Principe de la dominance*). Une solution $x^{(i)}$ *domine* une autre solution $x^{(j)}$ si les deux conditions suivantes sont vérifiées :

1 . $f_m(x^{(i)}) \leq f_m(x^{(j)})\ \forall m \in \{1, ...M\}$

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

Si $x^{(i)}$ domine $x^{(j)}$, cette relation est notée $x^{(i)} \prec x^{(j)}$.
D'autres relations qui peuvent être dérivées de la relations de dominance sont énoncées ci-dessous.

Définition (*Dominance Stricte*) Une solution $x^{(i)}$ *domine strictement* une solution $x^{(j)}$ si et seulement si : $f_m(x^{(i)}) < f_m(x^{(j)})\ \forall m \in \{1, ...M\}$. Cette relation est notée $x^{(i)} \prec\prec x^{(j)}$

Définition (*Dominance Faible*) Une solution $x^{(i)}$ *domine faiblement* une solution $x^{(j)}$ si et seulement si : $f_m(x^{(i)}) \leq f_m(x^{(j)})\ \forall m \in \{1, ...M\}$. Cette relation est notée $x^{(i)} \preceq x^{(j)}$

Définition (*ε-Dominance*) Une solution $x^{(i)}$ *ε-domine* une solution $x^{(j)}$ si et seulement si : $f_m(x^{(i)}) \leq \varepsilon \cdot f_m(x^{(j)})\ \forall m \in \{1, ...M\}$. Cette relation est notée $x^{(i)} \preceq_\varepsilon x^{(j)}$

Définition (*ε-Dominance additive*) Une solution $x^{(i)}$ *ε-domine additivement* une solution $x^{(j)}$ si et seulement si : $f_m(x^{(i)}) \leq \varepsilon + f_m(x^{(j)})\ \forall m \in \{1, ...M\}$. Cette relation est notée $x^{(i)} \preceq_{\varepsilon+} x^{(j)}$

Il est évident que si une solution A $\preceq$ B le décisionnaire préfèrera la solution A à la solution B.

2.2.3 Optimalité de Pareto

Pour un ensemble de solutions fini, toutes les solutions peuvent être comparées deux à deux selon le principe de dominance, et nous pouvons déduire quelle solution domine l'autre. A la fin, nous obtenons un ensemble où aucune des solutions ne domine l'autre, cet ensemble est appelé *ensemble des solutions non dominées.*

Définition (*Ensemble des solutions non-dominées*). Soit P un ensemble de solutions, l'ensemble des solutions non dominées $P^{'}$ est l'ensemble des solutions non-dominées par aucun autre membre de l'ensemble P.

Si l'ensemble P représente la totalité de l'espace de recherche S, l'ensemble des solutions non-dominées $P^{'}$ est appelé *ensemble Paret-Optimal* dans l'espace de décision ou *front de Pareto* dans l'espace des objectifs. Il s'agit de l'ensemble des solutions que l'utilisateur cherche à obtenir à travers l'optimisation.

Définition (*Ensemble Pareto-Optimal*). l'ensemble Pareto-optimal est l'ensemble des solutions non-dominées de l'espace de recherche faisable S .

2.3 Méthodes classiques

Nous proposons dans cette section de survoler rapidement quelques méthodes classiques de l'optimisation multi-objectif non-évolutionnaire. A noter que toute ces méthodes reposent sur la transformation du problème initial en un problème d'optimisation mono-objectif.

2.3.1 Méthode d'agrégation

Cette méthode populaire consiste à ramener un problème multi-objectif à un problème d'optimisation d'une combinaison linéaire des objectifs initiaux. Cette combinaison est caractérisée souvent par une pondération qui traduit l'importance relative des objectifs.

$$F_m(x) = \sum_{m=1}^{M} w_m f_m(x) \tag{2.1}$$

où les $w_m \leq 0$ sont les points respectifs des M objectifs à optimiser.
Cependant, cette méthode permet de résoudre uniquement des problèmes simples

avec des front de Pareto convexe. Pour la plus part des problèmes non-linéaires ou des problèmes dont la surface de Pareto contient des régions non-convexes cette méthode ne permet pas de trouver les solutions de Pareto.

2.3.2 Méthode de ε-contraintes

Cette méthode proposée dans [Haimes 1986] suggère de reformuler le problème d'optimisation sous la forme suivante : $\begin{cases} min_{x \in S} f_\mu(x) \\ f_m(x) \leq \varepsilon_m \quad j = 1, ..., M, m \neq \mu \end{cases}$

où $\mu \in \{1, ..., M\}$ et les paramètres ε_m sont à définir par l'utilisateur.
En d'autres termes, une des fonctions à optimiser est retenue comme unique objectif, tandis que les objectifs restants sont transformés en contraintes. Contrairement à la méthode d'agrégation pondérée, la méthode de ε-contraintes est capable de trouver des solutions de Pareto appartenant à des régions non-convexes. Cependant, le choix du vecteur ε est très important et lié à certaines difficultés relatives à sa position par rapport au front de Pareto.

2.3.3 Mesure de Chebyshev

La mesure de Chebyshev permet de résoudre le problème pondéré pour lequel la fonction objectif à minimiser est :

$$Argmin \{max_m(w_m|f_m(x) - z_m|)\} \tag{2.2}$$

où les w_m sont les poids de pondération des M objectifs, et z est le vecteur idéal qui représente les bornes inférieures de chaque objectif dans l'espace faisable.
Cette méthode, tout comme la méthode de ε-contraintes permet d'accéder aux parties non-convexes du front de Pareto. Cependant, avec cette méthode la fonction objectif est non différentiable, ce qui pose problème lors de l'utilisation des méthodes de gradient. De plus, cette méthode requiert l'optimisation de chaque objectif séparément afin de définir le vecteur idéal.

2.4 Algorithmes évolutionnaires

Les algorithmes évolutionnaires (AE) sont des algorithmes stochastiques d'optimisation inspirés du paradigme darwinien de l'évolution naturelle. Selon la théorie darwinienne, les individus les plus aptes survivent à la sélection naturelle et se reproduisent d'une génération à l'autre. En termes d'optimisation, l'évolution se traduit par un processus itératif de recherche de l'optimum dans l'espace de recherche.
Il existe plusieurs familles historiques d'algorithme évolutionnaires qui se sont développées de façon indépendante : La Programmation Evolutionnaire (EP) introduite par L.J. Fogel [Fogel 1966] dans les années 60, les algorithmes génétiques (GA) proposés par J.Holland [Holland 1975] aux USA, et popularisés un peu plus de dix ans plus tard par son élève D.E. Goldberg [Goldberg 1989], les stratégies

d'évolutions (ES) inventées par I.Rechenberg [I.Rechenberg 1973] dans les années 70 à Berlin, et enfin la programmation génétique (GP pour *Genetic Programming*) proposée par J.Koza [Koza 1992] [Koza 1994].

2.4.1 Vocabulaire et Terminologie

Nous présentons dans cette section le vocabulaire spécifique aux algorithmes évolutionnaires, inspiré du parallèle réalisé avec les principes de l'évolution naturelle.
- Les points de l'espace de recherche D sont appelés des *individus* ;
- Un ensemble fini d'individus est appelé *population* ;
- La fonction objectif à optimiser est appelée fonction *performance*, ou fonction *fitness* ;
- Le calcul de la performance d'un individu est appelé *évaluation* ;
- La *génération* correspond à une population en une certaine itération ;
- L'*évolution* est un processus itératif de recherche des individus optimaux ;
- Les *opérateurs de variation* sont utilisés pour générer de nouveaux individus et sont le plus souvent catégorisés en deux types d'opérateurs : le *croisement* qui consiste à échanger des parties composantes (gènes) entre deux ou plusieurs individus, et la *mutation* qui consiste à la modification d'un ou plusieurs gènes d'un individu ;
- La *sélection* est le processus de choix des individus, basé sur leur performance ;
- Le *remplacement* est le processus de formation d'une nouvelle population à partir des parents et des enfants.

2.4.2 Principe de fonctionnement

Le processus d'optimisation évolutionnaire commence par l'étape de l'**initialisation** : un nombre fini d'individus p choisis généralement par tirage aléatoire uniforme dans D forment la population initiale P_0. Après **évaluation** de la population initiale (calcul de la performance), certains individus (les plus performants) sont choisis lors de l'étape de la **sélection**. L'application des **opérateurs de variation** (croisement et mutation) permet de créer un nouvel ensemble d'individus, appelé "population d'enfants" (à noter que cette étape est toujours stochastique). Ces enfants vont être évalués à leurs tour et combinés avec leur parents afin de décider lesquels d'entre eux vont remplacer certains parents pour faire partie de la génération suivante, il s'agit de l'étape de **remplacement**. la figure 2.1 illustre le principe général de fonctionnement d'un algorithme évolutionnaire.

A noter que dans la plupart des applications réelles (chapitre 6, 7), le coût de calcul des algorithmes évolutionnaires provient essentiellement de l'étape d'évaluation. A titre d'exemple, si l'on souhaite faire évoluer une population de quelques

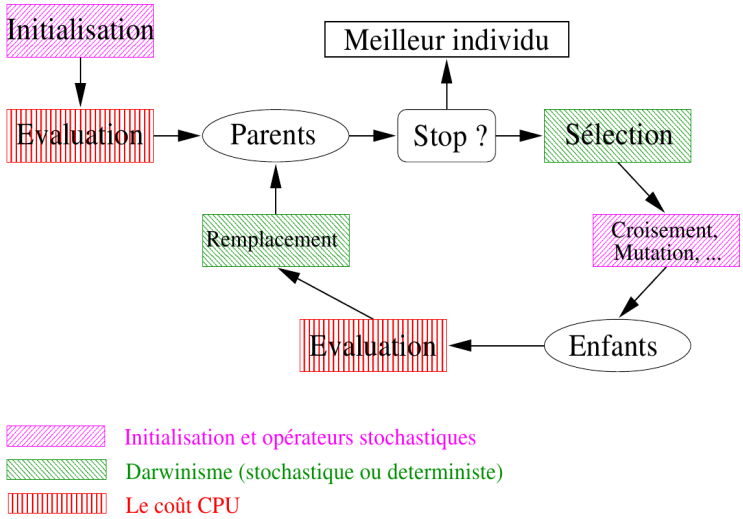


FIGURE 2.1 – Principe de fonctionnement d'un AE [Schoenauer 2003]

dizaine d'individus pendant quelques dizaine de générations, quelques milliers de calculs de la fonction performance (ou fitness) doivent être réalisés souvent à travers des évaluations coûteuses.

2.4.3 Le dilemme exploration/exploitation

A chaque étape de l'algorithme évolutionnaire, un compromis entre l'**exploration** de l'espace de recherche et l'**exploitation** des meilleurs individus obtenus doit être réalisé.
L'exploitation des meilleurs individus obtenus revient à chercher dans leurs voisinages des individus qu'on espère encore meilleurs. Cependant, cette technique ne permet pas de préserver la diversité génétique dans la population, et guide l'algorithme vers une convergence qui peut être prématurée vers un optimum local. L'exploration quant à elle cherche à préserver la diversité de la population, et à explorer de nouvelles régions qui peuvent s'avérer prometteuses pour l'algorithme. Toutefois, une utilisation excessive de cette technique entraîne la non-convergence de l'algorithme.
Il est donc important de maintenir un certain équilibre entre l'exploration et l'exploitation afin de garantir une bonne efficacité de l'algorithme évolutionnaire. Malheureusement, il n'existe pas de règles universelles pour ce type de réglage et le dosage "exploitation/exploration" est souvent défini empiriquement à travers l'expérience et dépend du problème en question.

2.4.4 Les procédures de sélection

Le darwinisme artificiel comprend les deux étapes de sélection et de remplacement, qui sont totalement indépendantes de l'espace de recherche. Lors de la sélection, il s'agit de favoriser les meilleurs individus de manière stochastique (un individu peut être sélectionné plusieurs fois). Le remplacement peut impliquer soit la population des enfants seulement, soit la totalité des enfants et des parents. Plusieurs méthodes de sélections ont été proposées dans la littérature. Nous nous contenterons de présenter dans ce qui suit deux méthodes bien connues : le tirage de roulette et la sélection par tournoi.

2.4.4.1 Le tirage de roulette

C'est une méthode stochastique qui a été introduite par Holland [**?**]. Elle consiste à attribuer à chaque individu une probabilité d'être sélectionné proportionnellement à sa performance : La boule est lancée dans la roulette et l'individu représentant le secteur dans le quel la boule finit sa course sera choisi. Cette méthode présente toutefois plusieurs inconvénients, notamment le problème de mise à l'échelle de la fonction objective. A titre d'exemple, si dans une population, une solution a une fitness bien meilleure que le reste des individus, la probabilité de choisir cette "super-solution" est proche de 1, et par conséquent cette solution dominera l'ensemble des individus sélectionnés avec ses copies. De l'autre part, si toute les solutions ont plus au moins la même fitness, elles auront la même probabilité d'être sélectionnées [Deb 2001]. Pour y remédier, des procédures ne dépendant que de la comparaison entre les individus ont été proposées par la suite.

2.4.4.2 La sélection par tournoi

Dans la sélection par tournoi, des tournois sont joués entre deux ou plusieurs solutions et la meilleure solution est sélectionnée. La taille du tournoi T est donnée par le nombre de solutions retenues lors de cette procédure. Le choix de T permet de faire varier la *pression sélective* : une grande valeur de T augmente les chances de sélectionner les meilleurs individus, tandis qu'une petite valeur de T donne aux individus moins performants (au sens de la fitness) une chance d'être sélectionnés. La sélection par tournoi est de nos jours l'une des méthodes de sélection les plus utilisées dans le domaine des algorithmes évolutionnaires.

2.5 Les moteurs d'évolution

Un moteur d'évolution représente une réunion des procédures de sélection et de remplacement, deux étapes qui ne peuvent pas être dissociées lors de l'analyse des algorithmes évolutionnaires. Plusieurs types de moteurs d'évolution peuvent être obtenus selon la taille de la population, le nombre d'enfants et les procédures de sélection et de remplacement. Selon le nombre d'enfants impliqués dans la procédure

de sélection et de remplacement, deux variantes principales de l'état de l'art peuvent être distinguées : les algorithme générationnels et les algorithmes stationnaires.

2.5.1 Algorithmes générationnels

Dans les schémas d'évolutions générationnels, à chaque génération N parents sont sélectionnés par l'algorithme en utilisant une méthode de sélection stochastique (certains parents peuvent donc être sélectionnés plusieurs fois) pour donner naissance à exactement N enfants en appliquant les opérateurs de variations. Ces N enfants remplacent par la suite les N parents lors de la procédure de remplacement (déterministe).

2.5.2 Algorithmes stationnaires (*"steady-state"*)

Dans les algorithmes à schéma d'évolution stationnaire un seul enfant est généré à chaque génération en utilisant un ou deux parents sélectionnés généralement par tournoi. Après évaluation de l'enfant, ce dernier est intégré dans la population, afin de remplacer un parent en utilisant un tournoi inversé [Syswerda 1991]. Dans ce type de tournoi, l'individu le moins performant est sélectionné pour disparaitre (éventuellement le plus vieux).

2.6 Optimisation évolutionnaire multi-objectif

Comme décrit précédemment dans ce chapitre, les algorithmes évolutionnaires manipulent une population de solutions au lieu d'une seule solution. Cette propriété leur permet d'être bien adaptés à la résolution des problèmes d'optimisation multi-objectif où l'optimum est représenté par un ensemble de points (ensemble de Pareto) et non pas par une solution unique. Dans la littérature, plusieurs approches ont été proposées afin d'adapter les algorithmes évolutionnaires au cas de l'optimisation multi-objectif. L'objectif principal de ces différentes approches est le même : obtenir une bonne approximation du front optimal tout en assurant une bonne diversité des solutions le long de ce dernier. Ces méthodes diffèrent toutefois entre elles dans la façon d'aborder cette problématique en utilisant des procédure de sélection différentes. Avant de présenter principaux AEMO, nous nous focalisons dans un premier temps sur ce qui fait la différence entre ces algorithmes à savoir les différentes méthodes de comparaison pouvant être utilisées pour sélectionner les solutions. Comme ces méthodes de sélection font parfois recourt aux indicateurs de performance, nous commençons d'abord par présenter ces derniers.

2.6.1 Indicateurs de Performance

L'évaluation de la performance des AEMO peut avoir plusieurs aspects, comme par exemple la qualité des résultats obtenus, ou le temps de calcul nécessaire. Nous nous focalisons seulement sur le premier aspect étant donné que le temps d'exécution des algorithmes devient négligeable dès lors qu'il s'agit d'une application réelle

ou l'évaluation des objectifs peut durer de quelques minutes à quelques jours.
Plusieurs indicateurs de performance ont été proposés dans la littérature afin de comparer la qualité des AEMO. Parmi ces indicateurs, nous retrouvons ceux qui s'intéressent seulement à la diversité des solutions (*spacing* [Schott 1995], maximum spread [Zitzler 1999a]) et ceux qui s'intéressent plutôt à la convergence ou la proximité au front de Pareto (*Error Ratio* [Veldhuizen 1999], *Set Coverage Metric* [Zitzler 1999a]). Par la suite la communauté des AEMO a vu apparaitre de nouveaux indicateurs qui évaluent à la fois la convergence et la diversité des AEMO (L'indacteur de l'hypervoulume [Zitzler 1999b], l'indicateur Epsilon [Zitzler 2003], l'indicateur R [Hansen 1998]).
Les indicateurs de performance peuvent être classés aussi selon qu'ils sont unaires ou binaires : Un indicateur de qualité I unaire affecte à chaque ensemble de solutions une valeur réelle $I : \Omega \to \mathbb{R}$. Un indicateur binaire est un indicateur qui affecte une valeur réelle à une paire d'ensemble de solutions. Cette valeur réelle représente la mesure de qualité d'un ensemble de solutions par rapport à un autre $I : \Omega \times \Omega \to \mathbb{R}$.
Nous présentons dans ce qui suit les indicateurs les plus utilisés de l'état de l'art à savoir l'indicateur de l'hypervolume et l'indicateur Epsilon.

2.6.1.1 Indicateur Hypervolume

L'indicateur de l'hypervolume (I_H) [Zitzler 1999b] mesure le volume de la portion faiblement dominée par un ensemble de point A, dans l'espace des objectifs. Le calcul de ce volume nécessite la désignation d'un point de référence, qui soit de préférence dominé par la totalité des points de l'ensemble A (figure 2.2(a)). L'indicateur de l'hypervolume est souvent calculé par rapport à un ensemble de référence R (figure 2.2(b)), cet indicateur est noté I_H^- et est défini comme suit :

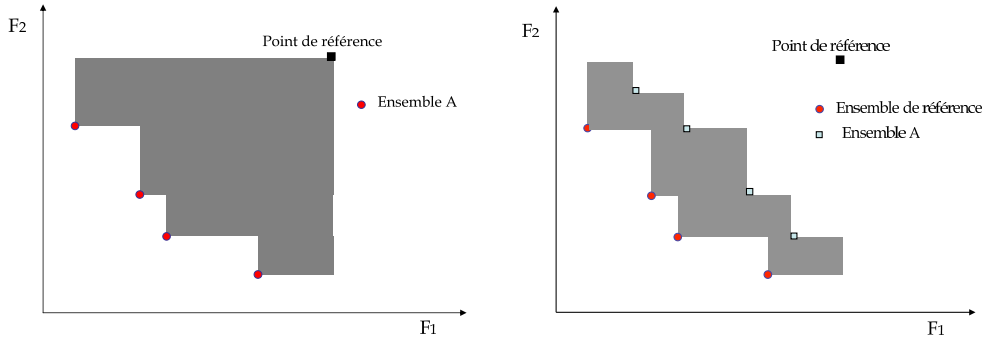
$$I_H^-(A) = I_H(R) - I_H(A) \tag{2.3}$$

où plus la valeur de I_H^- est petite, plus la qualité est meilleure. L'indicateur de l'hypervolume permet de prendre en compte à la fois la convergence de l'algorithme et la diversité des solutions trouvées. Toutefois, le coût de calcul est élevé : la complexité est exponentiellement proportionnelle au nombre d'objectifs. [Knowles 2006]

2.6.1.2 Indicateur Epsilon

L'indicateur Epsilon a été introduit dans [Zitzler 2003] et comprend deux variantes : multiplicative et additive. Pour les deux versions il existe une forme unaire et une autre binaire. Cet indicateur est basé sur la notion de l'ε-dominance définie dans la section (2.2.2). La forme binaire multiplicative de l'indicateur epsilon, notée $I_\varepsilon(A, B)$, calcule la valeur minimum du facteur ε par laquelle chaque point dans B peut être multiplié, pour que l'ensemble résultant de cette transformation soit faiblement dominé par A :

$$I_\varepsilon(A, B) = \inf_{\varepsilon \in \mathbb{R}} \left\{ \forall z^2 \in B, \exists z^1 \in A : z^1 \preceq_\varepsilon z^2 \right\} \tag{2.4}$$



(a) I_H (b) I_H^-

FIGURE 2.2 – Indicateur de l'hypervolume

Où z représente un vecteur objectif et $\preceq_\varepsilon$ la relation de l'epsilon-dominance multiplicative.
De la même façon un indicateur unaire d'un ensemble A, $I_\varepsilon^1(A)$ peut être défini comme suit :

$$I_\varepsilon^1(A) = I_\varepsilon(A, R) \tag{2.5}$$

Où R représente un ensemble Référence. Par analogie, l'indicateur epsilon unaire-additif $I_{\varepsilon+}^1$ peut être obtenu en remplaçant la relation de l'ε-dominance multiplicative $\preceq_\varepsilon$ par celle de l'ε-dominance additive $\preceq_\varepsilon$ dans l'équation 2.4.

2.6.2 Procédures de comparaison

Plusieurs procédures de sélection ont été proposées dans la littérature des AEMO. Cependant nous pouvons distinguer deux catégories principales : la première utilise une comparaison basée sur la dominance + un deuxième critère de diversité tandis que la deuxième catégorie utilise uniquement une mesure basée sur les indicateurs de performance qui permettent d'évaluer la convergence et la diversité à la fois (indicateur de l'hypervolume, indicateur epsilon).
Pour les algorithmes qui utilisent la comparaison basée sur la dominance, comme cette dernière est une relation d'ordre partiel, nous nous retrouvons avec une catégorie d'individus qui sont non-comparables entre eux. Pour cela, un deuxième critère de comparaison est utilisé afin de départager les individus qui sont non-dominés entre eux. Ce critère concerne la densité des solutions et a pour but de favoriser la diversité du front de Pareto. Nous présentons dans ce qui suit les techniques utilisées pour comparer les individus au sens de la dominance ainsi que celles relatives au deuxième critère qui est la densité des solutions, et nous finissons cette section par la procédure de comparaison basée uniquement sur les indicateurs de performance.

2.6.2.1 Comparaison basée sur la dominance

Rang de Pareto [Deb 2002] Avec ce critère de comparaison, les individus non-dominés d'un ensemble A se voient attribuer une valeur de rang égale à 1. Par la suite, en considérant les individus non dominés de l'ensemble A sauf les individus du 1er rang, on obtient les individus du 2ème rang. La procédure est répétée jusqu'à ce que tous les individus soient attribués à un rang.

Le *Strength* [Zitzler 2001] Avec ce critère le classement des individus en utilisant le principe de dominance se fait via la valeur de *Strength* S_i qui représente pour chaque individu i appartenant à un ensemble A, le nombre de solutions qu'il domine :

$$S(i) = |\{j : j \in A \ \ et \ \ i \prec j\}| \tag{2.6}$$

2.6.2.2 Critères de densité

Critère de surpeuplement Ce critère proposé dans [Deb 2002] permet d'ordonner des individus non-départageables par la relation de dominance (non-dominés entre eux) selon une mesure de densité qui permet d'évaluer leur degré de contribution à la diversité.
La distance de surpeuplement d_i d'une solution i est une mesure de l'espace de recherche autour de i qui n'est pas occupé par aucune autre solution dans la population.

Critère du k^{ime} voisin Ce critère proposé dans [Zitzler 2001] est une adaptation de la technique du "k^{ime} plus proche voisin" [Silverman 1986]. La densité d'un point est une fonction décroissante de la distance le séparant du k^{ime} plus proche point. le critère de densité est défini dans [Zitzler 2001] comme étant l'inverse de la distance au k^{ime} point voisin. Plus précisément, pour chaque individu i, les distances (dans l'espace des objectifs) le séparant de tous les autres individus j de la population sont calculées et ordonnées dans une liste. Le k^{me} élément correspond à la distance recherchée, dénotée σ_i^k. La valeur de k est définie habituellement comme étant la racine carrée de la taille de l'ensemble des individus.

Critère de contribution à l'hypervolume Ce critère proposé dans [Igel 2007a] repose sur l'utilisation de l'indicateur de l'hypervolume décrit ci-dessus. Ce critère mesure la qualité d'un point en calculant sa contribution à l'hypervolume de l'ensemble de la population A. La mesure de la contribution d'un individu a à l'hypervolume d'une population A est donnée par :

$$\Delta_{I_H^-}(a, A) = I_H^-(A) - I_H^-(A \backslash \{a\}) \tag{2.7}$$

2.6.2.3 Indicateurs de performance

Une autre possibilité pour comparer les individus entre eux est d'utiliser les indicateurs de performance comme mesure de comparaison des individus. L'utilisation d'un indicateur évaluant la convergence et la diversité des individus en même temps (Indicateur de l'hypervolume, indicateur epsilon) permet de considérer une mesure unique pour établir un ordre total dans l'ensemble des individus de la population.

2.7 Etat de l'art des algorithmes évolutionnaires multi-objectifs

Nous présentons dans cette section l'état de l'art des algorithmes évolutionnaire multi-objectif en se focalisant seulement sur les principaux derniers travaux dans ce domaine à savoir : NSGA-II, SPEA2, IBEA, ε-MOEA et MO-CMA-ES.

2.7.1 NSGA-II

NSGA-II (*Non-dominated Sorting Genetic Algorithm*) [Deb 2002] a longtemps été l'algorithme phare dans le domaine de l'optimisation évolutionnaire multi-objectif. Il tient son appellation de l'algorithme NSGA qui a été proposé auparavant par les mêmes auteurs [Srinvas 1994]. l'algorithme NSGA reprend l'idée proposée par Goldberg sur l'utilisation du concept de classement par dominance dans les algorithmes génétiques [Goldberg 1989]. Dans la plus part des aspects NSGA-II est très différent de NSGA, cependant le nom a été gardé pour indiquer les origines de cette approche. Nous nous contenterons dans ce qui suit de présenter les différentes étapes de l'algorithme NSGA-II.
Dans NSGA-II la population des enfants Q_t est d'abord créée en utilisant la population des parents P_t. Les deux populations sont ensuite réunies pour former la population mixte R_t de taille $2N$. Cette population est triée selon le critère du rang de Pareto décrit ci-dessus pour former les fronts successifs : le premier front $\mathcal{F}_1$ correspond à l'ensemble des solutions non-dominées de R_t. En considérant le reste des individus dans R_t après avoir enlever ceux de $\mathcal{F}_1$, et après avoir réaliser un nouveau tri de dominance, nous obtenons le deuxième front $\mathcal{F}_2$ constitué des individus non-dominés de l'ensemble $(R_t \backslash \mathcal{F}_1)$. Cette procédure est répétée jusqu'à ce que tous les individus de R_t soient attribués à un front. Par la suite, la nouvelle population P_{t+1} est créée et est remplie au fur et à mesure avec les différents fronts successifs. Comme la taille de la population R_t est $2N$, les fronts successifs ne peuvent pas intégrer en totalité la nouvelle population qui doit être de taille N. Ces fronts seront tout simplement éliminés. Cependant, la taille du dernier front considéré peut être supérieure aux nombres de cases vides à remplir dans la nouvelle population. Dans ce cas, le critère de surpeuplement décrit dans la section précédente sera utilisé pour choisir parmi les solutions du dernier front, celles qui vont intégrer la nouvelle population afin de favoriser les solutions dans les régions les moins peuplées du front

considéré dans le but d'améliorer la diversité des solutions.
La procédure de NSGA-II est résumée dans l'algorithme 1.

Algorithm 1 NSGA-II

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0|$=N
2: **repeat**
3: $Q_t \leftarrow$ variation (Select(P_t)). // sélection parentale :Tournoi de taille 2 ; variation : croisement, mutation.
4: $R_t \leftarrow P_t \cup Q_t$
5: $\mathcal{F} \leftarrow$ Tri-dominance(R_t) // $\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, ...)$
6: $P_{t+1} \leftarrow \emptyset$, $i \leftarrow 1$
7: **while** $|P_{t+1}| + |\mathcal{F}_i| < N$ **do**
8: $P_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i$; $i \leftarrow i + 1$
9: **end while**
10: $P_{t+1} \leftarrow$ Tri-surpeuplement(P_{t+1})
11: $P_{t+1} \leftarrow P_{t+1}[0 : N]$ // choisir les N premiers éléments
12: $t \leftarrow t + 1$
13: **until** Critère d'arrêt rencontré

2.7.2 SPEA2

SPEA2 (*Strenght Pareto Evolutionary Algorithm*) [Zitzler 2001] représente une version corrigée de l'algorithme SPEA [Zitzler 1998] proposé auparavant par les mêmes auteurs. SPEA2 repose sur l'utilisation d'une archive représentée par une population externe A_t de taille $N_{archive}$. Cette archive de taille fixe est destinée à contenir un nombre limité de solutions non-dominées trouvées par l'algorithme au cours de l'optimisation. A chaque itération, les nouveaux individus non-dominés de la population P_t sont comparés aux membres de l'archive A_t en utilisant le critère de dominance. Si le nombre d'individus non-dominés n'est pas suffisant, l'archive est complétée par les meilleurs individus dominés.
La comparaison des individus dans SPEA2 se fait comme dans le cas de NSGA-II en utilisant la relation de dominance + un critère de diversité. Le classement des individus en utilisant le principe de dominance se fait via la valeur de *strength* S_i décrite dans la section précédente :

$$S(i) = |\{j : j \in P_t \cup A_t \ \ et \ \ i \prec j\}| \tag{2.8}$$

Sur la base des valeurs de *strength* calculées, un classement entre les individus est effectué sur la base de la grandeur R_i calculée pour chaque individu i et définie par :

$$R(i) = \sum_{j \in P_t \cup A_t, j \prec i} S(j) \tag{2.9}$$

$R(i) = 0$ correspond à un individu non dominé, tandis que une valeur élevée de $R(i)$ correspond à un individu dominé par plusieurs individus. Ce critère de comparaison

trouve sa limite dans le cas où plusieurs individus ne dominent pas les uns les autres. Dans ce cas, le critère du kième voisin décrit précédemment sera utilisé comme critère de diversité, afin de départager les individus avec les mêmes valeurs de $R(i)$ (non dominés entre eux). La densité de chaque solution i est définie par :

$$D(i) = \frac{1}{\sigma_i^k + 2} \tag{2.10}$$

Où σ_i^k est la distance recherchée et k est définie habituellement comme étant la racine carrée de la taille de l'ensemble $P_t \cup A_t$ i.e. $k = \sqrt{N + N_{archive}}$. Finalement, La valeur globale du critère de comparaison est donnée par :

$$F(i) = R(i) + D(i) \tag{2.11}$$

Le schéma générale de l'algorithme SPEA2 est résumé dans l'algorithme 2.

Algorithm 2 SPEA2

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0|=N$, $A(t) \leftarrow \emptyset$ $|A(t)|= N_{archive}$
2: **repeat**
3: $A_{t+1} \leftarrow Nondom(P_t \cup A_t)$
4: $Tri_F(A_t)$ // Trier en utilisant la formule donnée par l'équation 2.11
5: **if** $|A_{t+1}| > N_{archive}$ **then**
6: $A_{t+1} \leftarrow A_{t+1}[0 : N_{archive}]$ // tronquer l'archive
7: **else**
8: $A_{t+1} \leftarrow A_{t+1} \cup Tri_F(Dom(P_t \cup A_t)[0 : N_{archive} - |A_{t+1}|])$ // rajouter les meilleurs individus dominés
9: **end if**
10: $P_{t+1} \leftarrow$ variation(select(A_{t+1}))
11: $t \leftarrow t + 1$
12: **until** Critère d'arrêt rencontré

2.7.3 IBEA

IBEA (*Indicator Based Evolutionary Algorithm*) est un AEMO qui a été proposé dans [Zitzler 2004]. Cet algorithme a pour objectif d'apporter une meilleure solution au dilemme "exploration vs exploitation" auquel sont confrontés les AEMO en proposant une généralisation du concept de dominance. L'idée principale de IBEA est de formaliser les préférences entre individus en utilisant un indicateur de performance binaire de qualité I arbitrairement choisi. Avec cette option, l'algorithme n'aura pas besoin d'utiliser une technique de préservation de la diversité en tant que deuxième critère de comparaison. Cependant, la question qui se pose est de savoir comment I pourrait être intégré dans le calcul de la *fitness* dans un AEMO.
Une possibilité serait d'attribuer à chaque individu une valeur de *fitness* $F(x)$ correspondant à la mesure de la "perte en qualité" si cet individu avait été retiré de la

population.

$$F^{'}(x^1) = \sum_{x^2 \in P \setminus \{x^1\}} I(\{x^2\}, \{x^1\}) \tag{2.12}$$

Les auteurs retiennent finalement une formule modifiée qui amplifie l'influence des solutions non-dominées sur les solutions dominées :

$$F(x^1) = \sum_{x^2 \in P \setminus \{x^1\}} -e^{-I(\{x^2\},\{x^1\})/k} \tag{2.13}$$

Où $k > 0$ est un facteur d'échelle qui dépend de l'indicateur I utilisé et du problème à optimiser. Pour le choix de l'indicateur, les auteurs de l'algorithme comparent deux exemples : l'indicateur de l'hypervolume et l'indicateur epsilon. Les résultats obtenus par les auteurs ont montré que les deux variantes permettent d'obtenir de meilleurs résultats que ceux obtenus avec NSGA-II et SPEA2.
L'algorithme 3 résume les principales étapes de IBEA.

Algorithm 3 IBEA

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0|$=N,
2: **repeat**
3: $Q_t \leftarrow$ variation(Select(P_t)) // sélection parentale ; variation : croisement, mutation
4: $P_{t+1} \leftarrow P_t \cup Q_t$
5: $Tri_F(P_{t+1})$, // Trier en utilisant la formule donnée par l'équation 2.13
6: **while** $|P_{t+1}| > N$ **do**
7: $P_{t+1} \leftarrow P_{t+1} \setminus \{x^* \leftarrow \arg\max_x F(x)\}$
8: $Update_F(P_{t+1})$ // actualiser les valeurs de F
9: **end while**$t \leftarrow t + 1$
10: **until** Critère d'arrêt rencontré

2.7.4 ε-MOEA

ε-MOEA est un AEMO qui a été proposé dans [Deb 2003b]. Cet algorithme utilise un schéma d'évolution stationnaire et est basé sur le concept de l'ε-dominance introduit dans [Laumanns 2002] (voir section 2.2.2) . L'idée principale de ε-MOEA est proposer une approche qui permet d'assurer la diversité des solutions obtenues avec un plus faible coût de calcul par rapport à NSGA-II et SPEA2. L'espace de recherche est divisé en un certain nombre de boites, et la diversité est maintenue en s'assurant qu'une boite ne contient qu'une seule solution au plus.
ε-MOEA considère deux populations qui évoluent en même temps : une population principale de l'algorithme évolutionnaire $P(t)$ et une population archive $A(t)$ (où t représente le compteur des générations). A chaque itération de l'algorithme, deux solutions sont choisies, la première p à partir de la population $P(t)$ et la deuxième a à partir de l'archive $A(t)$. Par la suite, λ solutions sont créées en combinant les solutions p et a dénotées : $(c_i, i = 1, 2, ..., \lambda)$. Chacune des solutions créées est

comparée avec les membres de l'archive $A(t)$ et la population $P(t)$ dans le but de tester leur possible inclusion. Plusieurs stratégies sont utilisées selon le cas : Pour intégrer l'archive $A(t)$, chaque solution c_i est comparée avec chaque membre de l'archive au sens de l'ε-dominance. Pour intégrer la population $P(t)$, chaque solution c_i est comparée aux membres de la population principale au sens de la dominance. Si la solution c_i domine une ou plusieurs solutions de la population, elle remplace une de ces solutions aléatoirement. Si la solution c_i est dominée par au moins une solution de la population, elle sera rejetée. Dans le troisième cas ou les deux premiers test échouent, la solution c_i remplace une solution choisie aléatoirement. Les principales étape de ε-MOEA sont résumées dans l'algorithme 4

Algorithm 4 ε-MOEA

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0|$=N, $A_0 \leftarrow \varepsilon$-nondominés($P_0$).
2: **repeat**
3: $\{p_1, p_2\} \leftarrow U(P_t)$
4: $p \leftarrow$ Select-dominance(p_1, p_2); $a \leftarrow U(A_t)$
5: $\{c_i, i = 1, 2, ..., \lambda\} \leftarrow$ variation(p, a)
6: **for** i=1,2, . . . ,λ **do**
7: Remplacement-archive(c_i, A_t)
8: Remplacement-population(c_i,P_t)
9: **end for**$t \leftarrow t + 1$
10: **until** Critère d'arrêt rencontré

Algorithm 5 Remplacement-archive(c_i, A_t)

1: **for** $i = 1, 2, ..., \lambda$ **do**
2: **if** $\exists a \in A(t) : c_i \preceq_\varepsilon a$ **then**
3: $A_{t+1} \leftarrow (A_t \cup c_i) \backslash \{a \in A_t : c_i \preceq_\varepsilon e\}$
4: **else**
5: **if** $\exists a \in A_t : a \preceq_\varepsilon c_i$ **then**
6: c_i est rejetée
7: **else**
8: Remplacement-population(c_i, A_t)
9: **end if**
10: **end if**
11: **end for**

2.7.5 MO-CMA-ES

L'algorithme MO-CMA-ES (*MultiObjective-Covariance Matrix Adaptation-Evolution Strategy*) proposé Dans [Igel 2007a] est une extension de l'algorithme mono-objectif CMA-ES [Hansen 2001] proposé auparavant. C'est l'un des algorithmes les plus récents de l'état de l'art des AEMO. MO-CMA-ES transpose au

Algorithm 6 Remplacement-population(c_i,P_t)

1: **for** $i = 1, 2, ..., \lambda$ **do**
2: **if** $\exists p \in P_t : c_i \prec p$ **then**
3: $P_{t+1} \leftarrow (P_t \cup c_i) \backslash \{p\}$
4: **else**
5: **if** $\exists p \in P_t : p \prec c_i$ **then**
6: c_i est rejetée
7: **else**
8: $P_{t+1} \leftarrow (P_t \cup c_i) \backslash \{p \leftarrow U(P_t\}$
9: **end if**
10: **end if**
11: **end for**

contexte multi-objectif, les principes de l'algorithme CMA-ES basé sur l'utilisation des stratégies d'évolutions. L'algorithme CMA-ES repose sur l'adaptation, au cours des itérations, de la matrice de co-variance de la distribution et des pas de transition aux caractéristiques du paysage de la fitness locale.
Plusieurs configurations de stratégies d'évolution peuvent être utilisées dans MO-CMA-ES. Celle que nous présentons ici est la variante $\mu \times (1 + 1)$-MO-CMA-ES. L'algorithme commence par évaluer une population initiale générée aléatoirement. A chaque génération t, chacun des μ parents, génère $\lambda = 1$ enfant. Après évaluation des enfants générés, les parents et les enfants sont combinés pour former l'ensemble Q_t. La taille du pas de mutation d'un parent et de son enfant est actualisée si les mutations ont permis d'avoir un enfant meilleur que le parent selon le principe de dominance $\prec$. La matrice de covariance de l'enfant généré est actualisée par la suite en prenant en compte la mutation récemment effectuée. L'ensemble Q_t est trié selon le principe de dominance, et les meilleurs μ individus sont sélectionnés pour former la population des parents de la génération suivante. Cependant, dans le cas où le principe de dominance ne permet pas de départager les individus entre eux (les individus sont non-dominés entre eux), un deuxième critère de comparaison est utilisé afin de préserver la diversité des solutions. Deux variantes de MO-CMA-ES peuvent dans ce cas être distinguées selon le deuxième critère de comparaison choisi : La variante c-MO-CMA-ES qui utilise le critère de surpeuplement proposé dans NSGA-II et la variante s-MO-CMA-ES qui utilise la mesure de contribution à l'hypervolume comme second critère de comparaison.
L'algorithme 7 retrace les principales étapes de MO-CMA-ES décrites ci-dessus.

2.8 Fonctions tests

Plusieurs fonctions analytiques de tests ont été proposées dans la communauté de l'optimisation évolutionnaire multi-objectif dans le but de pouvoir comparer les AEMO entre eux. Nous citons à titre d'exemple les problèmes de Schaffer SCH1 et SCH2 [Schaffer 1984], le problème FON de Fonseca et Fleming [Fonseca 1995] et le

Algorithm 7 MO-CMA-ES

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0|$=N
2: **repeat**
3: **for** $i = 1, ..., N$ **do**
4: $q_i^{(t)} \leftarrow p_i^{(t)}$
5: **end for**
6: $Q_t \leftarrow \left\{ q_i^{(t)}, p_i^{(t)} \mid 1 \leq i \leq N \right\}$
7: Actualiser le pas de mutation
8: Actualiser la matrice de covariance
9: $Q_t \leftarrow$ Tri-dominance(Q_t)
10: $Q_t \leftarrow$ Tri-diversité(Q_t) // critère de surpeuplement ou de l'hypervolume
11: $P_{t+1} \leftarrow Q_t[0 : N]$
12: $t \leftarrow t + 1$
13: **until** Critère d'arrêt rencontré

problème KUR de Krusawe [Kursawe 1990].
Par la suite, plusieurs benchmarks de fonctions test représentant différentes formes du front de Pareto ont été proposés. Dans le cadre de cette thèse, nous retenons les deux benchmarks connus de l'état de l'art : ZDT et IHR.
Le benchmark ZDT tient son appellation des initiales des trois chercheurs qui l'ont proposé, en l'occurrence Zitzler, Deb et Thiele [Zitzler 2000]. Ce benchmark comporte des problèmes de test bi-objectifs avec des formes différentes du front de Pareto (Table 2.1). Le benchmark IHR (Igel, Hansen et Roth) proposé plus tard dans [Igel 2007a], propose une généralisation du benchmark ZDT en considérant une rotation de l'espace de recherche. La fonction à minimiser est $y = Ox$, où O est une matrice de rotation orthogonale (Table 2.2).

Les fonctions auxiliaires h, h_f et h_g utilisées par les problèmes du benchmark IHR sont données par les équations 2.14 , 2.15, et 2.16 respectivement :

$$h : \mathbb{R} \rightarrow [0, 1] \qquad x \mapsto (1 + exp\frac{-x}{\sqrt{n}})^{-1} \tag{2.14}$$

$$h_f : \mathbb{R} \rightarrow \mathbb{R} \qquad x \mapsto \begin{cases} x & Si \ |y_1| \preceq y_{max} \\ 1 + |y_1| & ailleurs \end{cases} \tag{2.15}$$

Sachant que $y_{max} = 1/max_j(|o_{1j}|)$

$$h_g : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0} \qquad x \mapsto \frac{x^2}{|x| + 0.1} \tag{2.16}$$

Table 2.1 – Les fonctions du benchmark ZDT

Problème	fonctions objectif
ZDT1	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - \sqrt{x_1/g(x)}]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$
ZDT2	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - (x_1/g(x))^2]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$
ZDT3	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - \sqrt{x_1/g(x)} - \frac{x_1}{g(x)} sin(10\pi x_1]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$
ZDT6	$f_1(x) = 1 - exp(-4x_1)sin^6(6\pi x_1)$ $f_2(x) = g(x)[1 - (f_1(x)/g(x))^2]$ $g(x) = 1 + 9[(\sum_{i=2}^{n} x_i)/(n-1)]^{0.25}$

Table 2.2 – Les fonctions du benchmark IHR

Problème	fonctions objectif
IHR1	$f_1(x) = \|y_1\|$ $f_2(x) = g(y)h_f(1 - \sqrt{h(x_1)/g(x)})$ $g(y) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$
IHR2	$f_1(x) = \|y_1\|$ $f_2(x) = g(y)h_f(1 - (y_1/g(x))^2)$ $g(y) = 1 + 9(\sum_{i=2}^{n} h_g(y_i))/(n-1)$
IHR3	$f_1(x) = \|y_1\|$ $f_2(x) = g(y)h_f(1 - \sqrt{h(y_1)/g(x)} - \frac{h(y_1)}{g(x)} sin(10\pi y_1)$ $g(y) = 1 + 9(\sum_{i=2}^{n} h_g(y_i))/(n-1)$
IHR6	$f_1(x) = 1 - exp(-4\|y_1\|)sin^6(6\pi y_1)$ $f_2(x) = g(y)(1 - (f_1(x)/g(y))^2)$ $g(y) = 1 + 9[(\sum_{i=2}^{n} h_g(y_i))/(n-1)]^{0.25}$

2.9 Comparaisons des algorithmes de l'état de l'art

Après avoir présenté les principaux AEMO, les indicateurs de performance ainsi que les benchmark de fonctions analytiques de l'état de l'art, nous consacrons cette section à la comparaison de AEMO présentés dans ce chapitre. Avec l'apparition des AEMO en fin des années 90 et en début des années 2000, plusieurs travaux dans la littérature se sont intéressés à la comparaison empirique des AEMO sur des benchmarks de fonctions analytiques comme dans [Zitzler 2000] [Tan 2002] où certains anciens algorithmes telle que SPEA et NSGA ont été comparés. Les comparaisons

réalisées par la suite sur les nouveaux AEMO, sont souvent celles réalisées dans les papiers de référence des AEMO décrit ci-dessus. Cependant, ces comparaisons concernent seulement quelques algorithmes de l'état de l'art, et ne comparent pas tous les principaux algorithmes à la fois. L'objectif de cette étape est de comparer tous les algorithmes entre eux en utilisant les benchmarks ZDT et IHR.

2.9.1 Réglages

Chaque algorithme est utilisé dans sa configuration de base proposée par ses auteurs. A noter qu'une étude préliminaire de réglage automatique des paramètres tels que les probabilités de croisement et de mutation a été menée et a conduit à des réglages similaires à ceux proposés par les auteurs des algorithmes. Une compagne de 30 essais est réalisée pour chaque algorithme. Pour tous les algorithmes, la taille de population est fixée à 100. A chaque génération, 100 enfants sont générés sauf pour ε-MOEA qui est un algorithme à schéma d'évolution stationnaire, et par conséquent un seul enfant est généré par génération. Le critère d'arrêt des algorithmes est le nombre d'évaluations qui est fixé à 50000 évaluations soit 500 générations pour les quatre algorithmes : NSGA-II, SPEA2, IBEA et MO-CMA-ES. Pour l'algorithme stationnaire ε-MOEA le nombre de générations est égale à 50000 (1 évaluation par génération). Les algorithmes seront testés sur les différentes fonctions des benchmarks ZDT et IHR (voir section 2.8). L'indicateur de l'hypervolume (voir section 2.6.1.1) sera utilisé comme indicateur de performance afin de comparer les résultats obtenus. Pour chaque algorithme l'indicateur de l'hypervolume est calculé par rapport à un ensemble de référence qui représente le front de Pareto optimal qui peut être calculé exactement pour les problèmes des benchmarks ZDT et IHR.

2.9.2 Résultats

Les résultats des expérimentations sont représentés dans la figure 2.9.3. Pour chaque problème test, les hypervolumes moyens de chaque algorithme sont comparés. Pour le benchmark ZDT, nous constatons que MOCMAES atteint globalement un niveau d'hypervolume plus bas que les autres algorithmes (sauf pour ZDT6) après 50000 évaluations. Nous pouvons relever aussi de l'analyse des résultats que l'algorithme IBEA échoue sur la fonction ZDT2. Ce résultat traduit une convergence prématurée de cet algorithme vers un seul point du front de Pareto.
En analysant les résultats après un budget relativement faible de nombre d'évaluations (5000 évaluations par exemple) nous constatons que le classement entre les algorithmes diffère d'une fonction à l'autre. Par conséquent, nous ne pouvons pas dégager un meilleur candidat à ce budget.
Pour la comparaison des algorithmes après 50000 évaluations, une interprétation des différences exprimées par l'indicateur de l'hypervolume est nécessaire pour l'analyse des résultats. En prenant comme exemple le problème ZDT1, la comparaison des solutions finales trouvées par les différents algorithmes sur un essai typique, montre que tous ces derniers arrivent à converger le long du front de Pareto (voir figure

2.3-gauche). Cependant, les différences exprimées par l'indicateur de l'hypervolume traduisent uniquement une différence dans la distribution des points sur le front de Pareto.

Pour le benchmark IHR, une convergence prématurée sur une partie du front de Pareto est constatée sur tous les algorithmes avec les deux problèmes IHR1 et IHR3. Pour IHR3 qui possède un front de Pareto discontinu, seule la grande portion du front est trouvée au détriment des autres petites parties du front de Pareto (voir figure 2.3-droite). Toutefois, MOCMAES semble atteindre des niveaux d'hypervolume plus bas (meilleure performance) en comparaison avec les autres algorithmes sur l'ensemble des problèmes du benchmark IHR. Par ailleurs, Pour un budget faible de nombre d'évaluations (5000 évaluations), ε-MOEA semble avoir une meilleure "convergence" que les autres algorithmes sur les fonctions IHR, mais se fait rattraper par ces derniers par la suite.

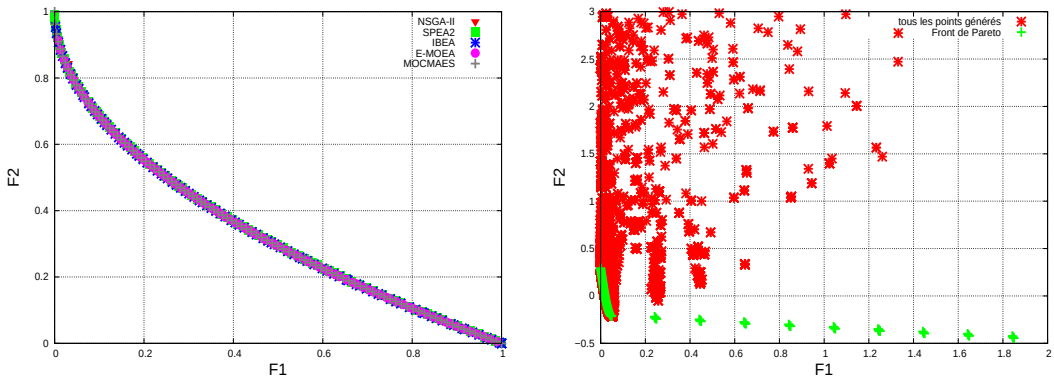


FIGURE 2.3 – Gauche : front de Pareto trouvé par les AEMO pour ZDT1 après 50000 évaluations. Droite : phénomène de convergence prématurée de la fonction IHR3 avec l'algorithme NSGA-II

2.9.3 Discussions

L'objectif principale de cette étape de l'étude était de faire un choix judicieux concernant l'AEMO à retenir pour réduire le coût global des optimisations du monde réel. Le coût très élevé des évaluations de certaines applications réelles rend parfois le processus d'optimisation irréalisable. Dans ce contexte, le choix de la méthode l'optimisation doit se faire dans l'optique de réduire considérablement le coût global de l'optimisation. Cependant après comparaison des différents AEMO de l'état de l'art, nous constatons qu'aucun de ces algorithmes ne permet de réaliser ce gain "considérable" en termes de rapidité de convergence vers le front de Pareto. D'une part, Les différences en termes de qualités des solutions constatées à la fin du budget alloué (50000 évaluations) expriment une différence dans la distribution des points une fois le front de Pareto est atteint, or dans les problématiques coûteuses du monde réel le défi principal de l'optimisation est d'abord de converger vers le front de Pareto. D'autres part, le budget des optimisations réelles couteuses ne dépassera

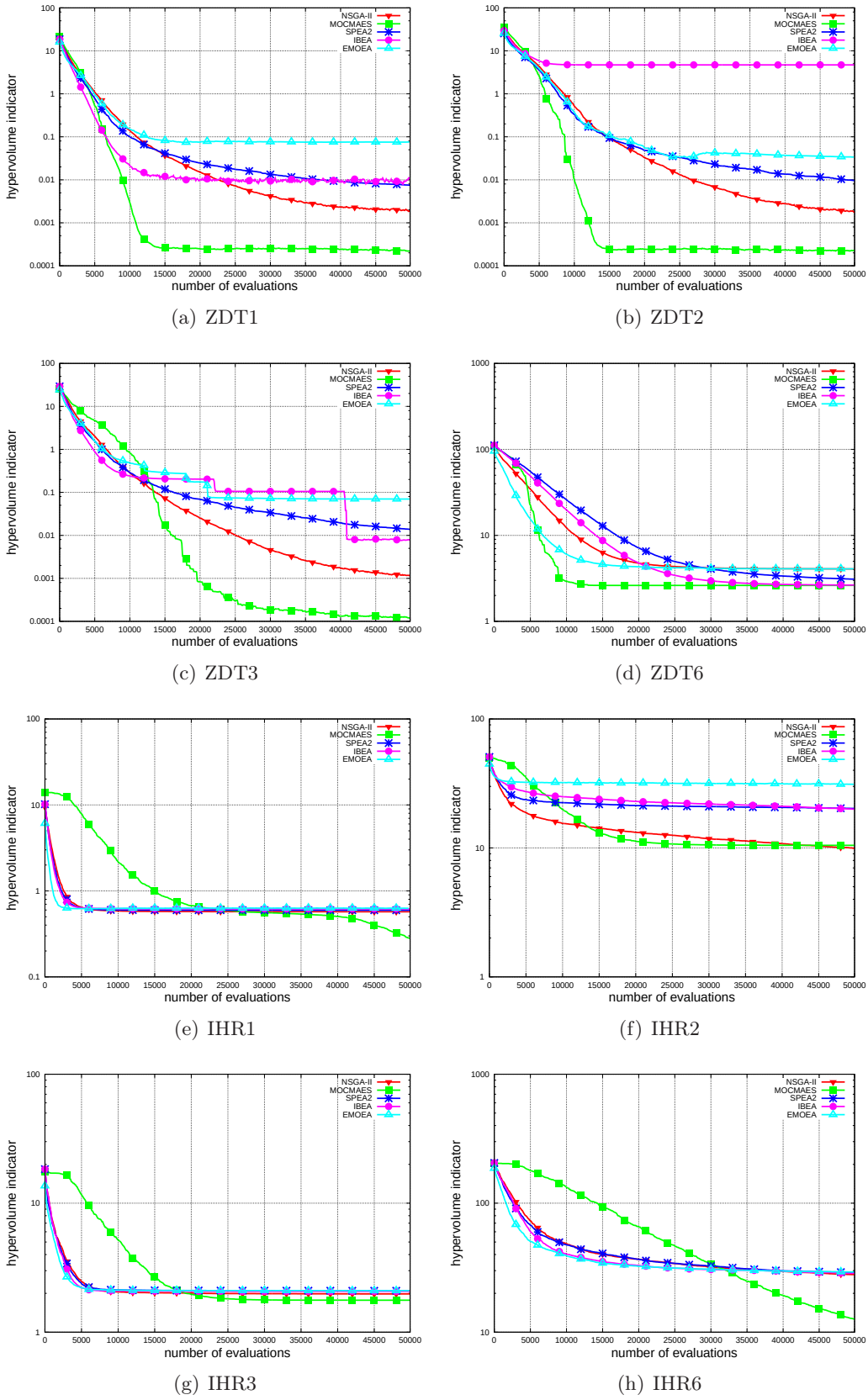
pas, quelques centaines d'évaluations ou au mieux quelques milliers d'évaluations. De la comparaison des AEMO à faibles budgets, nous pouvons constater qu'il est difficile de départager les algorithmes entre eux, car le classement diffère selon la fonction test choisie.
Dans ce contexte, d'autres axes de recherche, motivés par des phénomènes issus d'applications réelles ont été retenus afin de répondre à notre problématique.

2.10 Algorithmes évolutionnaires et modèles parallèles

Après avoir présenté les principaux AEMO de l'état de l'art, nous nous intéressons à présent à leur utilisation dans le contexte des optimisations du monde réel. Dans ce type d'applications, l'évaluation est souvent effectuée à travers de lourdes simulations très couteuses. Ces algorithmes nécessitent donc d'être parallélisés afin de pouvoir réaliser les optimisations en un temps raisonnable. Plusieurs modèles de parallélisation des algorithmes évolutionnaires ont été proposés dans la littérature. Ces modèles qui diffèrent dans le cœur de la parallélisation, peuvent être classés en trois catégories : les modèles "maîtres-esclaves", les modèles en îlots et les modèles totalement distribués (voir [Cantu-Paz 2000], [Alba 1999])

2.10.1 Modèle Maître-esclave

Dans le modèle "maître-esclave", qui est le modèle le plus simple à mettre en œuvre, le processus maître distribue l'évaluation de la fonction objective sur les différents nœuds esclaves, et effectue toutes les étapes de l'algorithme évolutionnaire (sélection, croisement, mutation). La communication entre les individus n'a lieu qu'après que les différends esclaves retournent l'évaluation qui leur a été assignée. Il est de ce fait identique algorithmiquement parlant, à un algorithme évolutionnaire séquentiel. Le modèle maître esclave a été largement utilisé dans la littérature. Les premiers travaux remontent aux années 70 avec Bethke [Bethke 1976] qui a été le premier à décrire une implémentation parallèle d'un AE. Par la suite, Grefenstette [Grefenstette 1981] proposa plusieurs prototypes des AE parallèles représentant plusieurs variantes du modèles maître esclave.
Sachant que le coût d'optimisation global avec ce modèle comporte deux temps, celui de l'évaluation des différents enfants et le temps de communication entre les différents nœuds esclaves et le processus maître, des travaux plus récents dans la littérature se sont intéressés à la problématique du coût de communication élevé qui peut nuire à l'efficacité de l'algorithme([Forgarty 1991] [Abramson 1992] [Hauser 1994]). Dans ce modèle, l'accélération en temps de calcul est prouvée être linéaire relativement au nombre de processeurs, jusqu'à une certaine limite (quand les temps des communications deviennent plus important que le temps d'une évaluation [Cantu-Paz 2000]). A noter que dans le cas des applications réelles où l'évaluation de la fonction objectif est très couteuse, les temps des communications sont souvent négligés.



(a) ZDT1 (b) ZDT2

(c) ZDT3 (d) ZDT6

(e) IHR1 (f) IHR2

(g) IHR3 (h) IHR6

FIGURE 2.4 – Evolution de l'indicateur de l'hypervolume moyen pour les différents AEMO avec les benchmark ZDT et IHR

2.10.2 Modèle en îlots

Ce modèle de parallélisation concerne les algorithmes évolutionnaires avec plusieurs populations. Il tient son appellation du parallèle fait avec les populations naturelles qui sont relativement isolées les unes des autres, formant des "iles". Toutefois, les individus peuvent "migrer" occasionnellement à une autre population. Les premiers travaux dans ce domaine remontent aux années 60 avec Bossert [Bossert 1967] qui fut probablement le premier à avoir proposé un algorithme évolutionnaire avec des populations multiples afin d'améliorer la qualité des solutions dans un problème d'optimisation. D'autres travaux qui ont été proposés par la suite ([?],[Grosso 1985],[Braun 1990]) différent selon les choix faits sur les nombreux paramètres à piloter avec ce modèle comme par exemple : le nombre et la taille des îles, la fréquence de migration, le nombre et la destination des migrants, et enfin la méthode utilisée pour sélectionner quel individu va migrer. D'ailleurs c'est la raison pour laquelle ce type de parallélisation reste difficile à contrôler

2.10.3 Modèle totalement distribué

Dans le modèle totalement distribué, la population est structurée en une grille rectangulaire bi-dimensionnelle où chaque noeud comprend un ou très peu d'individus, et l'évaluation de la fonction objectif est réalisée simultanément pour tous les individus. les opérations de sélection et de croisement sont réalisées avec les noeuds voisins. Ce type de parallélisation des algorithmes évolutionnaires peut être croisé dans la littérature sous d'autres appellations comme par exemple : AE de diffusion [Pettey 1997], AE cellulaires [Whitley 1993] [Tomassini 1999] ou encore AE massivement parallèles [Pettey 1992]. Cependant, ce dernier modèle reste peu étudié dans la littérature en comparant avec les deux premiers modèles.

2.11 Hétérogénéité des coûts d'évaluation et asynchronicité des AEMO

La motivation principale de ce travail trouve son origine dans la problématique réelle qui représente le contexte industriel de cette thèse et qui concerne l'optimisation de la combustion Diesel. Les modèles de simulations utilisés reposent sur le couplage de plusieurs outils qui permettent de modéliser les différents phénomènes physiques au sein de la chambre de combustion tels que : le mélange, la combustion, l'injection ...etc (cf chapitre 6, 7). Cependant, ce type de problématiques réelles est caractérisé par un temps de restitution hétérogène des individus appartenant à la même population. La figure 2.5 illustre les durées de calcul des individus appartenant à la même génération initiale (cette figure concerne l'application qui sera détaillée dans le chapitre 6). Nous pouvons constater que les coûts des évaluations des individus sont très hétérogènes. Globalement, l'hétérogénéité des durées d'évaluation des problématiques réelles peut avoir plusieurs sources : elle peut être liée au problème d'optimisation lui même (certaines régions de l'espace

de recherche sont plus coûteuses que d'autres en raison de la non-linéarité), comme elle peut être liée à la puissance des processeurs utilisés pour l'évaluation parallèle des individus (les évaluations en parallèle des individus peuvent se faire sur des machines de puissances différentes).
Comme nous venons de voir dans la section précédente, plusieurs modèles ont été proposés dans la littérature afin de paralléliser les AEMO. Le modèle maître-esclave reste toutefois le modèle le plus utilisé pour sa simplicité et facilité de mise en œuvre, c'est le modèle qui a été retenu dans le cadre de cette thèse.
L'implémentation simple et directe du paradigme maître-esclave utilise la version *standard générationnelle* des AEMO. Dans cette version, les étapes de sélection et de variation (mutation, croisement) sont appliquées après que tous les individus d'une génération donnée ont été évalués sur les processeurs esclaves. Dans le cas où toutes les évaluations nécessitent exactement le même temps CPU sur tous les processeurs disponibles, la population est uniformément distribuée sur les processeurs esclaves, il s'agit du cas *homogène*. A noter que le temps d'échange des données dans le réseau est très petit comparé au temps requis pour réaliser l'évaluation, et par conséquent il est supposé être négligeable.
Cependant comme nous venons de voir, dans les problèmes d'optimisation du monde réels particulièrement, les coûts CPU des évaluations sont rarement homogènes. Dans ce contexte, l'utilisation d'un algorithme avec un modèle générationnel ralentit considérablement l'avancement de l'optimisation : plusieurs processeurs disponibles dans la grille resterons inoccupés en attendant la fin des évaluations les plus longues, et ceci à chaque génération.
Une solution bien connue à ce problème est d'utiliser un algorithme *stationnaire* : dans le cas séquentiel, l'algorithme stationnaire génère un seul enfant, l'évalue, et le replace dans la population en éliminant le plus mauvais des parents. Cependant, l'utilisant de ce modèle un contexte hétérogène rend l'algorithme *asynchrone* car les individus ne regagnent plus la population dans le même ordre que celui dans lequel ils ont été crées. Dans cette thèse, nous tenterons de répondre à cette problématique en utilisant les moteurs d'évolution stationnaires qui génèrent un seul enfant par génération. Cependant, dans le but d'assurer une utilisation efficace de la grille de calcul, en ne laissant aucun des processeurs inoccupé, le processus d'évolution doit devenir asynchrone car les individus reviennent dans un ordre différent que celui de leur création (du fait de leur hétérogénéité de temps de calcul).

2.12 Démarche

La démarche adoptée dans le but de valider l'approche "stationnaire asynchrone" sur l'optimisation évolutionnaire parallèle des problèmes du monde réel est résumée par les trois points suivants :

1. Tout d'abord nous étudions le comportement stationnaire asynchrone sur des benchmarks de problèmes analytiques (cf. benchmark ZDT et IHR). Pour

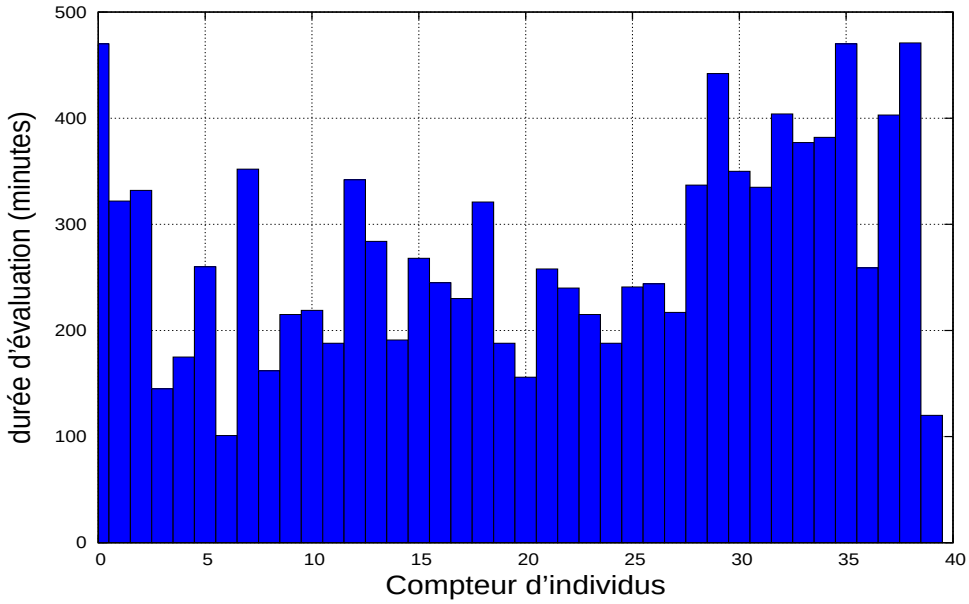


FIGURE 2.5 – Durées des évaluations des individus de la première génération aléatoire

ce type de problèmes, le temps d'évaluation est de l'ordre de quelques milisecondes et peut de ce fait être supposé négligeable. Cependant, comme la problématique de l'hétérogénéité des coût de calcul est étroitement liée aux applications réelles dans lesquelles les évaluations ont un coût significatif, l'étude du comportement asynchrone des AEMO dans un contexte de problème analytique nécessite une certaine adaptation dans l'implémentation de l'optimisation. Cette adaptation concerne la création d'un modèle "artificiel" qui permet de simuler le comportement de l'AEMO sur une grille de calcul parallèle avec un temps de calcul hétérogène.
D'un point de vue algorithmique, les deux volets suivants seront retenus dans cette première étape de la démarche.

(a) Le premier volet concerne l'étude du comportement asynchrone stationnaire sur les AEMO dans l'optique de réaliser des gains considérables en utilisant directement les AEMO, ce volet fera l'objet du chapitre 3

(b) Dans le deuxième volet nous nous focalisons sur l'utilisation des méta-modèles dans l'optique de réduire le coût de l'optimisation en minimisant le nombre des évaluations de la fonction objectif. Nous combinerons par la suite les aspects méta-modèles avec les aspects stationnaires asynchrones pour s'adapter aux problématiques à coût d'évaluations hétérogène. Cette partie fera l'objet du chapitre 4.

2. Après avoir validé l'utilisation des schémas asynchrones stationnaires sur des problèmes analytiques, la deuxième étape consiste à l'appliquer sur une problé-

matique réelle de la combustion Diesel en utilisant une modélisation 0D. Cette dernière est caractérisée par un temps de restitution plutôt "raisonnable" en comparaison avec le temps de calcul des modèles 3D (quelques heures pour le 0D contre quelques jours pour le 3D). Avec ce type de modélisation nous pouvons par exemple nous permettre d'effectuer une comparaison entre l'approche développée dans cette thèse et celle standard de l'état de l'art. Un deuxième objectif de l'utilisation de la modélisation 0D est d'ordre applicatif : cette étape consiste aussi à démontrer l'utilité d'utiliser la modélisation 0D en analysant les résultats des optimisations d'un point de vue physique. Cette première application sera détaillée dans le chapitre 6.

3. la troisième et dernière étape consiste à réaliser une optimisation en utilisant la modélisation 3D complète qui représente l'objectif principale du volet industriel de la thèse. Au vu des résultats obtenus lors des deux derniers points, cette ultime étape qui fera l'objet du chapitre 7 consiste à valider l'utilisation de l'approche stationnaire asynchrone proposée qui nous permettra de réduire considérablement le coût global de l'optimisation sur un cas réel complet où la simple évaluation d'un individu coûte quelques jours.

2.13 Conclusion

Nous avons présenté dans ce chapitre l'état de l'art de l'optimisation évoulutionnaire multi-objectif. Après avoir rappelé quelques notions préliminaires de l'optimisation évolutionnaire, nous avons passé en revue les principaux algorithmes AEMO de l'état de l'art ainsi que les différents indicateurs de performance en décrivant le principe de fonctionnement de chacun. Par la suite, les différents AEMO ont été comparés en utilisant les benchmarks de fonctions analytiques de l'état de l'art. Les résultats de comparaison montrent que le gain considérable espéré ne peut s'obtenir en faisant un choix d'un algorithme ou d'un autre. Afin de répondre à la problématique de réduction du coût global de l'optimisation et motivés par le phénomène d'hétérogénéité des coûts d'évaluations relevé sur la problématique de la combustion Diesel qui représente le contexte industriel de cette thèse, nous nous somme proposés d'étudier les schémas stationnaires asynchrones. Après avoir présenté brièvement les différents modèles de parallélisation des AEMO, et justifié le choix du modèle "maître-esclave", nous avons détaillé la démarche que nous proposons pour valider l'utilisation des algorithmes stationnaires asynchrones pour l'optimisation multi-objectif parallèle.
le prochain chapitre sera consacré à la première étape de la démarche, qui consiste à étudier les AEMO stationnaires asynchrones en utilisant les fonctions analytiques des benchmarks de l'état de l'art en simulant "artificiellement" le contexte parallèle des grilles de calcul.

Chapitre 3

AEMO asynchrones avec coûts d'évaluation hétérogènes

Sommaire

3.1 Introduction **35**
3.2 Parallélisation des AEMO avec le modèle Maître-esclave . **37**
3.2.1 NSGA-II Stationnaire 37
3.2.2 MO-CMA-ES Stationnaire 37
3.2.3 AEMO Asynchrones Stationnaires 37
3.3 Implémentation des modèles de test artificiels **38**
3.3.1 L'hétérogénéité aléatoire "Rand-VC" 40
3.3.2 L'hétérogénéité proportionnelle aux valeurs des objectifs "Eps-VC" 40
3.3.3 L'hétérogénéité liée à une région "Region-VC" 41
3.4 Conditions expérimentales générales **41**
3.4.1 Réglages algorithmiques 41
3.4.2 Représentation graphique des résultats 42
3.5 Accélération de convergence asynchrone **42**
3.6 Contexte hétérogène **46**
3.6.1 Le modèle Eps-VC 46
3.6.2 Le modèle Région-VC 48
3.7 Conclusion **53**

Les travaux présentés dans ce chapitre ont fait l'objet des publications [Yagoubi 2011] et [Yagoubi 2012].

3.1 Introduction

Pour les AEMO, le calcul parallèle représente une piste très intéressante afin de faire face au problème du coût d'évaluation très important des applications réelles. Plusieurs modèles de parallélisation des algorithmes évolutionnaires (voir section 2.10) ont été proposés dans la littérature. Cependant, le modèle Maître-esclave est de loin le modèle le plus simple à mettre en œuvre. Il a été utilisé dans divers domaines d'application. L'utilisation de ce modèle de parallélisation peut être étendue au cas multi-objectifs car, identiquement au cas d'un algorithme séquentiel, toutes les étapes du cycle évolutionnaire sont effectuées au niveau du processus maître.

D'un point de vue parallèle, il est important de noter que les modèles en îlots et les modèles totalement distribués sont basés sur une topologie sous-jacente du graphe des nœuds. Dans ce type de topologie, la disposition de la grille de calcul permet d'optimiser les coûts des communications entre les différents nœuds imposant une restriction qui limite les échanges aux nœuds voisins. Le modèle maître-esclave par contre est basé sur une topologie en étoile, dans laquelle toutes les communications transitent par le processus maître. Ceci a été considéré comme une limitation majeure dans diverse applications parallèles. Cependant, dans un contexte d'application du monde réel où le coût d'une évaluation est très élevé par rapport au coût des communications (quelques jours vs quelques milisecondes), cette limitation n'est plus valable. D'un autre coté, comme les algorithmes multi-objectifs nécessitent une synchronisation globale au moment de la sélection, les deux modèles de parallélisation (modèle en îlots et le modèle totalement distribué) doivent être adaptés pour être appliqués dans un contexte multi-objectifs. A titre d'exemple, une approche parallèle de NSGA-II a été proposée dans [Deb. 2003a]. Cette approche est basée sur le principe de "dominance guidée" : chaque processeur de la grille est impliqué dans la recherche d'une portion particulière du front de Pareto. [Branke. 2004] propose une approche géométrique nommée "séparation des cônes" qui subdivise l'espace de recherche en plusieurs sous-régions, et afin d'avoir chaque processeur focalisé dans la recherche dans un domaine bien délimité, les limites de chaque région sont traitées en tant que contraintes. Dans [Asouti 2009], un mécanisme complexe qui gère de multiples populations via des processeurs hétérogènes a été mis en place. Ce mécanisme est basé sur le calcul des agents mobiles.
Dans ce chapitre nous nous proposons d'étudier la parallélisation des AEMO artificiellement (en utilisant des fonctions analytiques), en tenant compte des spécificités pouvant être rencontrées dans le monde des applications réelles telle que l'hétérogénéité des temps de calcul et qui représentent la motivation de ce travail. En prenant en compte ces considérations, le reste de ce chapitre sera organisé comme suit : Après avoir décrit en détail les versions stationnaires asynchrones issues des AEMO parallélisés avec le modèle mâitre/esclave (section 3.2) ainsi que les modèles artificiels de simulation de l'hétérogénéité de calcul (section 3.3), nous nous intéressons dans la section 3.5 à l'étude de l'accélération liée à l'accroissement de la taille de la grille de calcul dans le contexte des AEMO asynchrone. Dans un deuxième temps, nous nous focaliserons sur l'étude de l'hétérogénéité des durées d'évaluation (section 3.6) que nous étudierons selon deux modèles : dans le premier modèle nous supposons que les durées d'évaluations sont proportionnelles à la distance séparant les individus du front de Pareto (section 3.6.1), tandis que dans le deuxième modèle nous étudierons le cas où certaines régions dans l'espace objectif sont plus coûteuses que d'autre. Dans ce deuxième modèle, nous étudierons d'abord les effets néfastes d'un tel phénomène, et nous proposons par la suite la "sélection basée sur la durée" afin d'y faire face (section 3.6.2).

3.2 Parallélisation des AEMO avec le modèle Maître-esclave

Nous présentons dans cette section les versions parallèles des AEMO. Dans le cadre de ce travail, deux algorithmes parmi ceux présentés dans le chapitre 2 seront retenus : NSGA-II qui reste aujourd'hui l'algorithme le plus populaire de l'état de l'art pour sa robustesse à travers des domaines d'applications divers, et le plus récent MO-CMA-ES considéré comme l'un des algorithmes les plus performants dans l'optimisation continue. Avant de présenter le modèle de parallélisation basé sur paradigme "maître-esclave", nous décrivons d'abord les versions stationnaires de NSGA-II et MO-CMA-ES.

3.2.1 NSGA-II Stationnaire

Une version stationnaire de l'algorithme NSGA-II peut être obtenue en considérant simplement une population d'enfant de taille 1. Dans ce schéma, un seul enfant est généré à chaque génération, et remplace un parent en utilisant un tournoi inversé dans lequel l'individu le moins performant est éliminé.

3.2.2 MO-CMA-ES Stationnaire

Comme pour NSGA-II, une version stationnaire de MO-CMA-ES peut être obtenue en utilisant un schéma $\mu + 1$. Deux variantes possibles de MOCMA-ES stationnaire ont été proposées [Igel 2007b] : la première variante utilise la totalité de la population pour sélectionner le parent à faire varier, cette variante est notée $(\mu+1)$. La deuxième variante $(\mu_{\prec} + 1)$ quant à elle, considère seulement les individus non-dominés lors de la sélection du parent. De nouvelles améliorations concernant le choix de la méthode de sélection du parent ont été publiées récemment [Loshchilov 2011]. Cependant, seule la variante $\mu_{\prec} + 1$ sera considérée dans ce travail.

3.2.3 AEMO Asynchrones Stationnaires

Les algorithmes décrits dans l'état de l'art ainsi que les versions stationnaires présentées ci-dessus sont des algorithmes séquentiels : un processeur donné effectue le calcul de toutes les évaluations les unes après les autres. Dans un contexte stationnaire, ceci revient à dire que les enfants regagnent les populations dans le même ordre que celui dans lequel ils ont été générés. Dans le cas où ces algorithmes stationnaires sont parallélisés avec un modèle maître-esclave, le processus maître prend en compte toutes les opérations évolutionnaires (initialisation, sélection et variation), et envoie toutes les évaluations aux processus esclaves.
Dans le cas *homogène*, c'est à dire quand toutes les évaluations ont le même temps de retour, les algorithmes auront un comportement similaire à ce qui pourrait être obtenu avec leurs versions séquentielles. Cependant, la situation n'est plus la même dans le cas *hétérogène*, où le temps de calcul de certaines évaluations est beaucoup plus important que d'autres. L'algorithme générationnel doit attendre que toutes

les évaluations soient effectuées avant de passer à la nouvelle génération, et par conséquent certains processeurs vont restés inoccupés pour de longues durées, attendant la fin de l'évaluation la plus longue. Le résultat sera le même que celui obtenu avec un algorithme séquentiel mais avec un temps de calcul beaucoup plus important.
Afin d'utiliser plus efficacement les processeurs disponibles dans la grille de calcul, les algorithmes stationnaires peuvent être modifiés pour devenir *asynchrones* : les enfants sont envoyés sur la grille pour l'évaluation, et le retour à la population est réalisé selon le principe "premier arrivé-premier servi". Avec cette configuration, aucun processeur ne reste inoccupé pendant l'optimisation, ce qui représente la motivation principale de l'utilisation des AEMO parallèles asynchrones. Les algorithmes résultant de l'application de ce modèle sont baptisés : AS-NSGA-II et AS-MO-CMA-ES. (AS en anglais, pour *Asynchronous Steady-state*) Il est à noter que les schémas de sélection stationnaire ont été appliqués dans plusieurs AEMO de la littérature : l'algorithme ε-MOEA [Deb 2003b] (voir section 2.7.4 pour plus de détail) utilise un modèle de sélection stationnaire. SMS-EMOA proposé dans [Beume 2007] est un algorithme stationnaire qui utilise l'indicateur de l'hypervolume comme un second critère de sélection. Cependant, ces travaux ne considèrent que le cas séquentiel. Les versions parallèles de l'algorithme NSGA-II ont été étudiées en détail dans [Durillo 2008] sur une application réelle, dans la quelle la version asynchrone permettait d'avoir de meilleurs résultats. Cependant, l'objectif principal du travail était d'évaluer le temps total écoulé pour les grandes plateformes *hardware* (réseaux *peer-to-peer*) et l'hétérogénéité dans l'évaluation provenait essentiellement des processeurs, et ne dépendait pas particulièrement de l'individu évalué.
Il est à noter que la différence dans le comportement des algorithmes parallèles et séquentiels dépend de la distribution des coûts des évaluations sur l'espace de recherche : une distribution uniforme (c'est à dire le coût d'évaluation d'un individu ne dépend pas de ce dernier) donne lieu à un comportement similaire des algorithmes parallèles et séquentiels : les enfants regagnent la population dans un ordre aléatoire, et les différences de coût moyennées sur l'optimisation entière, auront tendance à se neutraliser. Toutefois, dans le cas où le coût d'évaluation dépend des caractéristiques des individus, certaines régions de l'espace de recherche risquent d'être moins explorées que d'autres, et certaines parties du front de Pareto vont être découvertes tardivement ...voire jamais.

3.3 Implémentation des modèles de test artificiels

L'implémentation du modèle asynchrone est faite comme suit : L'algorithme maintient une queue additionnelle qui contient les enfants évalués classés dans un ordre similaire à celui de leur ordre d'insertion dans la population à la fin de la boucle stationnaire. Tout d'abord, pour des raisons de simplification, la population

initiale est évaluée et l'hétérogénéité des coûts d'évaluation n'est pas prise en compte. A noter que les résultats préliminaires des expérimentations montrent qu'il n'y a pas de différence significative avec le cas où la population initiale est remplie dans l'ordre de retour des individus après l'évaluation.
La boucle principale de l'algorithme hétérogène stationnaire commence par la sélection d'un parent (en utilisant la sélection parentale), et l'envoie de ce dernier pour effectuer l'évaluation. Selon les valeurs de la fonction objective, le "temps de calcul" de cette évaluation est calculé en utilisant un modèle artificiel de coût d'évaluation. Le temps de retour attendu de l'enfant est le temps actuel + le temps qu'a pris l'évaluation. L'enfant est inséré par la suite dans la queue d'attente et placé selon le temps de retour calculé auparavant (la queue d'attente comporte tous les enfants évalués, classé par rapport à leur temps de retour). L'enfant figurant en tête de la queue d'attente est inséré dans la population, pour procéder à l'étape de remplacement.
Cette variante a été appliquée aux deux algorithmes retenus pour cette étude : NSGA-II et MO-CMA-ES. Les pseudo-codes des versions asynchrones stationnaires des deux algorithmes sont décrites dans les algorithmes 8 et 9 respectivement. Les lignes grisées, représentent les étapes artificielles permettant la simulation du comportement asynchrone sur une grille de calcul.

Algorithm 8 NSGA-II stationnaire asynchrone

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0|=N$, Eval(P_0)
2: $B_0 \leftarrow$ Variation (P_0) Simule la queue d'attente
3: Calcul-Durée (B_0)
4: **repeat**
5: $(p_1^t, p_2^t) \leftarrow$ Select(P_t). // sélection de deux parents :Tournoi de taille 2
6: $(q_1^t, q_2^t) \leftarrow$ Variation (p_1^t, p_2^t) // croisement, mutation
7: $q^t \leftarrow U(q_1^t, q_2^t)$ // choix aléatoire d'un des deux enfants
8: Lancer-Eval$\{q^t\}$
9: Calcul-Durée $\{q^t\}$
10: $i \leftarrow$ Index de l'individu avec la durée la plus courte B_t
11: $B_{t+1} \leftarrow (B_t \cup \{q^t\}) \backslash \{b_i^t\})$
12: $R_t \leftarrow P_t \cup \{b_i^t\}$
13: Sort(R_t) // Principe de dominance + critère de surpeuplement
14: $P_{t+1} \leftarrow R_t[0 : N]$ // choisir les N premiers éléments
15: $t \leftarrow t + 1$
16: **until** Critère d'arrêt rencontré

Toutes les expérimentations dans le reste de ce chapitre ont été réalisées sur les benchmark de fonctions test analytiques ZDT et IHR (voir section 2.8 pour les détails). Les évaluations à travers ces fonctions sont très rapides, et sont homogènes en termes de temps de calcul. Cependant, afin de simuler le comportement asynchrone des évaluations sur la grille de calcul, ces fonctions tests nécessitent une certaine

Algorithm 9 MO-CMA-ES stationnaire asynchrone

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0|$=N
2: $B_0 \leftarrow$ Variation (P_0) Simule la queue d'attente
3: Calcul-Durée (B_0)
4: **repeat**
5: $p^t \leftarrow U(P_t)$ // choisir aléatoirement un parent
6: $q^t \leftarrow$ Variation(p^t) // mutation
7: Lancer-Eval(q^t)
8: Calcul-Durée (q^t)
9: $i \leftarrow$ Index de l'individu avec la durée la plus courte B_t
10: $B_{t+1} \leftarrow (B_t \cup \{q^t\}) \backslash \{b_i^t\})$
11: $Q_t \leftarrow P_t \cup \{b_i^t\}$
12: Actualiser le pas de mutation
13: Actualiser la matrice de covariance
14: Sort(Q_t) // principe de dominance + critère de surpeuplement ou hypervolume
15: **for** $i = 1, ..., N$ **do** $P_{t+1} \leftarrow Q_t[0 : N]$
16: $t \leftarrow t + 1$
17: **until** Critère d'arrêt rencontré

adaptation. Trois modèles artificiels de fonctions test (que nous présenterons dans cette section) ont été définis afin de simuler différentes variantes du comportement asynchrone liées aux différents cas de figures considérés.

3.3.1 L'hétérogénéité aléatoire "Rand-VC"

Le modèle de base qui a été défini suppose qu'il n'existe pas de corrélation entre le coût d'évaluation et l'individu concerné. Ce modèle couvre le cas où la seule source d'hétérogénéité provient des processeurs disponibles sur la grille comme par l'exemple l'utilisation de processeurs de type différent. Pratiquement, l'implémentation de ce premier modèle revient à insérer le nouveau enfant crée de façon aléatoire dans la queue d'attente. Dans la suite de ce chapitre, le problème test résultant de l'application de ce modèle à une fonction test ALPHA sera noté Rand-VC-ALPHA (pour *Random Variable cost*).

3.3.2 L'hétérogénéité proportionnelle aux valeurs des objectifs "Eps-VC"

Dans ce modèle, le coût d'évaluation de l'individu est inversement proportionnel à la distance qui le sépare au front de Pareto. Ce cas de figure représente le cas où la précision des résultats doit être de plus en plus fine à l'approche du front de Pareto (par exemple, plus d'itérations au sein de l'évaluation, ou un maillage plus fin ...etc). Pratiquement, il existe plusieurs méthodes pour estimer la distance qui sépare

l'individu du front de Pareto. Pour notre implémentation, nous utilisons l'indicateur epsilon dans sa forme unaire additive $I^1_{\varepsilon+}$ [Zitzler 2003] (voir section 2.6.1.2 pour plus de détails). Le front de Pareto quant à lui, peut être calculé analytiquement pour les fonctions tests. A partir de là, le coût d'évaluation d'un individu est défini comme étant l'inverse de la racine carrée de la valeur de l'ε-Indicateur calculé. La fonction test ALPHA qui suit ce modèle sera notée Eps-VC-ALPHA.

3.3.3 L'hétérogénéité liée à une région "Region-VC"

Dans ce dernier modèle, le coût des évaluations est aussi lié aux valeurs des objectifs. Chaque individu se voit attribuer une durée, qui simule un comportement hétérogène dans lequel une certaine région de l'espace de recherche est plus coûteuse à évaluer que les autres régions. (ce comportement a été relevé dans l'application réelle qui sera présentée dans le chapitre 6). L'implémentation de ce phénomène revient à assigner un coût d'évaluation différent entre les individus présents dans une région pré-définie dans l'espace des objectifs (définie par des bornes sur les différents objectifs) et les individus du reste de l'espace. La durée de calcul peut prendre 2 valeurs : une petite valeur pour les individus de l'ensemble de l'espace et une large valeur pour les individus de la région coûteuse. Pour ce modèle, une fonction test ALPHA sera notée Region-VC-ALPHA.

3.4 Conditions expérimentales générales

Nous présentons dans cette section quelques réglages globaux qui seront valables pour toutes les expérimentations présentées dans la suite de ce chapitre ainsi que les choix concernant la représentation des résultats.

3.4.1 Réglages algorithmiques

Comme mentionné précédemment dans ce chapitre, les benchmark ZDT [Zitzler 2000] et IHR [Igel 2007a] ont été retenus pour la réalisation des expérimentations. Tous les résultats qui suivent sont moyennés sur 30 réalisations indépendantes. Le budget en termes de nombre d'évaluations est fixé à 50000 évaluations. La taille de population a été fixé à 100, tout comme la taille de la queue d'attente des enfants évalués, ce qui correspond à une simulation d'une optimisation parallèle sur 100 processeurs sur une grille de calcul (sauf pour les expérimentations portant sur l'étude de l'accélération dans lesquelles la taille de la queue d'attente varie entre 50 et 1000)
L'indicateur de performance retenu pour évaluer la qualité des solutions est l'indicateur de l'hypervolume [Zitzler 1999b] (voir section 2.6.1.1 pour plus de détails).

3.4.2 Représentation graphique des résultats

Afin de comparer les différents algorithmes entre eux, nous proposons d'utiliser deux modes de représentation graphique :
Le premier mode de représentation consiste à comparer l'évolution de l'indicateur de l'hypervolume moyen en fonction du nombre d'évaluations, il s'agit de la comparaison classique de la qualité des solutions obtenues au cours de l'évolution.
Le deuxième mode de représentation tient compte du critère le plus important à prendre en compte lors des optimisations réelles avec des évaluations couteuses : le temps global écoulé. Ce critère peut être calculé en utilisant les modèles de coût artificiels définis précédemment.

3.5 Accélération de convergence asynchrone

Avant de se focaliser sur la problématique des coûts d'évaluation hétérogènes, nous nous intéressons d'abord à l'étude de l'accélération liée à la variation de la taille de la grille qui peut être obtenue par les variantes asynchrones des deux algorithmes présentés dans la section 3.2.3 dans un cadre globalement homogène, c'est à dire que l'hétérogénéité (due principalement aux processeurs) est uniformément distribuée. Nous utilisons dans ces expériences les benchmarks artificiels : Rand-VC-ZDT et Rand-VC-IHR décrit dans la section 3.3.1. Différentes valeurs de la taille de la queue d'attente seront balayées et comparées au cas référence d'un algorithme asynchrone steady-state sur une grille de taille égale à 1 processeur (c'est à dire le même algorithme avec une queue de taille 1).
Les figures 3.1 et 3.2 illustrent l'évolution de l'indicateur de l'hypervolume médian pour différentes valeurs de taille de queue pour les deux algorithmes AS-NSGA-II et AS-MO-CMA-ES respectivement, avec le problème artificiel Rand-VC-ZDT3. Les graphes de gauche expriment cette évolution en termes de nombre total d'évaluations (indépendamment de la parallélisation) tandis que les graphes de droite illustrent cette évolution en termes de temps écoulé. Pour le second type de graphe (graphes de droite), l'axe des x a été arbitrairement mis à l'échelle en considérant que les 50000 évaluations réalisées en utilisant l'algorithme avec 1 processeur (cas référence représenté par des triangles noirs dirigés vers le haut) correspondent à 1000 unités de temps (les courbes du cas de référence ne sont que partiellement visibles afin de rendre les autres courbes visibles).

Les tables 3.1 et 3.2 résument les résultats obtenus sur les benchmarks artificiels Rand-VC-ZDT et Rand-VC-IHR respectivement en utilisant des tailles de queue d'attente différentes. Chacune des colonnes représente un niveau d'hypervolume donné. Pour chaque colonne, la valeur inscrite dans la ligne $nProc$ est le ratio entre le temps écoulé médian des algorithmes asynchrones utilisant $nProc$ et 1 processeur respectivement.
De ces résultats, il ressort que les accélérations en termes de temps écoulé sont pour ces fonctions tests, clairement non linéaires. En comparant les queues d'attente de taille 50 et 100, un gain d'un facteur avoisinant 2 est globalement observé sur l'accé-

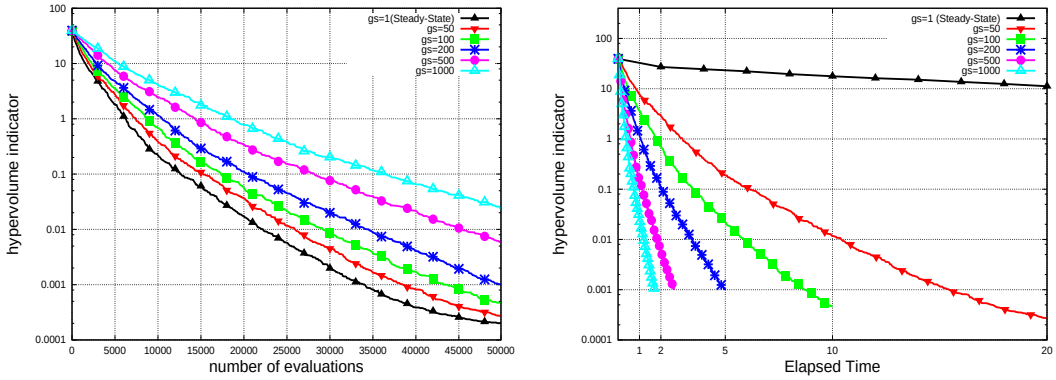


FIGURE 3.1 – Évolution de l'indicateur de l'hypervolume médian pour AS-NSGA-II sur Rand-VC-ZDT3 pour différentes tailles de queue. Gauche : en termes de nombre d'évaluations. Droite : en termes de temps écoulé.

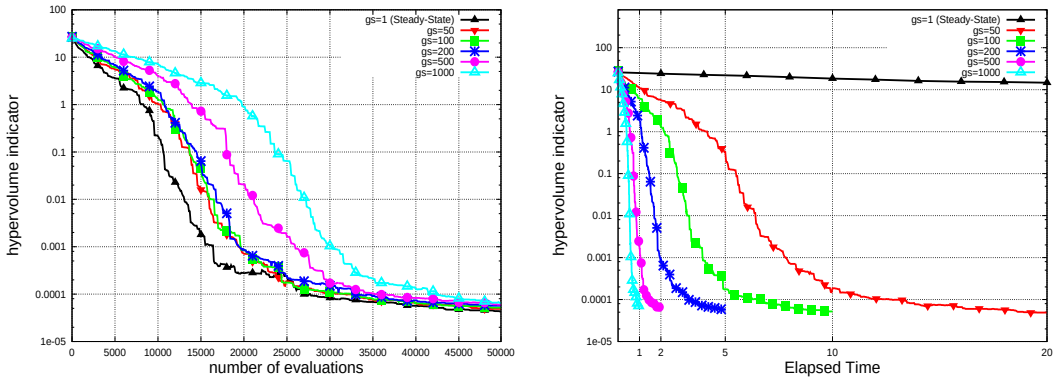


FIGURE 3.2 – Évolution de l'indicateur de l'hypervolume médian pour AS-MO-CMA-ES sur Rand-VC-ZDT3 pour différentes tailles de queue. Gauche : en termes de nombre d'évaluations. Droite : en termes de temps écoulé.

lération. Cependant, ce facteur ne dépasse pas 1.5 en comparant des grandes valeurs de taille de queue (500 et 1000).
Une autre observation peut être faite en analysant les figures : le nombre total d'évaluation (indépendamment de la parallélisation) nécessaire pour atteindre un certain niveau d'hypervolume décroit inversement proportionnellement avec le nombre de processeurs utilisés dans la grille. Ce phénomène peut être expliqué comme suit : en utilisant un schéma d'évolution stationnaire, un enfant est crée et inséré dans la queue d'attente à chaque génération. En supposant que les individus récemment créés sont meilleurs que leurs parents avec une certaine probabilité, la qualité des individus demeurant dans la queue d'attente est en moyenne proportionnelle à leur "âge génétique" : les individus les plus jeunes sont meilleurs que les individus âgés. Dans ce contexte, plus la taille de la queue d'attente est grande et plus la qualité moyenne diminue.

Par ailleurs, une autre analyse peut être faite cette fois-ci en comparant les com-

TABLE 3.1 – accélérations de la convergence pour un niveau d'hypervolume donnée sur les fonctions Rand-VC-ZDT (ratio des temps écoulés des algorithmes asynchrones entre $nProc$ et 1 processeur).

Algo.	AS-NSGAII			AS-MOCMA-ES		
HV cible	0.1	0.01	0.001	0.1	0.01	0.001
Rand-VC-ZDT1						
50	41.91	43.08	45	45.65	44.27	45.98
100	74.50	77.94	81.61	82.89	84.15	90.35
200	120	132.91	137.6	168	177.08	183.92
500	276.92	244.23	263.86	318.18	345.52	360.13
1000	343.37	378.57	410.6	414.47	461.95	507.39
Rand-VC-ZDT2						
50	48.58	42.75	45.63	52.4	44.03	44.3
100	82.69	75.84	81.13	98.86	84.95	87.2
200	141.56	129.16	143.33	187.09	162.7	161.48
500	259.81	240.77	263.62	365.54	335.66	340.62
1000	391.79	366.32	411.7	514.79	480	500
Rand-VC-ZDT3						
50	42.15	43.07	44.3	44.56	48.13	47.93
100	74.13	76.71	78.8	89.13	94.51	93.93
200	126.47	129.85	136.8	169.65	184.52	190.76
500	228.72	247.24	262.26	339.77	363.84	356.32
1000	353.42	379.01	404.73	518.98	567.76	613.86
HV cible	10	1	min	10	1	min
Rand-VC-ZDT6						
50	37.05	-	-	19.09	-	-
100	61.94	-	-	33.13	-	-
200	101.84	-	-	49.54	-	-
500	178.87	-	-	70.33	-	-
1000	266.88	-	-	98.21	-	-

portements des algorithmes AS-MO-CMA-ES et AS-NSGA-II précisément sur les fonctions Rand-VC-ZDT. Pour MO-CMA-ES, le "retard" en convergence discuté dans le paragraphe précédent est rattrapé relativement plus vite en comparant avec l'algorithme NSGA-II (figure 3.1-gauche et 3.2-gauche). Ce phénomène est aussi illustré dans la table 3.1 où l'accélération observée pour atteindre des niveaux bas de l'indicateur de l'hypervolume est plus grande pour AS-MO-CMA-ES que pour AS-NSGA-II. Ceci peut être expliqué par la plus grande pression sélective utilisée dans les variantes de NSGA-II qui consiste à choisir les parents pour le croisement en utilisant la sélection par tournoi, tandis que les variantes de l'algorithme MO-CMA-

TABLE 3.2 – accélérations de la convergence pour un niveau d'hypervolume donnée sur les fonctions Rand-VC-IHR (ratio des temps écoulés des algorithmes asynchrones entre $nProc$ et 1 processeur).

Algo.	AS-NSGAII			AS-MOCMA-ES		
HV cible	10	5	1	10	5	1
Rand-VC-IHR1						
50	40	36.36	-	35.71	36.76	24.06
100	66.66	61.53	-	78.12	85.22	52.95
200	100	94.11	-	172.41	178.57	83.73
500	250	190.47	-	409.83	407.6	206
1000	307.69	258.06	-	735.29	728.15	254.32
Rand-VC-IHR2						
50	54.97	-	-	-	-	-
100	242.01	61.53	-	-	-	-
200	361.94	229	-	-	-	-
500	1294	1178	-	-	-	-
1000	3029	2692	-	-	-	-
HV cible	10	5	min	10	5	min
Rand-VC-IHR3						
50	22.72	25.86	25	27.17	29.27	29.62
100	41.66	42.85	35.89	60.24	63.57	60.37
200	62.5	69.76	71.79	104.16	124.47	118.51
500	131.57	144.23	155.55	268.81	313.38	283.68
1000	185.18	283.01	224	520.83	589.4	544.21
Rand-VC-IHR6						
50	54.97	-	-	-	-	-
100	242.01	61.53	-	-	-	-
200	361.94	229	-	-	-	-
500	1294	1178	-	-	-	-
1000	3029	2692	-	-	-	-

ES choisissent aléatoirement un parent avant d'appliquer la mutation. De ce fait, les enfants résultant qui vont intégrer la queue d'attente sont en moyenne d'une qualité meilleure pour NSGA-II que pour MO-CMA-ES, et par conséquent MO-CMA-ES "souffre" moins si les individus insérés sont d'une qualité moindre, qui sont de plus en plus présents dans les grandes tailles des queues d'attente.
Les courbes des expérimentations pour toutes les fonctions des bencchmarks ZDT et IHR sont disponibles dna l'annexe B.1.
A noter que deux autres séries d'expériences ont été réalisées : dans la première la taille de population a été fixée à 500 tandis que dans la seconde série un algorithme

séquentiel a été utilisé à la place de la variante asynchrone stationnaire. Dans la version séquentielle, les enfants sont classés dans la queue d'attente dans le même ordre que celui dans lequel ils ont été crées. Cependant, l'analyse de ces expériences va dans le même sens que l'analyse faite dans cette section. Ces résultats sont représentés dans l'annexe B.2.

3.6 Contexte hétérogène

Cette section est dédiée à l'étude de la problématique d'hétérogénéité des coûts d'évaluation. Selon le modèle artificiel des variabilités de coût choisi, deux types d'expérimentation peuvent être distingués : la première série sera consacrée à l'étude du modèle de variabilité proportionnelle au valeurs des objectifs (modèle Eps-VC), alors que la deuxième série se focalisera sur l'étude du modèle de variabilité de coût d'évaluation dans lequel une région spécifique est plus coûteuse que le reste de l'espace (modèle Region-VC).

3.6.1 Le modèle Eps-VC

3.6.1.1 Réglages

Dans cette première série, 4 variantes des algorithmes NSGA-II et MO-CMA-ES sont considérées : la version générationnelle standard, la version stationnaire synchrone nommée (S-NSGA-II ou S-MO-CMA-ES), et deux variantes de la version asynchrone stationnaire qu'on note (AS-NSGA-II et AS-MO-CMA-ES) qui utilisent chacun un modèle artificiel de variabilité de calcul différent : la première utilise le modèle Rand-VC tandis que la deuxième variante utilise le modèle Eps-VC.
Les deux modes de représentations graphiques décrits précédemment seront utilisés pour comparer les différentes variantes des deux algorithmes. Pour le mode de représentation (hypervolume vs nombre d'évaluations) toutes les variantes seront comparées. Pour le deuxième mode (hypervolume vs temps écoulé) seulement deux variantes seront prises en considération : la version générationnelle standard et la version asynchrone stationnaire en utilisant le modèle Eps-VC. Pour ce dernier, chaque individu évalué se voit attribuer une durée artificielle d'évaluation calculée à l'aide du modèle artificiel basé sur la valeur de l'indicateur epsilon ; le temps écoulé de l'optimisation est calculé en prenant en compte le cumul des différentes évaluations. Pour la version générationnelle, le même modèle de coût (Eps-VC) est utilisé (pour des raisons de comparaison) pour calculer la durée d'évaluation des individus : après que tous les enfants d'une même génération aient été évalués, celui qui possède la durée d'évaluation la plus longue (calculée avec le modèle artificiel Eps-VC) détermine la durée d'évaluation de la génération. Le temps écoulé global de l'optimisation est obtenu par la suite en cumulant les durées d'évaluation des différentes générations.
Les deux autres variantes (stationnaire et asynchrone) utilisant le modèle Rand-VC ne sont pas visualisables en termes de temps écoulé. D'une part, la variante station-

naire est séquentielle, et par conséquent le modèle de coût ne peut pas être appliqué dans ce cas (pas de parallélisation). D'autre part, dans la variante asynchrone avec un modèle Rand-VC l'hétérogénéité provient seulement de la grille de calcul et ne dépend pas de l'évaluation de l'individu : l'hétérogénéité est simulée par le tirage aléatoire des individus dans la grille, sans prendre en compte le coût de l'évaluation de l'individu via une durée artificielle.
A noter que le modèle Région-VC n'a pas été retenu dans cette série d'expérimentations car la définition de la région coûteuse dépend de la forme de l'espace des objectifs de la fonction test en question, ce qui rend les différentes instances des benchmarks ZDT et IHR incomparables avec ce modèle.

3.6.1.2 Résultats

Les tables 3.3 et 3.4 résument les résultats obtenus sur les 8 problèmes test. Les chiffres représentent les valeurs moyennes de l'indicateur de l'hypervolume à 50000 évaluations (budget final) avec la signification statistique des différences obtenues en utilisant le test de Kruskal-Wallis.
Certains résultats clairs se dégagent pour NSGA-II avec le benchmark ZDT. S-NSGA-II dépasse toutes les autres variantes sur ZDT1 et ZDT2. Sur ZDT3, S-NSGA-II et les deux variantes de AS-NSGA-II (une avec le modèle Eps-VC et l'autre avec Rand-VC) sont équivalentes, et sont significativement meilleures que que la version générationnelle. Cependant, AS-NSGA-II avec le modèle Eps-VC est aussi le plus lent en termes de vitesse de convergence. Ceci est du au fait que les meilleurs individus sont plus "chers" à calculer avec ce modèle. Toutefois, la variante AS-NSGA-II avec le modèle Eps-VC rattrape l'algorithme générationnel après un certains nombre d'évaluations (30000 pour ZDT1 et ZDT3, 45000 pour ZDT2). Ces résultats confirment ceux obtenus dans [Chafekar 2003] et [Durillo 2008] selon lesquels les schémas d'évolution stationnaires sont meilleurs que les schémas générationnels dans plusieurs contextes, et particulièrement dans une configuration multi-objectif. Cette situation est typiquement visible dans la figure 3.3 : le graphe de droite montre le retard de NSGA-II en utilisant le modèle Eps-VC, tandis que le graphe de gauche illustre le bénéfice connu et attendu de l'utilisation d'un schéma stationnaire asynchrone en termes de temps global écoulé.
Les résultats obtenus sur les problèmes IHR avec NSGA-II montrent que la variante asynchrone stationnaire utilisant le modèle Eps-VC est meilleure que les autres algorithmes sur IHR1 et IHR2, par contre sur IHR3 et IHR6 tous les algorithmes sont équivalents. Notre interprétation de ce phénomène est la suivante : sur les problèmes IHR, une convergence prématurée vers une petite portion du front de Pareto est observée avec la version générationnelle standard de NSGA-II. Avec la variante AS-NSGA-II utilisant le modèle Eps-VC, les meilleurs individus sont "pénalisés" car ils sont plus coûteux, et paradoxalement ce phénomène permet de ralentir le processus de convergence prématurée, et par conséquent l'algorithme réussit à obtenir un front de Pareto plus diversifié.
Concernant MO-CMA-ES, les variantes basées sur un schéma stationnaire sont

meilleures que la version générationnelle standard sur toutes les fonctions sauf pour ZDT6. Aussi, ces résultats confirment ceux publiés dans [Igel 2007b] qui démontrent que la version stationnaire est meilleure que la version générationnelle. De plus, les variantes synchrones et asynchrones (avec les deux modèles artificiels) semblent avoir un comportement similaire en termes de vitesse de convergence. Pour toutes les fonctions ZDT, nous pouvons remarquer que la variante AS-MO-CMA-ES avec le modèle Eps-VC est la plus lente de tous (voir Annexe A où est représentée la totalité des résultats obtenus avec toutes les fonctions tests), de façon similaire et pour les mêmes raisons que celles observées sur NSGA-II. Cependant, sur la fonction IHR1 nous relevons un comportement un peu différent où le modèle Rand-VC pénalise l'algorithme AS-MO-CMA-ES (figure 3.4-gauche). Toutefois, la figure hypervolume vs temps écoulé (figure 3.4-droite) montre que la variante asynchrone stationnaire requiert beaucoup moins de temps de calcul que la variante générationnelle.

3.6.1.3 Discussion

Les expérimentations menées dans cette partie représentent la première partie de l'étude du phénomène d'hétérogénéité du temps de calcul dans les AEMO parallèles. Dans ce premier volet, l'hétérogénéité est simulée via le modèle Eps-VC dans lequel la durée de calcul est proportionnelle à la distance du front de Pareto. Ces expérimentations ont démontré que la variante asynchrone est meilleure que la variante générationnelle standard en termes des deux critères de comparaison retenus : la qualité de convergence et le temps global de l'optimisation. Obtenir un coût de calcul plus faible était bien entendu un résultat attendu, car la variante asynchrone assure une utilisation ininterrompue des processeurs sur la grille de calcul, alors que l'algorithme générationnel a besoin d'effectuer une synchronisation à chaque génération. En ce qui concerne la qualité de convergence (mesurée avec l'indicateur de l'hypervolume), les meilleurs résultats obtenus avec la variante asynchrone sont dus à son schéma de sélection qui est stationnaire, indépendamment du modèle artificiel de coût de calcul.

3.6.2 Le modèle Région-VC

La deuxième partie des expérimentations concernant la problématique du coût d'évaluation hétérogène qui traite le cas où une certaine région est plus coûteuse que le reste de l'espace de recherche. Le modèle artificiel utilisé pour simuler un tel comportement est le modèle Région-VC présenté dans la section 3.3.

3.6.2.1 Effets néfastes de l'asynchronicité

Dans cette section nous présentons les expérimentations relatives à l'effet de l'hétérogénéité sur les algorithmes asynchrones et qui peut conduire ces derniers à "rater" des régions entières du front de Pareto, du fait du coût de calcul important de ces régions. Pour illustrer ce phénomène, nous utilisons le modèle artificiel Région-VC sur la fonction ZDT3. La fonction Region-VC-ZDT3 possède un front de Pareto

Table 3.3 – Résultats moyens de l'indicateur de l'hypervolume obtenus sur 30 runs après 50000 évaluations avec les variantes de NSGA-II. Les exposants indiquent la signification statistique de la différence avec la colonne correspondante en utilisant le test de Kruskal-Wallis avec $\alpha = 0.01$.

fonction	Indicateur de l'hypervolume			
	Séquentiel		AS-NSGA-II	
	NSGA-II	S-NSGA-II	Eps-VC	Rand-VC
ZDT1	0.001923	$0.000123^{1,3,4}$	$0.000228^{1,4}$	0.000231^{1}
ZDT2	0.001865	$0.000431^{1,3,4}$	0.001121^{1}	0.000853^{1}
ZDT3	0.001265	0.000199^{1}	0.000422^{1}	0.000377^{1}
ZDT6	4.072532	4.060275	4.092183	4.080733
IHR1	0.572032^{4}	0.580593^{4}	$0.423853^{1,2,4}$	0.825886
IHR2	9.929486	11.652031	$5.091023^{1,2}$	$6.299612^{1,2}$
IHR3	1.979927	2.041639	1.796320	1.975204
IHR6	27.96567	28.82385	30.33832	28.46332

Table 3.4 – Résultats moyens de l'indicateur de l'hypervolume obtenus sur 30 runs après 50000 évaluations avec les variantes de MO-CMA-ES. Les exposants indiquent la signification statistique de la différence avec la colonne correspondante en utilisant le test de Kruskal-Wallis avec $\alpha = 0.01$.

fonction	indicateur de l'hypervolume			
	Séquentiel		AS-MO-CMA-ES	
	MO-CMA-ES	S-MO-CMA-ES	Eps-VC	Rand-VC
ZDT1	0.000223	0.000078^{1}	0.000065^{1}	0.000074^{1}
ZDT2	0.000223	0.000088^{1}	0.000073^{1}	0.000088^{1}
ZDT3	0.000118	0.000045^{1}	0.000041^{1}	0.000041^{1}
ZDT6	2.617498	2.617398	2.617390	2.617328
IHR1	0.279109	$0.173253^{1,3,4}$	0.191495	0.323821
IHR2	10.480423	10.474721	10.476632	10.480034
IHR3	2.111123	1.730129^{1}	1.556394^{1}	1.641038^{1}
IHR6	12.572234	$11.34743^{1,3,4}$	14.875943	13.075328

discontinu (clairement visible sur la figure 3.5-droite) et une région plus coûteuse du front de Pareto définie par $f_1 \in [0.3, 0.5]$ (un rectangle dans l'espace des objectifs) qui englobe une sous-partie du front de Pareto. En utilisant AS-NSGA-II, la région coûteuse n'est pas découverte même après 50000 évaluations alors que les autres sous-parties du front sont atteintes (Figure 3.5-gauche). Un comportement similaire, mais moins pénalisant est visible pour AS-MO-CMA-ES (Figure 3.5-droite) : on peut voir que significativement moins de points sont visités dans la région coûteuse. L'algorithme arrive toutefois à identifier la région coûteuse du front de Pareto.

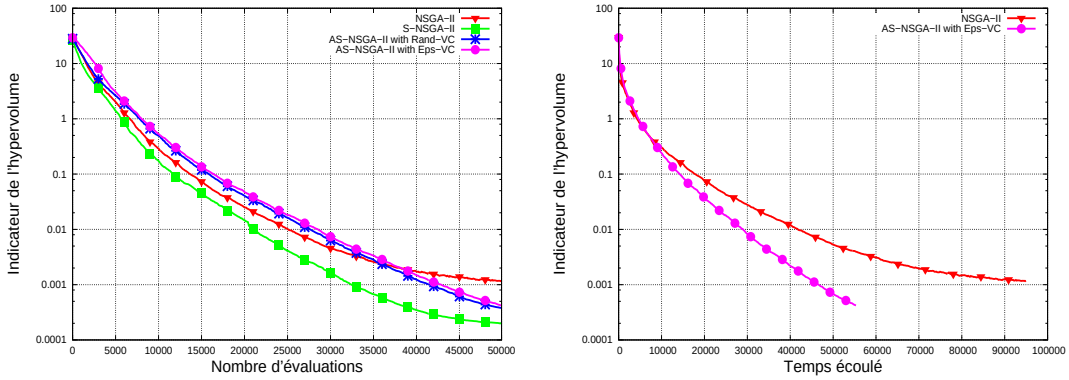


FIGURE 3.3 – Evolution de l'indicateur de l'hypervolume moyen pour les variantes de NSGA-II sur ZDT3. gauche : toutes les 4 variantes vs nombre d'évaluation. droite : variantes générationnelle et asynchrone stationnaire avec le modèle Eps-VC.

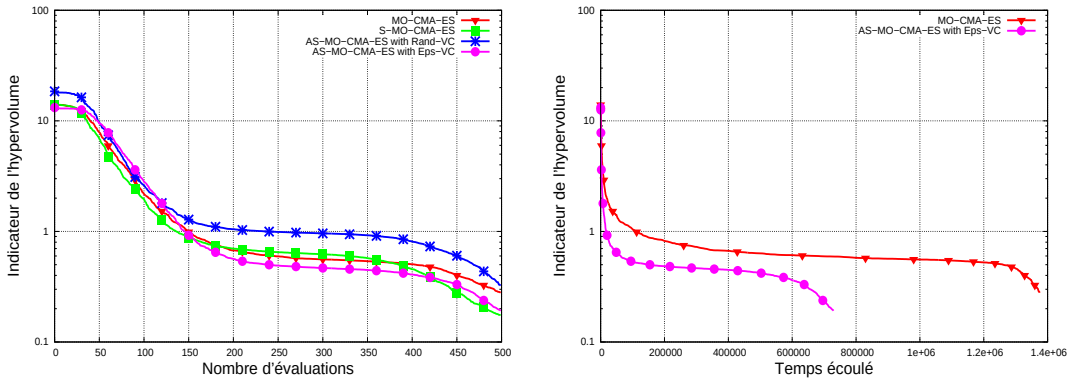


FIGURE 3.4 – Evolution de l'indicateur de l'hypervolume moyen pour les variantes de NSGA-II sur IHR1. gauche : toutes les 4 variantes vs nombre d'évaluation. droite : variantes générationnelle et asynchrone stationnaire avec le modèle Eps-VC.

3.6.2.2 La sélection basée sur la durée (DBS)

Comme nous avons pu le constater dans la section précédente, l'hétérogénéité des coûts d'évaluations peut avoir des conséquences dramatiques sur l'identification du front de Pareto, et ce phénomène s'est produit lors d'une application réelle comme nous allons le voir dans le chapitre 6. Une idée naturelle qui irait dans le sens d'essayer de contrebalancer les évaluations lentes ou coûteuses serait de modifier la procédure de sélection et d'intégrer au sein de cette dernière une procédure qui prend en considération la durée d'une évaluation.
L'idée de base de la sélection basée sur la durée (DBS pour *Duration Based Selection*) est d'introduire une probabilité (qui sera définie par l'utilisateur) $P_s \in [0, 1]$ dans la sélection parentale : avec une probabilité P_s la sélection se ferait de manière standard basée sur le principe de dominance, et avec une probabilité $1 - P_s$ la sélection serait basée sur le coût d'évaluation. Une question importante serait bien sur d'ajuster la

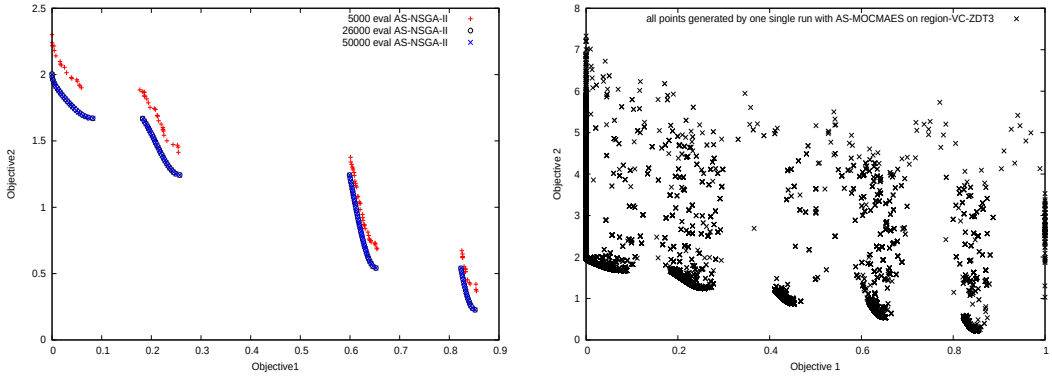


FIGURE 3.5 – Effets néfastes de l'asynchronicité sur Region-VC-ZDT3 (région coûteuse : $f_1 \in [0.3, 0.5]$). gauche : captures après 5000, 25000 et 50000 évaluations avec AS-NSGA-II. droite : tous les individus visités lors d'un essai avec AS-MOCMA-ES.

valeur de P_s de telle sorte que les individus coûteux ne soient pas outrageusement favorisés. Toutefois, les individus très coûteux vont aussi subir la sélection basée sur la dominance et vont être par conséquent éliminés s'ils représentent de mauvaises solutions d'un point de vue de Pareto.
Il est à noter que seule la sélection parentale est modifiée. Pour AS-NSGA-II, la sélection parentale est un tournoi (de taille 2 dans toutes les expériences présentées ici). Elle sera remplacée ici avec une probabilité de $1 - P_s$ par un tournoi de la même taille mais basé cette fois sur la durée de l'évaluation. Pour AS-MO-CMA-ES, il n'existe pas de sélection parentale en soi. De ce fait, avec une probabilité P_s, le parent sera choisi aléatoirement et avec une probabilité $1 - P_s$ un tournoi basé sur les durées des évaluations (de taille 2 aussi) est réalisé.

3.6.2.3 Expérimentations avec DBS

La première expérimentation consiste à valider l'utilisation de la sélection DBS comme un mécanisme de réparation de la perte de certaines parties du front de Pareto due au coût élevé de certaines régions (section 3.6.2.1). En effet, la figure 3.6-gauche montre que l'utilisation de l'algorithme DBS-AS-NSGA-II avec P_s=0.5 (des résultats similaires sont obtenus avec d'autres valeurs de P_s) permet à l'algorithme de trouver quelques points dans la région coûteuse après 26000 évaluations (les petits cercles noirs) avant de compléter par la suite le front avec les solutions optimales. Cette observation est généralisée dans la figure 3.6-droite qui illustre l'évolution de l'indicateur de l'hypervolume moyen en utilisant 3 différentes valeurs de P_s : augmenter la valeur de P_s est équivalent à utiliser davantage la sélection parentale classique basée sur la dominance. Par conséquent, le retard dans la découverte de la région coûteuse devient plus grand.

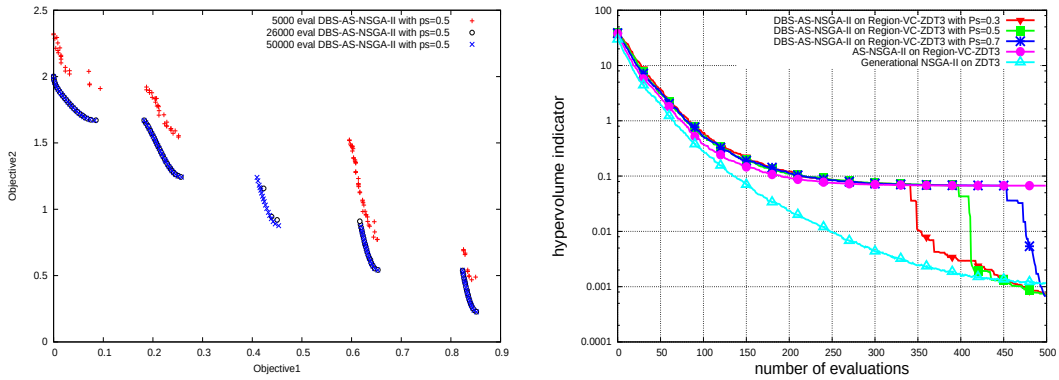


FIGURE 3.6 – Sélection basée sur la durée :résultats de DBS-AS-NSGA-II sur Region-VC-ZDT3 (région coûteuse : $f_1 \in [0.3, 0.5]$. gauche : population après 5000, 25000 and 50000 évaluations pour $P_s = 0.5$. droite : évolution de l'indicateur de l'hypervolume moyen pour différentes valeurs de P_s

3.6.2.4 DBS sans région coûteuse ?

D'autres expérience ont été menées sur la sélection DBS, en particulier pour vérifier que cette procédure ne va pas nuire à la convergence au front de Pareto dans le cas où l'hétérogénéité est uniformément distribuée (ceci revient à utiliser le modèle Rand-VC à la place de Region-VC). Les résultats montrent que le prix à payer pour l'utilisation de la sélection DBS est plutôt faible, comme le montre la figure 3.7 dans la quelle différentes variantes de DBS-AS-NSGA-II sont comparées sur Rand-VC-ZDT3. Nous pouvons remarquer cependant que les variantes DBS sont un peu dépassées par l'algorithme AS-NSGA-II mais restent quand même meilleures que la version générationnelle standard en termes de qualités.

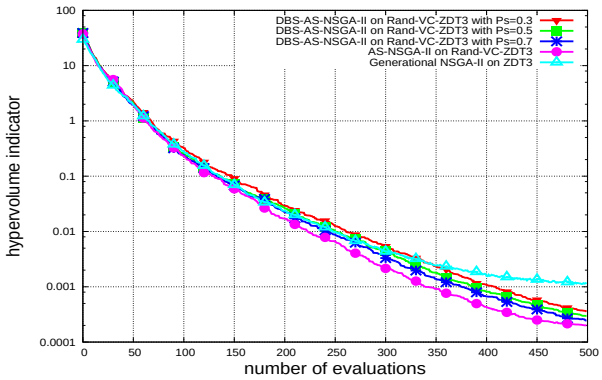


FIGURE 3.7 – évolution de l'indicateur de l'hypervolume moyen pour DBS-AS-NSGA-II avec différentes valeurs de P_s sur Rand-VC-ZDT3.

Un résultat plutôt surprenant a été obtenu avec la fonction Region-VC-ZDT6 avec une région coûteuse définie par $f_1 \in [0.8, 1]$.(voir figure 3.8-gauche). Dans ce cas,

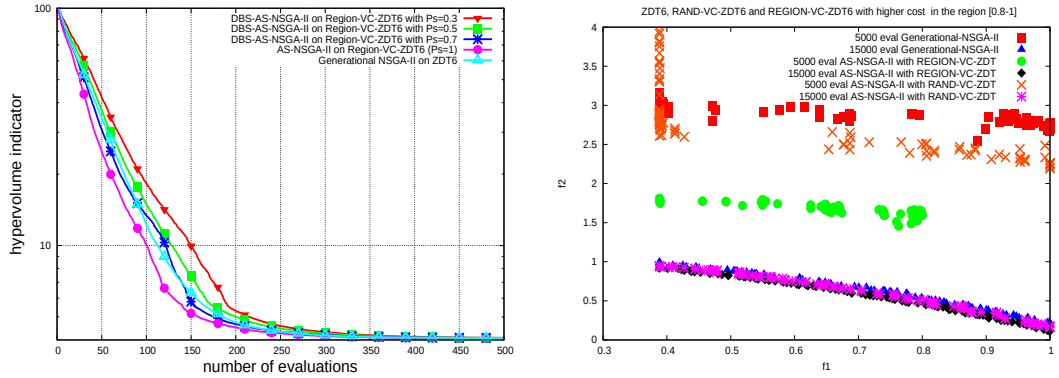


FIGURE 3.8 – Résultats des variantes de DBS-AS-NSGA-II avec des valeurs différentes de P_s sur Region-VC-ZDT6 avec la région coûteuse $f_1 \in [0.8, 1]$. gauche : évolution de l'indicateur de l'hypervolume moyen. droite : population après 5000, 15000 and 50000 évaluations des essais typiques.

l'utilisation de l'opérateur de sélection basée sur la durée DBS retarde la convergence plutôt que de l'accélérer. La visualisation des populations après différents nombre d'évaluations (figure 3.8-droite) confirment que AS-NSGA-II (équivalent à DBS-AS-NSGA-II avec $P_s = 1$) converge plus rapidement que les variantes DBS-AS-NSGA-II. Ce phénomène peut être expliqué par le fait que la sélection DBS retarde la convergence dans ce cas car elle essaye d'explorer les régions coûteuses. Les résultats finaux sont toutefois d'une qualité similaire pour tous les algorithmes en termes d'hypervolumes même pour des petites valeurs de P_s.

3.7 Conclusion

Ce chapitre a été consacré à l'étude des algorithmes asynchrones stationnaires dans un contexte de problèmes analytiques. Le premier objectif de cette étude a été de valider l'utilisation de ces algorithmes dans un contexte d'une problématiques réelle. Pour ce faire, des modèles artificiels traduisant des comportements différents ont été définis dans le but de simuler plusieurs cas de figures qui peuvent être rencontrés dans une application réelle. Ainsi, nous nous sommes intéressés dans un premier temps à l'étude de l'accélération qui peut être obtenue sur les algorithmes asynchrones en faisant varier la taille de la grille de calcul. Nous avons montré que ces accélérations sont non-linéaires avec ce types d'algorithmes, en balayant plusieurs valeurs de taille de grille de calcul. Dans un second temps, nous nous somme focalisés sur l'étude de l'hétérogénéité des évaluations en considérant deux types d'expérimentations utilisant chacun un modèle artificiel différent. Les expérimentations ont d'abord démontré l'effet néfaste de l'hétérogénéité : certaines régions du front de Pareto peuvent être complètement loupées à cause de son coût d'évaluation élevé (une situation qui a eu lieu lors d'une application réelle). Dans le but de faire face à ce problème nous avons proposé un opérateur de sélection basé sur la durée

DBS. Les résultats présentés ont montré la capacité des AEMO à découvrir ces régions coûteuses en utilisant l'opérateur DBS.
Dans le prochain chapitre, nous nous intéressons toujours à l'étude des algorithmes asynchrones stationnaires mais dans le contexte de l'optimisation en utilisant les méta-modèles afin de bénéficier doublement du gain qui peut être réalisé avec ces deux approches.

Chapitre 4

AEMO asynchrones et méta-modèles

Sommaire

4.1 Introduction **55**
4.2 Modèles d'approximation : État de l'art **56**
4.2.1 Approximation quadratique des moindres carrés 57
4.2.2 Modèles de krigeage 57
4.2.3 SVM (*Support Vector Machines*) 58
4.3 Méta-modèles & AEMO générationnels **61**
4.4 Méta-modèles & asynchronicité **62**
4.5 Réglages expérimentaux **63**
4.5.1 Réglages du méta-modèle 63
4.5.2 Réglages de l'AEMO 65
4.6 Résultats **65**
4.6.1 Réglage de N_{iter} et N_{eval} 65
4.6.2 Comparaison des algorithmes 66
4.7 Conclusion **67**

4.1 Introduction

Plusieurs approches basées sur l'utilisation méta-modèles dans le domaine de l'optimisation évolutionnaire ont été proposées dans les dernières décennies. L'objectif principal de ces approches est de réduire le coût global de l'optimisation en réduisant le nombre des évaluations effectuées en utilisant la fonction objectif réelle. Le principe de base de ces méthodes repose sur le remplacement du vecteur objectif F par un estimateur ou un méta-modèle $\hat{F}$. Un méta-modèle permet ainsi dans le cas des problématiques réelles d'optimisation d'évaluer les individus avec un coût négligeable comparé avec celui de l'évaluation de la fonction objectif réelle réalisée souvent à travers de lourdes simulations qui peuvent être de l'ordre de quelques jours. Il est clair que la qualité des résultats obtenus en utilisant cette approche dépend essentiellement de la capacité du méta-modèle à "bien" prédire la valeur de la fonction objectif. Dans ce cadre, plusieurs techniques d'approximation ont été proposées dans la littérature afin de construire les méta-modèles pour réduire le coût de l'optimisation évolutionnaire. Ces travaux bien qu'ils soient nombreux [Jin 2005], concernent

principalement le cas de l'optimisation évolutionnaire mono-objectif, le domaine des AEMO reste cependant faiblement exploré. [Voutchkov 2006] fut l'un des premiers à avoir utiliser les méta-modèles dans le contexte de l'optimisation évolutionnaire multi-objectif en utilisant l'algorithme NSGA-II et plusieurs méta-modèles basés sur différentes techniques d'approximation (krigeage, plusieurs variantes de la régression polynomiale). Les auteurs conclurent que le choix de la méthode d'approximation dépend de la nature de la fonction à approximer, mais que néanmoins la méthode du krigeage semble être la meilleure globalement.
Dans [Pilát 2011] plusieurs méthodes d'approximation ont été testées pour construire des méta-modèles (régression linéaire, régression SVM, réseaux de neurones) en utilisant les deux algorithmes de l'état de l'art NSGA-II et IBEA. Ces AEMO sont couplés à une méthode de recherche locale utilisée pour améliorer les enfants récemment crées.
Les travaux cités dans ce paragraphe utilisent pour toutes les méthodes d'approximation considérées un méta-modèle pour chaque objectif. Une nouvelle approche utilisant un méta-modèle unique pour l'ensemble des objectifs a été proposée récemment dans [Loshchilov 2010] basée sur la méthode de classification "Rank-SVM". Le méta-modèle de cette façon permet de différencier les individus dominés des individus non-dominés et est utilisé pendant l'évolution afin de pré-évaluer les individus et éliminer ceux qui ne sont pas prometteurs.

Dans ce chapitre, nous nous proposons d'étudier l'utilisation des méta-modèles dans le domaine des AEMO en utilisant la régression SVM. Nous commencerons d'abord par un survol rapide des principales méthodes d'approximation pouvant être utilisées en optimisation évolutionnaire multi-objectif (section 4.2). Par la suite, nous décrirons les différentes étapes de l'algorithme basé sur la version générationnelle standard de NSGA-II et la régression SVM en tant que méta-modèle (section 4.3) que nous proposons. L'utilisation des méta-modèles dans un contexte asynchrone stationnaire est étudiée dans la section 4.4 en couplant l'algorithme AS-NSGA-II proposé dans le chapitre précédent avec un méta-modèle de régression SVM. Les réglages effectués ainsi que les résultats des expérimentations seront présentés dans les sections 4.5 et 4.6 respectivement. La section 4.7 concluera ce chapitre.

4.2 Modèles d'approximation : État de l'art

Nous décrivons dans cette section, de façon succincte (pour plus de détails voir [Jin 2005]) les principales méthodes d'approximation utilisées pour construire des méta-modèles dans l'optimisation évolutionnaire à savoir : l'approximation quadratique, le krigeage et finalement la régression SVM.

4.2.1 Approximation quadratique des moindres carrés

Dans ce modèle, l'approximation est polynomiale avec un modèle du second ordre :

$$\hat{F}(X) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i,j=i}^{n} \beta_{ij} x_i x_j \tag{4.1}$$

où β_0, β_i et β_{ij} sont les coefficients à estimer .
L'estimation des coefficients du modèle peut être réalisée en utilisant la méthode des moindre carrés : on cherche à minimiser l'écart entre les valeurs estimées et les valeurs réelles. Autrement dit, on cherche à minimiser $\mathcal{X}$ donné par l'équation suivante :

$$\mathcal{X}^2 = \sum_{k=1}^{N} \left[F(X_k) - \hat{F}(X_k) \right]^2 \tag{4.2}$$

où N est le nombre de points échantillonnés, $F(X_k)$ est la valeur réelle de la réponse du système.

4.2.2 Modèles de krigeage

Le krigeage est une méthode d'estimation issue de la géostatistique. Elle porte le nom de son précurseur l'ingénieur minier D.G.Krige. Dans les années 50, Krige a développé une série de méthodes statistiques empiriques afin de déterminer la distribution spatiale des minerais à partir d'un ensemble de forage [Krige 1951]. Matheron a formalisé par la suite l'approche en utilisant les corrélations entre forages pour estimer la répartition spatiale [Matheron 1963]. Le modèle de krigeage repose sur l'estimation d'une fonction en un point inconnu en utilisant une combinaison d'un modèle global et d'une déviation locale :

$$y(X) = g(X) + Z(X) \tag{4.3}$$

où $g(X)$ représente le modèle global de la fonction d'origine et $Z(X)$ est une fonction gaussienne aléatoire avec une moyenne nulle et une covariance non nulle qui représente une déviation localisée par rapport au model global.
Généralement, $g(X)$ est polynomiale voire réduite à une constante β.
La matrice de covariance de $Z(X)$ quant à elle est définie comme suit :

$$Cov\left[Z(X^{(j)}), Z(X^{(k)})\right] = \sigma^2 \mathbf{R}\left[R(X^{(j)}, X^{(k)})\right] \qquad j,k = 1,...,N. \tag{4.4}$$

où R est une fonction de corrélation entre n'importe quelle paire de N échantillons et $\mathbf{R}$ est une matrice de corrélation symétrique de dimension $N \times N$ avec des valeurs égales à l'unité sur la diagonale. La forme de la matrice de corrélation peut être choisie par l'utilisateur. A titre d'exemple, la forme suivante a été utilisée dans plusieurs travaux [Brooker 1998, Giunta 1998] :

$$R\left(X^{(j)}, X^{(k)}\right) = exp\left[-\sum_{i=1}^{n} \theta_i |x_i^{(j)} - x_i^{(k)}|^2\right] \tag{4.5}$$

où les θ_i sont les paramètres de corrélation inconnus, $x_i^{(j)}$ et $x_i^{(k)}$ sont les i-ième éléments de l'échantillon de points X^j et X^k respectivement. La prédiction de $y(X)$ est une fonction des paramètres inconnus β et θ_i $i = 1, 2, ..., n$:

$$\hat{y} = \hat{\beta} + r^T(X)\mathbf{R}^{-1}(y - \beta\mathbf{I}) \tag{4.6}$$

où $\hat{y}$ est la valeur estimée de y , $\hat{\beta}$ est la valeur estimée de β, $\mathbf{I}$ est le vecteur unité et r est le vecteur de correlation .

4.2.3 SVM (*Support Vector Machines*)

Les *Support Vector Machines* sont un algorithme d'apprentissage qui s'inspire des techniques utilisées dans le domaine de l'apprantissage statistique depuis les années 1960. Dans sa forme actuelle, l'algorithme SVM a été largement développé au sein du laboratoire *AT&T Bell* par Vapnik et ses collaborateurs (voir [Boser 1992]). L'algorithme SVM, originalement développé pour des problèmes de classification, a été étendu aux problématiques de regression par la suite. Dans le cadre de ce travail, nous nous focaliserons uniquement sur la régression SVM dont nous présentons dans ce qui suit les principes de base. Cependant, une représentation détaillée peut être trouvée dans [Smola 2004].

4.2.3.1 Idée de base

Supposons un échantillon ou ensemble d'apprentissage :

$$S = \{(x_1, y_1), ..., (x_l, y_l)\} \subset (\mathcal{X} \times \mathbb{R}) \tag{4.7}$$

où $\mathcal{X}$ représente l'espace des paramètres ($\mathcal{X} = \mathbb{R}^d$). Dans la régression ε-SV [Vapnik 1995] l'objectif est de trouver une fonction $f(x)$ qui possède au pire une déviation ε par rapport aux valeurs cibles obtenues y_i pour toutes les données, et en même temps soit la plus régulière possible. En d'autres termes, les erreurs qui sont inférieures à ε seront tolérées, tandis que toute déviation plus grande que ε sera pénalisée. Nous commençons par décrire le cas des fonctions linéaires f qui sont de la forme :

$$f(x) = \langle w, x \rangle + b \qquad avec \quad w \in \mathcal{X}, b \in \mathbb{R} \tag{4.8}$$

où $\langle ., . \rangle$ est le produit scalaire dans $\mathcal{X}$. La régularité dans le cas de l'équation 4.8 se traduit par la minimisation de $||w||^2$. Nous pouvons exprimer ce problème comme un problème d'optimisation :

$$\begin{array}{ll} \text{minimiser} & \frac{1}{2}\|w\|^2 \\ \text{sous les contraintes} & \begin{cases} y_i - \langle w, x_i \rangle - b & \leq \varepsilon \\ \langle w, x_i \rangle + b - y_i & \leq \varepsilon \end{cases} \end{array} \tag{4.9}$$

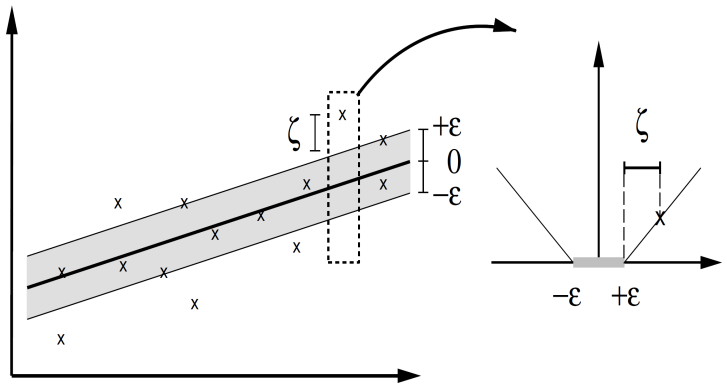


FIGURE 4.1 – réglage de la fonction perte dans le cas d'un SVM linéaire [Scholkopf 2002]

La supposition implicite dans le système 4.9 est traduite par l'existence d'une fonction f qui permet d'approximer toutes les pairs (x_i, y_i) avec une précision ε. En d'autres termes, ceci revient à dire que le problème d'optimisation convexe a une solution faisable. Cependant, dans certains cas où on souhaite tolérer certaines erreurs dans l'estimation, des variables de relaxation (ξ_i,ξ_i^*) peuvent être introduites afin de faire face à des données qu'on ne peut pas approcher linéairement à ε-près dans le problème d'optimisation 4.9. Ainsi, nous obtenons la formulation proposée dans [Vapnik 1995] :

$$\begin{array}{ll} \text{minimiser} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ \text{sous les contraintes} & \begin{cases} y_i - \langle w, x_i\rangle - b & \leq \varepsilon + \xi_i \\ \langle w, x_i\rangle + b - y_i & \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq 0 \end{cases} \end{array} \tag{4.10}$$

La constante $C > 0$ détermine le compromis entre la régularité de la fonction f et le degré de tolérance des déviations qui dépassent ε.
La figure 4.1 illustre le cas d'une fonction linéaire. Seulement les points en dehors de la région grisée (comprise entre $+\varepsilon$ et $-\varepsilon$) sont pénalisés d'une façon linéaire (la pénalisation est proportionnelle à la distance séparant le point en question de la limite de tolérance ε).

4.2.3.2 Problème dual et formulation quadratique

En associant à chacune des contraintes décrites ci-dessus un multiplicateur de Lagrange, le problème initial peut être décrit par son problème dual, qui est un problème d'optimisation quadratique sans contraintes.
Le lagrangien du système s'exprime comme suit :

$$\begin{aligned} L &:= \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) - \sum_{i}^{l}(\eta_i\xi_i + \eta_i^*\xi_i^*) \\ &- \sum_{i=1}^{l}\alpha_i(\varepsilon + \xi_i - y_i + \langle w, x_i\rangle + b) \\ &- \sum_{i=1}^{l}\alpha_i^*(\varepsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b) \end{aligned}$$

où η_i,η_i^*,α_i,α_i^* sont les multiplicateurs de Lagrange. A partir de cette formulation et en utilisant les dérivées partielles du Lagrangien L, la résolution n'utilise que des produit scalaires $\langle x_i, x\rangle$. La fonction f peut s'écrire :

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\langle x_i, x\rangle + b \qquad avec \qquad w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)x_i \tag{4.11}$$

Avec cette formulation w peut être décrit comme étant une combinaison linéaire des éléments x_i, aussi appelés vecteurs support, qui sont un sous ensemble des x_i initiaux.

4.2.3.3 Calcul de b

Le calcul du terme b dans l'équation 4.11 peut se faire en utilisant les conditions de Karush-Kuhn-Tucker :

$$\begin{aligned} \alpha_i(\varepsilon + \xi_i - y_i + \langle w, x_i\rangle + b) &= 0 \\ \alpha_i^*(\varepsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b) &= 0 \\ (C - \alpha_i)\xi_i &= 0 \\ (C - \alpha_i^*)\xi_i^* &= 0 \end{aligned}$$

En exploitant les équations du système 4.12 nous obtenons :

$$\begin{aligned} max\{\varepsilon + y_i - \langle w, x_i\rangle | \alpha_i < C \quad ou \quad \alpha^* > 0\} &\leq b \leq \\ min\{-\varepsilon + y_i - \langle w, x_i\rangle | \alpha_i > 0 \quad ou \quad \alpha^* < C\} & \end{aligned}$$

4.2.3.4 Non-linéarité et noyaux

L'étape suivante consiste à adapter les SVM au cas non-linéaire afin d'approcher des données fortement non-linéaires. Ceci peut être fait en utilisant une transformation de l'espace de recherche initial via une fonction $\Phi : \mathcal{X} \Rightarrow \mathcal{F}$ dans un espace des caractéristiques $\mathcal{F}$.

Le choix de la fonction Φ est crucial, et pour ne pas avoir à l'utiliser explicitement, et travailler dans un espace de très grande dimension, on se restreint aux fonctions Φ pour lesquelles il existe un noyau K tel que $K(x, x^{'}) := \langle \Phi(x), \Phi(x^{'}) \rangle$.
Comme la résolution du problème dual n'utilise que des produits scalaires, on peut réaliser tous les calculs dans l'espace initial avec K. L'extension de l'équation 4.11 au cas non-linéaire en utilisant le noyau K s'écrit donc :

$$f(x) = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) K(x_i, x) + b \tag{4.12}$$

4.3 Méta-modèles & AEMO générationnels

Après avoir présenté les principales méthodologies utilisant les méta-modèles dans le domaine de l'optimisation évolutionnaire, nous nous focalisons dans cette section sur l'utilisation de la régression SVM pour réduire le nombre d'évaluations des AEMO. L'algorithme proposé est basé sur NSGA-II. Le modèle régression SVM permettra d'estimer les valeurs de la fonction objectif. L'algorithme utilisera à la place le modèle basé sur la régression SVM qui va apprendre au fur et à mesure de l'évolution en enrichissant le méta-modèle avec les individus réellement évalués.
Cependant, la question clé reste de savoir *quand* ? et *combien* ? d'individus devraient être évalués avec la fonction objectif réelle. Il est évident qu'une utilisation excessive du modèle d'estimation pendant plusieurs générations risque de guider l'algorithme vers des zones non prometteuses à cause de l'erreur du méta-modèle, tandis qu'une utilisation trop faible de l'estimateur SVM ne permet pas de réaliser un gain considérable en termes de réduction du nombre d'évaluations. Il n'existe pas malheureusement des règles universelles dans ce sens, et seule l'expérience permet de faire un réglage en fonction des résultats obtenus. Dans ce qui suit, la notation utilisée pour désigner ces deux paramètres est la suivante : A chaque N_{iter} générations, N_{eval} individus seront évalués avec la fonction objectif réelle, le reste des individus sera évalué en utilisant le modèle de régression SVM.
Dans l'algorithme Reg-SVM-NSGA-II que nous proposons dans cette section, pour chaque objectif un modèle de régression SVM est construit afin d'approcher cet objectif. Nous aurons de ce fait autant de modèles de régression SVM que d'objectifs à optimiser.
L'algorithme 10 résume les principales étapes de l'algorithme Reg-SVM-NSGA-II que nous décrivons dans ce qui suit. L'algorithme débute comme l'algorithme NSGA-II (chapitre 2, algorithme 1) avec l'étape d'initialisation : une population P_0 de taille N est créée et est évaluée en utilisant la fonction objectif réelle. L'archive initiale A_0 est initialisée avec les valeurs de P_0 (lignes 1-3). Un méta-modèle basé sur la régression SVM est construit en utilisant l'archive A_0 pour chaque objectif à optimiser (ligne 4). A chaque génération, la population d'enfants Q_t est générée à partir de la population des parents P_t. Ces enfants sont évalués à l'aide du méta-modèle construit auparavant (lignes 6-7). A chaque N_{iter} générations, N_{eval} individus sont

choisis aléatoirement parmi les individus présents dans la population des enfants Q_t (lignes 9-12) . Ces N_{eval} individus sont évalués avec la fonction objectif réelle et sont intégrés dans l'archive A_t, qui verra les N_{eval} plus mauvais individus disparaitre (lignes 14-16). Par la suite, le méta-modèle est reconstruit en utilisant l'archive actualisée, et les valeurs des objectifs estimés des N_{eval} individus dans la population Q_t seront remplacés par les valeurs réelles qui viennent d'être calculées.
La suite de l'algorithme reprend les étapes classiques de remplacement de la population des parents par celle des enfants de l'algorithme de base NSGA-II (lignes 20-22)

Algorithm 10 Reg-SVM-NSGA-II

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0| \leftarrow N$
2: Eval-Real(P_0)
3: $A_0 \leftarrow P_0$ // archive initiale
4: Reg-SVM(A_0)
5: **repeat**
6: $Q_t \leftarrow$ variation (Select(P_t)) // sélection parentale :Tournoi de taille 2 ; variation : croisement, mutation.
7: Eval-SVM(Q_t)
8: **if** # générations $\equiv 0[N_{iter}]$ **then**
9: **for** i=1,... N_{eval} **do**
10: $q_t^i \leftarrow U(Q_t)$
11: Real-Eval(q_t^i)
12: **end for**
13: Update-Eval(Q_t) // Actualiser les valeurs de la fonction objectif des individus réellement évalués.
14: $A_t \leftarrow A_t \cup \{q_t^i, 1 \leq i \leq N_{eval}\}$
15: $A_t \leftarrow$ Tri-dominance(A_t)
16: $A_{t+1} \leftarrow A_t[0 : N]$
17: Reg-SVM(A_t)
18: **end if**
19: $R_t \leftarrow P_t \cup Q_t$
20: Tri-dominance(R_t) // Principe de dominance + critère de surpeuplement
21: $P_{t+1} \leftarrow R_t[0 : N]$ // choisir les N premiers éléments
22: $t \leftarrow t + 1$
23: **until** Critère d'arrêt rencontré

4.4 Méta-modèles & asynchronicité

Dans cette section nous nous focalisons sur l'utilisation des méta-modèles dans un contexte stationnaire asynchrone. L'algorithme 10 décrit dans la section précédente utilise un schéma d'évolution générationnel. Il est évident que l'utilisation des méta-modèles dans le domaine de l'optimisation évolutionnaire a pour objectif

principal la réduction du coût des optimisations réelles pour lesquelles une évaluation coûte quelques heures voire quelques jours. Cependant, les applications réelles sont souvent caractérisées par une hétérogénéité du coût des évaluations. Dans ce contexte, comme nous l'avons vu dans le chapitre précédent, l'utilisation des schémas générationnels augmente considérablement le coût de l'optimisation. L'utilisation des méta-modèles avec un AEMO générationnel bien qu'elle permette de réduire le nombre total des évaluations, ne permet pas une utilisation efficace des processeurs disponibles sur la grille de calcul car certains processeurs vont rester inoccupés à chaque génération en attendant l'individu le plus lent. Afin de tirer bénéfice des deux approches, c'est à dire d'une part utiliser les méta-modèles pour réduire le nombre des évaluations et d'autres part utiliser les AEMO stationnaires asynchrones pour réduire le coût de l'optimisation en utilisant efficacement les processeurs sur la grille, nous proposons de coupler ces deux concepts en rendant l'algorithme Reg-SVM-NSGA-II décrit ci-dessus stationnaire asynchrone.
L'algorithme Reg-SVM-AS-NSGA-II (algorithme 11) commence par générer une population aléatoire de parents P_0. L'archive initiale A_0 est remplie avec les individus de P_0 sera utilisée pour construire un méta-modèle en utilisant la régression SVM (lignes 1-4). Parallèlement, la queue d'attente est initialisée en appliquant les opérateurs de variation sur la population P_0. A chaque génération, un enfant q^t est choisi aléatoirement entre les deux enfants générés à partir de deux parents sélectionnés par tournoi (schéma d'évolution stationnaire) et est évalué à l'aide du méta-modèle construit précédemment. A chaque N_{iter} générations, l'algorithme lance l'évaluation de l'enfant généré sur la grille de calcul et récupère le premier individu ayant terminé sa propre évaluation. Comme au chapitre 3, pour simuler ce phénomène on utilise le modèle artificiel de coût qui maintient une queue d'attente simulant la grille de calcul : l'enfant généré sera échangé avec l'individu ayant la durée la plus courte présent dans la queue d'attente b_i^t et sera inséré dans la queue B après avoir calculé sa propre durée artificielle (lignes grisées dans l'algorithme 11). le nouveau enfant b_i^t regagnera l'archive A_t et le plus mauvais élément de l'ensemble $(A_t \cup \{b_i^t\})$ sera éliminé. Par la suite, le méta-modèle est reconstruit en utilisant l'archive actualisée A_t. La suite de l'algorithme est identique aux étapes de remplacement de l'enfant dans la population des parents dans un algorithme NSGA-II stationnaire.

4.5 Réglages expérimentaux

Nous décrivons dans cette section les différents réglages effectués pour la réalisation des expérimentations. Nous distinguons les réglages sur les paramètres du méta-modèle et ceux relatifs à l'AEMO.

4.5.1 Réglages du méta-modèle

Le premier réglage à effectuer sur le méta-modèle est le choix de la fonction noyau qui permet de prendre en compte la non-linéarité de l'espace de recherche. La

Algorithm 11 Reg-SVM-AS-NSGA-II

1: $t \leftarrow 0$, $P_0 \leftarrow$ random(), $|P_0| \leftarrow N$
2: Eval-Real(P_0)
3: $A_0 \leftarrow P_0$ // archive initiale
4: Reg-SVM(A_0)
5: $B_0 \leftarrow$ Variation (P_0) Simule la queue d'attente
6: Calcul-Durée (B_0)
7: **repeat**
8: $(p_1^t, p_2^t) \leftarrow$ Select(P_t). // sélection de deux parents :Tournoi de taille 2
9: $(q_1^t, q_2^t) \leftarrow$ Variation (p_1^t, p_2^t) // croisement, mutation
10: $q^t \leftarrow U(q_1^t, q_2^t)$ // choix aléatoire d'un des deux enfants
11: Eval-SVM$\{q^t\}$
12: **if** # générations $\equiv 0[N_{iter}]$ **then**
13: Lancer-Eval$\{q^t\}$
14: Calcul-Durée $\{q^t\}$
15: $i \leftarrow$ Index de l'individu avec la durée la plus courte B_t
16: $B_{t+1} \leftarrow (B_t \cup \{q^t\}) \backslash \{b_i^t\})$
17: $A_t \leftarrow A_t \cup \{b_i^t\}$
18: $A_t \leftarrow$ Tri-dominance(A_t)
19: $A_{t+1} \leftarrow A_t[0 : N]$
20: Reg-SVM(A_t)
21: **end if**
22: $R_t \leftarrow P_t \cup \{b_i^t\}$
23: Tri-dominance(R_t) // Principe de dominance + critère de surpeuplement
24: $P_{t+1} \leftarrow R_t[0 : N]$ // choisir les N premiers éléments
25: $t \leftarrow t + 1$
26: **until** Critère d'arrêt rencontré

fonction noyau que nous utilisons est basée sur les noyaux Gaussiens et est donnée par :

$$k(x, x^{'}) = e^{-\frac{\|x - x^{'}\|^2}{2\sigma^2}} \tag{4.13}$$

Les paramètres à régler pour le méta-modèle sont de ce fait au nombre de trois : C, ε pour le calcul de b et σ pour le calcul de la fonction noyau. Plusieurs techniques de réglages telles que la validation croisée peuvent être utilisées dans ce contexte. Le principe de base de cette méthode est de construire plusieurs fois le méta-modèle en utilisant dans chaque réalisation une configuration spécifique des paramètres à fixer sur un ensemble de points dont on connait les valeurs réelles de la fonction objectif. L'objectif est de choisir à la fin la configuration qui permet de réduire la somme des erreurs quadratiques sur tous les points de l'ensemble (la différence entre les valeurs estimées et les valeurs réelles).
Les valeurs des paramètres du méta-modèle utilisées dans le cadre des expérimenta-

tions menées dans ce chapitre reposent sur un réglage manuel réalisé au préalable : $C = 100$, $\varepsilon = 0.1$ et $\sigma = 0.2$.

4.5.2 Réglages de l'AEMO

Dans toutes les expérimentations présentées dans ce chapitre la taille de la population est fixée à 100. Le réglage des deux paramètres N_{iter} et N_{eval} feront l'objet de la première série d'expérimentations qui sera présentée dans la section suivante. Les algorithmes utilisés sont Reg-SVM-NSGA-II et Reg-SVM-AS-NSGA-II présentés ci-dessus. Pour l'algorithme Reg-SVM-AS-NSGA-II, le modèle artificiel utilisé pour calculer la durée de l'évaluation est le modèle Rand-VC présenté dans le chapitre précédent : l'enfant récemment crée par l'algorithme est inséré de façon aléatoire dans la queue d'attente. Le nombre total des évaluations est fixé à 10000 (soit 100 générations dans le cas de l'algorithme générationnel Reg-SVM-NSGA-II).

4.6 Résultats

4.6.1 Réglage de N_{iter} et N_{eval}

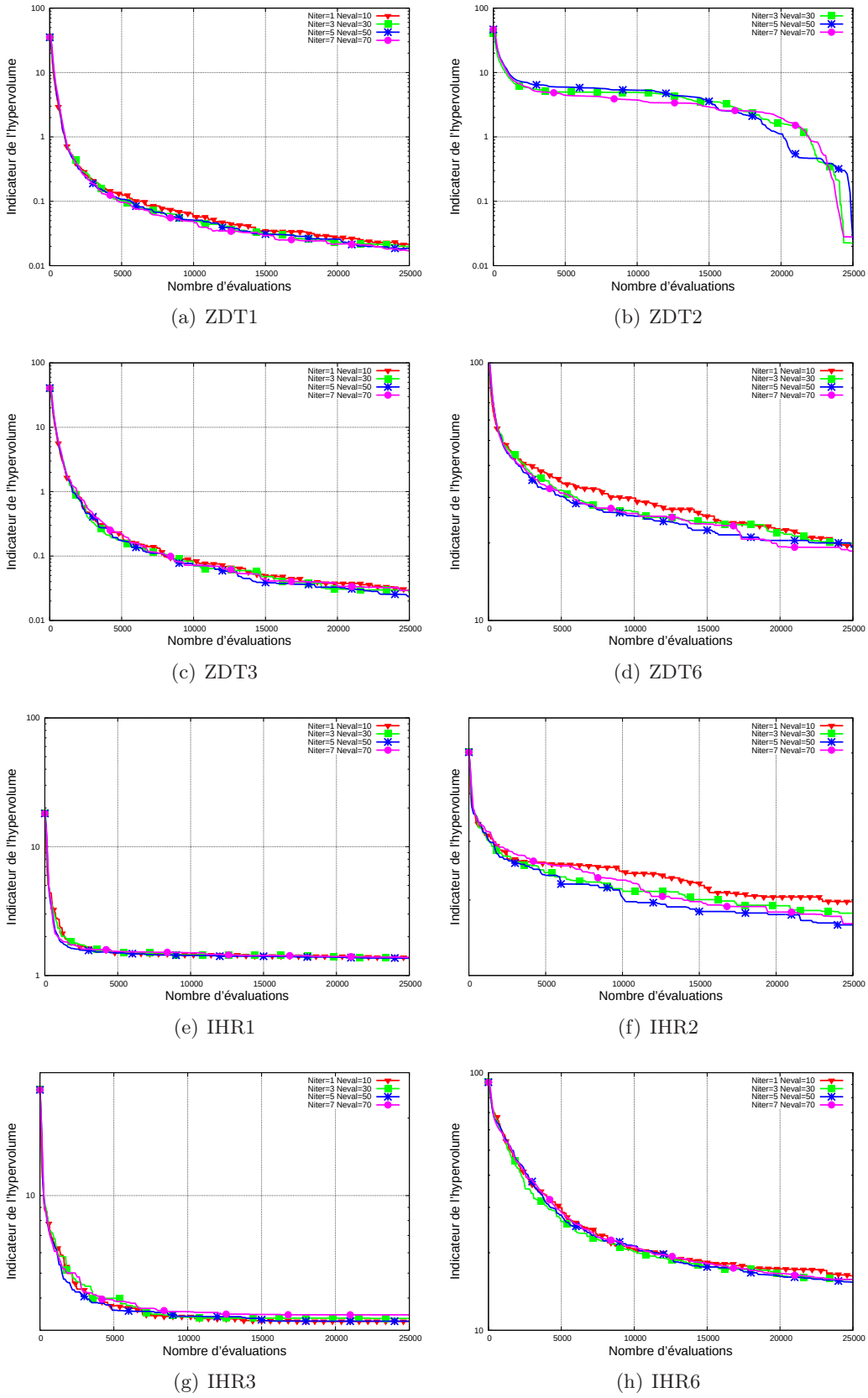
La première série d'expérimentations concerne le réglage des paramètres N_{iter} et N_{eval}. Pour cela, quatre configurations différentes représentant chacune un réglage spécifique de ces deux paramètres ont été testées. La figure 4.2 illustre l'évolution de l'indicateur de l'hypervolume moyen pour chacune des configurations en utilisant l'algorithme Reg-SVM-NSGA-II sur les benchmarks ZDT et IHR. Nous pouvons constater que l'évolution de l'hypervolume moyen est assez robuste à la variation du couple N_{iter}-N_{eval} pour l'ensemble des fonctions. Dans ce qui suit, compte tenu des résultats obtenus dans cette section, nous fixons les valeurs de N_{iter} et de N_{eval} à 5 et 50 respectivement, c'est à dire qu'à chaque 5 générations de l'AEMO, 50 individus vont être évalués en utilisant la fonction objectif réelle, le reste des évaluations étant réalisé évidemment en utilisant le modèle de régression SVM. Il est important de noter ici, que dans le contexte de l'utilisation de l'algorithme Reg-SVM-NSGA-II sur des problématiques réelles d'optimisation, le nombre d'individus à évaluer N_{eval} est déterminé par le nombre de processeurs disponible sur la grille de calcul, autrement certains processeurs resterons inoccupés tout au long de l'optimisation dans le cas où la capacité de la grille est supérieure à N_{eval}. Cependant, ce constat n'est plus valable avec l'algorithme stationnaire asynchrone Reg-SVM-AS-NSGA-II car aucun processeur ne reste inoccupé. Par conséquent, le choix du nombre d'individus à évaluer N_{eval} peut se faire indépendamment du nombre des processeurs disponibles sur la grille de calcul (en supposant que le temps d'évaluation en utilisant le méta-modèle est négligeable comparant avec le temps de l'évaluation réelle : quelques secondes contre quelques jours).

4.6.2 Comparaison des algorithmes

La deuxième série d'expérimentations est dédiée à la comparaison des deux algorithmes basés sur l'utilisation des méta-modèles (régression SVM) présentés dans ce chapitre à savoir : Reg-SVM-NSGA-II et Reg-SVM-AS-NSGA-II. Ces deux algorithmes sont comparés à la version standard générationnelle de l'algorithme NSGA-II. La figure 4.3 illustre l'évolution de l'hypervolume moyen des trois algorithmes en fonction du nombre d'évaluations. Nous pouvons constater globalement que les deux variantes utilisant la régression SVM permettent d'avoir une meilleure convergence en début d'évolution par rapport à la version standard de NSGA-II (sauf pour la fonction IHR2 où le constat est inversé). Cependant sur quelques fonctions, les algorithmes utilisant un méta-modèle semblent être rattrapés, et parfois dépassés par la variante standard de NSGA-II. Comme expliqué lors de la comparaison des AEMO effectuée dans le deuxième chapitre (section 2.9), les différences entre les valeurs de l'hypervolume calculées après un grand nombre d'évaluations (quelques miliers pour les benchmarks ZDT et IHR) traduisent uniquement une différence dans la distribution des points tout au long du front. Or, dans le contexte des optimisations réelles, l'objectif principal reste de converger d'abord le long du front de Pareto en un nombre très réduit d'évaluations, ce qui correspond sur les courbes de la figure 4.3 à la première partie de la courbe où le speed-up de convergence est grand. Toutefois, l'explication de ce phénomène peut être liée à l'erreur du méta-modèles qui ne permet pas d'avoir une distribution aussi bonne que celle trouvée par l'algorithme standard.
D'autre part, une différence entre les algorithmes Reg-SVM-NSGA-II et Reg-SVM-AS-SGA-II peut être relevée. La variante générationnelle du méta-modèle permet d'avoir une meilleure accélération que la version stationnaire asynchrone. Cette dernière semble être dépassée par la version standard de NSGA-II sur quelques fonctions du benchmark IHR (IHR3 et IHR6) mais reste tout de même meilleure que NSGA-II standard sur le benchmark ZDT (sauf ZDT2 où les courbes se croisent à 5000 évaluations). Ce phénomène peut être expliqué par le retour des enfants dans un ordre différent que celui dans lequel ils ont été générés. Ce constat est similaire à celui fait dans le chapitre précédent (chapitre 3) où la version utilisant modèle artificiel Rand-VC semblait être légèrement dépassée par la variante standard. Ce phénomène qui semble être accentué dans le cas présent, peut être expliqué par l'erreur liée à l'utilisation du méta-modèle. Cependant, tenant compte de l'objectif principal de l'utilisation des méta-modèles qui est la réduction du coût global des optimisations réelles, la variante stationnaire asynchrone dépassera la variante générationnelle standard car, comme expliqué, aucun processeur dans la grille de calcul ne reste inoccupé en attendant la fin de toutes les évaluations de l'actuelle génération.

4.7 Conclusion

Dans ce chapitre nous avons étudié l'utilisation des méta-modèles dans l'optimisation évolutionnaire multi-objectif. L'objectif de ce couplage est de réduire le nombre total des évaluations ce qui permet de réduire le coût global de l'optimisation. Dans un premier temps, la régression SVM a été utilisée pour construire un méta-modèle sur un algorithme générationnel. Dans un second temps, la régression SVM a été utilisée dans un contexte stationnaire asynchrone qui traduit le comportement des AEMO dans le cas d'une application réelle parallèle. La comparaison des deux algorithmes avec la version standard de l'état de l'art montrent une certaine amélioration dans le speed-up de convergence au début du processus évolutionnaire. La version stationnaire asynchrone, bien qu'elle soit moins meilleure que la version générationnelle en termes de qualité de convergence, elle permet cependant de réduire considérablement le coût global de l'optimisation grâce à sa meilleure utilisation des processeurs présents sur la grille de calcul.
Ce chapitre représente le dernier chapitre de développements algorithmiques dans cette thèse. Les deux chapitres suivants seront consacrés à l'application des approches algorithmiques développés jusqu'à présent sur les problématiques réelles qui représente le cadre industriel de cette thèse.



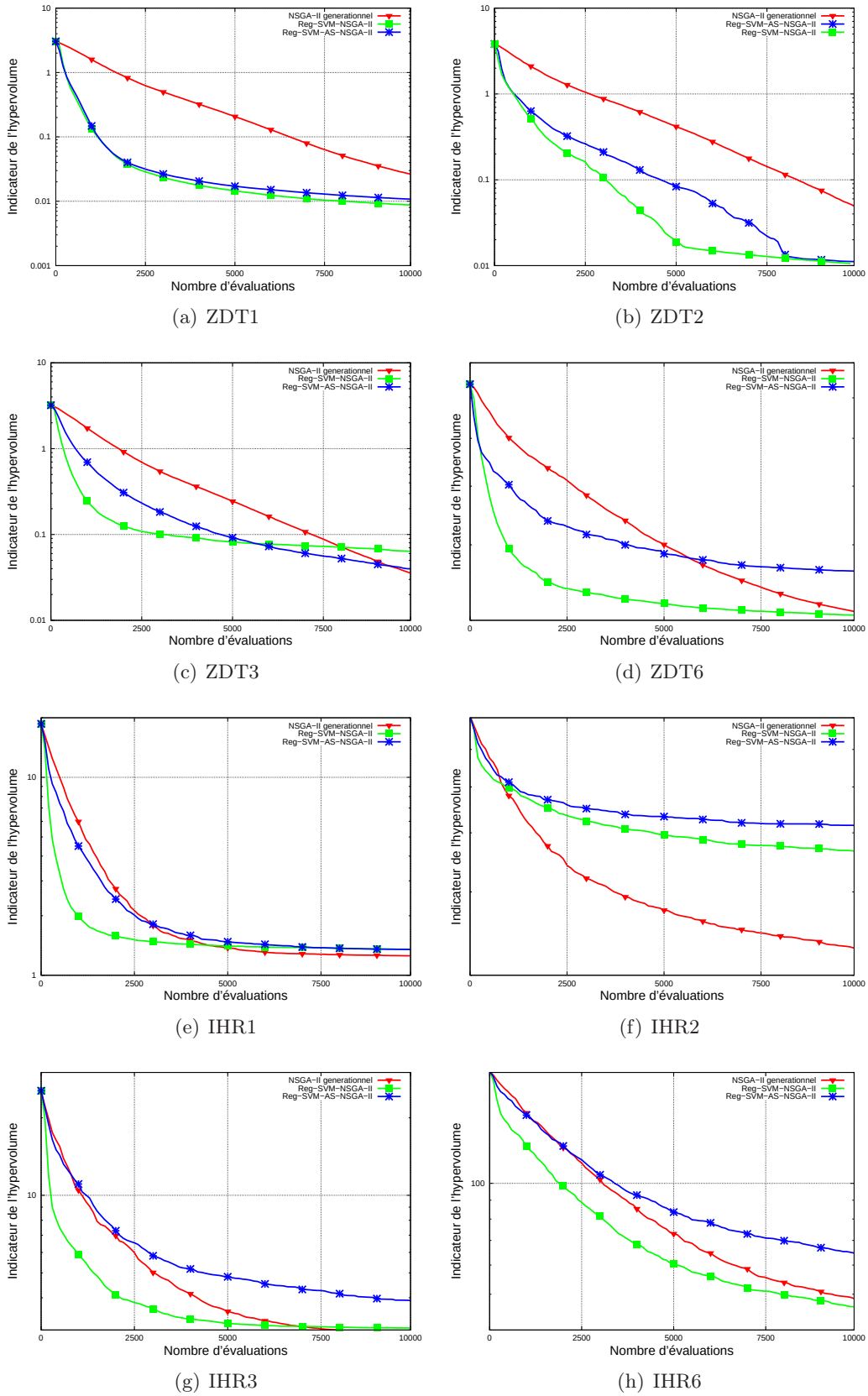
(a) ZDT1 (b) ZDT2

(c) ZDT3 (d) ZDT6

(e) IHR1 (f) IHR2

(g) IHR3 (h) IHR6

FIGURE 4.2 – Evolution de l'indicateur de l'hypervolume moyen pour différents couplets de valeurs de N_{iter} et N_{eval}



(a) ZDT1 (b) ZDT2

(c) ZDT3 (d) ZDT6

(e) IHR1 (f) IHR2

(g) IHR3 (h) IHR6

FIGURE 4.3 – Evolution de l'indicateur de l'hypervolume moyen pour les deux variantes utilisant la régression SVM et la version standard avec les benchmarks ZDT et IHR

Deuxième partie

Optimisation de la combustion Diesel

Chapitre 5

Généralités

5.1 Introduction

Avant de présenter les deux études d'optimisation réalisées dans cette deuxième partie applicative, nous commençons dans ce chapitre par rappeler quelques généralités sur les moteurs Diesel.
Les moteurs Diesel sont des moteurs alternatifs à combustion interne alimentés au gazole. Constitués de pistons coulissant dans des cylindres, leur fonctionnement repose sur l'auto-inflammation du gazole.
Les gaz pénètrent dans le cylindre à travers les soupapes d'admission. Lors de la combustion du mélange air/carburant, les gaz de combustion poussent le piston vers le bas. L'ensemble composé de la bielle et du vilebrequin permet de transformer le mouvement alternatif vertical du piston en un mouvement de rotation au niveau du vilebrequin. Ce mouvement de rotation est ensuite transmis au roues du véhicule par la boite e vitesses.

5.2 Cycle d'un moteur Diesel

Un cycle de fonctionnement correspond à deux tour de vilebrequin, c'est à dire deux montées et descentes du piston. Le cycle Diesel d'un moteur à quatre temps comporte les étapes suivantes (figure 5.1) :

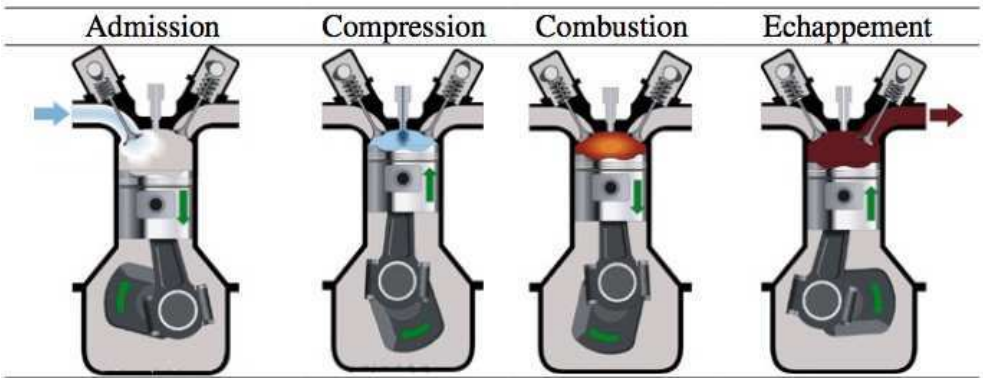


Figure 5.1 – Phases d'un cycle Diesel

1. **Admission** : la soupape d'admission s'ouvre et le piston descend en aspirant l'air frais.
2. **Compression** : La remontée du piston entraîne la compression de l'air avec un rapport volumétrique élevé. Malgré les pertes thermiques aux parois, la température de fin de compression est de 600 °C à 1500 °C. En fin de compression, le carburant est injecté sous forme d'un ou plusieurs jets pulvérisés dans le cylindre (au voisinage du point mort haut).
3. **Combustion et détente** : Sitôt injecté, le carburant s'enflamme presque instantanément, sans qu'il ne soit nécessaire de recourir à un allumage commandé par bougie. La combustion rapide qui s'ensuit constitue le temps moteur. En brûlant, le mélange augmente fortement la température et la pression dans le cylindre (60 à 200 bars), repoussant le piston qui fournit une force de travail sur une bielle, laquelle entraîne la rotation du vilebrequin (ou arbre manivelle faisant office d'axe moteur).
4. **Échappement** : Le piston remonte et évacue les gaz brûlés par la soupape d'échappement.

5.3 Mécanisme d'auto-inflammation

La combustion Diesel repose sur l'auto-inflammation du carburant en présence de l'air comprimé. Cette auto-inflammation a lieu lorsque la température du mélange air/carburant atteint un certain seuil appelé "température d'auto-inflammation". En effet, pour des températures inférieures à ce seuil, le mélange s'oxyde pour donner des peroxydes dont la concentration croît avec la température. A partir d'une concentration critique des peroxydes, les réactions chimiques deviennent instables et par un mécanisme de réactions en chaine, elles donnent lieu à une combustion.
A noter que cette inflammation n'est pas instantanée dans les moteurs Diesel. Le temps pendant lequel le système est maintenu aux même conditions de pression et de température avant d'observer l'inflammation proprement dite est appelé *délai d'auto-inflammation*. Ce dernier est composé d'un délai physique lié à la vaporisation des gouttelettes, et d'un délai chimique lié à la cinétique des processus chimiques.

5.4 Formation des polluants

Les polluants majeurs émis lors de la combustion Diesel sont les oxydes d'azote (NOx) et les particules de suies.
Les oxydes d'azote qui représentent l'ensemble des molécules NO, NO_2 et N_2O sont produits lors de la combustion dans le moteur. Le mécanisme de formation des NOx se produit par simple effet d'augmentation de la température d'air, qui contient l'azote et l'oxygène.
Concernant la formation des suies, comme le mélange air/carburant n'est pas homogène, il existe dans la chambre de combustion des zones plus riches en carburant qui

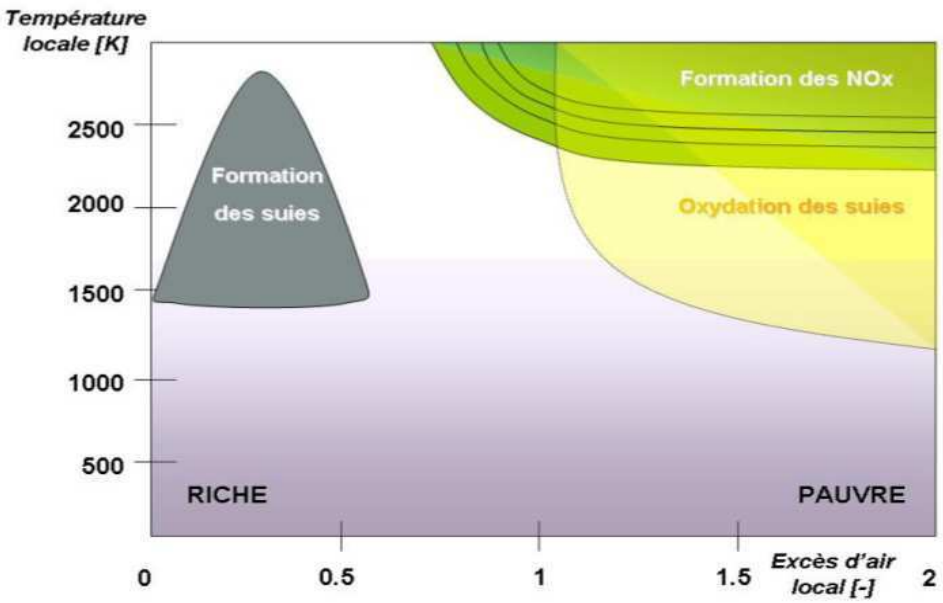


FIGURE 5.2 – Diagramme de Pischinger [Pischinger 1988]

sont plus favorables à la formation des suies. En revanche, la formation des NOx est favorisée par une élévation de la température dans les zones où l'excès d'air est plus important. Ceci donne lieu à un compromis NOx-suies illustré par le diagramme de Pischinger (figure 5.2). Ce compromis illustre aussi la difficulté du problème d'optimisation de la combustion Diesel que nous souhaitons étudier, du moment que ces deux grandeurs (NOx/suies) font partie des objectifs à optimiser.

5.5 Modélisation de la combustion

La simulation numérique prend une place de plus en plus importante dans la conception des moteurs de demain grâce au développement des moyens de calcul informatique, et à la mise au point des méthodologies avancées de simulation numérique de la combustion. Plusieurs types de modélisation ont été proposés dans la littérature. Ces modèles diffèrent dans le degré de prise en compte des différents phénomènes physiques qui ont lieu dans la chambre. Dans ce sens, deux familles de modélisation peuvent être distinguées : la modélisation phénoménologique 0D et la modélisation multidimensionnelle 3D.
La modélisation phénoménologique ou la modélisation 0D [Albrecht 2006, Lafossas 2007] est une modélisation qui ne prend pas en compte les dimensions spatiales dans la chambre, ce qui lui permet d'avoir un temps de retour "raisonable" par rapport à la modélisation 3D.
La modélisation numérique multidimensionnelle, appelée aussi modélisation 3D a été largement utilisée dans le domaine de la physique de la combustion. Trois grands

types de modèles peuvent être distingués : la simulation numérique directe, la simulation aux grandes échelles et la modélisation statistique. La simulation numérique directe (DNS pour *Direct Numerical Simulation*) consiste à résoudre numériquement toutes les échelles de la turbulence en résolvant numériquement les équations de Navier-Stockes [1]. Autrement dit, dans cette approche aucune modélisation n'est pratiquée. La simulation aux grandes échelles (LES pour *Large Eddy Simulation*) comme son nom l'indique consiste à résoudre les échelles les plus grandes de la turbulence tandis que les échelles les plus petites sont modélisées. La séparation entre les petites et les grandes échelles se fait via un filtre spatial. Dans le troisième type qui est la modélisation statistique (RANS pour *Reynolds-Avereged Navier-Stockes*), toutes les échelles de la turbulence sont modélisées.
Les trois approches citées ci-dessus ne requièrent pas les mêmes exigences en termes de coût CPU des simulations, du fait des différents niveaux d'information obtenus pour chaque méthode. Dans ce contexte, les approches DNS et LES s'avèrent très couteuses. D'une part, la modélisation DNS reste encore irréalisable dans le cas du moteur, son application se résume à des cas académiques très réduits. D'autre part, la simulation LES reste très couteuse en termes de coût CPU, et son application est réduite à des applications unitaires (un calcul LES sur un cycle moteur complet dure aux alentours de 1 mois) [Thobois 2006, Enaux 2010].
La modélisation RANS quant à elle reste à ce jour l'approche la plus utilisée dans le monde de la modélisation multidimensionnelle car elle offre un bon compromis entre précision recherchée et coût CPU.
Un autre aspect important à prendre en compte dans la modélisation de la combustion est le point de fonctionnement du moteur étudié. Nous distinguons dans ce sens le régime à "pleine charge" et les régimes à "charge partielle".
Tenant compte de la complexité des différents phénomènes qui ont lieu dans la chambre de combustion, et de l'antagonisme des objectifs à optimiser, nous nous focalisons dans les deux prochains chapitres sur l'application de l'approche stationnaire asynchrone présentée dans la première partie de cette thèse sur deux types de modélisation : la modélisation phénoménologique 0D (chapitre 6) et la modélisation multidimensionnelle 3D (chapitre 7).

1. Les équations de Navier-Stockes sont des équations aux dérivées partielles non linéaires qui décrivent le mouvement des fluides dans un mileu continu.

Chapitre 6

Modélisation phénoménologique (0D)

Sommaire

6.1 Introduction **77**
6.2 Présentation de l'outil PDF0D **78**
6.2.1 Equilibre thermodynamique 79
6.2.2 Mélange des particules 79
6.2.3 Pertes Thermiques aux Parois 80
6.2.4 Injection du carburant 80
6.2.5 Combustion 81
6.3 Définition du problème d'optimisation **81**
6.3.1 Paramètres de l'optimisation 81
6.3.2 Objectifs de l'optimisation 83
6.3.3 Contrainte de l'optimisation 83
6.4 Réglages **84**
6.5 Résultats **85**
6.5.1 Variabilité du temps de calcul 86
6.5.2 Coût global de l'optimisation 87
6.5.3 Comparaison des fronts de Pareto 88
6.5.4 Analyse physique 88
6.6 Conclusion **90**

Un partie des travaux présentés dans chapitre a été publiée dans [Yagoubi 2010]

6.1 Introduction

La modélisation phénoménologique ou la modélisation 0D a été largement utilisée dans le domaine de la simulation numérique de la physique de combustion [Albrecht 2006, Lafossas 2007]. Bien que ce type de modélisation ne permet pas de prendre en compte certains aspects de la modélisation multidimensionnelle comme la géométrie de la chambre par exemple, la modélisation 0D offre tout de même un bon compromis entre coût de calcul et qualité des résultats. Dans la présente étude, la modélisation 0D sert d'étape intermédiaire entre les différents benchmarks de fonctions analytiques testées, et la modélisation multidimensionnelle 3D. Le temps

de retour étant beaucoup plus réduit comparativement à la modélisation multidimensionnelle (3D), cette étape nous permet de tester les algorithmes sur une problématique qui met en jeu les phénomènes de la physique de la combustion.
Par ailleurs, cette étape a aussi pour objectif de démontrer la capacité des modèles phénoménologiques à bien prédire les phénomènes physiques dans la chambre de combustion. Dans le cadre de cette étude, nous utilisons un modèle phénoménologique qui permet de prendre en compte l'hétérogénéité dans la chambre via la stratification. Ce modèle nommé PDF0D est un outil interne qui a été développé au sein de PSA Peugeot Citroen.
Dans la littérature, il existe quelques travaux qui se sont intéressés au couplage entre les algorithmes évolutionnaires et les modèles 0D. Hiroyasu et al. présentent dans [Hiroyasu 2002] une étude de l'optimisation multi-objective appliquée sur un modèle de combustion 0D. Les objectifs à optimiser sont : les NOx , les suies et la consommation du carburant. Cependant, la minimisation de ces trois objectifs a été réalisée en variant seulement une variable de décision qui est le profil (la forme) de l'injection. Hiroyasu et al, proposent dans [Hiroyasu 2003] une extension des travaux précédents, en variant d'autres paramètres de décisions tels que : l'EGR [1] et les débuts des injections. Les paramètres retenus dans cette optimisation ne couvrent pas la totalité des leviers qui peuvent être utilisés avec la modélisation phénoménologique (0D) comme par exemple les paramètres de mélange.
Ce chapitre sera organisé comme suit : après une présentation succincte de l'outil PDF0D dans la section 6.2, nous définissons les différentes composantes du problème d'optimisation dans la section 6.3. Après avoir décrit les réglages effectués pour réaliser l'optimisation (section 6.4), nous présentons les résultats obtenus dans la section 6.5 et nous finissons ce chapitre par une conclusion.

6.2 Présentation de l'outil PDF0D

PDF0D est un outil qui a été développé au sein de PSA Peugeot Citroen pour la modélisation de la combustion Diesel. Il permet de calculer l'évolution thermochimique d'un mélange de gaz au cours d'un cycle moteur complet, en volume fermé. Cette évolution est basée sur un schéma cinétique choisi selon les conditions thermodynamiques comportant les éléments utilisés, les espèces chimiques, les réactions ainsi que les lois d'Arrhenius [2] leur correspondant. Le mélange de gaz est décrit par un ensemble de particules possédant chacune une composition et une température spécifique. PDF0D permet de calculer, à chaque pas de temps, l'évolution de la température et de la composition de toutes les particules en prenant en compte les

1. Recirculation des gaz d'échappement (en englais : Exhaust Gas Recirculation) est un système inventé au début des années 1970 qui consiste à rediriger une partie des gaz d'échappement dans le collecteur d'admission. Le but d'introduire ce système est de réduire les émissions polluantes dans la chambre de combustion [Heywood 1988]

2. La loi d'Arrhenius permet de décrire la variation et de la vitesse d'une réaction chimique. Cette loi a été énoncée par S .A. Arrhenius en 1889 et a pu être vérifée expérimentalement pour un grand nombre de réactions chimiques [Law 2006]

phases de compression et de détente, les injections au cours du cycle, le mélange entre les particules, la combustion et les pertes thermiques aux parois. Le mélange initial est imposé par une PDF (Fonction Densité de Probabilité) jointe de type gaussienne ou fonction β. Cette modélisation permet donc d'imposer une stratification de la richesse dans la chambre de combustion en début de simulation. L'injection est discrétisée en plusieurs sous-injections. A chaque sous-injection, une particule ne contenant que du fuel ayant une masse définie prend place dans la chambre de combustion.
Nous présentons dans ce qui suit, de façon succincte, les grandes étapes du calcul avec PDF0D allant de phase initiale (composition initiale) jusqu'à la phase finale (formation des polluants).

6.2.1 Equilibre thermodynamique

A chaque pas de temps, un bilan thermodynamique est établi pour chaque particule afin de réaliser le couplage thermodynamique des particules dû au confinement. Ce bilan est réalisé en utilisant la loi des gaz parfaits (équation 6.1) et l'équation de l'énergie interne d'une particule (équation 6.2). A partir de ces deux équations, les évolutions des grandeurs thermodynamiques telles que : la température, le volume sont calculées pour chaque particule. Les valeurs moyennes de ces grandeurs dans la chambre de combustion sont ensuite déduites.

$$PV = nRT \tag{6.1}$$

$$e = h - P/\rho = \int_{T_0}^{T} C_v dT - \frac{RT_0}{W} + \sum_{k=1}^{n} \Delta h_{f,k} Y_k \tag{6.2}$$

Où P est la pression, V est le volume, T est la température, R est la constance de Reynolds et n le nombre de moles et ρ est la densité.

6.2.2 Mélange des particules

Le mélange des particules représente une étape très importante dans la modélisation 0D. Il permet de prendre en compte l'aérodynamique de la chambre de combustion en simulant l'effet de la turbulence. En d'autres termes, le mélange des particules permet de prendre en compte un phénomène très important de la combustion Diesel qui est la turbulence, qui n'est pas modélisé en 0D en l'absence de notion d'espace (pas de coordonnées spatiales). Le modèle utilisé dans PDF0D pour mélanger les particules est celui de Curl [Janicka 1977]. Dans ce modèle, l'obtention du mélange turbulent passe par l'échange de masse entre un certain nombre de paires d'éléments sélectionnées au hasard parmi l'ensemble des particules. Si l'on note les deux éléments d'une paire par p et q et leurs compositions avant le mélange (instant t) par ϕ^p et ϕ^q respectivement, à l'issue du mélange (instant $t + dt$) on obtient :

$$\phi^p_{t+dt} = \phi^p_t + \frac{1}{2}\beta(\phi^q_t - \phi^p_t) \tag{6.3}$$

$$\phi^q_{t+dt} = \phi^q_t + \frac{1}{2}\beta(\phi^p_t - \phi^q_t) \tag{6.4}$$

où β est une valeur aléatoire uniformément distribuée entre 0 et 1.
Un autre modèle de mélange [Subramaniam 2004] propose de sélectionner les particules en prenant en compte leur "âge" qui correspond au temps écoulé depuis leur dernière interaction afin d'éviter que les particules plus jeunes ne soient mélangées une nouvelle fois. la procédure de mélange est faite selon les étapes suivantes : après une sélection (selon l'age) des particules devant être mélangées, ces dernières sont reliées entre elles de façon à ce que chaque particules soit mélangée avec une particule voisine dans l'espace des compositions.
Les tests réalisés au cours de cette étude révèlent qu'il n'existe pas de différence significative entre les deux modèles. Finalement, c'est le modèle de Curl qui sera retenu pour la problématique d'optimisation.

6.2.3 Pertes Thermiques aux Parois

PDF0D prend en compte le processus des transferts thermiques d'un mélange de gaz aux parois. Ces transferts sont modélisés dans les moteurs par convection en utilisant les corrélations de Woschni[Woschni 1998]. Le taux de transfert de chaleur d'un mélange de gaz à la paroi s'écrit comme suit :

$$q = h_c(T_{gaz} - T_{paroi}) \tag{6.5}$$

Où T_{gaz} est la température moyenne du mélange gazeux, T_{paroi} est la température de la paroi, et h_c est le coefficient du transfert de chaleur. A noter que le calcul de h_c prend en compte plusieurs grandeurs tels que : l'alésage, la vitesse, la pression ...etc.

6.2.4 Injection du carburant

Le module d'injection intervient à un moment précis du cycle de combustion défini préalablement par l'utilisateur. L'injection est modélisée par une forme imposée de la PDF de richesse dans la chambre de combustion au moment de l'injection. Cette PDF est imposée soit par modification de la composition des particules existantes (plusieurs formes sont possibles : gaussienne, β-PDF), soit par création de nouvelles particules de carburant pur qui vont être mélangées avec les particules déjà existantes dans la chambre.

6.2.5 Combustion

La combustion est modélisée à l'aide des schémas cinétiques de la chimie complexe. Ces schémas décrivent en détail l'oxydation du "n-heptane" (espèce chimique qui modélise le fuel) avec l'oxygène. Toutes les espèces des hydrocarbones imbrulés (HC) sont représentées, ainsi que les oxydes de carbone (CO et CO_2) et le monoxyde d'azote (NO). Il existe plusieurs schémas cinétiques dans la littérature, nous utilisons dans cette étude le schéma de Hewson [Bolling 1996]. A chaque pas de temps, les taux de réaction massiques sont calculés pour chaque espèce et pour toutes les particules à l'aide de la librairie Chemkin-II [Lutz 1988]. Pour une espèce k, le taux de réaction massique w_k est la somme des différents taux w_{kj} produits par toutes les M réactions.

6.3 Définition du problème d'optimisation

La qualité des résultats d'une optimisation ne dépend pas seulement de l'algorithme qui pilote ce processus. Au delà du choix judicieux de l'algorithme d'optimisation qui doit être fait, la problématique d'optimisation doit être bien définie de telle sorte que l'espace de recherche (espace de décision) soit efficacement parcouru. Ceci passe par un choix intelligent des paramètres à faire varier en évaluant leurs impacts respectifs sur les objectifs à optimiser. Les contraintes du problème doivent être aussi bien choisies afin de bien délimiter le cadre de l'étude.
En ce qui concerne l'optimisation avec PDF0D, cette étape consiste à définir les leviers qu'on va faire varier pendant l'optimisation (paramètres d'optimisation), les objectifs à optimiser et les contraintes qui doivent être respectées.

6.3.1 Paramètres de l'optimisation

Nous présentons dans ce qui suit les différents paramètres qui ont été choisis pour balayer l'espace de recherche. Ces paramètres, qui sont au nombre de dix, peuvent être classés en trois catégories : les paramètres d'injection, les paramètres de boucle d'air et les paramètres de mélange.

6.3.1.1 Paramètres d'injection

Nous retrouvons dans cette catégorie les paramètres liés au phénomène d'injection du carburant dans la chambre de combustion. Dans le cadre de notre étude, nous nous limitons à une injection principale (prénommée "injection main" dans le jargon des motoristes). Cette injection débute à un instant donné fixé au préalable par l'utilisateur exprimé en degré vilebrequin (degré d'avancement du cycle de combustion).
Trois paramètres d'injections seront retenus dans cette étude. **Le Début d'injection** (ou SOI pour "Start Of Injection") correspond au moment où débute l'injection du fuel dans la chambre. **La perméabilité de la buse d'injection** : ce paramètre

va avoir un impact sur le débit du carburant injectée dans la chambre. **La pression RAIL** ou la pression de l'injecteur : ce paramètre modélise le degré de pénétration de l'injection dans la chambre.

6.3.1.2 Paramètres de boucle d'air

Dans un moteur Diesel, la boucle d'air est le système qui permet d'alimenter en air la chambre de combustion. Dans cette catégorie, trois paramètres aussi sont retenus : **Le taux D'EGR** (Recirculation des gaz d'échappement), le principe de l'EGR consiste à recirculer une partie des gaz échappés dans la chambre de combustion via la ligne d'admission. Le but de l'utilisation de l'EGR réside dans sa contribution à la diminution de la production des oxydes d'azote (NOx). Les deux autres paramètres de boucle d'air sont : **la pression et la température d'admission** (pression et température en début de cycle).

6.3.1.3 Paramètres de mélange

Jusque là, les paramètres déjà retenus (de mélange et de boucle d'air) sont des paramètres qui représentent des grandeurs quasiment tout le temps dans la modélisation de la chambre de combustion. Il s'agit des paramètres fonctionnels du moteur. Les paramètres de mélange quant à eux, sont des paramètres qui ont été définis au cours de cette étude afin d'étudier l'impact que peut avoir la modélisation du mélange en 0D sur la formation des polluants.
En utilisant le modèle de Curl [Janicka 1977], une constante de temps de mélange doit être définie à chaque pas de temps en utilisant le temps turbulent dans la chambre. Cette constante est décomposée durant le cycle de combustion en trois paramètres afin de prendre en compte trois différentes sources de turbulences : la fréquence du mélange dans la chambre avant l'injection, ce paramètre modélise la turbulence dans la chambre avant l'injection qui est due essentiellement à la forme géométrique de la chambre **"ENGINE-FREQ"**. Le deuxième paramètre est la fréquence de mélange due à l'injection du carburant **"FACT-FREQ"** pour modéliser la turbulence créée par la pénétration du spray d'injection dans la chambre. Le troisième paramètre permet de piloter l'hétérogénéité de la fréquence du mélange due à la différence entre les zones à turbulence élevée (proches du spray d'injection) et le reste de la chambre **"FACT-INJ"**. Le quatrième paramètre de mélange défini est le **"RATE-MAX"** qui représente le taux maximum de masse qui peut être échangé entre deux particules lors du mélange.

Le tableau 7.1 résume tous les paramètres d'optimisation décrits ci-dessus, ainsi que leurs unités et bornes supérieures et inférieures. Les bornes ont été définies dans l'optique d'élargir au maximum l'espace de recherche sans pour autant tomber dans des zones d'infaisabilité. A noter que pour le paramètre "FACT-FREQ", la borne inférieure correspond à la valeur du paramètre "ENGINE-FREQ", partant du principe que la turbulence de la chambre ne peut qu'augmenter après l'injection.

Table 6.1 – Tableau récapitulatif des paramètres de l'optimisation 0D

Paramètres	Unités	Description	Borne inférieure	Borne supérieure
EGR	%	Taux de recirculation des gaz brulés rate	0	60
P2	Bar	Pression d'admission	1.6	3
T2	K	Température d'admission	380	480
SOI	Degré vilebrequin	Début d'injection	300	380
ENGINE-FREQ	Hz	Fréquence du mélange avant l'injection	500	50000
FACT-FREQ	Hz	Fréquence du mélange due à l'injection	ENGINE-FREQ	150000
FACT-INJ	/	Ratio de fréquence de mélange entre la zone du spray et le reste de la chambre	0	6
PERMEA	$cm^3/30$s	Flux thermodynamique de l'injection	300	600
PRAIL	Bar	Pression de l'injecteur	500	3000
RATE-MAX	%	Taux maximum de masse pouvant être échangé entre deux particules	10^{-3}	10^{-1}

6.3.2 Objectifs de l'optimisation

Les objectifs de l'optimisation sont la diminution des émissions polluantes et la consommation du carburant. Nous retrouvons alors trois objectifs à optimiser (à minimiser) qui sont : les oxydes d'azote (NOx), les suies (qui sont modélisées avec une variable nommé "HC-CO" combinant le monoxyde de carbone (CO) et imbrulés (HC) et enfin la consommation du carburant qui est décrite comme étant le rapport entre la masse injectée et la puissance moyenne délivrée par le cycle de combustion.

6.3.3 Contrainte de l'optimisation

Afin que les différents systèmes de combustion soient comparables en termes d'émissions polluantes et de consommation du carburant, ces derniers doivent délivrer la même puissance moyenne par cycle. A défaut, les meilleurs systèmes de pollution (en termes de consommation et de NOX surtout), seront ceux dont la configuration ne permet pas d'avoir un dégagement de chaleur suffisant (ou une com-

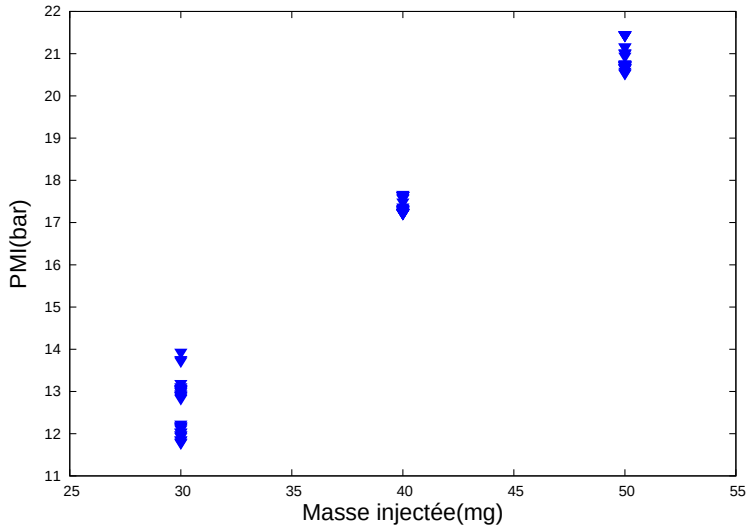


FIGURE 6.1 – Variation de la PMI en fonction de la Masse de carburant injectée sur plusieurs points de calcul

bustion). Autrement dit, la meilleure option pour minimiser les émissions polluantes serait de ne pas utiliser le moteur carrément. Dans ce contexte, une contrainte forte s'impose afin de garantir le respect de cette règle métier. La grandeur utilisée pour contrôler la puissance moyenne par cycle est la "PMI" (Pression Moyenne Indiquée). Chaque individu évalué doit avoir une valeur égale à la valeur de la PMI cible qui est celle du cas de référence à optimiser, avec une certaine tolérance fixée à 2 %. Pour cela, à chaque évaluation d'un individu, nous utilisons un algorithme de gradient pour faire converger ce dernier à la PMI cible. Nous choisissons comme paramètre de l'algorithme de gradient la masse de carburant injectée étant donné que c'est le paramètre qui a le plus d'influence sur la puissance délivrée par cycle, et que la variation de ce paramètre par rapport à la PMI est supposée linéaire. Afin de valider ce comportement linéaire, nous avons réalisé un plan d'expérience sur trois différentes valeurs de masse du carburant en balayant les autres paramètres et en observant la réponse obtenue sur la valeur de la PMI. la figure 6.1 illustre le comportement linéaire entre la PMI et la masse de carburant injectée.

6.4 Réglages

Nous décrivons dans cette section le couplage réalisé entre le modèle PDF0D et deux variantes de l'algorithme NSGA-II : La version standard de l'état de l'art [Deb 2002] et la version stationnaire asynchrone AS-NSGA-II présentée dans le chapitre 3. Une application directe du paradigme "maître-esclave" du calcul parallèle sur NSGA-II revient à évaluer les enfants générés à chaque génération en parallèle. Si la taille de la population est égale à N, le même nombre de processeurs

est requis sur la grille de calcul. L'outil PDF0D communique avec l'algorithme NSGA-II via un système de fichier. Initialement, le processus "maître" exécute le code de l'algorithme. Pour chaque individu, l'algorithme génère un fichier "input" et évalue la fonction objectif sur un processeur de la grille de calcul. Une fois l'évaluation achevée, PDF0D renvoie un fichier "output" avec les valeurs de la fonction objectif.
Pour la configuration de PDF0D, nous choisissons un point de calcul à charge partielle avec un régime de 2400 tour/min et une valeur de PMI cible estimée à 13.66 Bar. Pour l'algorithme du gradient, nous fixons à quatre le nombre maximum de sous-itérations de descente que pourra effectuer un individu afin de converger vers la valeur de la PMI cible. Si toutefois après les quatre sous-itérations, l'individu ne réussit toujours pas à converger, il sera pénalisé. Sachant qu'il s'agit d'un problème de minimisation, la pénalisation est réalisée en rajoutant une valeur de pénalité à tous les objectifs. Cette valeur rajoutée dépend de la distance qui sépare la meilleure PMI obtenue de la PMI cible : plus la différence entre la PMI de l'individu pénalisé et la PMI cible est grande, plus la valeur de pénalité est grande.
Dans le cas de l'algorithme standard générationnel, le temps nécessaire à l'évaluation de la population est celui de l'individu le plus lent de la génération. Compte tenu de l'hétérogénéité des durées d'évaluations, et afin de rendre le coût global raisonnable en termes de coût CPU, toutes les évaluations sont stoppées après un seuil fixé à 8 heures, en se basant sur une décision humaine : comme la simulation est un processus itératif, les individus qui semblent non prometteurs après 8 heures de calcul sont immédiatement stoppés, alors que les individus qui semblent être en bonne voix seront autorisés à continuer l'évaluation.
Dans le cas de l'algorithme stationnaire asynchrone, aucun individu n'est stoppé lors de l'évaluation. Comme l'utilisation de la grille de calcul est supposée ininterrompue (aucun processeur ne reste inoccupé), la contrainte sur le seuil de durée maximale d'évaluation est relaxée.
Dans cette étude, nous comparons ces deux algorithmes sur un problème réel de combustion en utilisant la modélisation 0D en réalisant deux optimisations distinctes : la première utilise une forme standard parallélisée de NSGA-II avec un modèle générationnel, tandis que la seconde utilise la version stationnaire asynchrone AS-NSGA-II présentée dans le chapitre 3.

Pour les deux approches, nous utilisons une taille de population égale à 40 (40 est le nombre des processeurs disponibles sur notre grille de calcul). Les deux algorithmes tournent pour un budget fixé à 1600 évaluations (c'est à dire, 40 générations dans le cas de l'algorithme générationnel).

6.5 Résultats

Nous présentons dans cette section les résultats des optimisations. Dans un premier temps, nous étudions l'hétérogénéité du temps de calcul et nous comparons le

coût global des optimisations. Nous nous focaliserons dans un second temps sur la qualité des résultats en comparant les fronts de Pareto respectifs. Enfin, nous analyserons les résultats du point de vue physique en discutant l'évolution des paramètres au cours de l'optimisation.

6.5.1 Variabilité du temps de calcul

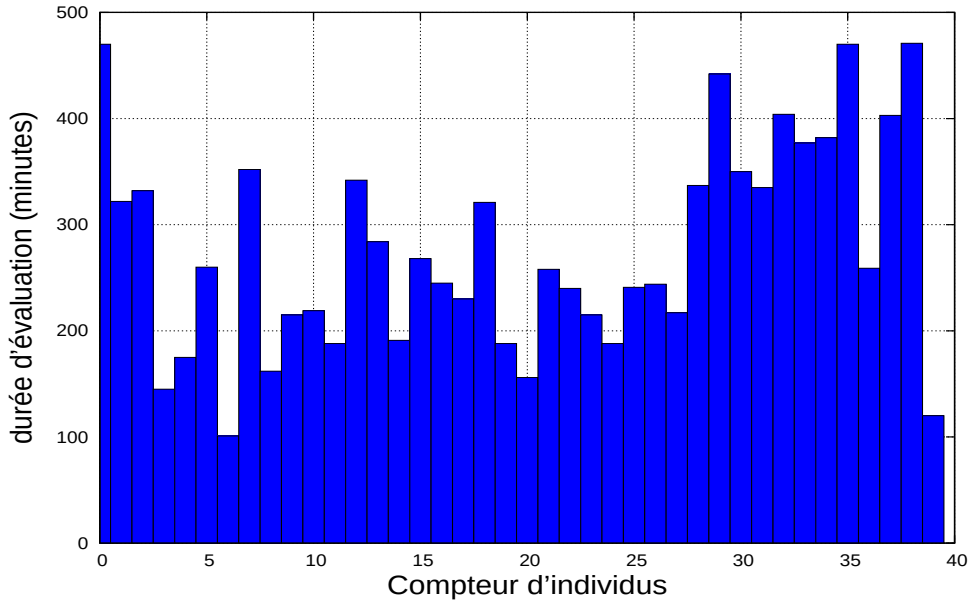


FIGURE 6.2 – Durées des évaluations des individus de la première génération aléatoire (en minutes)

La figure 6.2 décrit les différents temps d'évaluation des individus de la première génération aléatoire. Le graphe montre bien une hétérogénéité dans le temps de calcul entre les individus. Nous expliquons ce phénomène par le fait que certains paramètres de l'optimisation ont un impact direct sur le temps de restitution de l'évaluation. D'une part, une valeur très grande de la fréquence du mélange après l'injection (FACT-FREQ) ralentit significativement l'exécution de chaque pas de temps, et par conséquent augmente la durée de l'évaluation. D'autre part, certaines configurations d'individus ne permettent pas d'avoir une combustion (conditions thermodynamiques en termes de pression et de température non atteintes), nous citons à titre d'exemple le cas des individus avec des injections très tardives. De ce fait, la durée d'évaluation de ces individus est relativement courte.
Une autre source d'hétérogénéité provient de la procédure de convergence locale avec l'algorithme de gradient. Certains individus réussissent à satisfaire la contrainte dès la première sous-itération, d'autres auront besoin de deux, trois, voire utiliser le nombre maximum de sous-itérations qui est fixé à quatre. Sachant qu'à chaque sous-itération le calcul est entièrement relancé, la différence de rapidité de convergence

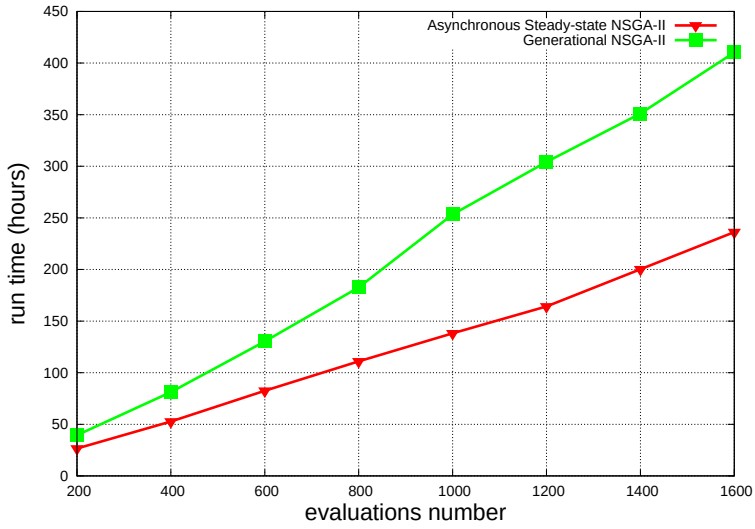


FIGURE 6.3 – Comparaison des durées d'optimisation à différents nombres d'évaluations

vers la valeur cible de la contrainte provoque de ce fait une hétérogénéité significative en termes de durée d'évaluation.

6.5.2 Coût global de l'optimisation

La figure 6.3 illustre l'évolution de la durée globale de l'optimisation en fonction du nombre d'évaluations pour les deux approches standard et stationnaire asynchrone de l'algorithme NSGA-II. A noter que le temps d'écriture et de lecture des différents fichiers de données est négligé (quelques milli-secondes), en comparaison avec celui de l'évaluation avec PDF0D (quelque heures).
Le coût global de l'optimisation observé avec l'algorithme NSGA-II standard est de l'ordre de 17 jours, alors que celui observé avec l'algorithme stationnaire asynchrone est inférieur à 10 jours, soit un gain de 42 %. Le premier résultat (attendu) est que l'algorithme asynchrone stationnaire permet d'atteindre le budget fixé en termes de nombre d'évaluations plus rapidement que l'algorithme standard. Autrement dit, l'algorithme stationnaire asynchrone permet de réaliser un nombre plus important d'évaluations comparativement à l'algorithme standard pour la même durée de calcul.
Ce résultat est attendu car dans le cas de l'algorithme standard générationnel, le temps nécessaire à l'évaluation de la population est le cumul des temps d'évaluations des individus les plus lents de chaque génération. Sachant que l'hétérogénéité de temps de calcul varie entre 1.5 et 24 heures, l'algorithme générationnel se voit distancé par l'algorithme stationnaire asynchrone au fur et à mesure que l'optimisation avance.

6.5.3 Comparaison des fronts de Pareto

Les figures 6.4, 6.5 et 6.6 représentent la projection du front de Pareto dans les trois plans : suies-NOx, conso-suies et conso-NOx respectivement. Sur chaque plan les individus issus des deux optimisations (l'une avec l'algorithme générationnel, l'autre avec l'algorithme asynchrone) sont représentés. Nous pouvons constater que les deux optimisations sont comparables en termes de qualité de convergence. Cependant, quelques points qui sont très intéressants en termes des valeurs de NOx sont trouvés dans le cas de l'optimisation avec l'algorithme stationnaire asynchrone et pas dans le cas de l'optimisation avec l'algorithme générationnel. Ces points correspondent à des individus avec une durée d'évaluation très élevée. Comme expliqué précédemment, les individus avec un coût d'évaluation très élevé sont stoppés dans le cas de l'optimisation avec l'algorithme générationnel. Ces individus sont caractérisés par une valeur très grande de la fréquence de mélange après l'injection (FACT-FREQ).

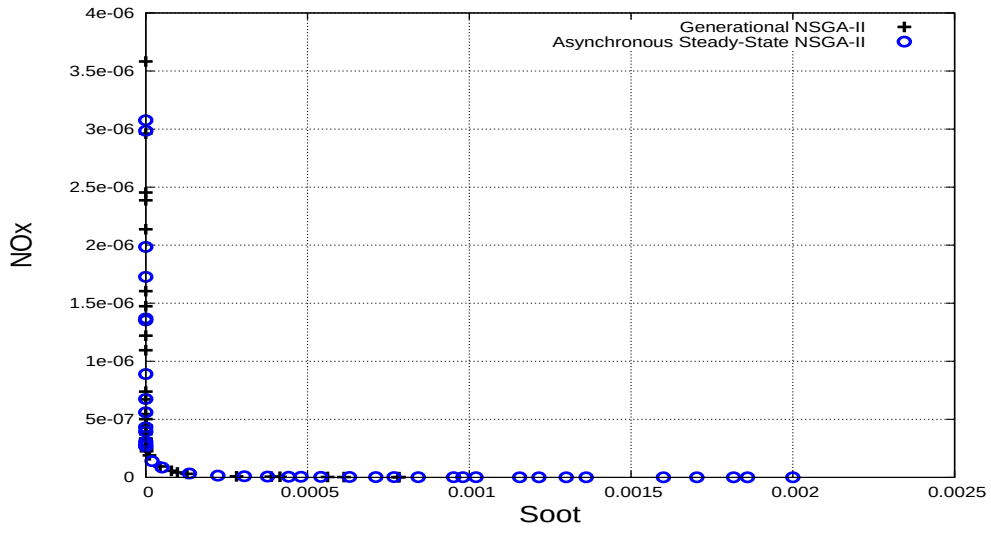


FIGURE 6.4 – Plan suies-NOx du front de Pareto

6.5.4 Analyse physique

Dans cette partie, nous analysons les résultats de l'optimisation du point de vue physique. Nous nous focaliserons sur l'analyse de l'évolution de certains paramètres (SOI, P2, T2, EGR) durant le processus d'optimisation avec l'algorithme NSGA-II stationnaire asynchrone. Les graphes B.8(g) et B.8(h) montrent que la pression d'admission (P2) tend au fur et à mesure que l'optimisation avance vers sa borne supérieure, tandis que la température d'admission tend plutôt vers sa borne inférieure. Nous expliquons ce phénomène par le fait que plus la pression d'admission est élevée, plus le travail mécanique est récupéré en dehors de la chambre de combustion, et par conséquent la consommation du carburant aura tendance à baisser. D'autre part,

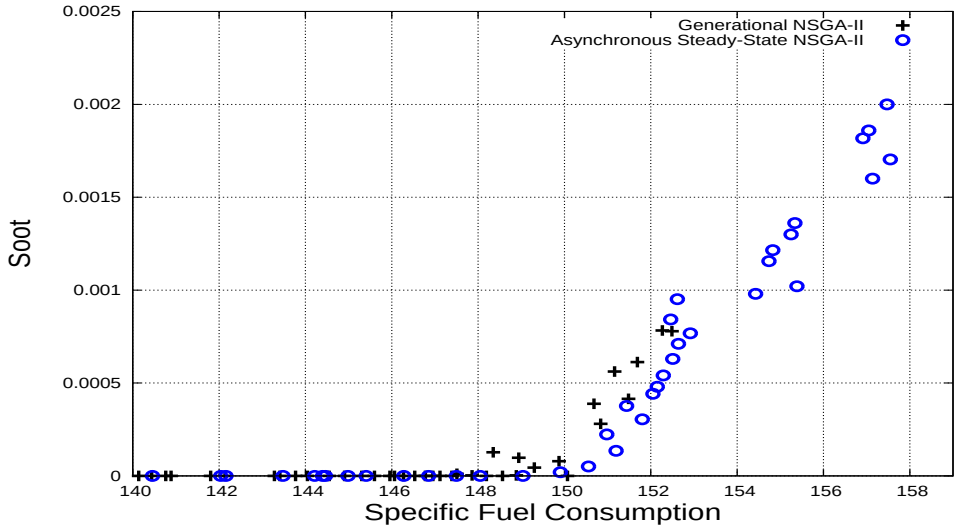


FIGURE 6.5 – Plan conso-suies du front de Pareto

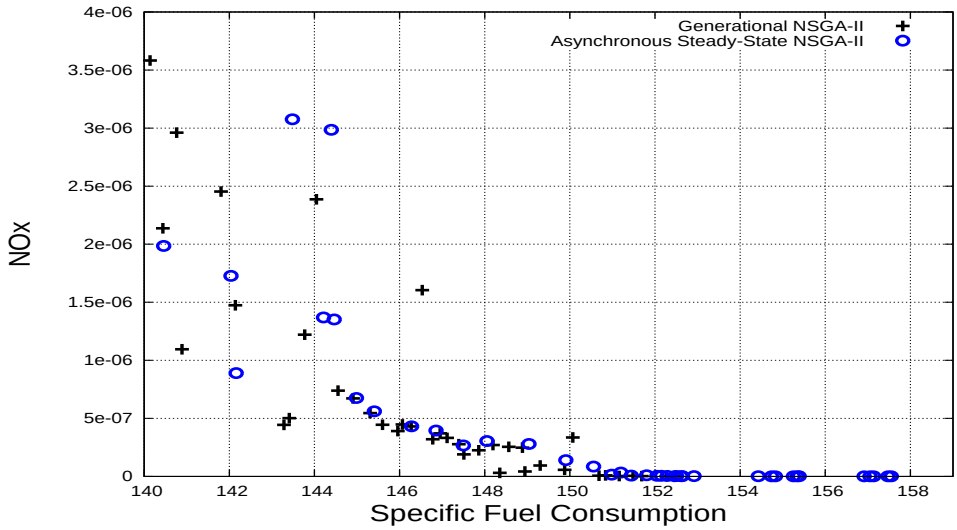


FIGURE 6.6 – Plan conso-NOx du front de Pareto

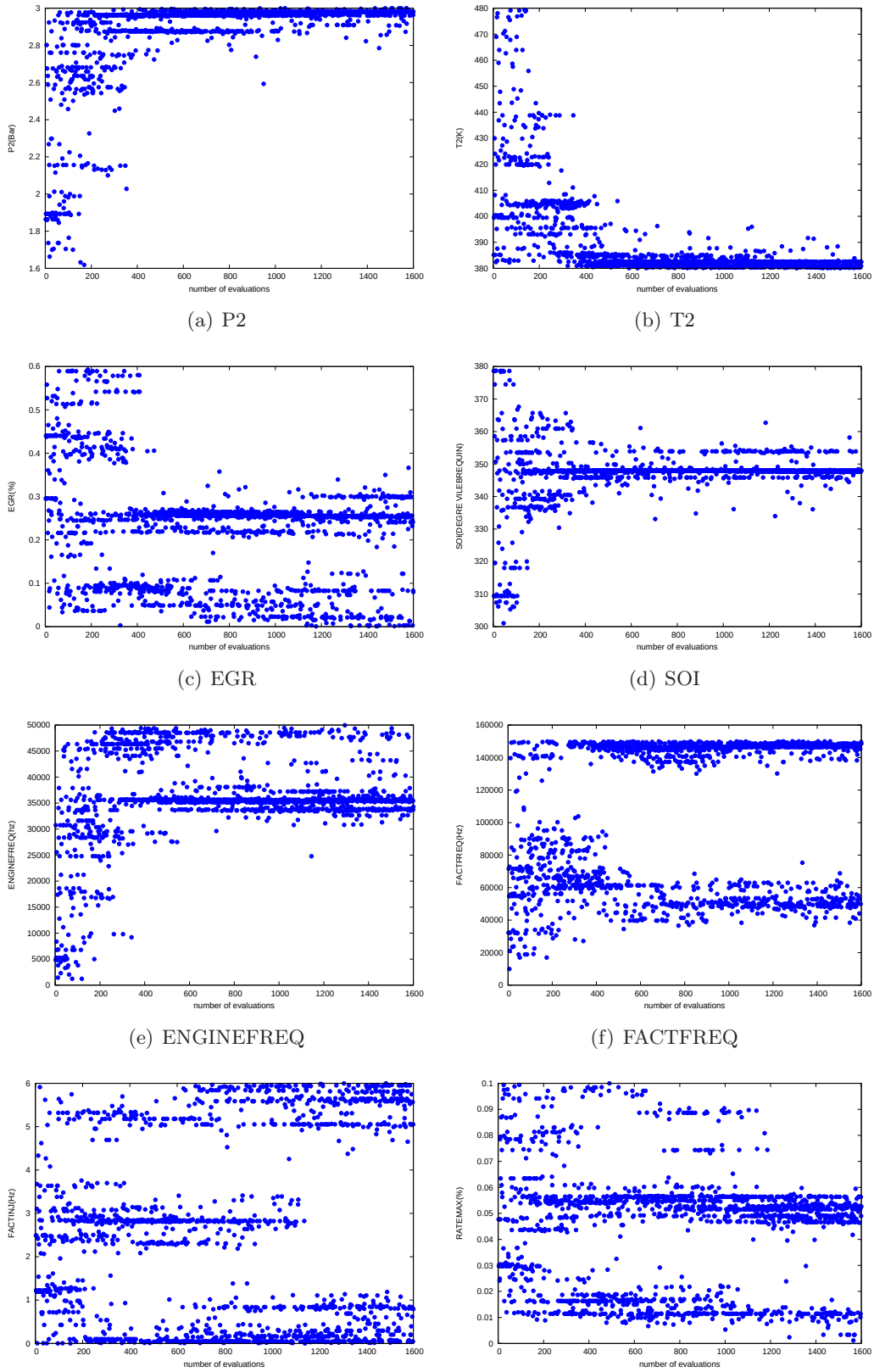
une température d'admission faible permet de diminuer la température maximale atteinte durant le cycle de combustion, ce qui permet de réduire l'émission des NOx. Le graphe B.4(c) qui illustre l'évolution de l'EGR permet de distinguer deux zones qui se forment au fur et à mesure de l'avancement de l'optimisation : une première zone avec des valeurs élevées de l'EGR (0.2-0.3), et une deuxième zone avec des valeurs faibles de l'EGR (0-0.1). Il s'agit ici d'un phénomène dans la physique de la combustion qui est celui de l'impact de l'EGR sur la formation des polluants. La combustion sans EGR atteint des températures très élevées comparativement à

celles en présence de l'EGR : ce dernier joue un rôle de buffer thermique et ralentit le processus de combustion dans le but de diminuer la formation NOx, ce qui conduit à l'augmentation de la formation des suies. A l'opposé, l'absence de l'EGR dans la chambre favorise l'oxydation rapide des suies et permet donc de diminuer l'émission des suies en fin de cycle, par contre cette configuration est favorable à la formation des NOx car les températures atteintes sont très élevées ce qui est favorable à la formation du monoxyde d'azote.
En ce qui concerne le début d'injection B.4(d), l'algorithme préfère des configurations avec des injections avant le Point Mort Haut (360 degrés vilebrequin) qui correspond au moment de la fin de la phase de compression du cycle de combustion. Les injections tardives (injections après 365 degré vilebrequin) sont vite abandonnées par l'algorithme car elle correspondent à des configurations d'individus qui ne respectent pas la contrainte d'égalité en PMI : une injection tardive se passe au cours de la phase de détente dans laquelle les conditions thermodynamiques en termes de pression et de température ne sont pas réunies pour avoir un dégagement de chaleur, et par conséquent la PMI délivrée est trop faible.
Les figures B.4(e), B.4(f), B.4(g) et B.4(h) retracent l'évolution des paramètres ENGINE-FREQ, FACT-FREQ, FACT-INJ et RATE-MAX respectivement. En analysant ces figures, nous distinguons deux grandes familles d'individus : la première famille regroupe des individus avec des fréquences de mélange après injection très élevées (proches de la borne supérieure), des valeurs très faibles de FACT-INJ et des taux d'échange de masse plutôt moyens. Ces individus qui tendent vers un mélange presque parfaitement homogène sont présents dans le front du Pareto grâce à leur très faibles émissions de NOx. Ce cas de figure se rapproche d'une configuration bien connue de la combustion Diesel qui est celle de la combustion "HCCI" ou la combustion homogène [Zhao 2003]. Néanmoins, ce type de combustion est appliqué seulement aux cas faiblement chargés. Dans notre cas, qui est relativement fortement chargé, l'application de ce type de combustion est difficilement réalisable (bruit très élevé) car la combustion HCCI suppose une injection très précoce du carburant et une homogénéisation du mélange air-carburant-EGR atteinte avant la compression. Inversement, la deuxième famille qui se dégage regroupe des individus avec des fréquences de mélange après injection plutôt faible avec des valeurs de FACT-INJ très élevées et des taux d'échange de masse très faibles. Cette configuration correspond à une chambre très hétérogène ce qui permet d'atteindre des températures plus élevées et par conséquent des valeurs en NOx très élevées. Par contre, cette configuration permet une meilleure catalyse du HC et du CO, et une réduction des émissions des suies ce qui explique la présence de ce genre d'individus dans le front du Pareto.

6.6 Conclusion

Dans ce chapitre, nous avons présenté la première application réelle réalisée en utilisant un outil de modélisation phénoménologique (PDF0D) de la combustion Diesel. Après avoir présenté rapidement les principales étapes de la modélisation en

utilisant PDF0D, nous avons décrit les différentes composantes du problème d'optimisation à savoir les paramètres, les objectifs et les contraintes. Par la suite, nous avons comparé l'algorithme stationnaire asynchrone AS-NSGA-II présenté dans le chapitre 3 avec la version standard générationelle de NSGA-II. Les résultats montre qu'un gain de 42% a été réalisé en utilisant l'algorithme AS-NSGA-II grâce à son plus efficace utilisation des processeurs sur la grille de calcul. Du point de vue physique de la combustion, les résultats de l'optimisation ont permis de dégager certains types de combustion définis par des familles d'individus.
D'un point de vue général, la modélisation 0D nous a permis de valider l'approche que nous proposons pour faire face à l'hétérogénéité du temps de calcul en la comparant avec la version standard de l'état de l'art. Cette comparaison a été possible grâce au temps de retour raisonnable de l'évaluation en utilisant le modèle 0D, bien ce dernier ne permet pas de prendre en compte certains phénomènes physiques tels que l'aspect multidimensionnel de la chambre de combustion. Le prochain chapitre sera consacré à cette problématique plus complète et plus coûteuse pour la quelle ne nous pouvons pas réaliser des comparaisons pour des questions de budgets de temps de calcul et qui fera donc l'objet d'application réalisée directement avec l'algorithme asynchrone stationnaire.



(a) P2 (b) T2

(c) EGR (d) SOI

(e) ENGINEFREQ (f) FACTFREQ

(g) FACTINJ (h) RATE-MAX

FIGURE 6.7 – Evolution des paramètres : Pression d'admission (g), Température d'admission (h), EGR (c) et début d'injection (d) au cours de l'optimisation

Chapitre 7

Modélisation multidimensionnelle (3D)

Sommaire

7.1 Introduction **93**
7.2 Travaux connexes **94**
7.3 Spécificités de la modélisation Diesel **95**
7.3.1 Symétrie de la chambre de combustion 95
7.3.2 Injection à soupapes fermées 96
7.4 Présentation de la chaine de modélisation 3D **96**
7.5 Définition du problème d'optimisation **97**
7.5.1 Les paramètres de l'optimisation 97
7.5.2 Les contraintes de l'optimisation 100
7.6 La chaine d'optimisation **103**
7.7 Implémentation de la chaine d'optimisation **104**
7.7.1 La plateforme CINES 104
7.7.2 Le serveur PSA 105
7.7.3 Discussion 105
7.8 Réglages **106**
7.9 Résultats **106**
7.9.1 Front de Pareto 106
7.9.2 Analyse physique 107
7.9.3 Méta-modèle : résultats préliminaires 111
7.10 conclusion **112**

7.1 Introduction

Dans ce chapitre, nous étudions la problématique de l'optimisation de la combustion Diesel en utilisant la modélisation multidimensionnelle 3D. Comme décrit précédemment dans le chapitre 5, il existe plusieurs types de modélisation 3D.
Dans le cadre de cette étude, nous utilisons la modélisation RANS pour réaliser l'optimisation 3D. Cependant, le coût de calcul en utilisant la modélisation RANS peut varier considérablement selon le point de fonctionnement du moteur étudié.
Dans le cas d'un point à "plein charge" le régime du moteur est très élevé (oscillant autour de 4000 tr/min), et la masse du carburant injectée est grande. Ce point de

fonctionnement correspond aux grandes vitesses qui peuvent être atteintes par un véhicule. De ce fait, la seule grandeur à optimiser dans ce cas de figure est la puissance délivrée par le moteur.
Contrairement au point "plein charge", le point à "charge partielle" correspond à des régimes moteur moyens ou faibles (entre 1500 tr/min et 2500 tr/min) et sa modélisation est beaucoup plus compliquée que celle de la "pleine charge". Cette complexité est liée aux phénomènes physiques qui rentrent en jeu dans le cas d'un point à charge partielle : la stratégie d'injection qui se fait en 3 sous-injections au lieu d'une seule à plein charge, l'auto-inflammation en présence de faibles températures et des taux d'EGR élevés, le processus de formation des polluants (Nox et suies)... Sachant que la prise en compte de ces différents phénomènes se fait au détriment du coût CPU, le coût de calcul à "charge partielle" devient beaucoup plus important par rapport à celui de la "pleine charge" : le calcul à pleine charge dure entre 12 heures et 1 jours tandis que le calcul à charge partielle oscille entre 2 et 5 jours (cette variabilité est due à l'hétérogénéité du temps calcul).
D'autre part, comme le point de fonctionnement à charge partielle correspond à des vitesses pratiquées par les véhicules dans la ville, plusieurs objectifs seront pris en compte dans l'optimisation tels que : les émissions polluantes (NOx et Suies) et la consommation du carburant. Dans l'optimisation que nous nous proposons de réaliser dans cette étude, le point de fonctionnement est un point à charge partielle (2400 tr/min). L'algorithme AS-NSGA-II (NSGA-II stationnaire asynchrone) présenté dans le chapitre 3 sera utilisé afin de faire face à la problématique de l'hétérogénéité du coût CPU dans le but de réduire le coût de l'optimisation. Une deuxième optimisation utilisant l'algorithme REG-SVM-AS-NSGA-II (NSGA-II stationnaire asynchrone avec méta-modèles) présenté dans le chapitre 4 a été mise au point dans le but d'étudier l'effet des méta-modèles sur une problématique réelle. Malheureusement, pour des raisons limitantes de temps, seulement quelques évaluations ont pu être réalisées. Nous nous contenterons donc de comparer les résultats préliminaires.
Ce chapitre sera organisé comme suit : après un survol des principaux travaux de l'état de l'art dans la section 7.2, nous décrivons les spécificités du calcul Diesel dans la section 7.3. La section 7.4 sera dédiée à la présentation de la chaine de modélisation 3D, et la section 7.5 se focalisera sur la définition des paramètres et des contraintes de l'optimisation. Les sections 7.6 et 7.7 seront consacrées aux détails de l'implémentation du couplage réalisé entre l'algorithme et l'outil de modélisation. Nous présentons les résultats de l'optimisation dans la section 7.9 et les résultats préliminaires de l'optimisation avec les méta-modèles dans la section 7.9.3, avant de finir ce chapitre par une conclusion.

7.2 Travaux connexes

Les premiers travaux dans le domaine de l'optimisation des modèles de combustion 3D, ont utilisé l'amélioration manuelle en étudiant l'impact de certains paramètres sur la combustion.

[Tsao 1990] utilisèrent un code de simulation 3D pour analyser l'influence de la profondeur du bol de combustion sur l'écoulement dans la chambre. Les auteurs de [Zhang 1995] étudièrent l'influence de la géométrie de la chambre de combustion sur la combustion Diesel en considérant 3 différentes formes géométriques du bol de combustion. Ce travail conclut qu'en modifiant la forme de la chambre, des gains peuvent être obtenus sur l'émission des suies.
Par la suite, plusieurs travaux se sont intéressés à l'utilisation des algorithmes évolutionnaires dans le cadre de l'optimisation automatique de la combustion en utilisant les modèles de simulation 3D :
Senecal et al. proposèrent dans [Senecal 2000] une méthodologie basée sur la modélisation multi-dimentionnelle dans le but d'optimiser les émissions polluantes. L'outil de simulation du cycle moteur fut couplé à un algorithm génétique multi-objectif. L'étude investigua l'effet de six paramètres du moteur sur les émissions et la performance sur un point de fonctionnement "plein charge".
Dans [Weber 1998], la caractérisation du spray d'injection dans un moteur Diesel a été étudiée en utilisant un algorithme génétique mono-objectif. L'objectif de l'optimisation était de calibrer le spray d'injection en diminuant l'erreur par rapport aux mesures expérimentales.
Une optimisation évolutionnaire multi-objective a été réalisée dans [Donateo 2006] en utilisant 4 modes de fonctionnement correspondant à des régimes moteurs différents. Cependant, le modèle de simulation 3D utilisé étant réduit, le coût d'une évaluation varie entre 10 minutes et 30 minutes selon le processeur utilisé, et l'optimisation globale dure un peu plus de 4 jours.

7.3 Spécificités de la modélisation Diesel

Avant de décrire les différents éléments du processus d'optimisation, nous présentons dans cette section quelques spécificités de la modélisation de la chambre de combustion Diesel en la comparant à celle relative à la combustion essence. Les spécificités décrites ci-dessous ont un impact direct sur le temps de restitution global du calcul Diesel.

7.3.1 Symétrie de la chambre de combustion

La chambre de combustion Diesel est caractérisée par une symétrie de la forme géométrique par rapport à l'axe principale du cylindre. Cette symétrie concerne aussi le phénomène d'injection du carburant qui se fait à travers les différents trous de l'injecteur. De ce fait, la chambre de combustion peut être divisée en N parties équivalentes où N représente le nombre de trous de l'injecteur. La simulation 3D peut être ainsi réalisée uniquement sur une portion de la chambre qui représente "1/nombre de trous" du volume global. La modélisation de la chambre complète peut être obtenue par la suite grâce aux propriétés de la symétrie.
Cette réduction permet de réaliser un gain important sur le temps de restitution

d'un calcul Diesel, qui devient beaucoup moins important que le calcul essence où la totalité de la chambre doit être modélisée du fait de la non-symétrie de la chambre.

7.3.2 Injection à soupapes fermées

L'injection dans un moteur Diesel se produit soit dans la phase de compression soit dans la phase de détente. Durant ces deux étapes du cycle Diesel, les soupapes d'admission et de d'échappement du moteur sont fermées contrairement à la combustion essence où l'injection peut avoir lieu durant la phase d'admission. De ce fait, la modélisation de la combustion Diesel peut être faite en prenant en considération uniquement l'intérieur de la chambre. L'effet des conduits est pris en compte dans la phase d'initialisation du calcul en imposant les paramètres qui décrivent l'aérodynamique générée par les conduites des soupapes. Cette réduction du volume global à modéliser pour simuler la combustion Diesel se traduit par un gain non négligeable sur le temps de restitution du calcul, en plus de celui obtenu grâce à la symétrie de la chambre.

Compte tenu des deux hypothèses décrites ci-dessus, le coût CPU d'un calcul Diesel est considérablement réduit. C'est la raison pour laquelle les systèmes de combustion essence ne sont pas utilisés dans le cadre des optimisations automatiques, car le coût d'une évaluation unitaire est de l'ordre de 2 semaines. Il est à rappeler que l'utilisation du calcul Diesel dans un processus d'optimisation reste toutefois un processus très coûteux à cause des nombreuses évaluations de l'algorithme évolutionnaire d'où la nécessite de faire des efforts afin de réduire le coût global de l'optimisation.

7.4 Présentation de la chaine de modélisation 3D

Dans cette section nous présentons la chaine de modélisation de la combustion 3D. Cette chaine comporte une série d'outils (codes de calcul) qui seront couplés afin de réaliser une évaluation unitaire d'un individu.
L'outil de modélisation 3D RANS utilisé dans cette thèse est FIRE dans sa version V2009.2, développé et commercialisé par la société AVL. Cet outil permet de calculer les écoulements instationnaires turbulents en 3 dimensions dans les géométries complexes en particulier dans les moteurs à combustion interne. Il s'appuie sur des principes généraux de conservation de propriétés décrivant le comportement de matière en interaction avec son entourage. Les unités élémentaires de calcul sont définies par des mailles qui échangent de l'information au cours du calcul. FIRE permet de réaliser le calcul en maillage mobile afin de permettre la simulation de la combustion en volume variable . Il est ainsi très utilisé dans le domaine automobile par plusieurs constructeurs en l'occurrence : PSA Peugeot Citroen, BMW, Toyota..etc.
Le code de calcul de FIRE sera couplé à un autre outil appelé ESED développé par la même société AVL. Le module ESED permet de générer la CAO et le maillage

des chambres de combustion de façon automatique, à partir des spécifications géométriques définies par l'utilisateur. Il est de ce fait, très adapté à l'utilisation dans un contexte d'optimisation ou toutes les étapes du calcul doivent être automatisées et pilotées par l'algorithme.

7.5 Définition du problème d'optimisation

Dans cette section nous décrivons les différentes composantes du problème d'optimisation. Nous rappelons que les objectifs à optimiser sont : les suies, les NOx et la consommation du carburant. Nous décrivons dans ce qui suit les différents paramètres retenus ainsi que les contraintes de l'optimisation.

7.5.1 Les paramètres de l'optimisation

Dans cette partie, nous présentons les différents paramètres retenus dans le cadre de cette étude. Nous distinguons quatre familles de paramètres : les paramètres géométriques qui pilotent la forme de la chambre de combustion, les paramètres d'injection, un paramètre d'aérodynamique qui est le swirl, et enfin l'EGR qui est un paramètre de boucle d'air.

7.5.1.1 Paramètres géométriques

Les paramètres géométriques permettent de modifier la forme de la chambre de combustion. Durant l'optimisation, ces paramètres sont utilisés par le logiciel ESED pour générer la forme géométrique fermée de la chambre de combustion. ESED utilise des *templates* dans lesquels sont définis les paramètres géométriques figés ainsi que les paramètres géométriques qui peuvent être modifiés. L'utilisateur peut ainsi, selon le template choisi, définir le champ d'investigation des géométries des bols. Dans le cadre de notre étude, nous avons retenu six paramètres géométriques : le diamètre du bol, l'inclinaison de la chasse, la distance piston-culasse (DPC), le taux du réentrant, la distance injecteur-piston (NTP : *Nozzle To Piston*), et enfin la profondeur du bol. Les 5 premiers paramètres sont directement pilotés par l'algorithme tandis que le dernier (profondeur du bol) sera utilisé pour satisfaire la contrainte sur le volume du bol qui sera décrite par la suite (section 7.5.2.1). La figure 7.1 illustre ces paramètres sur la géométrie du cas de référence de l'optimisation.

7.5.1.2 Paramètres d'injection

Dans la modélisation 3D que nous utilisons, 3 injections par cycle seront utilisées contrairement à la modélisation 0D (chapitre précédent) où seulement l'injection principale *main* a été prise en compte. La première injection est effectuée avant l'injection *main* dans la phase de compression avec une faible quantité du carburant. Connue dans le jargon des motoristes sous le nom "injection *pilote*" ou "pré-injection", cette dernière est utilisée dans le but de réduire le bruit et les vibrations du moteur : l'injection précoce d'une faible quantité de combustible prépare

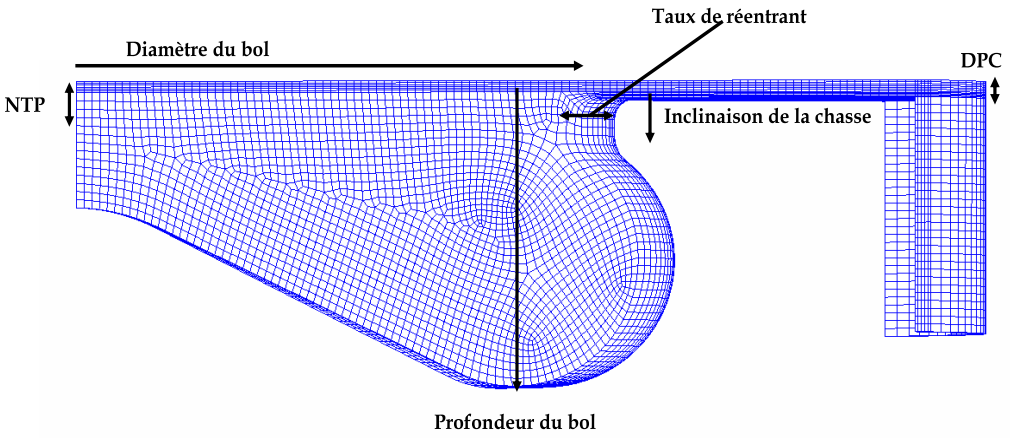


FIGURE 7.1 – Illustration des paramètres géométriques sur la géométrie du cas de référence

le mélange avant l'injection *main* en préchauffant les zones proches de l'injecteur ce qui implique la diminution du délai d'auto-inflammation de l'injection *main*. Ainsi, la combustion est moins violente [Tanaka 2002].
La deuxième injection est l'injection principale ou injection *main*, dans laquelle la grande partie de la quantité du carburant sera injectée.
La troisième injection, appelée "*split injection*" ou "post-injection" est effectuée dans la phase de détente après l'injection *main*. Cette injection peut être décrite comme un fractionnement de l'injection principale *main* en deux injections afin d'améliorer le mélange pendant la phase tardive de combustion, et réduire les émissions des suies suite à une meilleure utilisation de l'air dans la chambre [Choi 2002].
La variation du moment de début des trois injections décrites ci-dessus permet de modifier le phasage de ces dernières et donne lieu à des stratégies d'injections totalement différentes. Ainsi, compte tenu des effets que peut avoir les injections *pilot* et *split* sur la formation des polluants (objectifs de l'optimisation) , 3 paramètres liés au phasage des injections ont été définis :

1. le phasage pilote-main (ou "*dwell pilot-main*") : ce paramètre décrit le délai (en degré vilebrequin) entre le début de l'injection main et le début de l'injection pilot.

2. phasage main-split (ou "*dwell main-split*") : ce paramètre décrit le délai (en degré vilebrequin) entre le début de l'injection main et le début de l'injection split. Cependant, compte tenu des spécificités de notre application, ce paramètre a été reformulé pour représenter le délai entre la fin de l'injection main (plutôt que le début) et le début de l'injection split. (ce changement sera expliqué dans la section 7.5.2.2)

3. avance du train d'injection : Ce paramètre traduit l'avance globale du train d'injection (les 3 injections sont prises en compte). Comme les paramètres "dwell main-pilot" et "dwell main-split" sont calculés par rapport à l'injection main, le paramètre "avance du train" s'appliquera seulement sur l'injection main. Son application sur les deux autres injections est alors prise en compte car les dwell seront calculés par rapport à la nouvelle position de l'injection main.

Les deux derniers paramètres d'injection sont le nombre de trous qui permet de définir le volume du secteur de la chambre à modélisé, et la variation de l'angle de spray. Pour ce dernier paramètre, la variation se fait autour d'une valeur nominale de l'angle de spray qui est calculée pour chaque individu en fonction de sa géométrie. En effet, une règle métier préconise de diriger le spray d'injection vers la lèvre du bol de combustion afin de mieux répartir le carburant dans la chambre. Cependant, afin de créer plus de diversité dans les stratégies d'injection, l'angle de spray est légèrement modifié par la suite à l'aide du paramètre d'optimisation qui permet d'effectuer des variations autour de la valeur nominale.

7.5.1.3 Paramètres d'aérodynamique

L'aérodynamique de la chambre est prise en compte dans l'optimisation à travers un paramètre qui est le *swirl*. Le swirl peut être défini comme un mouvement de rotation autour de l'axe du cylindre (figure 7.2). Le swirl est créé en entrainant l'écoulement d'admission au sein du cylindre avec un moment angulaire initial. L'avantage d'utiliser ce mouvement est d'accélérer le mélange entre l'air présent dans la chambre et le fuel injecté, ce qui permet d'avoir une meilleure combustion [Heywood 1988].
Le paramètre d'optimisation lié au swirl est défini comme étant une variation autour de la valeur nominale du cas de référence.

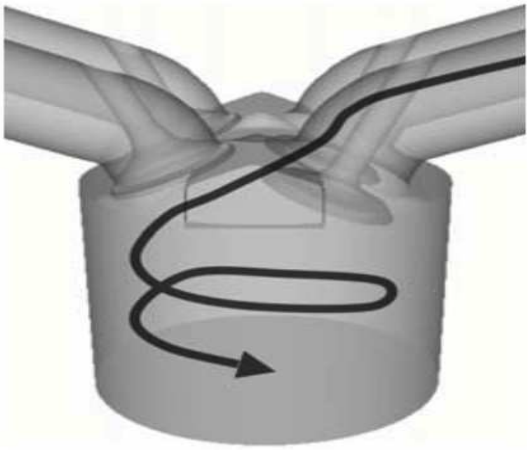


FIGURE 7.2 – Le mouvement de swirl

7.5.1.4 Paramètres de boucle d'air

Les paramètres de boucle d'air sont au nombre de trois : la température d'admission, la pression d'admission et le taux de l'EGR. Cependant, seulement le taux de l'EGR sera considéré comme paramètre d'optimisation. les deux autres paramètres seront recalculés à partir de l'EGR en utilisant une corrélation empirique obtenue à partir des essais réalisés sur banc moteur.

La table 7.1 récapitule les paramètres d'optimisation présentés ci-dessus en précisant leurs bornes inférieures et supérieures.

Table 7.1 – Tableau récapitulatif des paramètres de l'optimisation 3D

Paramètre	Unités	Description	borne inférieure	borne supérieure
Diamètre du bol	mm	/	46	60
Inclinaison de la chasse	mm	/	0	4
taux de réentrant	%	/	0	20
DPC	mm	Distance Piston-Culasse	0.6	2
NTP	mm	Distance Injecteur-Piston	-1.5	1
Avance du train	DV	/	355	370
dwell main-pilot	DV	phasage main-pilot	5	29
dwell main-split	DV	phasage main-split	2	20
Nombre de trous	/	/	5	12
variation de l'angle de spray	%	variation par rapport à la valeur nominale calculée	-10	10
EGR	%	Taux de recirculation des gaz brulés	0	30
Variation du swirl	%	variation par rapport au swirl du cas de référence	-0.5	1

7.5.2 Les contraintes de l'optimisation

Les contraintes de l'optimisation sont au nombre de deux et sont traitées à deux niveaux différents du processus d'optimisation. La première contrainte concerne le volume du bol de combustion, tandis que la deuxième est la même que celle imposée dans l'optimisation avec la modélisation 0D qui concerne la PMI.

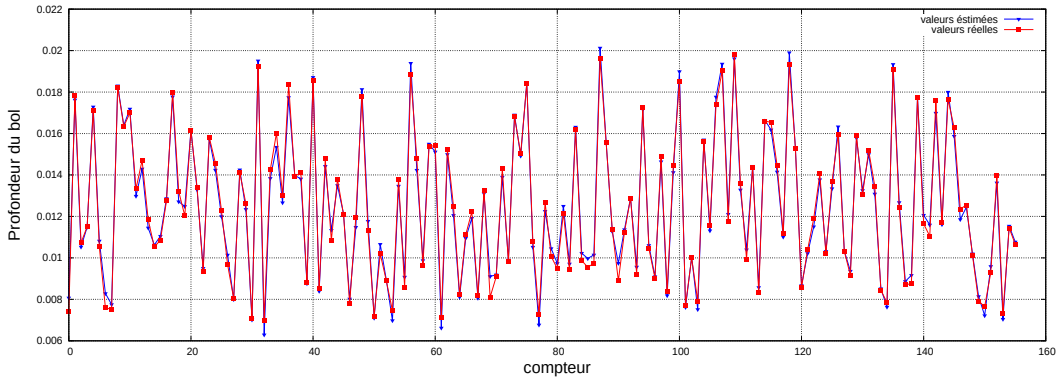


FIGURE 7.3 – Comparaison des valeurs estimées et des valeurs réelles du paramètre profondeur de bol

7.5.2.1 Contrainte sur le volume du Bol de combustion

Tous les individus issus de l'optimisation doivent avoir un volume de bol constant : la quantité d'air présente dans la chambre est un élément très important dans la combustion Diesel, car la qualité de la combustion dépend essentiellement du mélange air-carburant. Le fait que les individus puissent avoir des volumes de bols de combustion différents rend la comparaison de ces derniers en termes d'émissions polluantes biaisée.
Afin de satisfaire cette contrainte, le volume résultant de la combinaison des paramètres géométriques issues de l'algorithme doit rester constant (le plus proche possible du volume calculé pour le cas de référence). Pour cela, un plan d'expérience a été réalisé, dans lequel 200 différentes géométries de chambre de combustion ont été générées en utilisant le logiciel ESED. Par la suite, une surface de réponse basée sur une approximation quadratique est obtenue. Cette surface de réponse permet d'avoir une estimation du paramètre qui sera utilisé pour piloter le volume de la chambre, la profondeur du bol dans notre cas, en fonction des autres paramètres géométriques ainsi que du volume du bol calculé pour chaque individu issu du plan d'expérience.
La figure 7.3 compare les valeurs calculées réellement du volume de bol et les estimations obtenues en utilisant la formule de surface de réponse de la profondeur du bol. Nous pouvons constater que la formule permet d'avoir une bonne approximation du volume du bol sur les 156 géométries (ce nombre est inférieur au nombre fixé au préalable qui est de 200 car les géométries non fermées non pas été considérées). Au final, cette formule sera utilisée lors de l'optimisation en utilisant un volume de bol constant (celui du cas de référence), et les différents paramètres géométriques générés par l'algorithme.

7.5.2.2 contrainte sur la PMI

La deuxième contrainte à satisfaire concerne la PMI. Tout comme dans l'application précédente avec la modélisation 0D, les individus doivent être à iso valeurs de PMI afin de pouvoir être comparés en termes d'émissions polluantes. Un algorithme de gradient est également utilisé dans le but de faire converger les individus vers la valeur cible de la PMI. Seule la masse de l'injection main sera utilisée comme paramètre de l'algorithme de gradient, les masses des injections *pilot* et *split* resterons figées tout au long des sous-itérations. Le nombre maximum des sous-itérations est fixé à 4. A défaut, si l'individu ne parvient pas à converger vers la valeur cible de la PMI, il sera pénalisé.
Cependant, la définition actuelle des paramètres d'injections peut donner lieu à des cas non réalisables, compte tenu de l'utilisation des multi-injections : dans le jargon des motoristes, le phasage entre deux injections est défini comme étant le délai (exprimé généralement en degré vilebrequin) entre le début des deux injections. En utilisant cette définition, le paramètre *dwell-main-split* représente le délai entre le début de l'injection *main* et le début de l'injection *split*. Cependant, durant le processus de convergence avec l'algorithme de gradient afin de respecter la contrainte en PMI , la masse de l'injection *main* varie (augmente ou diminue). Dans certains cas où il faut augmenter la masse de l'injection *main*, la durée de cette dernière augmente étant donnée que la perméabilité de la buse est supposée constante. L'accroissement de la durée de l'injection *main* peut dans certains cas donner lieu à un chevauchement avec l'injection *split*, ce qui est du point de vue pratique irréalisable (figure 7.4). Afin d'éviter un tel scénario pendant l'optimisation, le paramètre *dwell-main-split* a été redéfini pour exprimer la distance qui sépare la fin de l'injection main du début de l'injection split. Dans ce cas de figure, l'injection split est redéfinie à chaque sous-itération de l'algorithme de gradient en fonction de la masse de l'injection main, et le risque de voir les deux injections se chevaucher est éliminé (figure 7.5).

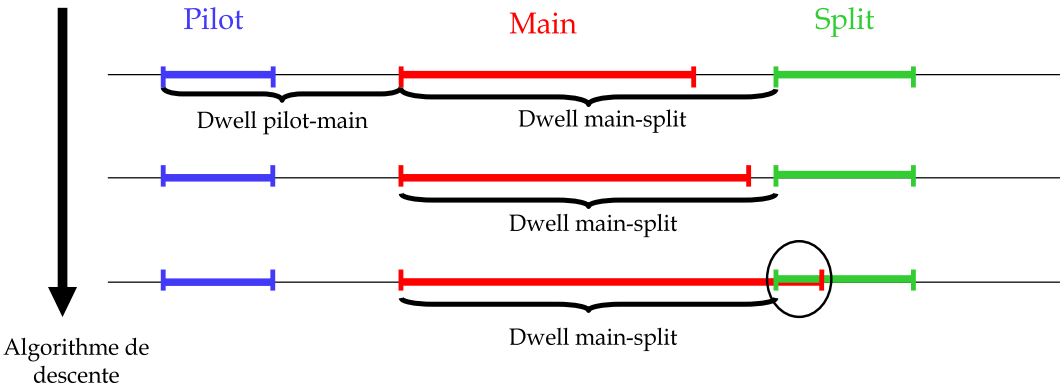


FIGURE 7.4 – Illustration du risque de chevauchement entre les injections main et split avec la définition classique du paramètre d'injection Dwell-main-split

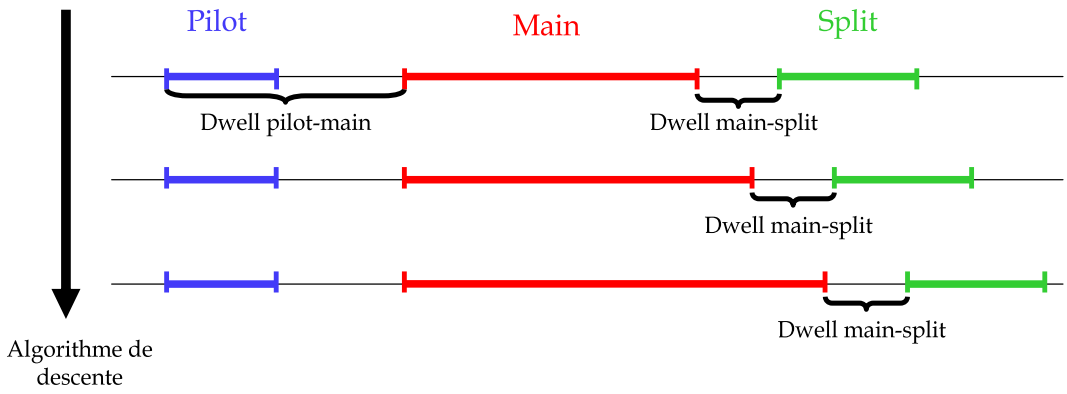


FIGURE 7.5 – Illustration de la nouvelle définition du paramètre d'injection Dwell-main-split

7.6 La chaine d'optimisation

Nous présentons dans cette section le schéma global de la chaine d'optimisation qui permet d'effectuer le couplage entre les outils de modélisation 3D et l'algorithme d'optimisation. L'algorithme utilisé durant cette étude est l'algorithme AS-NSGA-II (NSGA-II stationnaire asynchrone, voir chapitre 3 pour plus de détails). L'algorithme commence par générer une population initiale aléatoire, et fait évoluer cette population dans un schéma stationnaire asynchrone en générant un seul enfant par génération et en intégrant les individus dans la population dans un schéma "premier arrivé premier servi" afin de garantir une utilisation non interrompue de la grille de calcul.
A chaque itération, lors de l'étape d'évaluation, l'algorithme évolutionnaire passe la main à l'algorithme de descente qui veille sur le respect de la contrainte en PMI. L'algorithme de gradient commence l'évaluation par la première sous-itération en imposant une valeur nominale de la variable utilisée pour contrôler la PMI qui est la masse injectée de la main. A l'issue de la première sous-évaluation, l'algorithme de gradient adapte la valeur de l'injection main en fonction de la PMI calculée et relance la deuxième sous-itération. Cette procédure est répétée jusqu'à ce que la PMI calculée soit incluse dans l'intervalle de tolérance fixé à 1 % autour de la valeur cible. le budget maximum des sous-itérations pouvant être réalisées est fixé à 4. Au delà de ce chiffre, l'algorithme de gradient renvoie à l'algorithme évolutionnaire les résultats de la sous-itération avec la meilleurs valeur de PMI (la plus proche de la valeur cible). Une procédure de pénalisation sera utilisée par la suite pour pénaliser les individus selon leur éloignement de la valeur cible de la PMI. Les différents étapes de chaque sous-évaluation de l'algorithme de gradient peuvent être résumées comme suit :
Après que les différents paramètres d'optimisation aient été générés, un prétraitement est effectué afin de définir quelques paramètres auxiliaires à partir des paramètres d'optimisation (ces paramètres auxiliaires ne sont pas directement ex-

ploitables par l'algorithme). A titre d'exemple, la température initiale ainsi que la pression initiale sont recalculées à partir de la valeur de l'EGR. Cette étape de pré-traitement permet aussi d'estimer la valeur de la profondeur du bol pour le respect de la contrainte sur le volume de la chambre. Une fois l'étape de pré-traitement est achevée, ESED est lancé pour générer la géométrie ainsi que le maillage de l'individu à partir des spécifications géométriques. A l'issue de cette étape, ESED passe la main à l'outil FIRE qui, après la prise en compte des différents paramètres d'injections, d'aérodynamique et de boucle d'air, procède à la simulation numérique de l'individu. Cette simulation est effectuée en parallélisant le calcul sur plusieurs processeurs. Chaque processeur se voit assigné une partie de la chambre de combustion. A la fin de chaque sous-itération du calcul FIRE, les processeurs s'échangent les différentes informations afin de recalculer les différentes grandeurs sur toute la chambre. Une fois le calcul FIRE de l'individu est effectué, la procédure d'évaluation est terminée par le retour des valeurs des objectifs ainsi que celle des contraintes à l'algorithme.

7.7 Implémentation de la chaine d'optimisation

Après avoir défini les différents composants du problème d'optimisation, nous présentons dans cette section quelques détails concernant l'implémentation du workflow d'optimisation, en décrivant les spécificités des deux pistes retenues pour réaliser l'optimisation ainsi que le choix qui a été fait.
Comme indiqué précédemment, l'implémentation du problème d'optimisation est faite selon le modèle parallèle "maître-esclave" avec une topologie en étoile. La parallélisation intervient aussi au niveau de l'évaluation elle même : chaque individu est évalué sur plusieurs processeurs en utilisant une librairie de calcul parallèle. Le nombre de processeur à utiliser pour l'évaluation d'un individu avec FIRE varie entre 4 et 16 processeurs (au delà de 16 processeurs, le temps des communications augmente considérablement). Cependant, le choix du nombre de processeurs à utiliser par évaluation dépend de la capacité de la grille de calcul allouée à l'optimisation. Ainsi, deux pistes de plateformes de calcul différentes ont été considérées dans ce travail : le centre de calcul national CINES [1] et le serveur de calcul interne à PSA Peugeot Citroen. Dans ce qui suit, nous présentons les spécificités de chaque plateforme, et nous discutons par la suite le choix qui a été fait pour la réalisation de l'optimisation.

7.7.1 La plateforme CINES

Le CINES (Centre Informatique National de l'Enseignement Supérieur) est un centre de calcul national domicilié à Montpellier, qui offre aux académiques la possibilité de réaliser des calculs intensifs en mettant à leur disposition des milliers de processeurs de calcul. Ce centre a été retenu comme une piste intéressante pour

1. http ://www.cines.fr

réaliser l'optimisation 3D pour le nombre important des capacités de calcul qu'il offre. Le principale avantage du calcul au CINES est la non limitation du nombre de processeurs qui peuvent être utilisés en parallèle : l'utilisateur n'est pas limité dans le nombre de calculs à soumettre en même temps (en parallèle). Toutefois, l'utilisation des processeurs obéit à des règles de priorités et par conséquent, dans le cas d'une forte demande, une partie ou la totalité des calculs peut rester longtemps en queue d'attente. Ce phénomène bien qu'il contribue à l'augmentation du coût global de l'optimisation, reste gérable d'un point de vue algorithmique, car le schéma stationnaire asynchrone permet à l'algorithme de s'adapter à cette hétérogénéité et de poursuivre l'optimisation sans attendre les individus les plus lents, bloqués dans les queues d'attente dans ce cas de figure.
Une autre règle d'utilisation de la grille de calcul au CINES limite la durée globale que peut avoir un calcul à 120 heures (soit 5 jours). Cette limitation est appliquée pour tous les processus en cours d'exécution quelque soit leur nature.
Dans les premiers essais que nous avons conduits sur la plateforme de CINES, les calculs FIRE ont été réalisés sur 8 processeurs. Avec ce nombre de processeurs, la durée d'un calcul FIRE varie entre 2 et 4 jours.

7.7.2 Le serveur PSA

Le serveur interne à PSA dispose de 92 processeurs. Dans cette configuration, Les calculs FIRE sont parallélisés sur 4 processeurs seulement au lieu de 8 comme dans la plateforme du CINES, car le nombre d'individus pouvant être mis en parallèle avec 8 processeurs est petit (23 individus avec 4 processeurs contre 11 avec 8 processeurs). Cependant, le temps de calcul global reste semblable à celui obtenu avec les calculs au CINES c'est à dire variant entre 2 et 4 jours. A noter qu'en utilisant 8 processeurs, la durée du calcul varie entre 1 et 2 journée. Ce speed-up de facteur deux est du à la capacité de calcul des processeurs qui est deux fois plus rapide pour la grille de PSA par rapport à celle du CINES.
D'autre part, bien que la grille de PSA dispose d'un nombre limité de processeurs, un avantage qu'offre cette grille réside dans le fait qu'elle soit dédiée à l'optimisation que nous souhaitons réaliser. Autrement dit, aucun passage des individus à évaluer en queue d'attente n'est prévu.

7.7.3 Discussion

Le principale désavantage de l'utilisation de la plateforme du CINES reste son manque de flexibilité avec les problématiques d'optimisation. En effet, dans un schéma parallèle "maître-esclave", le processus maître doit rester en état d'exécution tout au long du processus d'optimisation. Or, une règle d'utilisation des machines au CINES préconise l'arrêt automatique de tous les processus en cours au bout de 120 heures d'exécution (5 jours), quelque soit la nature de ce dernier.
Bien que nous avons tenté de contourner ce problème en exécutant le processus maître sur une machine extérieure au CINES, les temps de passage en queue d'at-

tente qui peuvent être parfois de l'ordre de quelques jours rendaient le coût de l'optimisation trop important.
Pour ces raisons, nous avons choisi de réaliser l'optimisation sur la grille de calcul à PSA, qui malgré un nombre réduit de processeurs offrait une utilisation non interrompue des processeurs sur la grille.

7.8 Réglages

Nous présentons à présent les réglages qui ont été effectués afin de réaliser l'optimisation 3D. L'algorithme stationnaire asynchrone AS-NSGA-II (cf chapitre 3) a été utilisé pour réaliser l'optimisation. A noter que la version standard générationelle n'a pas été retenue à cause du coût très élevé à prévoir pour cette dernière dans un contexte de temps d'évaluation hétérogène. La taille de la population est fixée à 50. Sachant que chaque évaluation est réalisée sur 8 processeurs, 23 individus peuvent être évalués en parallèle sur la grille de calcul. Cette différence entre la taille de la population et la taille de la grille n'a pas d'impact sur les algorithmes à schéma stationnaire : à chaque génération 1 enfant est généré et envoyé sur la grille de calcul. Le critère d'arrêt est donné par le coût de l'optimisation en termes de temps global écoulé : un budget de 3 mois de calcul a été alloué pour réaliser l'optimisation sur le serveur interne de PSA. Une deuxième optimisation utilisant l'algorithme Reg-SVM-AS-NSGA-II (cf chapitre 4) a été lancée. Malheureusement seulement 40 évaluations ont pu être réalisées, à cause de contraintes limitantes de temps. A noter que l'objectif principal de cette étude est de réaliser l'optimisation en utilisant l'algorithme stationnaire asynchrone de NSGA-II. Cependant, l'utilisation de Reg-SVM-AS-NSGA-II a pour but de comparer les deux algorithmes cette fois sur une problématique "grandeur nature" et de quantifier le gain qui peut être réalisé en utilisant les méta-modèles.

7.9 Résultats

La durée global de l'optimisation réalisée avec AS-NSGA-II s'élève à environs 3 mois. Ce chiffre correspond à un budget avoisinant les 400000 heures CPU. La figure 7.6 illustre le phénomène d'hétérogénéité des durées d'évaluation pour les individus de la population initiale. Dans ce qui suit, après une analyse du front de Pareto nous nous focalisons sur l'aspect physique en analysant les résultats d'un point de vue de combustion.

7.9.1 Front de Pareto

Les figures 7.7, 7.8 et 7.9 illustrent les plans 2 dimensions de l'espace objectif pour tous les points faisables évalués lors de l'optimisation. Les points représentés par des ronds correspondent aux points appartenant au front optimal de Pareto. Des figures 7.7 et 7.8 nous pouvons constater des compromis dans les plans "suies-NOx" et "consommation-NOx" respectivement. Nous pouvons constater aussi de la

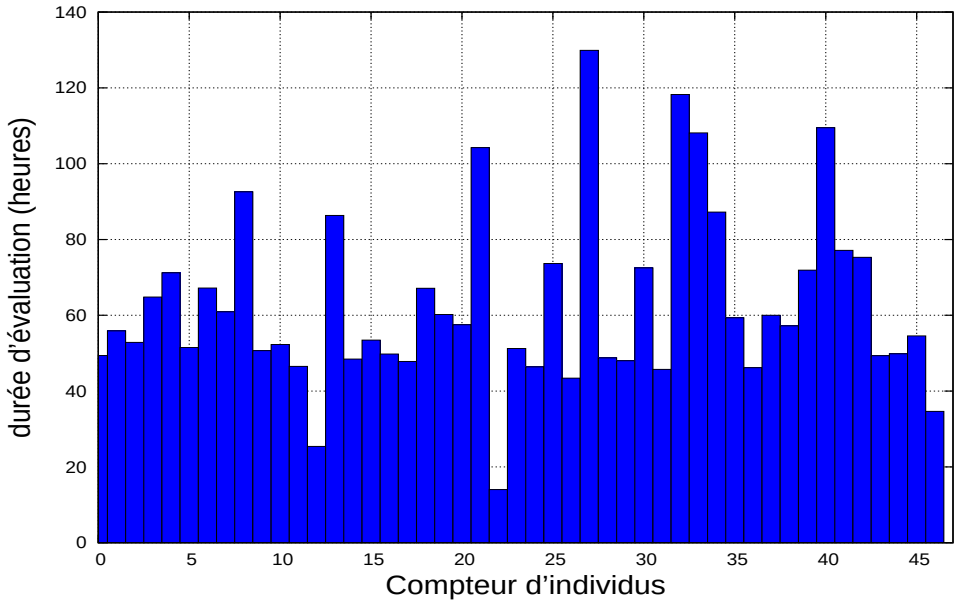


FIGURE 7.6 – Durées des évaluations des individus de la première génération aléatoire (heures)

figure 7.9 que les deux objectifs suies et consommation évoluent plutôt dans la même direction. Ceci peut être expliqué par le fait que la consommation est directement liée à la masse de carburant injectée, c'est à dire que plus la masse injectée est élevée, plus la consommation du carburant augmente. D'autre part, on sait que la formation des suies est directement liée à la quantité du carburant présente dans la chambre.

7.9.2 Analyse physique

Nous proposons d'étudier dans cette section l'optimisation d'un point de vue physique. A partir du front de Pareto nous avons choisi un individu dans la zone de compromis que nous allons analyser. L'analyse repose sur la comparaison des phénomènes physiques qui se sont produits dans la chambre de combustion de l'individu optimal et le cas de référence dans un espace 3 dimensions.
Les figures 7.10 et 7.11 illustrent les formes géométriques de l'individu optimal que nous proposons d'analyser et du cas de référence de l'optimisation respectivement. Nous pouvons remarquer que l'individu optimal est caractérisé par un bol de forme plutôt profonde avec faible taux de ré-entrant et une chasse inclinée. La table 7.2 résume les différents paramètres d'optimisation de l'individu optimal que nous proposons d'analyser et du cas de référence par rapport auquel nous souhaitons le comparer. L'individu optimal est caractérisé par une stratégie d'injection semblable à celle du cas de référence ainsi qu'un taux d'EGR presque similaire. Cependant, le nombre de swirl est plus important pour l'individu optimal par rapport au cas

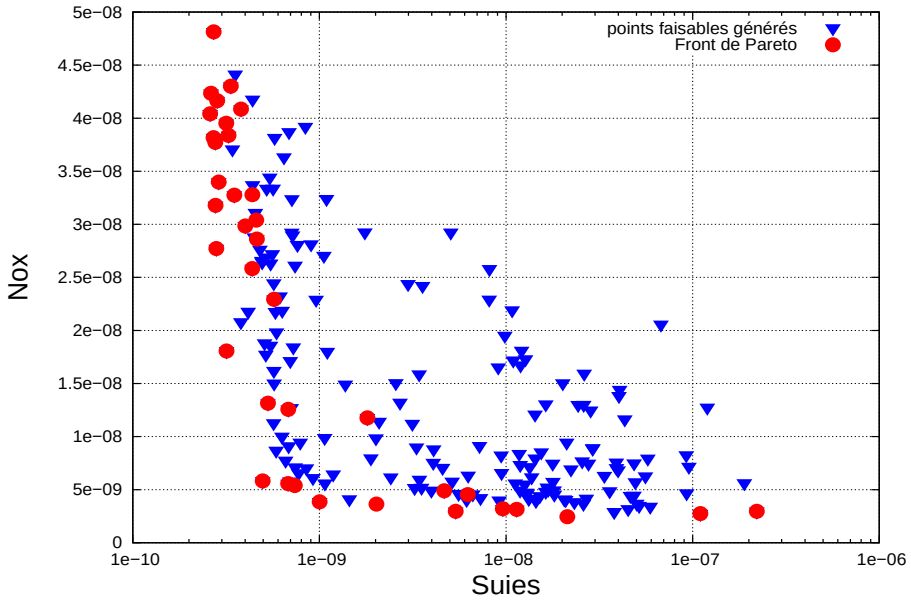


FIGURE 7.7 – Plan suies-NOx du front de Pareto

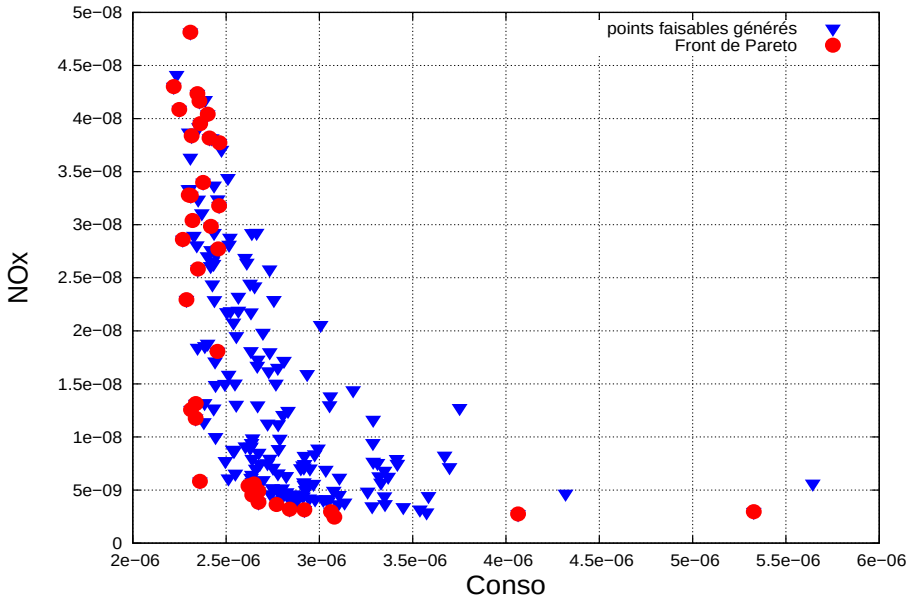


FIGURE 7.8 – Plan consommation-NOx du front de Pareto

référence, et le nombre de trous est égale à 7 ce qui donne lieu à un secteur un peu plus grand que celui du cas de référence (8 trous). En termes d'objectifs, l'individu optimal que nous proposons d'analyser permet d'améliorer les trois objectifs à la fois. Les gains par rapport au cas de référence s'élèvent à 3.41 %, 14,43 % et 0.41 %

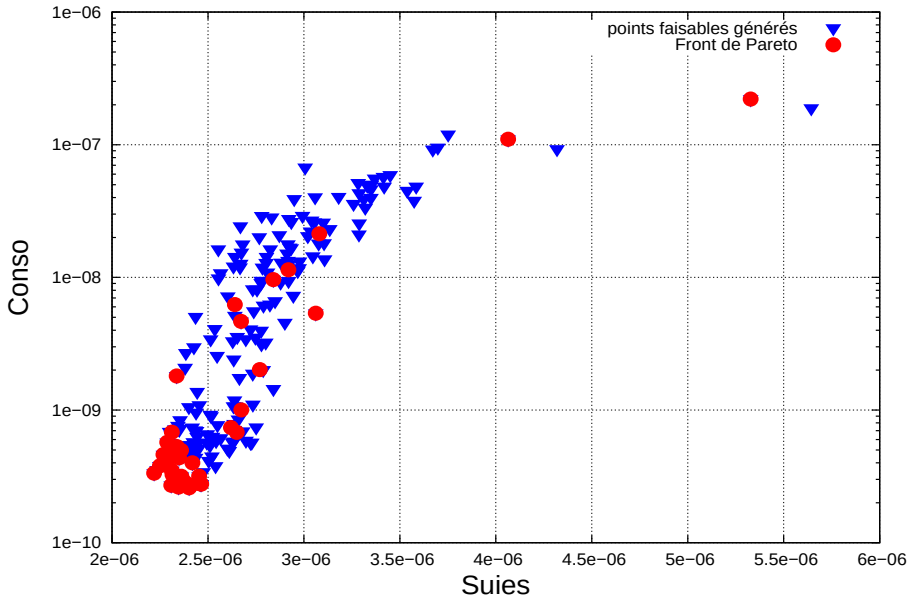


FIGURE 7.9 – Plan suies-consommation du front de Pareto

pour les trois objectifs : suies, NOx et consommation respectivement.

TABLE 7.2 – Comparaison des paramètres du cas de référence et du cas optimal

Paramètre	Unités	Cas référence	Individu optimal
Diamètre du bol	mm	56	51.8
Inclinaison de la chasse	mm	0	1.49
taux de ré-entrant	%	10	0.52
DPC	mm	0.8	0.824
NTP	mm	-1.02	-1.28
Avance du train	DV	362.2	359.28
dwell main-pilot	DV	28.6	26.27
dwell main-split	DV	2.52	6.76
Nombre de trous	/	8	7
variation de l'angle de spray	%	0	1.04
EGR	%	22	21
Nombre du swirl	%	2	2.39

Notre objectif à travers cette analyse physique est d'essayer d'expliquer ce gain en s'appuyant d'une part sur les différentes grandeurs physiques moyennées sur la chambre (pression, température, dégagement de chaleur, fractions massiques des émissions polluantes), et d'autre part sur l'analyse multidimensionnelle de la chambre de combustion (analyse 3D). La figure 7.9.2 illustre l'évolution des

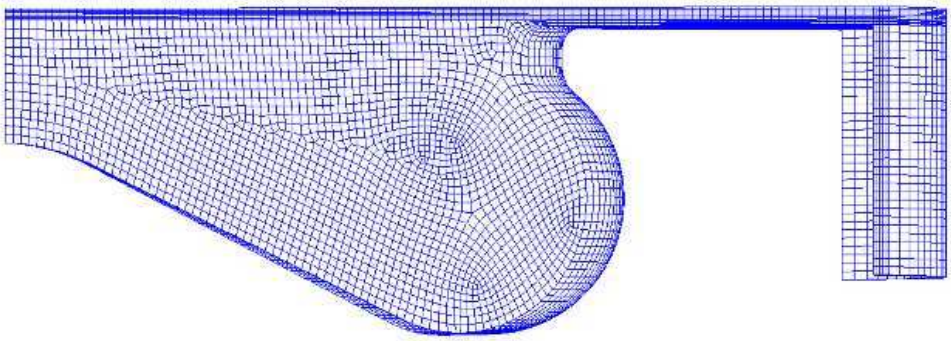


FIGURE 7.10 – Forme géométrique du cas de référence

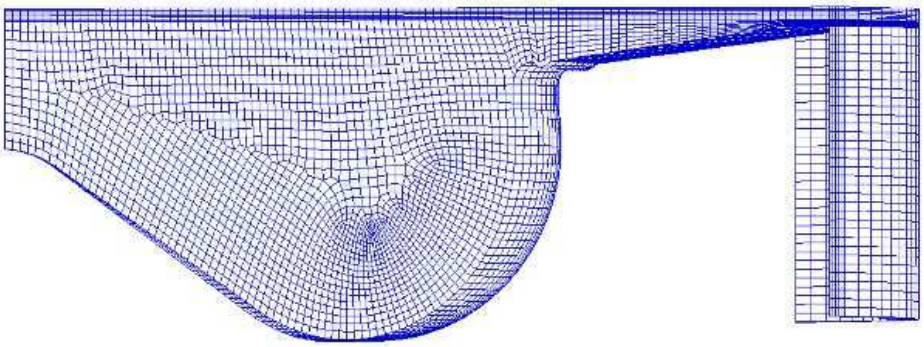


FIGURE 7.11 – Forme géométrique de l'individu optimal

différents paramètres physiques du cas de référence et de l'individu optimal au cours du cycle moteur. Tout d'abord, nous pouvons remarquer que les courbes de température moyenne (7.12(b)) sont quasiment semblables. Sur la courbe de pression (figure 7.12(a)) nous constatons une légère différence au voisinage du point mort haut [2] due au phasage légèrement différent de l'injection main des deux cas. Cette différence est visible aussi sur la courbe de dégagement de chaleur (figure 7.12(e)). Concernant les NOx, nous pouvons constater que le gain réalisé avec l'individu optimal est du essentiellement à une production inférieure des NOx lors de l'injection pilote (Figure 7.12(c)). Les deux individus semblent produire la même

2. Point mort haut : Dans un moteur a combustion interne, le piston est au point mort haut lorsque le volume de la chambre et le plus faible. Ce point correspond dans notre cas à la fin de la phase de compression (360 degrés vilebrequin)

quantité par la suite (injection main et split). Pour comprendre ce phénomène nous nous appuyons sur l'analyse 3D de la chambre de combustion. La figure 7.13 montre l'effet du swirl sur les injections dans le cas des deux individus. Un swirl plus élevé dans le cas de l'individu optimal permet de mieux répartir la masse injectée dans la chambre, et par conséquent une combustion "moins locale" que celle avec le cas de référence ce qui explique une réduction des NOx produits lors de l'injection Pilote. En ce qui concerne les suies (figure 7.12(d)), nous remarquons que malgré une plus grande production de suies avec l'individu optimal qu'avec le cas de référence, le processus d'oxydation par la suite permet d'oxyder plus rapidement pour descendre à la fin du cycle à un niveau plus bas que ce celui du cas de référence. Une fois de plus, nous nous référons à l'analyse 3D pour expliquer ce phénomène. Sur La figure 7.14, capturée à 380 degrés vilebrequin en début de détente, la fraction massique des suies est illustrée sur le plan vertical, tandis que l'iso-surface en rouge représente le taux d'oxygène équivalent à 17 % dans la chambre (17% correspond à un taux élevé d'oxygène). Pour le bol de référence, nous constatons que les suies sont situés principalement dans la partie du fond du bol. Ceci est du à sa forme géométrique caractérisée par une inclinaison de chasse égale à zéro. Pour l'individu optimal, les suies formés sont repartis en deux parties distinctes équivalentes en taille : la première en zone de chasse grâce à la chasse inclinée tandis que la deuxième est dans le fond du bol. En observant les iso-surfaces du taux d'oxygène, nous remarquons que grâce à la répartition équilibrée entre zone de chasse et fond du bol dans le cas de l'individu optimal, les suies sont "mieux" mélangés à l'oxygène qu'avec le cas de référence. Ceci permet d'expliquer l'oxydation plus efficace dans le cas de l'individu optimal. La figure 7.15 capturée à 420 degré vilebrequin confirme cette tendance. On y voit clairement que les deux parties des suies sont mieux disposées pour entrer en contact avec l'oxygène (processus d'oxydation) que dans le cas de référence où les suies sont totalement dans la zone du fond du bol et sont en contact avec une petite partie de l'oxygène, alors qu'une grande partie de ce dernier se situe dans la zone de la chasse et ceci permet d'expliquer l'accélération d'oxydation observée sur la courbe 7.12(d).

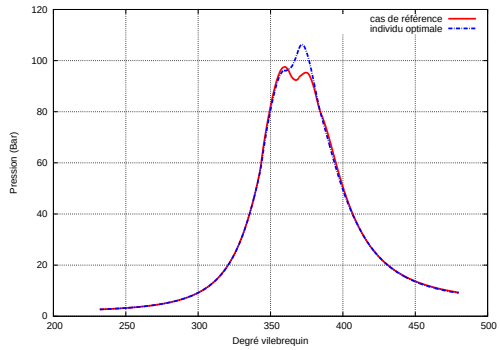
7.9.3 Méta-modèle : résultats préliminaires

Nous présentons dans cette section une comparaison des résultats préliminaires obtenus de la deuxième optimisation 3D réalisée avec l'algorithme Reg-SVM-NSGA-II avec l'optimisation réalisée avec AS-NSGA-II détaillée dans ce chapitre. Comme nous n'avons pas pu réaliser plus que 40 évaluations avec la deuxième optimisation, nous proposons de comparer les résultats préliminaires obtenus à ce budget. Nous pouvons constater de la comparaison des valeurs de l'indicateur de l'hypervolume des deux algorithmes à un budget de 40 évaluations (figure 7.16), que l'utilisation du méta-modèle permet d'avoir une certaine accélération. Ce constat conforte les résultats obtenus sur les benchmarks de fonctions analytique (cf. chapitre 4) dans lesquels une accélération nette est obtenue avec les variantes utilisant un méta-

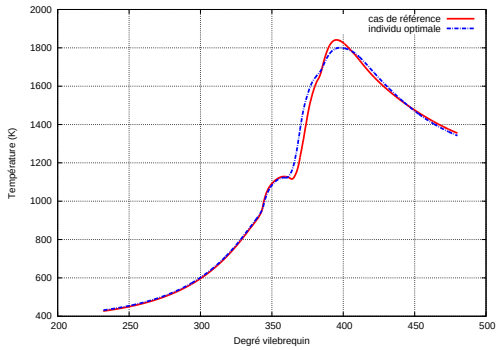
modèle dans les toutes premières évaluations.

7.10 conclusion

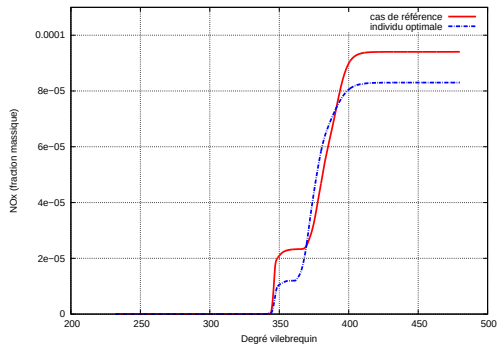
Ce chapitre a été consacré à la deuxième application réelle étudiée dans le cadre de cette thèse qui concerne l'optimisation multidimensionnelle 3D. Après avoir validé



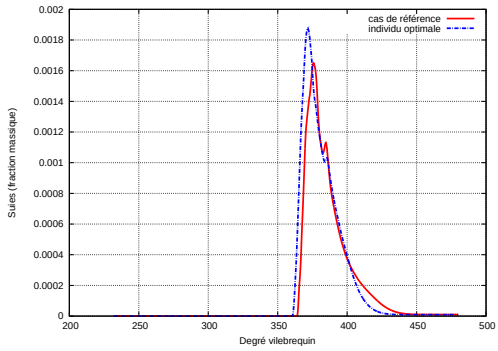
(a) Pression



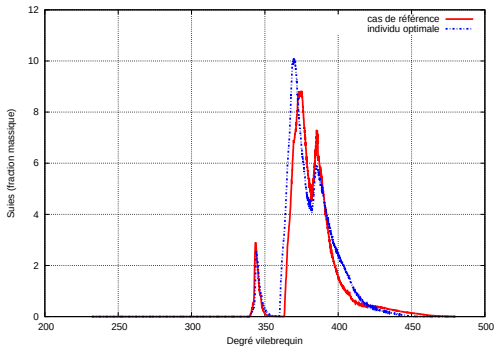
(b) Température



(c) NOx



(d) Suies



(e) Dégagement de chaleur

FIGURE 7.12 – Comparaison des grandeurs physiques de l'individu de référence et de l'individu optimal

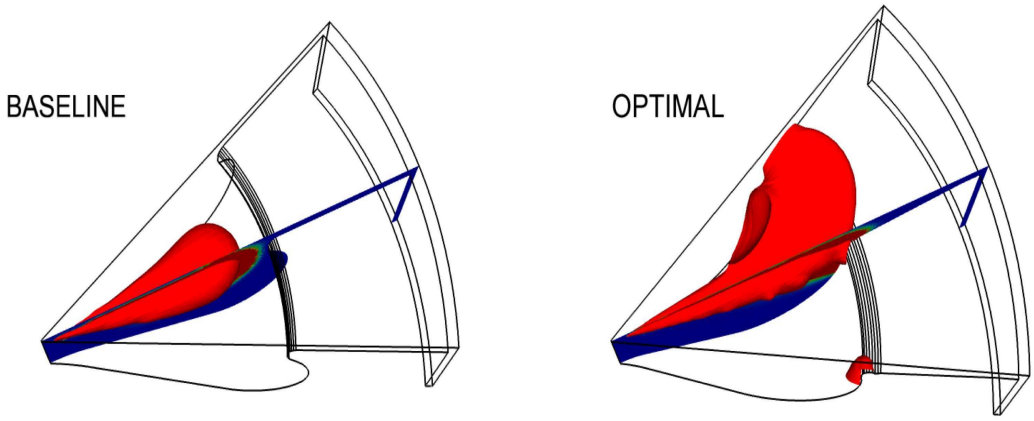


FIGURE 7.13 – Effet du swirl sur l'injection

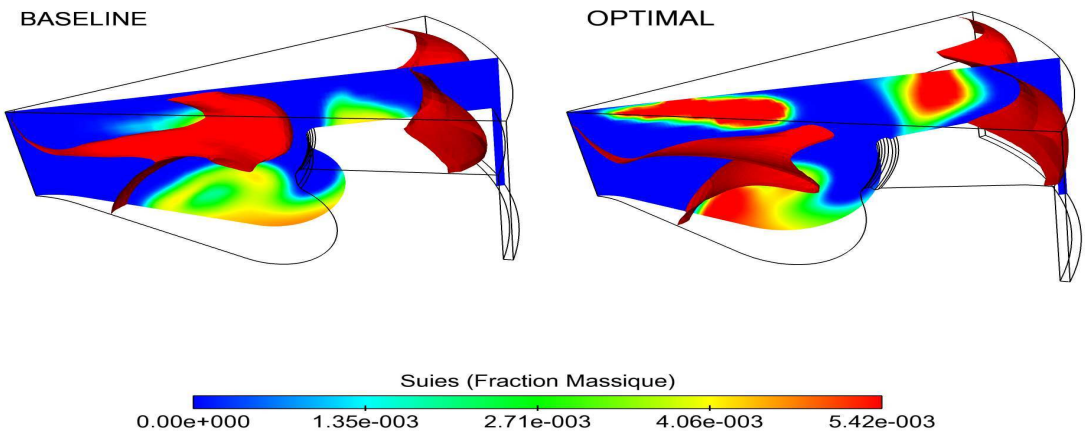


FIGURE 7.14 – Fraction massique des suies (plan vertical) avec une iso-surface du taux d'oxygène à 17 %- 380 degrés vilebrequin

l'utilisation de l'algorithme stationnaire asynchrone dans les chapitres précédents, d'abord sur les fonctions analytiques puis sur une problématique réduite de la combustion Diesel (modélisation 0D), nous avons couplé ce dernier à une chaine de modélisation 3D. Après implémentation du workflow d'optimisation sur le serveur interne de PSA, nous avons réaliser une optimisation avec un budget avoisinant les 300 évaluations, pour un coût global d'environs trois mois en termes de temps écoulé. D'un point de vue algorithmique, l'approche stationnaire asynchrone nous a permis de faire face au problème de l'hétérogénéité du temps de calcul en assurant une utilisation non interrompue des serveurs disponibles sur le serveur. A noter que cette réalisation n'était pas envisageable avec un algorithme générationnel à cause du coût très élevé qu'aurait engendrait l'attente à chaque génération de l'individu

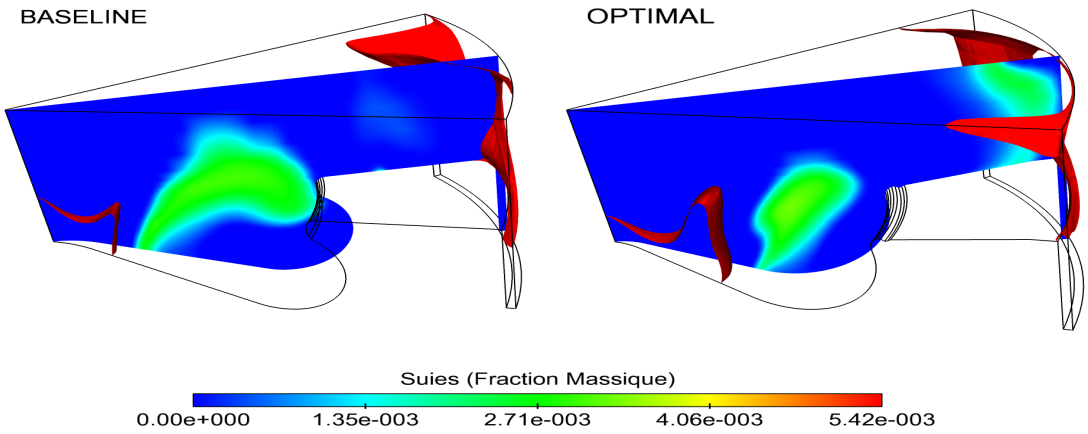


FIGURE 7.15 – Fraction massique des suies (plan vertical) avec une iso-surface du taux d'oxygène à 17 %- 420 degrés vilebrequin

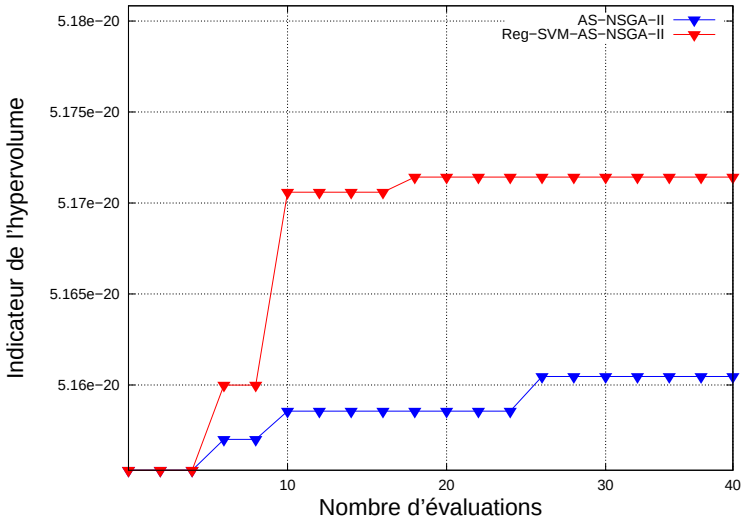


FIGURE 7.16 – Indicateur de l'hypervolume calculé pour les deux algorithmes AS-NSGA-II et Reg-SVM-AS-NSGA-II pour un budget de 40 évaluations

le plus long.
D'autre part, l'analyse multi-dimensionnelle des résultats obtenus nous a permis de dégager un concept qui semble être prometteur pour la combustion Diesel à charge partielle : le concept de chasse inclinée, qui permet de réduire les émissions polluantes grâce à une combustion plus efficace notamment en termes de processus d'oxydation de suies, qui grâce à la chasse inclinée permet d'avoir une meilleure répartition des suies formées dans la chambre.

Chapitre 8

Conclusion

Les travaux décrits dans ce manuscrit de thèse se trouvent au carrefour de deux domaines : celui de l'optimisation évolutionnaire multi-objectif et celui de la physique de la combustion. Les algorithmes évolutionnaires multi-objectif connus pour leur flexibilité et leur robustesse, ont été choisis comme méthode de résolution du problème d'optimisation de la combustion Diesel, caractérisé par une forte non-linéarité de l'espace de recherche, et des objectifs antagonistes. Cependant, comme les AEMO nécessitent de réaliser un grand nombre d'évaluations des fonctions objectif, l'objectif principal de ce travail a été de contribuer à la réduction du coût des optimisation réelles caractérisées par des évaluations très coûteuses. La motivation principale de ce travail trouve son origine dans les problèmes réels que nous avons étudiés dans ce cette thèse et qui représentent le contexte industriel de ce travail. Nous avons relevé, en comparant les différents algorithmes de l'état de l'art de l'optimisation évolutionnaire multi-objectif, que les performances respectives de ces derniers diffèrent selon le problème traité, et qu'aucun d'eux ne se détache des autres. Par conséquent, le gain escompté par rapport à l'algorithme NSGA-II qui est souvent utilisé dans le contexte des problématiques réelles ne pourra pas être réalisé en faisant un simple choix d'un algorithme ou d'un autre.
La piste qui a été finalement retenue concerne l'utilisation de ces algorithmes dans un premier temps sur les grilles de calcul parallèles. Motivés par le phénomène d'hétérogénéité du coût des évaluations constaté sur les deux problèmes réels traités dans cette thèse, et qui provoque l'inactivité de plusieurs serveurs sur la grille de calcul avec les algorithmes générationnels standards, nous nous somme focalisés sur l'étude des algorithmes stationnaires asynchrones. Ces algorithmes permettent d'utiliser plus efficacement les grilles de calcul en assurant une utilisation sans interruption des processeurs disponibles.
Nous avons commencé tout d'abord par étudier les algorithmes stationnaires asynchrones sur des benchmarks de fonctions analytiques (chapitre 3). Pour reproduire un comportement similaire à celui du contexte parallèle hétérogène nous avons construits des modèles artificiels de coûts de calcul qui représentent des configurations différentes pouvant être observées sur une grille de calcul. Outre le gain en terme de coût global de l'optimisation obtenu grâce à l'utilisation non interrompue de la grille, les résultats obtenus ont montré que les algorithmes stationnaires asynchrones offrent aussi un niveau de convergence comparable à celui obtenu avec les algorithmes générationnels.
Un perspective à ce travail d'étude du contexte hétérogène des évaluations serait de considérer d'autres types de parallélisation afin s'adapter en particulier au

développement des moyens de calcul à très grande échelle (millier de processeurs), où le simple fait d'augmenter la taille de la population risque de ne pas suffire. Dans cette direction, une adaptation des AEMO en utilisant le modèle des îlots peut être considérée comme une bonne approche.

Dans un deuxième temps, et toujours dans le cadre de la problématique de réduction du coût global des optimisation, nous nous sommes intéressés à l'utilisation des méta-modèles qui permettent de réduire le nombre total des évaluations de la fonctions objectif en approchant cette dernière (chapitre 4). Dans ce cadre, nous avons proposé une adaptation des algorithmes utilisant les méta-modèles sur le contexte parallèle hétérogène qui représente le cadre applicatif de cette thèse. Les résultats ont d'abord montré que l'utilisation des méta-modèles permet de réaliser un gain considérable par rapport aux algorithmes standard, notamment dans la première partie de l'évolution où l'accélération de convergence est nettement meilleure. De plus, les expérimentations menées avec les algorithmes stationnaires asynchrones à l'aide des modèles artificiels de coût ont consolidé le constat déjà réalisé dans le chapitre 3 en montrant la capacité de ces derniers à maintenir le gain dû au méta-modèles tout en assurant un coût d'optimisation moins élevé grâce à la gestion plus efficace de la grille.
Une piste intéressante pour des travaux futurs sur l'utilisation des méta-modèles dans l'optimisation évolutionnaire multi-objectif serait de comparer la méthode utilisée dans cette thèse qui construit pour chaque objectif un méta-modèle avec la régression SVM avec d'autres approches comme par exemple celle basée sur le rank-SVM [Loshchilov 2010] ou les méthodes basées sur des méta-modèles locaux qui ont déjà été appliqués sur des problématiques mono-objectif avec l'algorithme CMA-ES [Bouzarkouna 2012].

Après avoir validé les approches stationnaires asynchrones sur les benchmarks de fonctions analytiques, nous nous sommes focalisés sur leur applications sur un problème réel de la combustion Diesel qui représentent le contexte industriel de cette thèse en utilisant deux types de modélisation.
Dans un premier temps (chapitre 6), nous avons utilisé un outil de modélisation phénoménologique (0D) de la combustion Diesel. Cet outil, bien qu'il ne permet pas de prendre en compte des aspects importants des phénomènes physiques telle que la géométrie de la chambre, présente, par son temps d'évaluation relativement réduit par rapport à la modélisation 3D, un bon moyen de validation de l'approche stationnaire asynchrone proposée en la comparant avec l'approche générationnelle standard sur une problématique réelle. Les résultats ont montré qu'un gain d'environs 42 % en termes de coût global d'optimisation à été réalisé avec l'approche asynchrone stationnaire grâce à une gestion plus efficace de la grille de calcul (aucun processeur ne reste inoccupé pendant l'optimisation). De plus, ce gain en temps de calcul ne se fait pas au détriment de la qualité des solutions trouvées puisque les deux approches sont comparables de ce point de vue, ce qui consolide les conclusions déjà obtenues sur les benchmarks de fonctions analytiques.

Le deuxième type de modélisation de la combustion Diesel considéré dans ce travail de thèse concerne l'optimisation multidimensionnelle (3D). Compte tenue du coût d'évaluation très élevé qui est de quelques jours, et du contexte hétérogène des coûts des évaluations, l'approche asynchrone stationnaire déjà validée sur les benchmarks de fonctions analytiques et sur une problématique réelle utilisant la modélisation 0D, a été directement appliquée. Le workflow d'optimisation parallèle à été mis en place sur un serveur de calcul interne à PSA. 300 évaluations ont pu être réalisées avec un budget avoisinant les 400000 heures CPU. L'analyse physique des résultats de l'optimisation a permis de dégager un concept intéressant pour la combustion Diesel à charge partielle qui est celui de la chasse inclinée du bol de combustion. En effet, cette forme permet de réduire les émissions des NOx et des suies grâce à un meilleur mélange air/carburant dans la chambre.
A noter que le concept de chasse inclinée a déjà fait l'objet d'une validation par des essais sur banc moteur sur un point de fonctionnement à charge partielle et les résultats des essais ont confirmé les résultats obtenus à travers la simulation numérique. Ces essais découlent d'une série d'essais précédents qui ont été conduits sur le même concept de bol de combustion mais sur un point de fonctionnement à pleine charge pour lesquels un gain a aussi été observé. Il est important de dire que les essais n'ont pas été conduits sur le bol optimal présenté dans ce manuscrit, mais sur un autre bol avec une géométrie semblable (ces essais ont été réalisés en parallèle avec l'optimisation réalisée dans cette thèse). Cependant, les résultats obtenus dans le cadre de cette thèse ont permis d'expliquer par la simulation numérique le gain observé sur le banc moteur, ce qui consolide les conclusions des résultats d'essais et ouvre les portes à de nouvelles investigations futurs.
Une perspective motivant les motoristes serait de réaliser une optimisation qui permet de prendre en compte les spécificités de plusieurs régimes moteurs à la fois. L'étude réalisée dans cette thèse concerne un régime moteur à charge partielle, pour lequel le problème d'optimisation est totalement différent du problème à pleine charge : à titre d'exemple, la PMI (puissance délivrée par la moteur) est un objectif en soi dans la problématique à plein charge alors qu'elle est traitée en tant que contrainte dans la problématique à charge partielle. Il serait intéressant de réfléchir à une reformulation du problème d'optimisation afin de prendre en compte tous les aspects qui peuvent entrer en jeu dans un moteur à des régimes moteur différents.

Une deuxième optimisation utilisant la modélisation 3D et l'algorithme stationnaire asynchrone avec méta-modèle a été mise en place dans le but de la comparer avec l'optimisation 3D déjà réalisée afin de quantifier le gain pouvant être réalisé en utilisant un méta-modèle sur une problématique grandeur nature. Malheureusement, pour des raisons limitantes de temps cette dernière n'était pas terminée à l'échéance du contrat. Cependant, une comparaison préliminaire sur les quelques évaluations réalisées à été effectuée et a révélé une accélération certaine avec l'algorithme utilisant un méta-modèle.
Dans cette direction, une perspective évidente de ce travail de thèse est de réaliser une optimisation avec quelques centaines d'évaluations en utilisant les

méta-modèles, afin de pouvoir la comparer avec l'approche asynchrone stationnaire sur un plus grand budget d'évaluations.

Ce travail de thèse a démontré la capacité de l'optimisation évolutionnaire à résoudre des problèmes réels liés à la vie courante. Dans le cas du moteur diesel, sachant que la chambre de combustion est la source principale des émissions polluantes des véhicules automobiles, ce travail s'inscrit dans la démarche de conception des moteurs automobiles de demain, polluant et consommant moins. La consolidation par des essais sur bancs-moteurs des conclusions physiques obtenues numériquement via l'optimisation constitue un élément important dans la démarche d'intégration du concept de bol à chasse inclinée dans les moteurs futurs de PSA PEUGEOT CITROEN, afin de conforter la position de cette dernière en tant que leader mondial en matière de réductions des émissions polluantes.
Enfin, le développement des capacités de calcul parallèle permettra sans doute d'utiliser l'optimisation évolutionnaire pour résoudre des problèmes considérés aujourd'hui encore comme irréalisables. L'optimisation évolutionnaire est d'ores et déjà prête à jouer pleinement ce rôle.

Bibliographie

[Abramson 1992] D. Abramson et J. Abela. *A parallel genetic algorithm for solving the school timetabling problem.* In In Proceedings of the Fifteenth Australian Computer Science Conference (ACSG-15), Volume 14, pages 1–11, 1992. 29

[Alba 1999] E. Alba et J.M. Troya. *A survey of parallel ditributed genetic algorithms.* Complexity, vol. 4, no. 4, pages 31–52, 1999. 29

[Albrecht 2006] A. Albrecht, J. Chauvin, F.A. Lafossas, S. Potteau et G. Corde. *Development of highly premixed combustion Diesel engine model : from simulation to control design.* SAE Paper No 2006-01-1072, 2006. 75, 77

[Asouti 2009] Varvara G. Asouti et Kyriakos C. Giannakoglou. *Aerodynamic optimization using a parallel asynchronous evolutionary algorithm controlled by strongly interacting demes.* Engineering Optimization, vol. 41, no. 3, pages 241–257, March 2009. 36

[Bethke 1976] A.D. Bethke. *Comparison of genetic algorithms and gradient-based optimizers on parallel processors :Efficiency of use of processing capacity.* Rapport technique, (Tech. Rep. No.197). University of Michigan, 1976. 29

[Beume 2007] Nicola Beume, Boris Naujoks et Michael Emmerich. *SMS-EMOA : Multiobjective selection based on dominated hypervolume.* European Journal of Operational Research, vol. 181, no. 3, pages 1653–1669, 2007. 38

[Bolling 1996] M. Bolling, H. Pitsch, J.C. Hewson et K. Seshadri. *Reduced n-heptane mechanism for non-premixed combustion with emphasis on pollutant-relevant intermediate species.* Symposium (international) on Combustion, vol. 26, pages 729–737, 1996. 81

[Boser 1992] B.E. Boser, I.M. Guyon et V.N. Vapnik. *A training algorithm for optimal margin classifiers.* In In D.Haussler, éditeur, Proceedings of the 5th annual ACM Workshop on Computational Lea rning Theory, pages 144–152. ACM Press, 1992. 58

[Bossert 1967] W. Bossert. *Mathematical optimization : Are there abstract limits on natural selection ?* Mathematical Challenges to the Neo-Darwinian Interpretation of Evolution. Philadelphia, PA : The Wistar Institute Press, pages 35–46, 1967. 31

[Bouzarkouna 2012] Z. Bouzarkouna. *Well placement optimization.* PhD thesis, Université Paris Sud, 2012. 116

[Branke. 2004] J. Branke., H. Schmeck et K. Deb. *Parallelizing Multi-Objective Evolutionary Algorithms : Cone Separation.* In IEEE Congres on Evolutionary Computation 2004, pages 1952–1957, 2004. 36

[Braun 1990] H.C. Braun. *On solving travelling salesman problems by genetic algorithms.* In in Parallel Problem Solving from Nature (PPSN), pages 129–133, 1990. 31

[Brooker 1998] A.J. Brooker, J. Dennis, P.D. Frank, D.B. Serafini adn V. Torczon et M. Trosset. *A rigourous framework for optimization of expensive functions by surrogates.* Structural Optimization, 17, pages 1–13, 1998. 57

[Cantu-Paz 2000] E. Cantu-Paz. Efficient and accurate parallel genetic algorithms. Kluwer Academic Pub, 2000. 29

[Chafekar 2003] Deepti Chafekar, Jiang Xuan et Khaled Rasheed. *Constrained Multi-objective Optimization Using Steady State Genetic Algorithms.* In Cantú-Paz, Erick et al., editeur, Genetic and Evolutionary Computation, volume 2723, pages 201–201. Springer Verlag, 2003. 47

[Choi 2002] C.Y. Choi et R.D.Reitz. *An experimental study on the effects of oxygenated fuel blends and multiple injection strategies on DI diesel engine emissions* . Fuel 78-11, pages 1303–1317, 2002. 98

[Coello 2002] C. A. Coello Coello, D. A. Van Veldhuizen et G. B. Lamont. Evolutionary algorithms for solving multi-objective problems. Kluwer Academic Publishers, 2002. 1

[Deb 1995] K. Deb. Optimization for engineering design : Algorithms and examples. New Delhi : Prentice-Hall, 1995. 9

[Deb 2001] Kalyanmoy Deb. Multi-objective optimization using evolutionary algorithms. Wiley, Chichester, UK, 2001. 1, 14

[Deb 2002] K. Deb, A. Pratap, S. Agarwal et T. Meyarivan. *A fast and elitist multiobjective genetic algorithm : NSGA-II.* IEEE Transactions on Evolutionary Computation, vol. 6, pages 182–197, 2002. 18, 19, 84

[Deb. 2003a] K. Deb., P. Zope et A. Jain. *Distributed Computing of Pareto-Optimal Solutions with Evolutionary Algorithms.* In C. Fonseca et al., editeur, Evolutionary Multi-objective Optimization – EMO'03, pages 534–549. LNCS 2632, Springer Verlag, 2003. 36

[Deb 2003b] Kalyanmoy Deb, Manikanth Mohan et Shikhar Mishra. *Towards a Quick Computation of Well-Spread Pareto-Optimal Solutions.* In C. Fonseca et al., editeur, EMO'03, pages 222–236. LNCS 2632, Springer Verlag, 2003. 22, 38

[Donateo 2006] T. Donateo, D. Laforgia, G. Aloisio et S. Mocavero. *Evolutionary Algorithm as a Tool for Advanced Designing of Diesel Engines.* International Journal of Computational Intelligence Research, pages 169–180, 2006. 95

[Durillo 2008] J.J. Durillo, A.J. Nebro, F. Luna et E. Alba. *A Study of Master-slave Approaches to parallelize NSGA-II.* In Proc. IEEE International Symposium on Parallel and Distributed Processing, pages 1–8, 2008. 38, 47

[Enaux 2010] B. Enaux. *Simulation aux grandes échelles d'un moteur à allumage commandé - Évaluations des variabilités cycliques.* PhD thesis, Institut national polytechnique de Toulouse, 2010. 76

[Fogel 1966] L.J. Fogel, A.J. Owens et M.J. Walsh. Artificial intelligence through simulated evolution. New York : John Wiley, 1966. 11

[Fonseca 1995] C.M. Fonseca et P.J. Fleming. *An overview of evolutionary algorithms in multi-objective optimization.* Evolutionary Computation, vol. 3, no. 1, pages 1–16, 1995. 24

[Forgarty 1991] T.C. Forgarty et R. Huang. *Implementing the genetic algorithm on transuter based parallel processing systems.* In in Parallel Problem Solving from Nature (PPSN), pages 145–149, 1991. 29

[Giunta 1998] A.A. Giunta et L. Watson. *A comparison of approximation modeling techniques : Polynomial versus interpolating models.* Rapport technique, Technical Report 98-4758 AIAA, 1998. 57

[Goldberg 1989] D. E. Goldberg, B. Korb et K.Deb. *Messy genetic algorithms : Motivation, analysis and first results.* Complex Systems, vol. 3, pages 493–530, 1989. 11, 19

[Grefenstette 1981] J.J. Grefenstette. *Parallel adaptive algorithms for function optimization.* Rapport technique, (Tech. Rep. No.1CS-81-19). Nashville, TN : Vanderblit University, Computer Science Department, 1981. 29

[Grosso 1985] P.B. Grosso. *Computer simulations of genetic adaptation : Parallel subcomponent interaction in a multilocus model.* PhD thesis, Unpublished doctoral dissertation, The University of Michigan, 1985. 31

[Haimes 1986] Y.Y Haimes, L.S. Lasdon et D.A. Wismer. *On a bicriterion formulation of the problems of integrated system identification and system optimization.* IEEE Trans. on systems, Man of Cybernetics, pages 122–128, 1986. 11

[Hansen 1998] M. P. Hansen et A.Jaszkiewicz. *Evaluating the quality of approxmations to the non-dominated set.* Rapport technique, Technical Report IMM-REP-1998-7, Technical University of Denmark, 1998. 16

[Hansen 2001] N. Hansen et A. Ostermeier. *Completely Derandomized Self-Adaptation in Evolution Strategies.* Evolutionary Computation, vol. 9, no. 2, pages 159–195, 2001. 23

[Hauser 1994] R. Hauser et R. Manner. *Implementation of standard genetic algorithm on MIMD machines.* In in Parallel Problem Solving from Nature (PPSN III), pages 504–513, 1994. 29

[Heywood 1988] J.B. Heywood. Internal combustion engine fundamentals. McGraw-Hill Book Company, 1988. 78, 99

[Hiroyasu 2002] T. Hiroyasu, M. Liki, J. Kamiura, S. Watanabe et H.Hiroyasu. *Multi-Objective Optimization of Diesel Engine Emissions and Fuel Economy Using Genetic Algorithms and Phenomenological Model.* SAE paper, 2002. 78

[Hiroyasu 2003] H. Hiroyasu, H. Miao, T.Hiroyasu, M.Miki, J. Kamiura et S. Watanabe. *Genetic algorithms Optimization of Diesel Engine Emissions and Fuel Efficiency with Air Swirl, EGR, Injection Timing and Multiple Injections.* SAE paper, 2003. 78

[Holland 1975] J.H. Holland. Adaptation in natural and artificail systems. University of Michigan Press, 1975. 11

[Igel 2007a] C. Igel, N. Hansen et S. Roth. *Covariance Matrix Adaptation for Multi-objective Optimization.* Evolutionary Computation, vol. 15, no. 1, pages 1–28, 2007. 18, 23, 25, 41

[Igel 2007b] C. Igel, T. Suttorp et N. Hansen. *Steady-state selection and efficient covariance matrix update in MO-CMA-ES.* In S. Obayashi et al., editeur, EMO'07, pages pp.171–185. LNCS 2632, Springer Verlag, 2007. 37, 48

[I.Rechenberg 1973] I.Rechenberg. Evolution strategie : Optimierung technisher systeme nach prinzipien des biologischen evolution. Fromman-Hozlboog Verlag, Stuttgart, 1973. 12

[Janicka 1977] J. Janicka, W. Kolbe et W. Kollmann. *Closure of the transport equation for the probability density function of turbulent scalar fields.* J. Non-Equilib. Thermodyn. 4, 47-66, 1977. 79, 82

[Jin 2005] Yaochu Jin. *A Comprehensive Survey of Fitness Approximation in Evolutionary Computation.* Soft Computing, pages 3–12, 2005. 55, 56

[Knowles 2006] J. Knowles, L.Thiele et E. Zitzler. *A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers.* TIK repport 214, (TIK), ETH Zurich, 2006. 16

[Koza 1992] J.R. Koza. Genetic programming : on the programming of computers by means of natural selection. MIT Press, Cambridge, MA, USA, 1992. 12

[Koza 1994] J.R. Koza. Genetic programming ii : Automatic discovery of reusable programs. MIT Press, Cambridge, MA, USA, 1994. 12

[Krige 1951] D.G. Krige. *A statistical approach to some basic mine valuation problems on the Witwatersrand.* J, of Chem.,Metal. and Mining Soc. of South Africa, pages 119–139, 1951. 57

[Kursawe 1990] F. Kursawe. *A variant of evolution strategies for vector optimization.* In in Parallel Problem Solving from Nature I (PPSN I), pages 193–197, 1990. 25

[Lafossas 2007] F.A. Lafossas, M. Marbaix et P.Menegazzi. *Development and application of a 0D D.I. Diesel cmbustion model for emissions prediction.* SAE Paper No 2007-01-1841, 2007. 75, 77

[Laumanns 2002] M. Laumanns, L. Thiele, K.Deb et E.Zitzler. *Combining Convergence and Diversity in Evolutionary Multiobjective Optimization.* Evolutionary Computation, vol. 10, pages 263–282, 2002. 22

[Law 2006] C.K. Law. Combustion physics. Cambridge University Press, 2006. 78

[Loshchilov 2010] I. Loshchilov, M. Schoenauer et M. Sebag. *Dominance-Based Pareto-Surrogate for Multi-Objective Optimization.* In R. Takahashi et al., editeur, Simulated Evolution and Learning (SEAL 2010), pages 230–239. LNCS 6457, Springer Verlag, December 2010. 56, 116

[Loshchilov 2011] I. Loshchilov, M. Schoenauer et M. Sebag. *Not all parents are equal for MO-CMA-ES.* In Ricardo H. C. Takahashi et al., editeur, Proc. EMO'2011, LNCS, pages 31–45. Springer Verlag, 2011. 37

[Lutz 1988] A. Lutz, R. Kee et J. Miller. *Senkin : a fortran program for predicting homogeneous gas phase chemical kinetics with sensitivity analysis.* Rapport technique, Tech.Rep.SAND87-8248.UC-4. Sandia National Laboratories report - SAND-8248, 1988. 81

[Matheron 1963] G. Matheron. *Principles of Geostatics.* Economic Geol, pages 1246–1268, 1963. 57

[Pettey 1992] C.B. Pettey. *A massively distributed parallel genetic algorithm (mdpGA).* Rapport technique, Tech.Rep.No. CMU-CS-92-196R). Pittsburgh, PA : Carnegie Mellon University, 1992. 31

[Pettey 1997] C.B. Pettey. *Population structures : diffusion (cellular) models.* Rapport technique, Handbook of evolutionary computation (pp. C6.4 :1-C6.4 :6). Bristol and New York : Institute of Physics Publishing and Oxford University Press., 1997. 31

[Pilát 2011] Martin Pilát et Roman Neruda. *ASM-MOMA : Multiobjective memetic algorithm with aggregate surrogate model.* In IEEE Congress on Evolutionary Computation, pages 1202–1208, 2011. 56

[Pischinger 1988] F. Pischinger, H. Schutle et J. Hansen. *The Diesel engine's future.* VDI-Congress, Wolfsburg, 1988. viii, 75

[Schaffer 1984] J.D. Schaffer. *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms.* PhD thesis, PH.D. Thesis, Nashville,TN :Vanderbilt University, 1984. 24

[Schoenauer 2003] M. Schoenauer. *Les Algorithmes évolutionnaires : état de l'art et enjeux.* Rapport technique, Algorithms seminar 2001-2002 INRIA(2003), pp 113-118.0, 2003. vii, 13

[Scholkopf 2002] B. Scholkopf et A.J. Smola. Learning with kernels. MIT Press, 2002. viii, 59

[Schott 1995] J. R. Schott. *Fault Tolerant Design Using Single and Multi-Criteria Genetic Algorithms.* Rapport technique, Master's Thesis, Boston, MA : MIT, 1995. 16

[Senecal 2000] P.K. Senecal, D.T. Montgomery et R.D. Reitz. *A methodology for engine design using multi-dimensional modelling and genetic algorithms with validation through experiments.* International Journal of Engine Research, vol. 1, 2000. 95

[Silverman 1986] B.W. Silverman. Density estimation for statistics and data analysis. London : Chapman and Hall, 1986. 18

[Smola 2004] A.J. Smola et B. Scholkopf. *A tutorial on support vector regression.* Statistics and Computing, pages 199–222, 2004. 58

[Srinvas 1994] N. Srinvas et K.Deb. *Multi-Objective function optimization using non-dominated sorting genetic algorithm.* Evolutionary Computation, vol. 2, pages 221–248, 1994. 19

[Subramaniam 2004] S. Subramaniam et S.B. Pope. *A mixing model for turbulent reactive flows based on Euclidean Minimum Spanning Trees.* Combustion and Flame, vol 117, pages 270–282, 2004. 80

[Syswerda 1991] G. Syswerda. *A study of reproduction in generational and steady state genetic algorithm.* In G. J. E. Rawlins, editeur, Foundations of Genetic Algorithms, pages 94–101. Morgan Kaufmann, 1991. 15

[Tan 2002] K.C. Tan, T.H. Lee et E.F. Khor. *Evolutionary algorithms for multi-objective optimization : performance assessments and comparisons.* Artificial Intelligence Review, vol 17n no.4, pages 251–290, 2002. 26

[Tanaka 2002] T. Tanaka, A. Ando et K.Ishizaka. *Study on pilot injection of DI Diesel engine using common-rail injection system.* JSAE Rewiew, no.23, pages 297–302, 2002. 98

[Thobois 2006] L. Thobois. *Intérêt et faisabilité de la simulation aux grandes échelles dans les moteurs automobiles.* PhD thesis, Institut national polytechnique de Toulouse, 2006. 76

[Tomassini 1999] M. Tomassini. *Parallel and distributed evolutionary algorithms : A review.* In In Evolutionary algorithms in Engineering and Computer Science . Chichester, UK : J.Wiley and Sons., pages 113–133, 1999. 31

[Tsao 1990] K.C. Tsao, Y. Dong et Y. Xu. *Investigation of Flow Field a,d Fuel Spray in a Direct-Injection Diesel Engine via Kiva-II Program.* SAE Paper 901616, 1990. 95

[Vapnik 1995] V. Vapnik. The nature of statistical learning theory. Springer, New York, 1995. 58, 59

[Veldhuizen 1999] D.V. Veldhuizen. *Multiobjective Evolutionary Algorithms : Classifications, analyses, and New Innovations.* PhD thesis, Dayton, OH, 1999. 16

[Voutchkov 2006] I. Voutchkov et A.Keane. *Multiobjective Optimization Using surrogates.* In In I.Parmee,editor,ACDM'06, pages 167–175. Institute for People-centred Computation, 2006. 56

[Weber 1998] J. Weber, N.peters, A.Pawlowski, R.Kneer, d S.H.EL Tahry C.A.Hergart a et A.Lippert. *Diesel Spray Characterization Using a Micro-Genetic Algorithm and Optical Measurements.* SAE Paper 790833, 1998. 95

[Whitley 1993] D. Whitley. *Cellular genetic algorithms.* In Proceedings of the Fifth International Conference on Genetic Algorithms (pp.658), 1993. 31

[Woschni 1998] G Woschni. *A Universally Applicable Equation for the Instantaneous Heat Transfer in the Cylinder of a High Speed Diesel engine.* SAE Paper 790833, 1998. 80

[Yagoubi 2010] M. Yagoubi, L. Thobois et M. Schoenauer. *An Asynchronous Steady-state NSGA-II Algorithm for Multi-Objective Optimization of Diesel Combustion.* In H. Rodrigues et al., editeur, Abstract. Proc. 2nd Intl Conf. on Engineering Optimization – EngOpt'2010, 2010. 77

[Yagoubi 2011] M. Yagoubi, L. Thobois et M.Schoenauer. *Asynchronous Evolutionary Multi-Objective algorithms with heterogeneous evaluation costs.* In IEEE Congress on Evolutionary Computation, pages 21–28, 2011. 35

[Yagoubi 2012] M. Yagoubi et M.Schoenauer. *Asynchronous Master/Slave MOEAs and heterogeneous evaluation costs.* In GECCO 2012 To Appear, 2012. 35

[Zhang 1995] L. Zhang, T. Ueda, T. Takatsuki et Y. Yokota. *A Study of the Effect of Chamber Geometries on Flame Behavior in a DI Diesel Engine.* SAE Paper 952512, 1995. 95

[Zhao 2003] F. Zhao, T.N. Asmus, D.N. Assanis, J.E. Dec, J.A. Eng et P.M. Najt. Homogeneous charge compression ignition (hcci) engines. SAE International, 2003. 90

[Zitzler 1998] E. Zitzler et L. Thiele. *An Evolutionary Approach for Multiobjective Optimization : The Strength Pareto Approach.* TIK Report 43, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Mai 1998. 20

[Zitzler 1999a] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization : Methods and Applications.* PhD thesis, ETH Zurich, Switzerland, 1999. 16

[Zitzler 1999b] E. Zitzler et L. Thiele. *Multiobjective Evolutionary Algorithms : A Comparative Case Study and the Strength Pareto Approach.* IEEE Trans. on Evolutionary Computation, vol. 3, no. 4, pages 257–271, 1999. 16, 41

[Zitzler 2000] E. Zitzler, K. Deb et L. Thiele. *Comparison of Multiobjective Evolutionary Algorithms : Empirical Results.* Evolutionary Computation, vol. 8, no. 2, pages 173–195, 2000. 25, 26, 41

[Zitzler 2001] Eckart Zitzler, Marco Laumanns et Lothar Thiele. *SPEA2 : Improving the Strength Pareto Evolutionary Algorithm.* Rapport technique 103, TIK, ETH Zurich, 2001. 18, 20

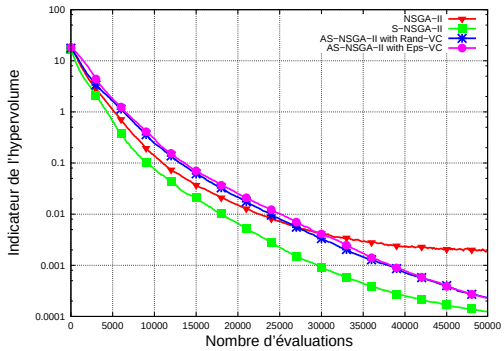
[Zitzler 2003] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca et V. Grunert da Fonseca. *Performance Assessment of Multiobjective Optimizers : An Analysis and Review.* IEEE Transactions on Evolutionary Computation, vol. 7, no. 2, pages 117–132, 2003. 16, 41

[Zitzler 2004] Eckart Zitzler et Simon Künzli. *Indicator-Based Selection in Multiobjective Search.* In Xin Yao et al., editeur, PPSN VIII, pages 832–842. LNCS 3242, Springer-Verlag, 2004. 21

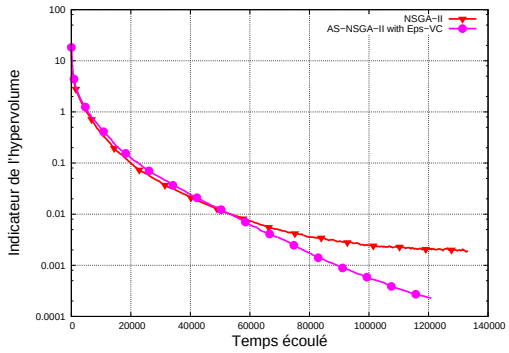
Annexe A

Courbes des expérimentations réalisées avec le modèle artificiel "Eps-VC"

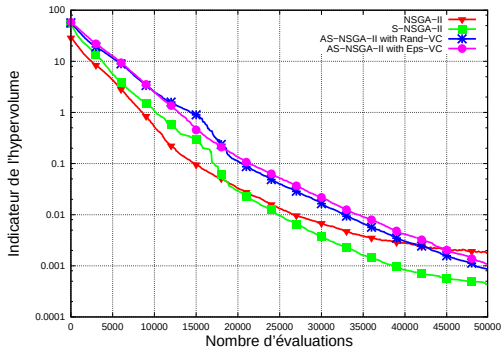
Danc cette première annexe, nous rapportons les courbes des expérimentations relatives au modèle d'hétérogénéité Eps-VC sur les différentes benchmarks de l'état de l'art (ZDT et IHR)



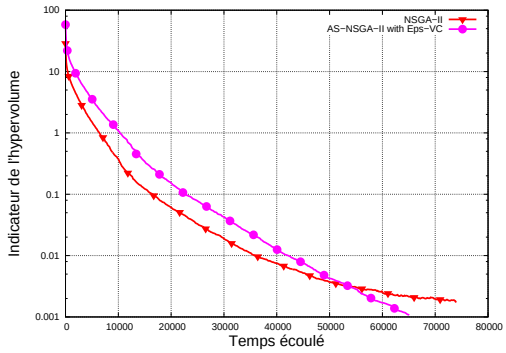
(a) ZDT1-NSGA-II



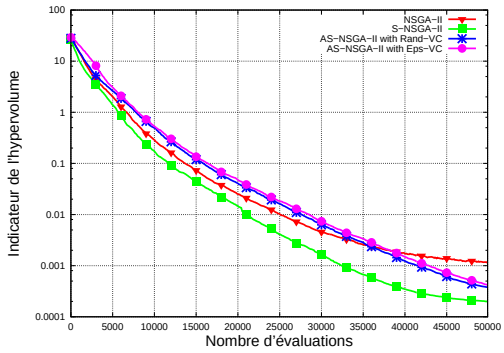
(b) ZDT1-NSGA-II



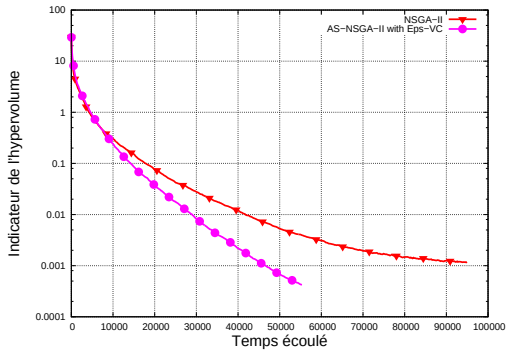
(c) ZDT2-NSGA-II



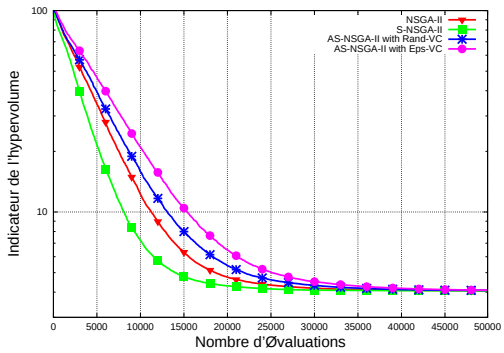
(d) ZDT2-NSGA-II



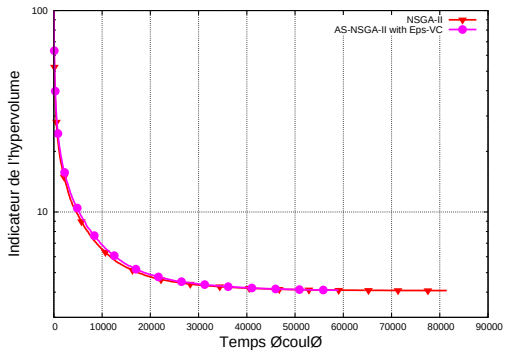
(e) ZDT3-NSGA-II



(f) ZDT3-NSGA-II

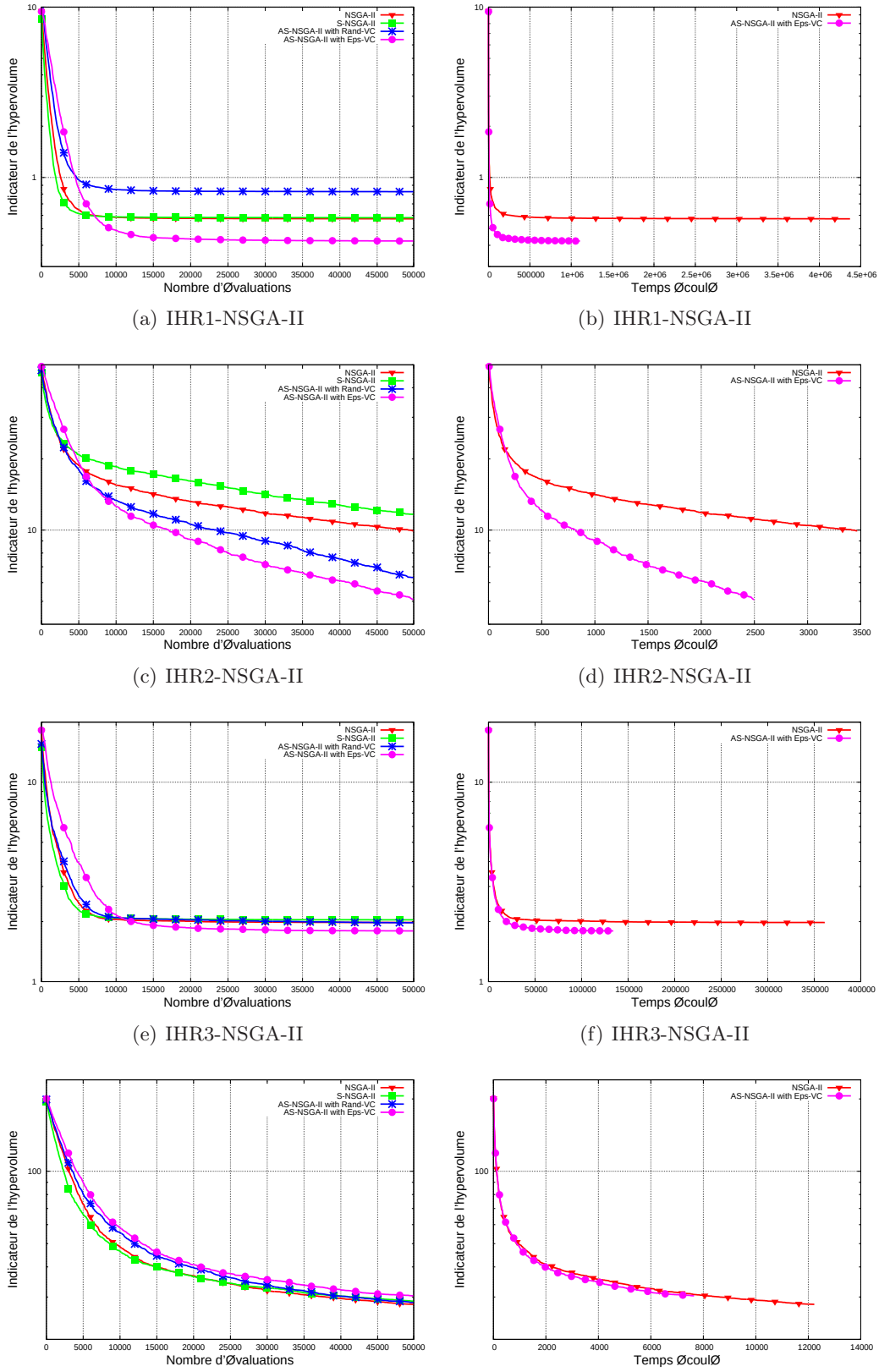


(g) ZDT6-NSGA-II



(h) ZDT6-NSGA-II

FIGURE A.1 – Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de NSGA-II avec le benchmark Eps-VC-ZDT



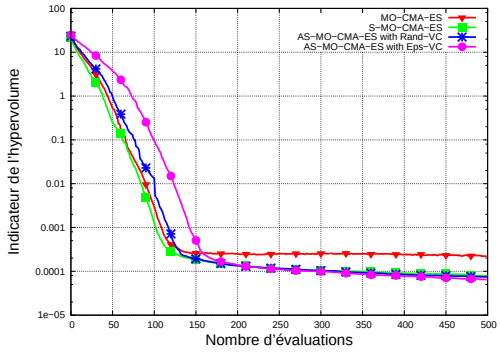
(a) IHR1-NSGA-II (b) IHR1-NSGA-II

(c) IHR2-NSGA-II (d) IHR2-NSGA-II

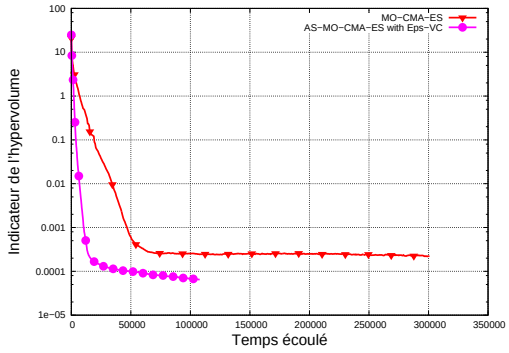
(e) IHR3-NSGA-II (f) IHR3-NSGA-II

(g) IHR6-NSGA-II (h) IHR6-NSGA-II

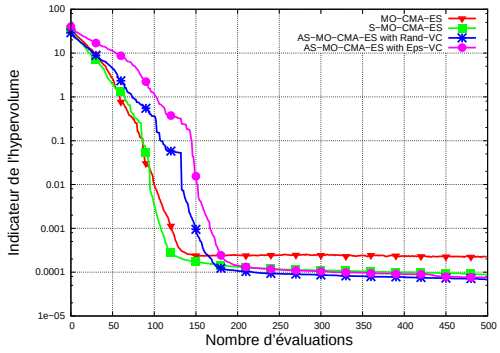
FIGURE A.2 – Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de NSGA-II avec le benchmark Eps-VC-IHR



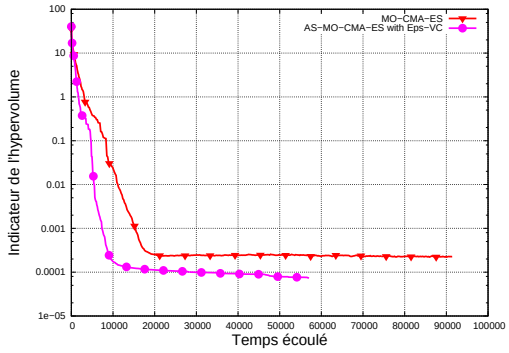
(a) ZDT1-MOCMAES



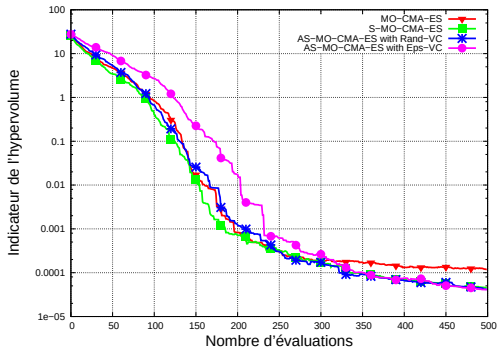
(b) ZDT1-MOCMAES



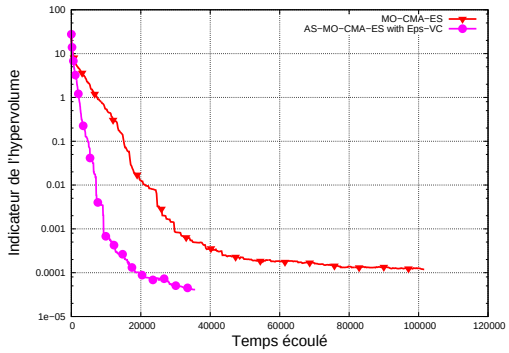
(c) ZDT2-MOCMAES



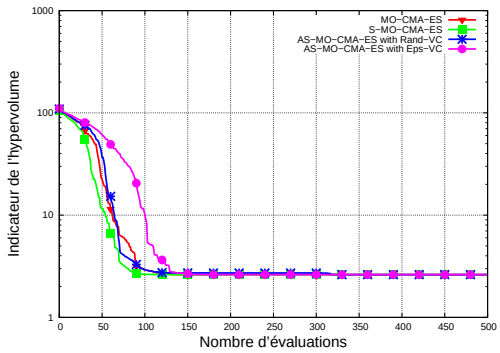
(d) ZDT2-MOCMAES



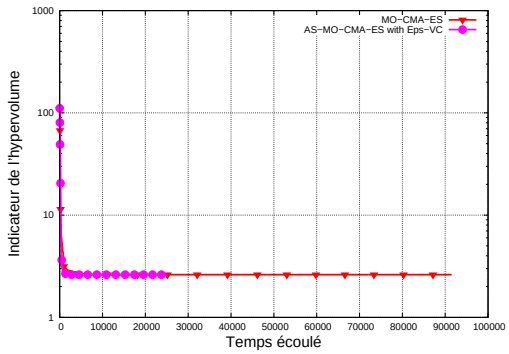
(e) ZDT3-MOCMAES



(f) ZDT3-MOCMAES

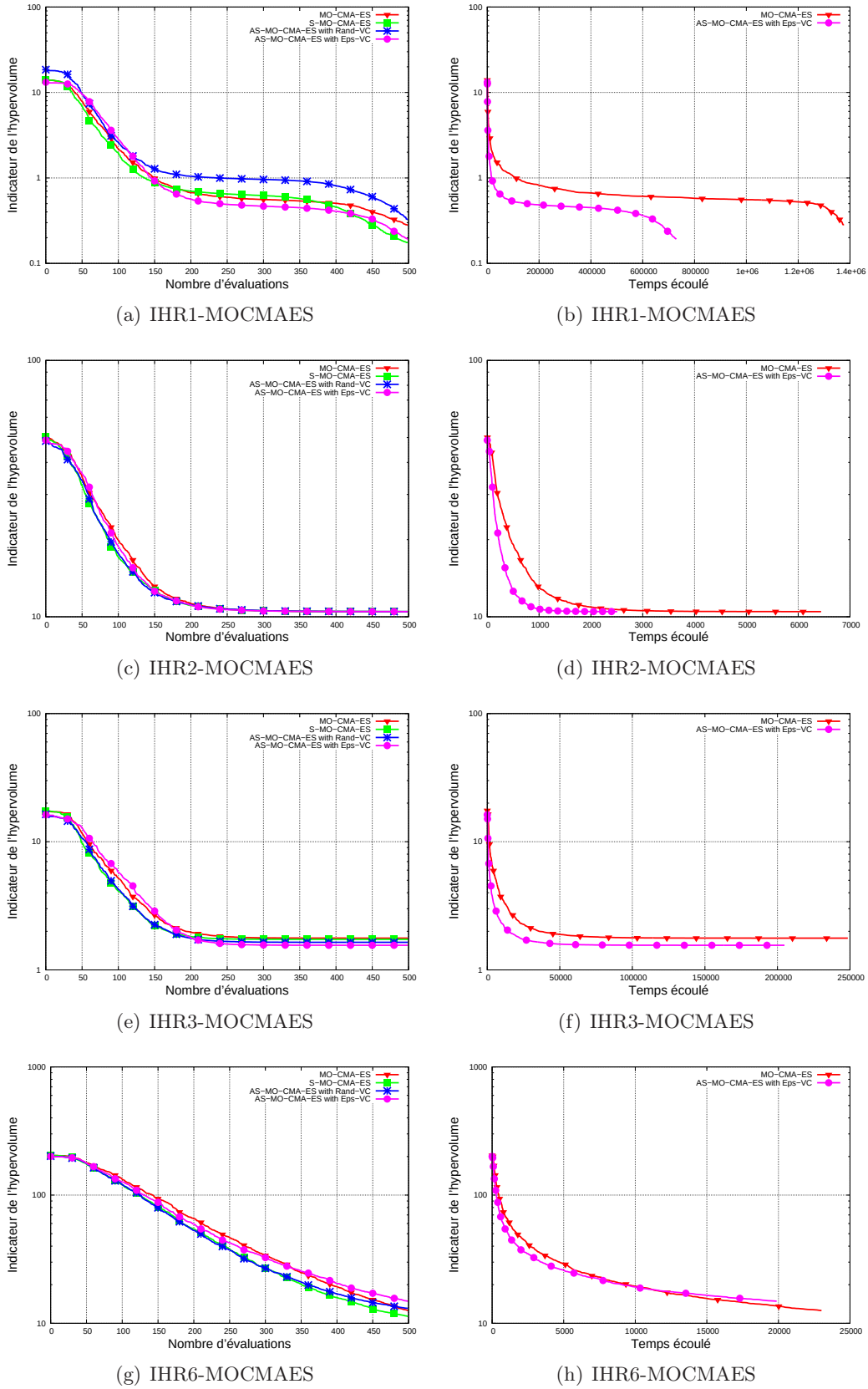


(g) ZDT6-MOCMAES



(h) ZDT6-MOCMAES

FIGURE A.3 – Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de MO-CMA-ES avec le benchmark Eps-VC-ZDT



(a) IHR1-MOCMAES (b) IHR1-MOCMAES

(c) IHR2-MOCMAES (d) IHR2-MOCMAES

(e) IHR3-MOCMAES (f) IHR3-MOCMAES

(g) IHR6-MOCMAES (h) IHR6-MOCMAES

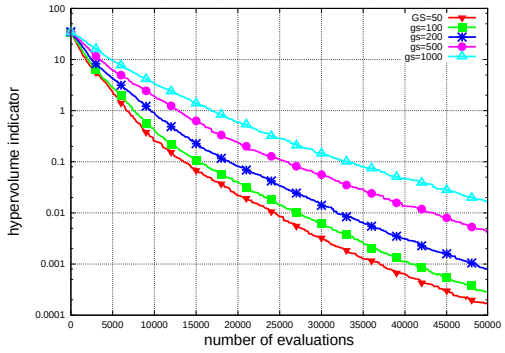
FIGURE A.4 – Evolution de l'indicateur de l'hypervolume moyen en termes de nombre d'évaluations et de temps écoulé pour les variantes de MO-CMA-ES avec le benchmark Eps-VC-IHR

Annexe B

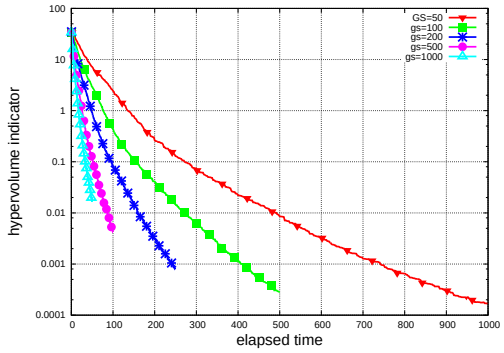
Courbes des expérimentations sur l'accélération avec différentes tailles de grille de calcul

B.1 accélération en fonctions de la taille de la grille

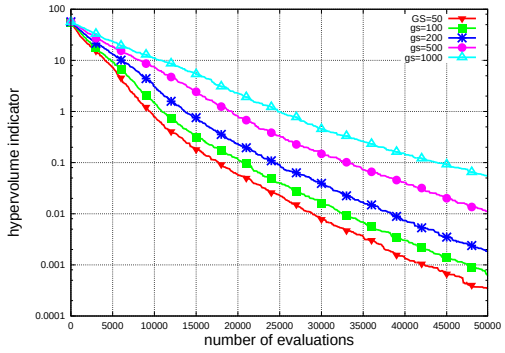
Danc cette deuxième annexe, nous rapportons les courbes des expérimentations relatives à l'accélération due aux différentes tailles de grille de calcul réalisées sur les benchmarks de l'état de l'art (ZDT et IHR).



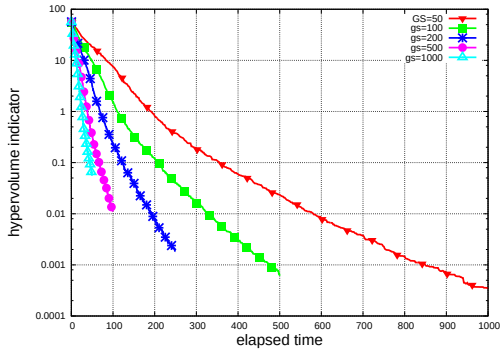
(a) ZDT1-NSGA-II



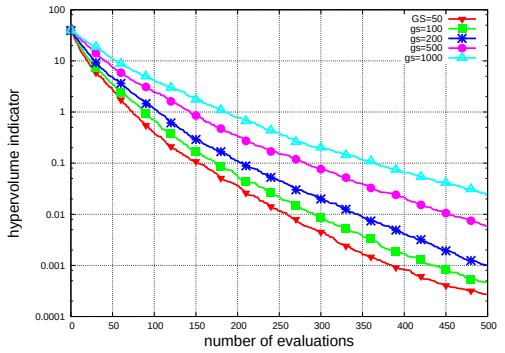
(b) ZDT1-NSGA-II



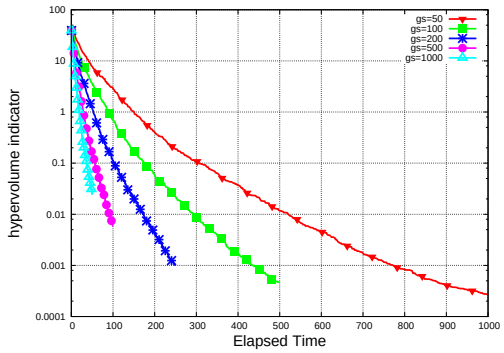
(c) ZDT2-NSGA-II



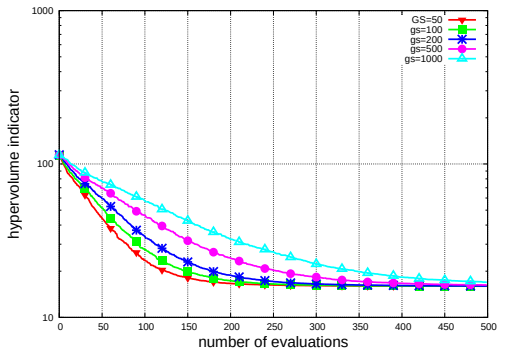
(d) ZDT2-NSGA-II



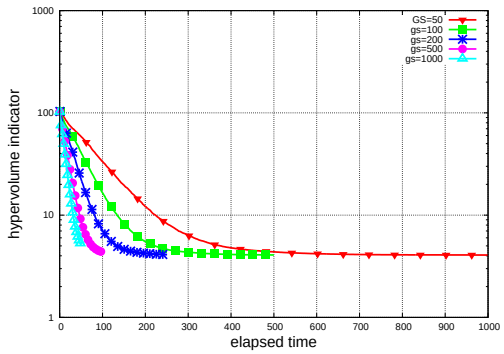
(e) ZDT3-NSGA-II



(f) ZDT3-NSGA-II

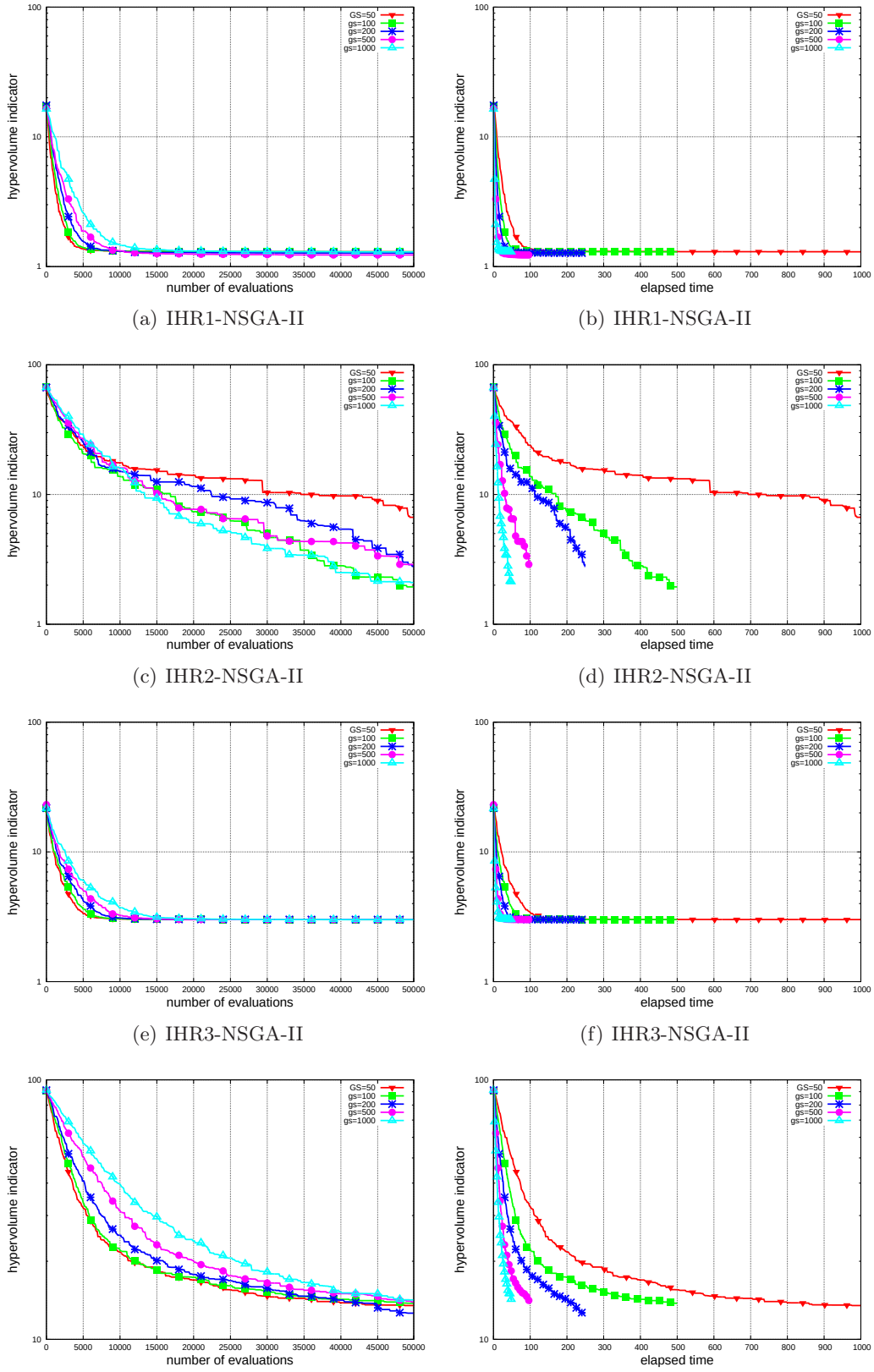


(g) ZDT6-NSGA-II



(h) ZDT6-NSGA-II

FIGURE B.1 – Evolution de l'indicateur de l'hypervolume moyen pour AS-NSGA-II avec le benchmark Rand-VC-ZDT pour différentes tailles de queue d'attente



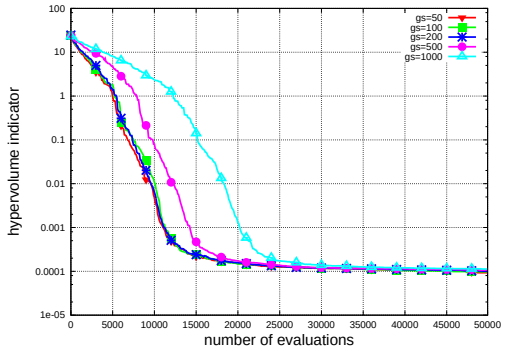
(a) IHR1-NSGA-II (b) IHR1-NSGA-II

(c) IHR2-NSGA-II (d) IHR2-NSGA-II

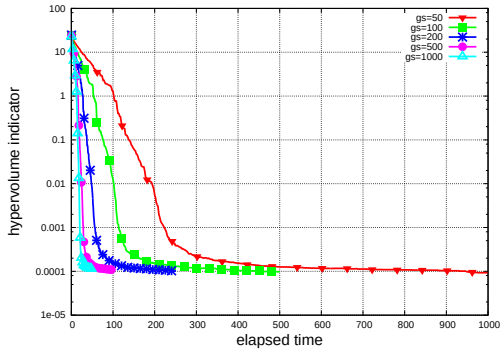
(e) IHR3-NSGA-II (f) IHR3-NSGA-II

(g) IHR6-NSGA-II (h) IHR6-NSGA-II

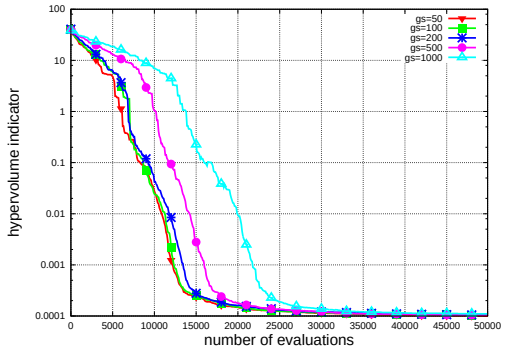
FIGURE B.2 – Evolution de l'indicateur de l'hypervolume moyen pour AS-NSGA-II avec le benchmark Rand-VC-IHR pour différentes tailles de queue d'attente



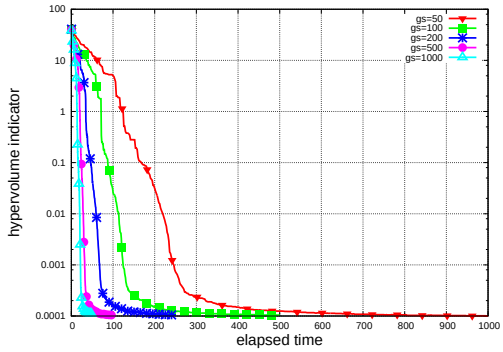
(a) ZDT1-MOCMAES



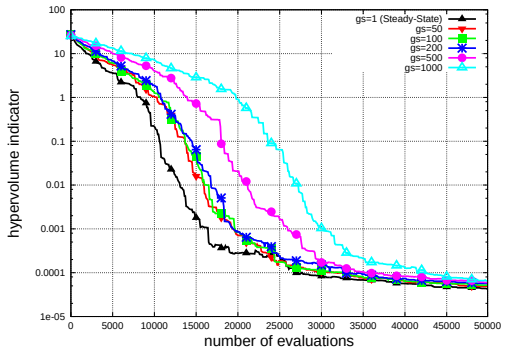
(b) ZDT1-MOCMAES



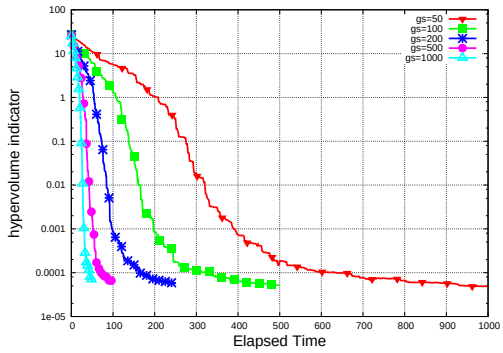
(c) ZDT2-MOCMAES



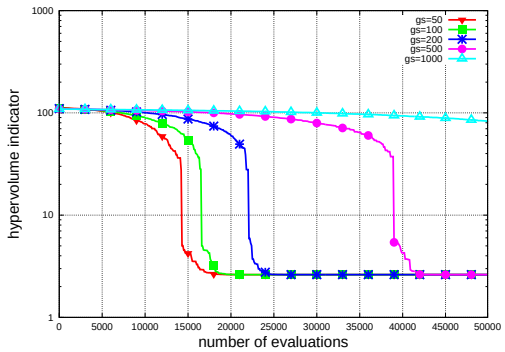
(d) ZDT2-MOCMAES



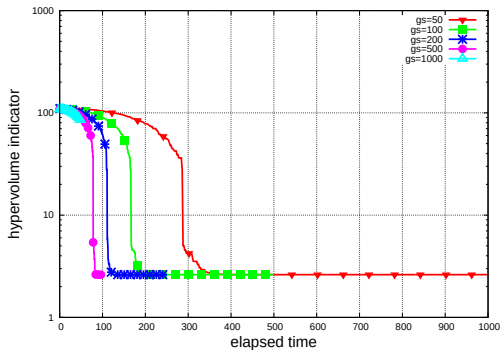
(e) ZDT3-MOCMAES



(f) ZDT3-MOCMAES

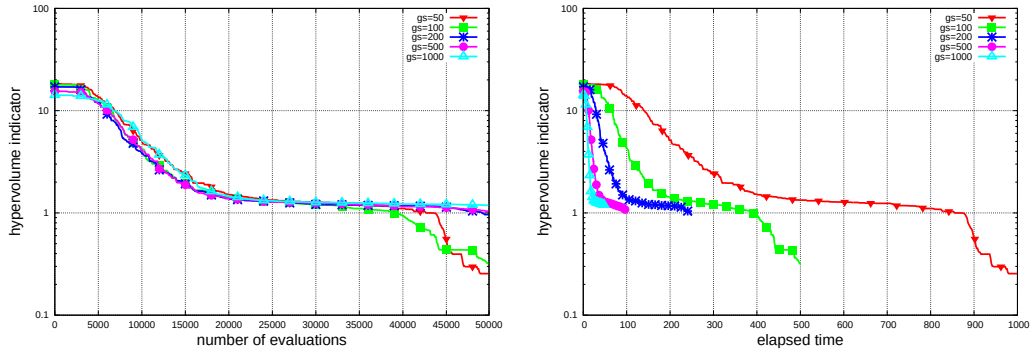


(g) ZDT6-MOCMAES



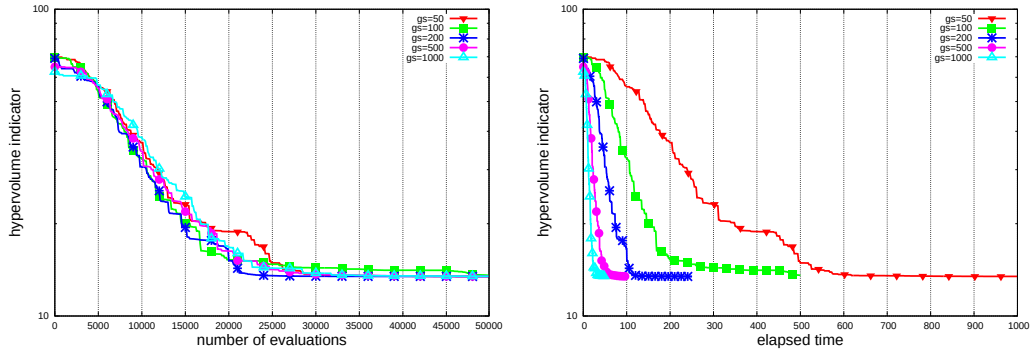
(h) ZDT6-MOCMAES

FIGURE B.3 – Evolution de l'indicateur de l'hypervolume moyen pour AS-MO-CMA-ES avec le benchmark Rand-VC-ZDT pour différentes tailles de queue d'attente



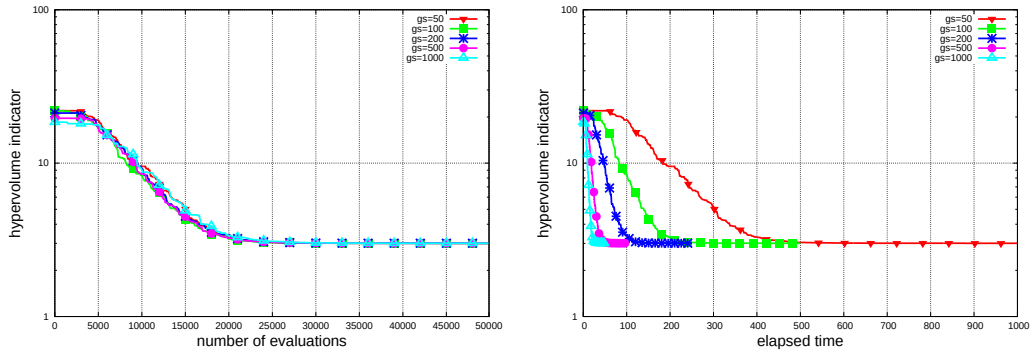
(a) IHR1-MOCMAES

(b) IHR1-MOCMAES



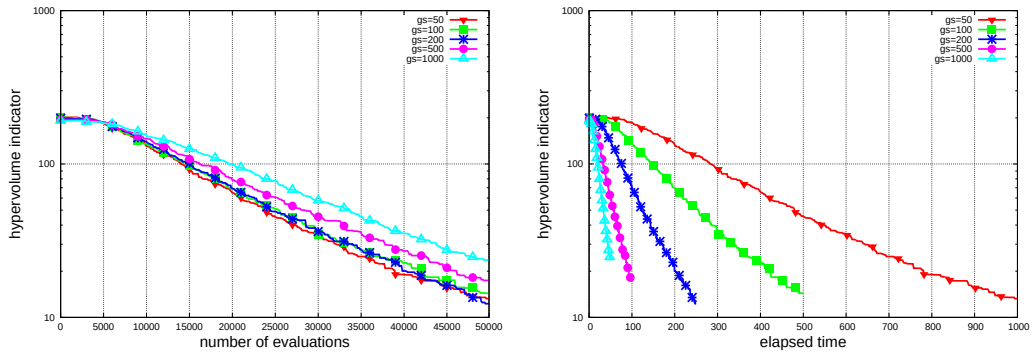
(c) IHR2-MOCMAES

(d) IHR2-MOCMAES



(e) IHR3-MOCMAES

(f) IHR3-MOCMAES



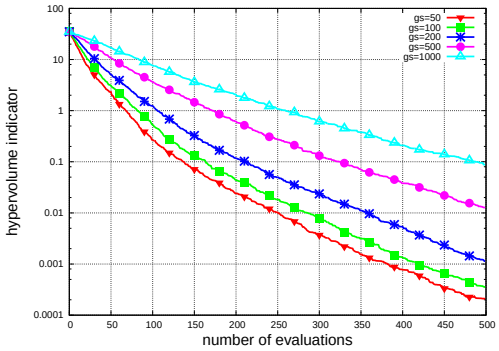
(g) IHR6-MOCMAES

(h) IHR6-MOCMAES

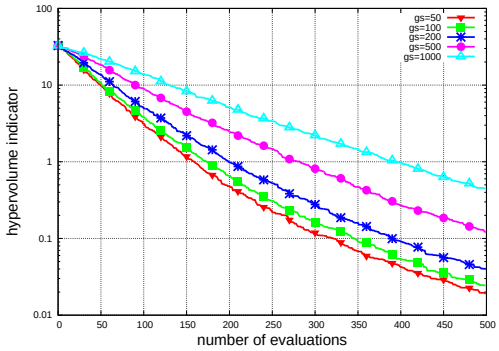
FIGURE B.4 – Evolution de l'indicateur de l'hypervolume moyen pour AS-MO-CMA-ES avec le benchmark Rand-VC-IHR pour différentes tailles de queue d'attente

B.2 Courbes des expérimentations réalisées avec le modèle séquentiel

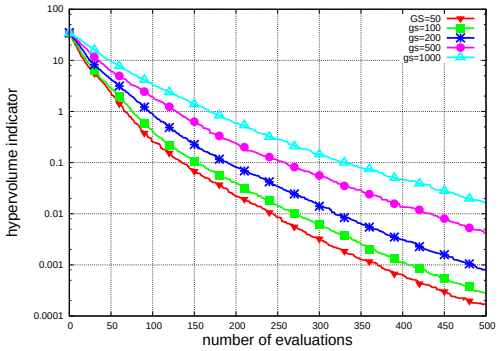
Dans cette deuxième série de courbes, nous comparons les courbes des expérimentations obtenues avec le modèle Rand-VC avec ceux obtenus avec le modèle séquentiel, ceci pour deux tailles de population différentes 100 et 500.



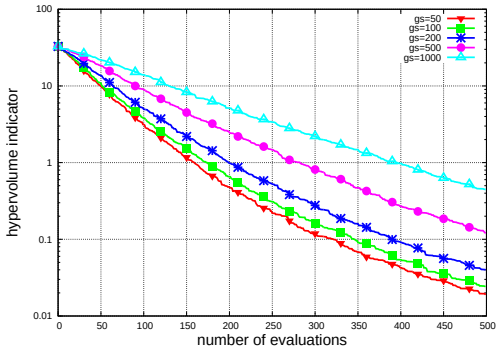
(a) ZDT1-SEQ-pop100-NSGA-II



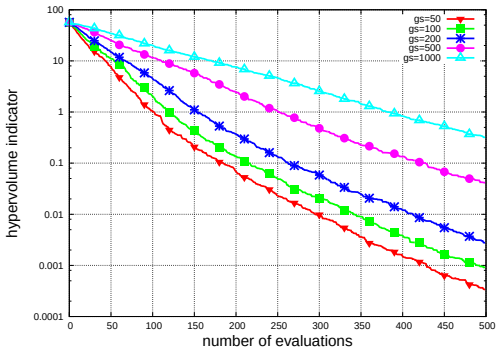
(b) ZDT1-SEQ-pop500-NSGA-II



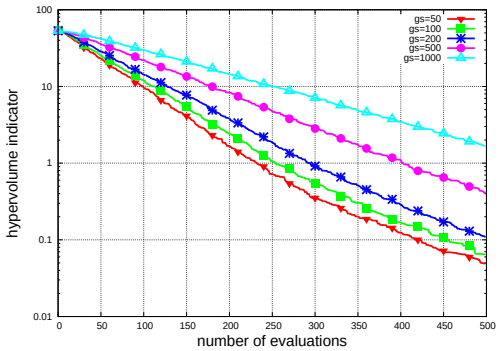
(c) ZDT1-ASSYNC-pop100-NSGA-II



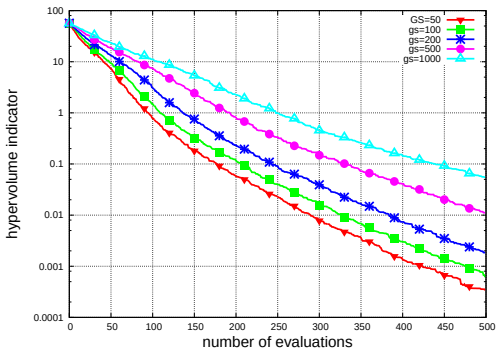
(d) ZDT1-ASSYNC-pop500-NSGA-II



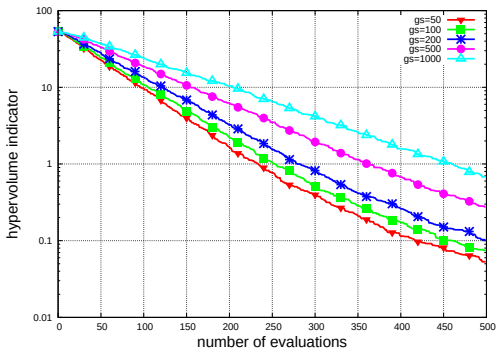
(e) ZDT2-SEQ-pop100-NSGA-II



(f) ZDT2-SEQ-pop500-NSGA-II

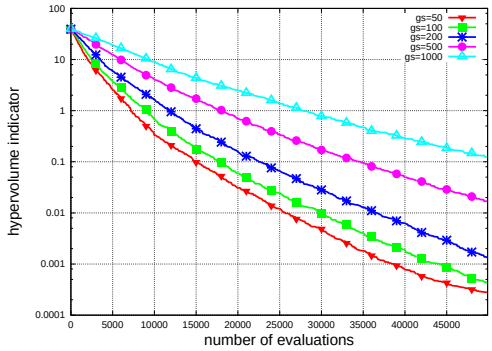


(g) ZDT2-ASSYNC-pop100-NSGA-II

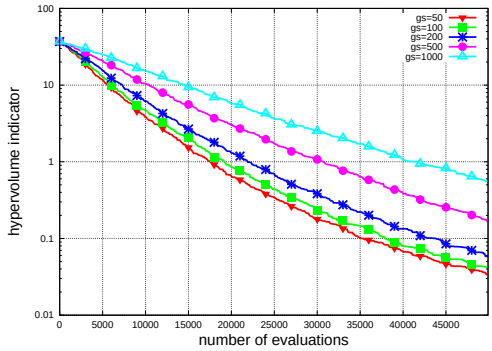


(h) ZDT2-ASSYNC-pop500-NSGA-II

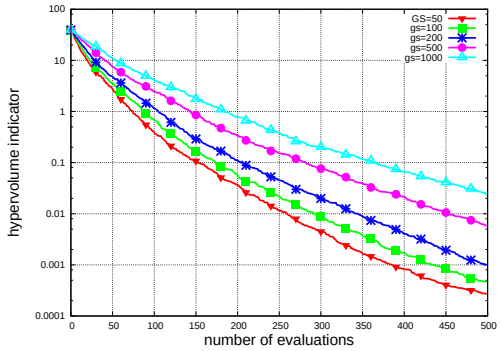
FIGURE B.5 – Évolution de l'indicateur de l'hypervolume moyen pour ZDT1 et ZDT2 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500



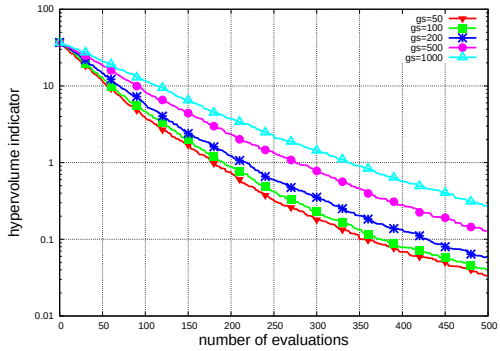
(a) ZDT3-SEQ-pop100-NSGA-II



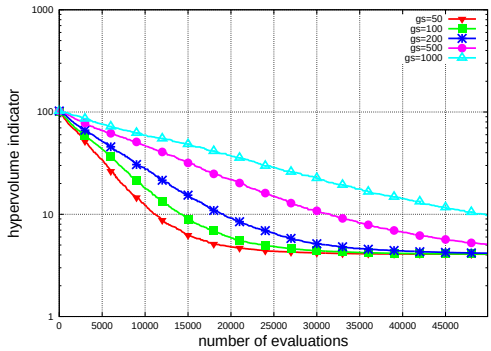
(b) ZDT3-SEQ-pop500-NSGA-II



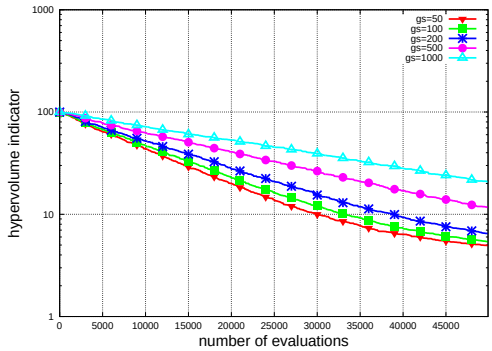
(c) ZDT3-ASSYNC-pop100-NSGA-II



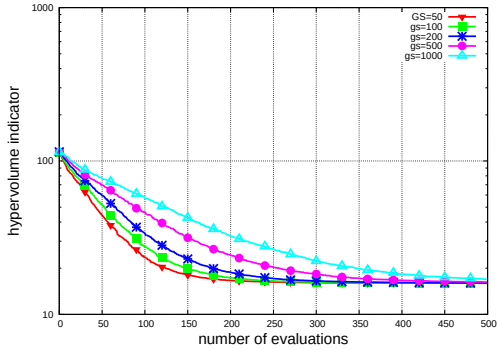
(d) ZDT3-ASSYNC-pop500-NSGA-II



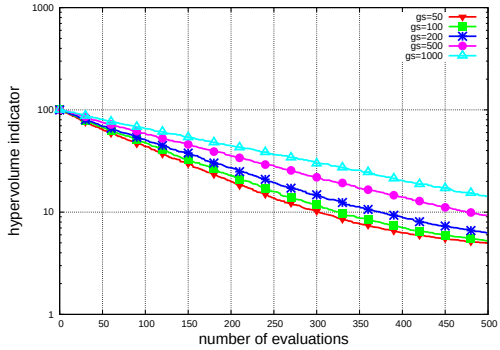
(e) ZDT6-SEQ-pop100-NSGA-II



(f) ZDT6-SEQ-pop500-NSGA-II

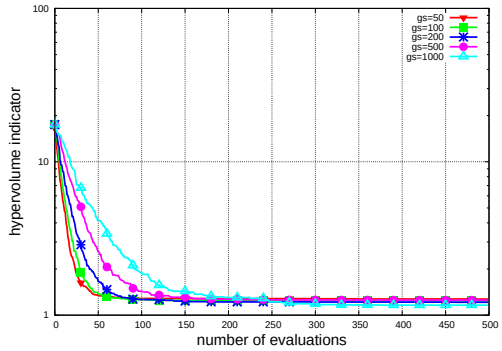


(g) ZDT6-ASSYNC-pop100-NSGA-II

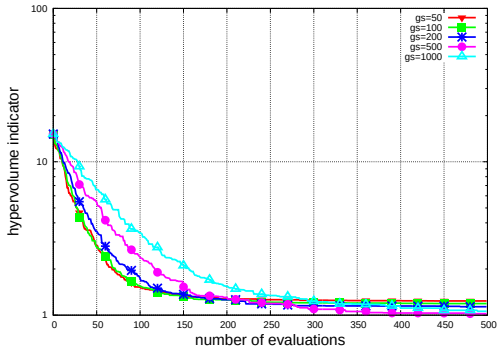


(h) ZDT6-ASSYNC-pop500-NSGA-II

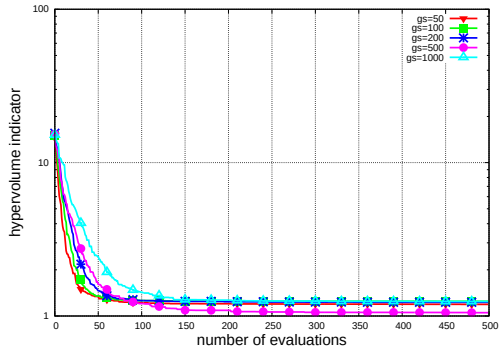
FIGURE B.6 – Évolution de l'indicateur de l'hypervolume moyen pour ZDT3 et ZDT6 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500



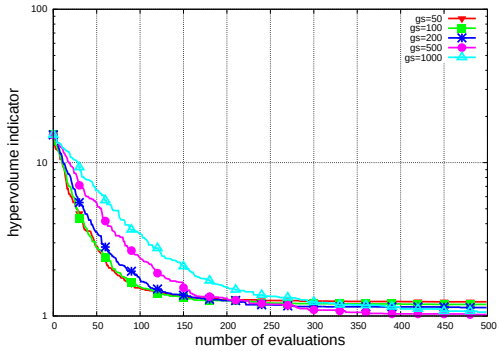
(a) IHR1-SEQ-pop100-NSGA-II



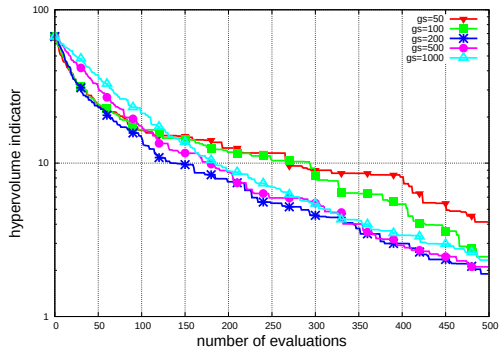
(b) IHR1-SEQ-pop500-NSGA-II



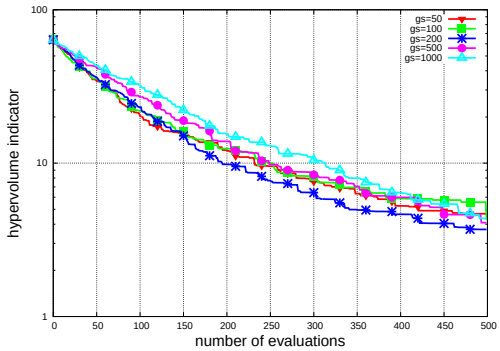
(c) IHR1-ASSYNC-pop100-NSGA-II



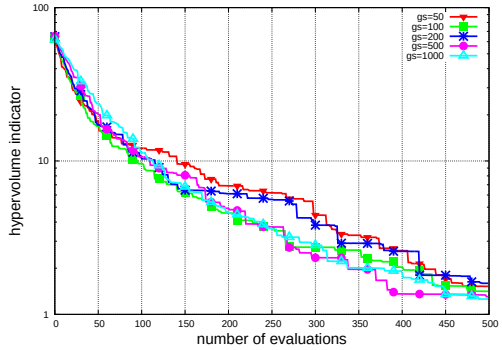
(d) IHR1-ASSYNC-pop500-NSGA-II



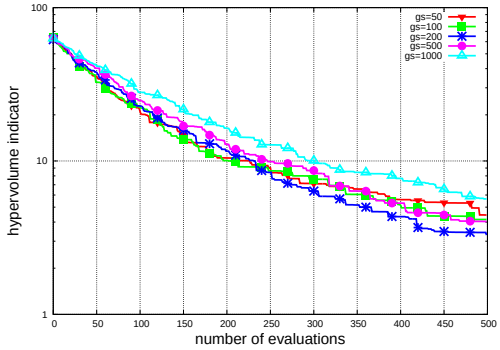
(e) IHR2-SEQ-pop100-NSGA-II



(f) IHR2-SEQ-pop500-NSGA-II

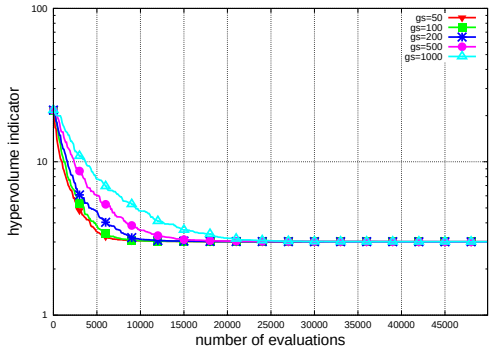


(g) IHR2-ASSYNC-pop100-NSGA-II

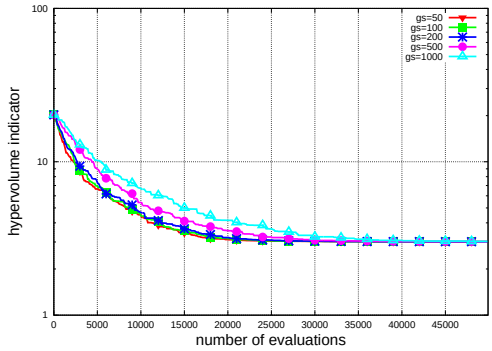


(h) IHR2-ASSYNC-pop500-NSGA-II

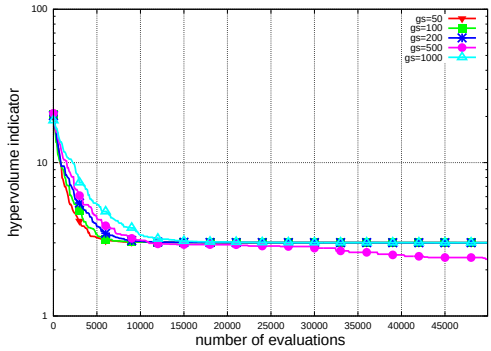
FIGURE B.7 – Évolution de l'indicateur de l'hypervolume moyen pour IHR1 et IHR2 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500



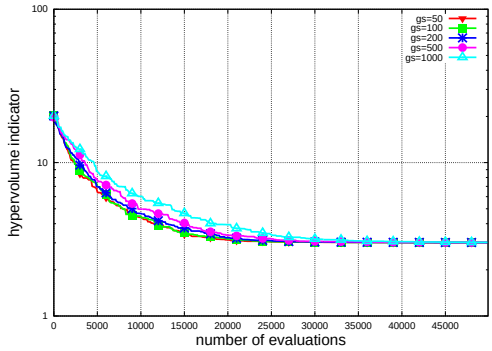
(a) IHR3-SEQ-pop100-NSGA-II



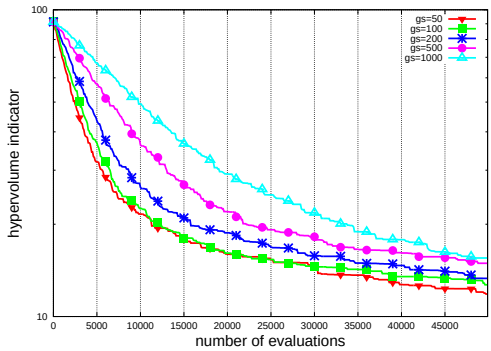
(b) IHR3-SEQ-pop500-NSGA-II



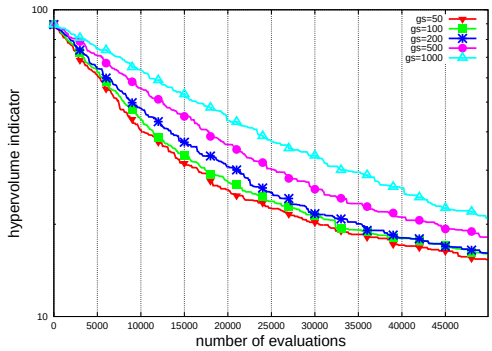
(c) IHR3-ASSYNC-pop100-NSGA-II



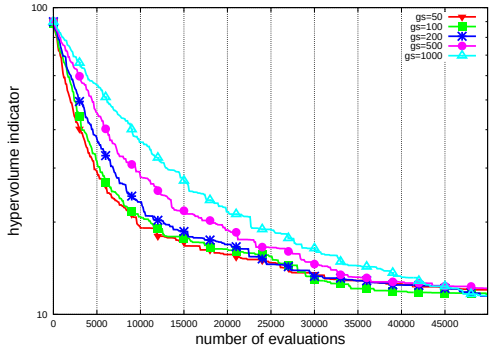
(d) IHR3-ASSYNC-pop500-NSGA-II



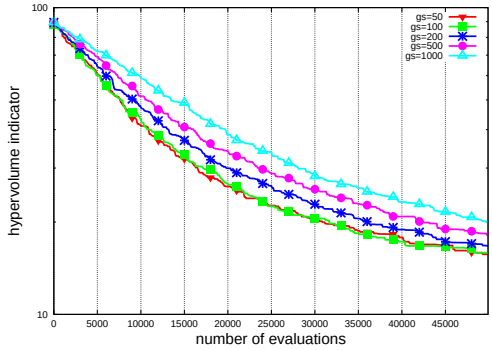
(e) IHR6-SEQ-pop100-NSGA-II



(f) IHR6-SEQ-pop500-NSGA-II



(g) IHR6-ASSYNC-pop100-NSGA-II



(h) IHR6-ASSYNC-pop500-NSGA-II

FIGURE B.8 – Évolution de l'indicateur de l'hypervolume moyen pour IHR3 et IHR6 avec les deux variantes asynchrone et séquentielle de NSGA-II en utilisant deux tailles de population différentes : 100 and 500