Optimisation robuste multiobjectifs par modèles de substitution

Vincent Baudoui

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par **l'Institut Supérieur de l'Aéronautique et de l'Espace**
Spécialité : Mathématiques appliquées et énergétique et transferts

---

Présentée et soutenue par **Vincent Baudoui**
le **7 mars 2012**

Optimisation robuste multiobjectifs par modèles de substitution

---

JURY

M.	Marc Schoenauer	président
M.	Rajan Filomeno Coelho	
M.	Kyriakos C. Giannakoglou	rapporteur
M.	Jean-Baptiste Hiriart-Urruty	directeur de thèse
Mme	Sophie Jan	
Mme	Patricia Klotz	co-directrice de thèse
M.	Rodolphe Le Riche	rapporteur
M.	Éric Walter	

École doctorale : **Aéronautique - Astronautique**

Unité de recherche : **Équipe d'accueil ISAE-ONERA MOIS**

Directeur de thèse : **M. Jean-Baptiste Hiriart-Urruty**
Co-directrice de thèse : **Mme Patricia Klotz**

Résumé

Cette thèse traite de l'optimisation sous incertitude de fonctions coûteuses dans le cadre de la conception de systèmes aéronautiques.

Nous développons dans un premier temps une stratégie d'optimisation robuste multiobjectifs par modèles de substitution. Au delà de fournir une représentation plus rapide des fonctions initiales, ces modèles facilitent le calcul de la robustesse des solutions par rapport aux incertitudes du problème. L'erreur de modélisation est maîtrisée grâce à une approche originale d'enrichissement de plan d'expériences qui permet d'améliorer conjointement plusieurs modèles au niveau des régions de l'espace possiblement optimales. Elle est appliquée à la minimisation des émissions polluantes d'une chambre de combustion de turbomachine dont les injecteurs peuvent s'obstruer de façon imprévisible.

Nous présentons ensuite une méthode heuristique dédiée à l'optimisation robuste multidisciplinaire. Elle repose sur une gestion locale de la robustesse au sein des disciplines exposées à des paramètres incertains, afin d'éviter la mise en place d'une propagation d'incertitudes complète à travers le système. Un critère d'applicabilité est proposé pour vérifier a posteriori le bien-fondé de cette approche à partir de données récoltées lors de l'optimisation. La méthode est mise en œuvre sur un cas de conception avion où la surface de l'empennage vertical n'est pas connue avec précision.

Mots clés : optimisation robuste, incertitudes, modèles de substitution, optimisation multiobjectifs, optimisation multidisciplinaire, plan d'expériences, combustion, conception avion

Abstract

Multiobjective robust optimization via surrogate models

This PhD thesis deals with the optimization under uncertainty of expensive functions in the context of aeronautical systems design.

First, we develop a multiobjective robust optimization strategy based on surrogate models. Beyond providing a faster representation of the initial functions, these models facilitate the computation of the solutions' robustness with respect to the problem uncertainties. The modeling error is controlled through a new design of experiments enrichment approach that allows improving several models concurrently in the possibly optimal regions of the search space. This strategy is applied to the pollutant emission minimization of a turbomachine combustion chamber whose injectors can clog unpredictably.

We subsequently present a heuristic method dedicated to multidisciplinary robust optimization. It relies on local robustness management within disciplines exposed to uncertain parameters, in order to avoid the implementation of a full uncertainty propagation through the system. An applicability criterion is proposed to check the validity of this approach a posteriori using data collected during the optimization. This methodology is applied to an aircraft design case where the surface of the vertical tail is not known accurately.

Keywords: robust optimization, uncertainty, surrogate models, multiobjective optimization, multidisciplinary optimization, design of experiments, combustion, aircraft design

Remerciements

Je tiens à remercier très chaleureusement toutes les personnes que j'ai eu le plaisir de côtoyer au cours de ces trois années de thèse, et qui ont contribué de près ou de loin à son aboutissement par l'aide et le soutien qu'elles m'ont apporté aussi bien dans mes travaux scientifiques que sur le plan humain.

Un immense merci tout d'abord à mes trois directeurs et encadrants de thèse qui m'ont accompagné au long de ce parcours : Patricia Klotz à l'Onera, et Jean-Baptiste Hiriart-Urruty et Sophie Jan à l'Institut de Mathématiques de Toulouse (IMT). Travailler avec vous a été un véritable plaisir. Chacun à votre manière, vous avez su me faire profiter de votre expérience de la recherche scientifique pour me permettre de mener à bien mes travaux. Patricia, merci pour ta profonde implication dans cette thèse et ton suivi quasi quotidien de mes recherches. J'ai été très heureux de partager ce bureau avec toi pendant ces trois ans à l'Onera. Merci pour tes compétences scientifiques et ton enthousiasme motivant que tu as su me transmettre, ainsi que la disponibilité dont tu as fait preuve afin de me venir en aide ou d'engager des discussions techniques passionnées source de grande inspiration. Sophie et Jean-Baptiste, merci pour vos précieux conseils sur les orientations à prendre, vos suggestions toujours judicieuses et vos relectures attentives de mes écrits qui ont grandement participé à leur amélioration. Je vous remercie tous les trois pour cet encadrement scientifique de qualité et espère avoir d'autres occasions de collaborer avec vous à l'avenir. Merci aussi à Renaud Lecourt et Franck Morel de l'Onera, qui nous ont prêté main forte dans le cadre des différentes applications de cette thèse, pour votre sympathie, votre disponibilité et votre patience face à mes nombreuses questions.

Merci aux rapporteurs Kyriakos C. Giannakoglou et Rodolphe Le Riche de m'avoir fait l'honneur d'accepter d'évaluer mes travaux de thèse, tâche qui fût probablement quelque peu fastidieuse au vu de la longueur de ce manuscrit (qui plus est en langue étrangère pour M. Giannakoglou) mais que vous avez su relever avec la plus grande attention. Merci également aux autres membres du jury Rajan Filomeno Coelho et Éric Walter pour le vif intérêt que vous avez porté à mes recherches, ainsi qu'à Marc Schoenauer qui a présidé ce jury d'exception avec brio. Merci à tous pour la qualité de vos questions et la pertinence de vos conseils lors de la soutenance. Je garde un très bon souvenir de nos échanges.

Merci à Bernard Lécussan, Directeur du Département « Traitement de l'Information et Modélisation » (DTIM) de l'Onera Toulouse de m'avoir permis de réaliser cette thèse grâce au soutien financier de l'Onera, ainsi qu'à François Rogier qui m'a accueilli au sein de son unité de recherche « Modélisation Mathématique et Simulation Numérique » (M2SN). Merci de même à tous les autres membres de l'équipe M2SN avec qui j'ai passé ces trois ans dans une ambiance de travail fort agréable : Nathalie Bartoli qui a suivi le début de ma thèse et qui est toujours restée disponible par la suite, Guillaume Dufour, Pierre Mazet,

Vincent Mouysset dont l'humour n'a pas manqué d'égayer le bureau d'en face, et Manuel Samuelides. Merci pour l'accueil chaleureux que vous avez réservé à un informaticien au sein de votre communauté mathématique. Merci aussi à Christiane et Josette pour votre aide administrative si précieuse. Je n'oublie pas non plus toute la bande des doctorants du DTIM : Cédric, Joël, Julien, Pierre, Sophie et Stéphanie déjà partis vers d'autres horizons, Dimitri, Florian, Jean-Baptiste, Mikel et Yann mes compagnons de labeur avec qui j'ai partagé les joies et les doutes d'une thèse, ainsi que les petits « jeunes » Anthony, Antoine, Charles, Christophe, Guillaume, Konstantinos, Maria, Matthias, Pierre, Pierre-Henri, Rémy, Thomasz et Vincent. Que de bons moments passés en votre compagnie, que ce soit autour d'une table du RU de Supaéro, d'un café-mots-croisés propice aux discussions érudites ou encore lors de la préparation des Journées des Thèses et du pique-nique. Je vous souhaite le meilleur pour la suite.

Ce travail n'aurait pu être réalisé sans la présence et le soutien permanent de mes proches. Merci aux amis toulousains Édith, Fabien, Florence, Guillaume, Jérôme, Julien, Pierre et Rémi pour vos encouragements bienveillants, ainsi qu'au fidèle groupe de Castres : Anouck, Carole, Cédric, Cyril, Julie, Laëtitia, Marjorie, Matthias, Nico, Stéphanie et Vanessa. Vous avez toujours été là, et, très sincèrement, merci. Merci enfin à mes parents et à mon frère pour toute votre affection, vous qui m'avez toujours donné les moyens d'arriver au bout de ce que je voulais faire.

Merci !

Table des matières

Introduction **1**

I Problèmes d'application

1 Conception d'une chambre de combustion **7**
1.1 Turbomachines 8
1.2 Réduction des émissions de polluants 9
1.3 Injection multipoints 10
1.4 Simulation numérique de la chambre de combustion 11
1.5 Conception d'un système d'injection optimal 16
1.6 Incertitude sur la répartition du carburant 19

2 Étude conceptuelle d'un design d'avion **21**
2.1 La conception avion 22
2.2 Simulation multidisciplinaire de l'avion 23
2.3 Définition du problème d'optimisation du design d'avion 27
2.4 Implémentation de la méthodologie d'optimisation multidisciplinaire 28
2.5 Incertitude sur la surface de l'empennage vertical 31

II État de l'art

3 Modèles de substitution **35**
3.1 Modèles de substitution et apprentissage 36
3.2 Polynômes 37
3.3 Réseaux de neurones 39
3.4 Krigeage 44
3.5 Autres modèles de substitution 50
3.6 Comparaison des types de modèles de substitution 51
3.7 Erreur d'un modèle de substitution 52
3.8 Plans d'expériences 57

4 Optimisation multiobjectifs **65**
4.1 Optimisation . . . 65
4.2 Optimisation multiobjectifs . . . 67
4.3 Algorithmes d'optimisation multiobjectifs . . . 69
4.4 Élagage du front de Pareto par classification . . . 75
4.5 Test des algorithmes d'optimisation . . . 77
4.6 Optimisation par modèles de substitution . . . 83

5 Optimisation sous incertitude **91**
5.1 Sources d'incertitude . . . 93
5.2 Analyse de sensibilité . . . 94
5.3 Modélisation de l'incertitude . . . 95
5.4 Mesures de robustesse . . . 97
5.5 Formulation d'un problème d'optimisation robuste . . . 104
5.6 Problèmes de test pour l'optimisation robuste . . . 108
5.7 Résolution d'un problème d'optimisation robuste . . . 112
5.8 Calcul des mesures de robustesse . . . 113
5.9 Plans d'expériences adaptatifs pour l'optimisation robuste . . . 124

6 Optimisation multidisciplinaire **127**
6.1 Optimisation multidisciplinaire . . . 127
6.2 Méthodes d'optimisation multidisciplinaire . . . 129
6.3 Modèles de substitution en optimisation multidisciplinaire . . . 132
6.4 Incertitudes en optimisation multidisciplinaire . . . 133

III Démarche

7 Méthodologie d'optimisation robuste **139**
7.1 Identification des incertitudes . . . 140
7.2 Choix des mesures de robustesse et formulation du problème robuste . . . 147
7.3 Choix des stratégies de calcul de la robustesse . . . 150
7.4 Exemple sur le cas test des deux barres . . . 151
7.5 Résolution du problème d'optimisation robuste . . . 159
7.6 Conclusion . . . 161

8 Plans d'expériences pour l'optimisation robuste **163**
8.1 Plans d'expériences adaptatifs pour l'amélioration globale de modèles . . . 164
8.2 Plans d'expériences pour l'optimisation déterministe (méthode PareBO) . . . 180
8.3 Plans d'expériences pour l'optimisation robuste (méthode PareBRO) . . . 189
8.4 Conclusion . . . 205

9 Optimisation robuste d'un système d'injection **207**
9.1 Bref rappel du problème . . . 208
9.2 Amélioration globale des modèles de substitution . . . 210
9.3 Formulation du problème d'optimisation robuste . . . 214
9.4 Optimisation robuste adaptative à l'aide de la méthode PareBRO . . . 218

9.5 Détermination des solutions optimales robustes . . . 223
9.6 Analyse des solutions optimales obtenues . . . 227
9.7 Conclusion . . . 231

10 Optimisation robuste mutidisciplinaire d'un avion 233
10.1 Bref rappel du problème . . . 234
10.2 Prise en compte simplifiée des incertitudes (méthode LOUP) . . . 236
10.3 Recherche de designs d'avion robustes avec la méthode LOUP . . . 240
10.4 Expression d'un critère d'applicabilité de la méthode . . . 242
10.5 Analyse des solutions optimales obtenues . . . 247
10.6 Conclusion . . . 249

Conclusion et perspectives 253

Annexes

A Mesure de l'espérance sur un modèle de krigeage . . . 259
B Mesure de la variance sur un modèle de krigeage . . . 263
C Mesure du quantile approché sur un modèle de krigeage . . . 273

Bibliographie 275

La seule certitude, c'est que rien n'est certain.
– Pline l'Ancien

Liste des symboles

$\boldsymbol{D}$	Espace des variables de décision
$\boldsymbol{d}$	Vecteur des variables de décision
d_i	i^{e} variable de décision
$D_{i,j}$	j^{e} sortie de la i^{e} discipline
$\boldsymbol{e}$	Vecteur des paramètres environnementaux
e_i	i^{e} paramètre environnemental
$\boldsymbol{f}$	Vecteur des fonctions objectif
f_i	i^{e} fonction objectif
$\hat{f}$	Modèle de substitution de la fonction f
$\hat{f}^{+\boldsymbol{x}}$	Modèle de substitution $\hat{f}$ auquel on a ajouté l'échantillon d'apprentissage $\boldsymbol{x}$
$\hat{F}$	Processus stochastique associé au modèle $\hat{f}$
$\boldsymbol{g}$	Vecteur des contraintes d'inégalité
g_i	i^{e} contrainte d'inégalité
$\boldsymbol{h}$	Vecteur des contraintes d'égalité
h_i	i^{e} contrainte d'égalité
IC	Intervalle de confiance à 95%
n_d	Nombre de variables de décision
n_e	Nombre de paramètres environnementaux
n_f	Nombre d'objectifs
n_g	Nombre de contraintes d'inégalité
n_h	Nombre de contraintes d'égalité
n_s	Nombre d'échantillons d'apprentissage
n_x	Nombre de variables d'entrée
$\boldsymbol{\Omega}$	Espace des paramètres incertains
p_χ	Densité de probabilité de la variable aléatoire χ
$\hat{\text{Q}}_k$	Quantile approché d'ordre k (équation (7.3))

Q_k	Quantile d'ordre k
$\mathbf{RC}$	Région de confiance (voir l'équation (4.17))
ρ	Mesure de robustesse
$\boldsymbol{S}$	Ensemble des échantillons d'apprentissage
$\boldsymbol{s}^i$	i^{e} échantillon d'apprentissage
s^i_j	j^{e} coordonnée du i^{e} échantillon d'apprentissage
S_i	Groupe de points similaires obtenu par classification
$\boldsymbol{u}$	Vecteur des images par f des échantillons d'apprentissage
u^i	Image du i^{e} échantillon d'apprentissage par la fonction de référence f
$\boldsymbol{X}$	Espace des variables d'entrée $\boldsymbol{x}$
$\boldsymbol{x}$	Vecteur des variables d'entrée
x_i	i^{e} variable d'entrée
$\boldsymbol{\chi}_x$	Vecteur des variables aléatoires associées aux variables $\boldsymbol{x}$
χ_{x_i}	Variable aléatoire associée à la variable x_i
$\prec$	Relation de dominance au sens de Pareto (voir l'équation (4.3))
$\prec_{\mathrm{RC}}$	Relation de dominance de Pareto modifiée pour prendre en compte les régions de confiance (voir l'équation (8.6))

Introduction

Cette étude s'intéresse au domaine de l'optimisation multiobjectifs pour la conception de systèmes dont l'évaluation est coûteuse. Il s'agit donc de minimiser simultanément plusieurs fonctions objectifs qui présentent des temps de réponse importants. De tels problèmes peuvent être rencontrés dans les différents secteurs de l'industrie où l'optimisation est utilisée pour aider au développement de systèmes plus performants, comme par exemple l'automobile, l'électronique ou encore la chimie. Les applications que nous allons traiter ici sont quant à elles issues du contexte aéronautique.

La conception de nouveaux produits s'appuie aujourd'hui sur des simulations numériques évoluées décrivant de plus en plus fidèlement les phénomènes physiques qui régissent le fonctionnement des systèmes. La place occupée par ces simulations dans les processus de développement ne cesse de croître, car elles permettent d'analyser et de comparer à moindre coût différentes configurations envisagées avant de mettre en œuvre des expérimentations plus onéreuses sur des prototypes réels. Elles peuvent de plus être automatisées et intégrées au sein de boucles d'optimisation en vue de déterminer la configuration du système la plus performante. Le nombre de simulations numériques qu'il est envisageable de réaliser en un temps d'étude raisonnable reste tout de même limité, car elles s'appuient sur des codes de calcul généralement complexes dont l'exécution est assez lourde.

Le problème rencontré lors de la conception d'un système optimal est que les configurations trouvées peuvent être relativement sensibles aux incertitudes. L'optimisation déterministe a en effet tendance à produire des solutions dépourvues de toute redondance superflue afin d'atteindre les meilleures performances possibles. Les systèmes ainsi conçus sont donc vulnérables aux fluctuations qui peuvent apparaître lors de leur réalisation ou de leur fonctionnement. Ces fluctuations proviennent par exemple des tolérances de fabrication qui font que le système réel ne correspond pas avec exactitude à la configuration idéale souhaitée, ou bien de l'environnement qui peut être soumis à des variations intrinsèques comme celles de la température de l'air ambiant. Si ces incertitudes ne sont pas considérées lors de l'optimisation, la solution obtenue pourra se révéler en pratique bien moins performante que prévue. À l'heure actuelle, les incertitudes qui affectent le système sont le plus souvent négligées, ou bien grossièrement prises en compte grâce à des marges de sécurité afin de ne pas complexifier les problèmes de conception.

Les recherches présentées dans ce mémoire ont pour point de départ deux applications qui évoluent dans ce contexte. La première concerne la conception d'un système d'injection pour une chambre de combustion de turbomachine aéronautique. L'objectif est de réduire les émissions polluantes de la chambre, sachant que sa simulation numérique nécessite deux jours de calcul. Les performances de ce système peuvent être altérées par l'obstruction aléatoire de certains points d'injection au cours de son fonctionnement. La

seconde application vise quant à elle à optimiser les performances d'un avion lors de son étude conceptuelle, en déterminant les principales caractéristiques de son design. La simulation du comportement de l'appareil fait intervenir différentes disciplines en interaction. La surface de l'empennage vertical de cet avion est dimensionnée de manière simplifiée, et elle peut donc ne pas correspondre exactement à celle qui sera définie dans les phases futures de sa conception. Ces deux cas d'application sont décrits en préambule dans les chapitres 1 et 2.

Pour résoudre ces deux problèmes, nous nous sommes appuyés sur les avancées réalisées dans les différents domaines impliqués, à savoir la gestion de fonctions coûteuses, l'optimisation multiobjectifs, la prise en compte d'incertitudes et le cadre plus général de l'optimisation multidisciplinaire.

Le temps de calcul conséquent nécessaire aux simulations numériques oblige à les utiliser avec parcimonie. Pour des études qui requièrent d'analyser un nombre important de configurations différentes (comme l'optimisation par exemple), il n'est donc pas envisageable de travailler directement avec ces simulations. Une solution est de les remplacer temporairement par des modèles approchés plus rapides, appelés modèles de substitution. Ces modèles sont construits à partir d'observations évaluées avec les simulations (et qui forment ce qu'on appelle un plan d'expériences), et permettent ensuite de prédire les performances de n'importe quelle configuration du système. Parmi les principaux types de modèles de substitution qui ont été développés, on peut citer les approximations polynomiales, les réseaux de neurones ou encore le krigeage (tous trois présentés dans [SPKA01]). Ces modèles sont assez génériques car ils sont établis indépendamment de la physique du système étudié, ce qui permet de les employer dans des domaines très variés. Ils sont ainsi relativement répandus aujourd'hui. Il ne faut par contre pas oublier qu'il ne s'agit que de modèles approchés, et la maîtrise de leur erreur est donc primordiale. Pour cela, on peut se baser sur approches statistiques de rééchantillonnage (voir [KS00]). Les modèles de substitution sont présentés dans le chapitre 3.

Les problèmes d'optimisation auxquels nous sommes confrontés font intervenir plusieurs objectifs à minimiser simultanément. De nombreux algorithmes ont été proposés pour résoudre ces problèmes multiobjectifs. Nous nous sommes particulièrement intéressés aux méthodes évolutionnaires, avec notamment l'algorithme génétique NSGA-II [DPAM02]. Ces méthodes sont des heuristiques qui s'adaptent relativement bien à une large palette de problèmes différents, quelles que soient les caractéristiques de leurs fonctions objectifs (en termes de convexité, de continuité, de dérivée, etc.). Les objectifs que nous considérons ici sont en effet définis à partir de simulations numériques complexes dont les propriétés sont le plus souvent inconnues. Les algorithmes évolutionnaires nécessitent par contre de réaliser un nombre important d'évaluations de ces objectifs, mais cela est moins problématique lorsqu'on se base sur des modèles de substitution très réactifs. L'erreur de ces modèles peut néanmoins venir biaiser les résultats de l'optimisation. Des stratégies de construction de plans d'expériences adaptatifs dédiés à l'optimisation ont ainsi été développées afin de limiter les évaluations des simulations aux zones supposées optimales, permettant ainsi de préciser les modèles aux endroits stratégiques. La méthode adaptative la plus répandue est l'algorithme EGO [JSW98], mais elle ne concerne que les problèmes mono-objectifs. Les approches adaptatives proposées dans le cadre de l'optimisation multiobjectifs sont plus récentes et donc moins éprouvées (voir [EGN06] par exemple). L'optimisation multiobjectifs est présentée dans le chapitre 4.

Les problèmes à résoudre présentent aussi des incertitudes dont il faut tenir compte. On souhaite ainsi concevoir des systèmes dont les performances ne seront pas trop sensibles aux fluctuations induites par les paramètres incertains. On parle d'optimisation robuste (voir [BS07]). La recherche de solutions robustes s'appuie sur des mesures de robustesse pour quantifier les variations des objectifs dues aux incertitudes. On peut par exemple mesurer la valeur moyenne de la performance d'une solution lorsque les paramètres incertains varient, ou bien encore sa variance. On recherchera ainsi le système qui possède la meilleure performance moyenne, ou bien celui dont la performance varie le moins, ou encore les deux à la fois. Différentes mesures de robustesse ont été exprimées, ainsi que différentes formulations de problèmes d'optimisation robuste associés. Il en résulte que les études sur le sujet paraissent relativement dispersées. Le calcul de ces mesures de robustesse nécessite par ailleurs de multiplier le nombre d'évaluations des objectifs afin d'observer l'effet des variations des paramètres incertains autour de chaque solution considérée. Cela justifie l'emploi de modèles de substitution. Nous nous sommes notamment intéressés aux possibilités de calcul analytique de mesures de robustesse sur certains modèles (voir [Jin04]). L'utilisation de modèles de substitution pose ici aussi la question de la gestion de leur erreur. Les quelques méthodes d'optimisation robuste adaptative proposées sont relativement récentes (voir [Jur07, JLR12]), et bien plus rares dans le cadre multiobjectifs. L'optimisation sous incertitude est présentée dans le chapitre 5.

Le second cas d'application à traiter est un problème d'optimisation multidisciplinaire, c'est-à-dire un problème dans lequel interviennent plusieurs disciplines interdépendantes. Différentes méthodes de résolution spécifiques à ce type de problèmes ont été développées. Il s'agit de stratégies mono-niveau où le système est considéré dans sa globalité, ou bien d'approches multiniveaux dans lesquelles plusieurs boucles d'optimisation interviennent à différents niveaux hiérarchiques (voir [TM06]). La résolution d'un problème d'optimisation multidisciplinaire peut se baser sur des modèles de substitution pour remplacer complètement des disciplines trop coûteuses, ou pour approcher les objectifs d'une boucle d'optimisation (voir [STBV08]). La prise en compte d'incertitudes dans le problème passe généralement par la mise en place d'une propagation d'incertitudes intrusive (toute approche globale étant bien souvent impossible du fait du coût de l'évaluation du système complet). Cette propagation consiste, comme son nom l'indique, à propager les incertitudes au sein du système afin de suivre leur évolution au travers des différentes disciplines et connaître leur effet final sur les objectifs du problème (voir [DWC00]). Son inconvénient est qu'elle est relativement coûteuse et complexe à mettre en œuvre sur un système multidisciplinaire existant. On retrouve ainsi les mêmes problématiques qu'en optimisation monodisciplinaire, mais les méthodes utilisées pour y répondre sont différentes. L'optimisation multidisciplinaire est présentée dans le chapitre 6.

Finalement, si la construction de modèles de substitution et l'optimisation multiobjectifs sont des domaines relativement bien maîtrisés, l'optimisation robuste est quant à elle plus récente. Différentes approches existent pour intégrer les incertitudes à un problème d'optimisation ainsi que pour calculer les mesures de robustesse. Par ailleurs, les stratégies d'enrichissement des modèles de substitution dans le cadre de l'optimisation multiobjectifs sont relativement peu éprouvées, surtout en optimisation robuste. Du côté de l'optimisation multidisciplinaire, la prise en compte des incertitudes par propagation n'est applicable qu'à un prix élevé dans les systèmes existants. Nous allons donc tenter de répondre à ces limitations.

La motivation de cette thèse est de définir des approches d'optimisation robuste multiobjectifs et multidisciplinaire afin de traiter les deux applications présentées plus haut. Ces approches seront basées sur des modèles de substitution permettant de représenter les fonctions coûteuses qui entrent en jeu.

Notre démarche a consisté dans un premier temps à synthétiser les différents travaux de l'état de l'art en optimisation robuste, dans le but de définir une méthodologie pratique pour poser puis résoudre des problèmes d'optimisation sous incertitude. Le chapitre 7 décrit ainsi les différentes étapes de cette méthodologie en l'illustrant sur plusieurs exemples simples.

Le calcul des mesures de robustesse utilisées dans un problème d'optimisation robuste est facilité par l'utilisation de modèles de substitution. Nous avons ainsi développé par la suite une approche d'enrichissement adaptatif de plan d'expériences pour l'optimisation robuste multiobjectifs. La définition de cette approche s'est faite par étapes en s'appuyant sur des exemples analytiques. Nous avons ainsi considéré dans un premier temps l'amélioration de modèles dans leur ensemble, avant de passer au cas de l'optimisation déterministe puis enfin robuste. Ce déroulement est présenté dans le chapitre 8. La méthode d'optimisation robuste adaptative ainsi définie a ensuite été appliquée au problème d'optimisation d'un système d'injection, dont la résolution est présentée dans le chapitre 9. Un article [BKHU+12a] présentant ces travaux est en cours de soumission.

Nous nous sommes finalement intéressés au cas plus général de la gestion des incertitudes en optimisation multidisciplinaire, en relation avec le problème de conception avion. Nous avons ainsi mis au point une méthode approchée de prise en compte des incertitudes afin d'éviter d'avoir à réaliser une propagation d'incertitudes coûteuse au sein du système. Le développement de cette méthode ainsi que la résolution de l'application concernée sont présentés dans le chapitre 10. Ces résultats ont fait l'objet d'un article [BKHU+12b] publié dans la revue « Structural and Multidisciplinary Optimization ».

Les différents chapitres de ce mémoire ont été construits de manière à pouvoir être lus plus ou moins indépendamment. Des rappels de notions déjà abordées auparavant sont donc placés aux endroits stratégiques. Nous faisons aussi des références régulières vers l'état de l'art afin de permettre au lecteur d'obtenir plus d'explications sur les notions qui ne lui sont pas familières.

Première partie

Problèmes d'application

Chapitre 1

Conception d'une chambre de combustion moins polluante

Sommaire

1.1 Turbomachines 8
1.2 Réduction des émissions de polluants 9
1.3 Injection multipoints 10
1.4 Simulation numérique de la chambre de combustion 11
1.4.1 Simulation de la phase gazeuse 11
1.4.2 Simulation de la phase liquide 13
1.4.3 Déroulement d'une simulation numérique 15
1.5 Conception d'un système d'injection optimal 16
1.6 Incertitude sur la répartition du carburant 19

Les préoccupations environnementales représentent un enjeu capital pour demain. La législation limite de plus en plus les émissions polluantes des avions et autres engins aéronautiques, ce qui encourage la conception de moteurs mettant en œuvre des techniques toujours plus sophistiquées. Les nouvelles générations de turbomachines doivent ainsi être à même de fournir la puissance nécessaire à la propulsion d'un appareil tout en rejetant le moins de polluants possible et en garantissant un bon fonctionnement du moteur.

Les travaux que nous présentons font suite à différents projets de recherche visant à l'amélioration des chambres de combustion des turbomachines afin de réduire leurs émissions polluantes. On peut notamment citer le projet européen TLC [1] (« Towards Lean Combustion ») et le projet interne Onera HEPHAISTOS qui se sont penchés sur le même procédé d'injection que celui que nous allons étudier. Les projets européens LOCOPOTEP et INTELLECT D.M. [2] ont aussi effectué des recherches avec le même objectif, tout comme le consortium INCA [3] qui continue encore aujourd'hui.

1. `http://ec.europa.eu/research/transport/projects/items/tlc_en.htm`
2. `www.intellect-dm.org`
3. `www.inca-combustion.fr`

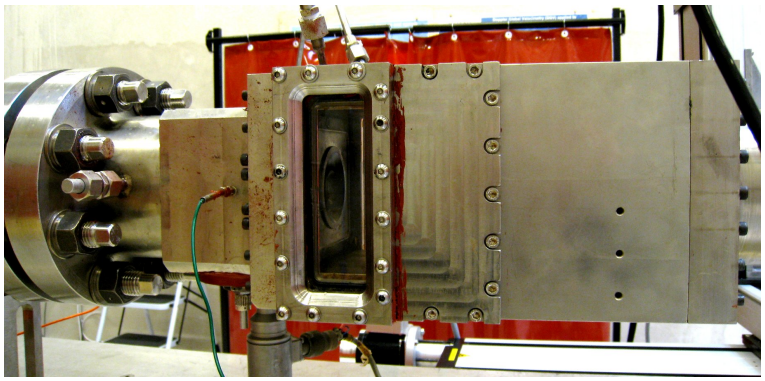


FIGURE 1.1 – Dispositif expérimental d'étude du système d'injection au banc LACOM.

Nous nous intéressons dans le cadre de cette étude à l'optimisation d'un système d'injection pour une chambre de combustion de turbomachine aéronautique. Le but de cette optimisation est de trouver la configuration optimale du système d'injection afin de réduire les émissions polluantes du moteur. Sa particularité est d'être doté d'un injecteur multipoints, c'est-à-dire d'un injecteur comportant plusieurs points d'injection répartis sur une couronne et dont nous présentons le principe dans ce chapitre. Nous détaillons aussi les simulations numériques utilisées pour modéliser le système en vue de son optimisation.

Durant le fonctionnement de la chambre de combustion, certains injecteurs peuvent se boucher de façon imprévisible, modifiant alors la qualité de la combustion à l'intérieur de la chambre de combustion. Il est souhaitable de prendre en compte cette incertitude durant l'optimisation afin de garantir de bonnes performances pour le moteur malgré cet aléa. Nous détaillons ainsi en fin de chapitre le problème d'optimisation sous incertitude de ce système d'injection, qui sera le sujet de la première partie de notre étude.

1.1 Turbomachines

Le système d'injection que nous allons étudier fait partie d'une turbomachine destinée à la propulsion d'engins aéronautiques dont le but est de transformer l'énergie du carburant en poussée.

Une turbomachine est composée tout d'abord d'un compresseur permettant d'augmenter la pression et de réchauffer l'air qui va entrer dans la chambre de combustion, puis d'un système d'injection permettant de vaporiser le carburant dans la chambre. Vient ensuite le foyer où se déroule la combustion et enfin un conduit d'échappement dans lequel une partie de l'énergie produite durant la combustion est récupérée grâce à une turbine pour entraîner le compresseur, l'énergie restante fournissant la poussée.

Le dispositif expérimental qui a servi à étudier le système d'injection que nous allons optimiser est présenté sur la photo de la figure 1.1. Il est implanté au banc LACOM (« LAboratoire de COmbustion Multiphasique ») de l'Onera. Ce banc permet de réaliser des expérimentations sur les dispositifs d'injection afin de mieux comprendre leur comportement et de mesurer leur efficacité dans les conditions réelles de fonctionnement des turboréacteurs. On y retrouve à gauche une arrivée d'air comprimé ainsi que de kérosène. Celui-ci est envoyé au système d'injection qu'on peut apercevoir au travers de la

vitre rectangulaire au centre. À droite se trouve la chambre de combustion dans laquelle est injecté le carburant. Ce dispositif expérimental permet de faire varier différents paramètres d'injection : on peut par exemple modifier le débit d'air ou de carburant entrant dans la chambre et visualiser leur effet sur la dispersion du carburant par le biais de diverses mesures optiques. On peut aussi modifier d'autres paramètres comme la position des injecteurs, mais cela nécessite l'usinage de nouvelles pièces avec un coût et des délais conséquents. La géométrie générale de la chambre de combustion est quant à elle fixée. La chambre que nous étudions fait 18 cm de long, ce qui correspond à un temps de traversée de 1.2 ms pour une vitesse axiale de l'écoulement de l'ordre de 150 $\mathrm{m.s^{-1}}$. Des mesures expérimentales réalisées sur cette chambre sont présentées dans [OGJ+09].

Les turbomachines aéronautiques utilisent un carburant liquide pour des raisons de sécurité et de faible encombrement. Ce carburant doit cependant retrouver sa forme gazeuse pour pouvoir être brûlé. Il est injecté sous forme de brouillard liquide dans la chambre de combustion et se disperse dans l'air chaud brassé par les entrées d'air. A la sortie de chaque point d'injection, le carburant liquide est donc pulvérisé en gouttelettes avant de s'évaporer dans le flux d'air qui l'enveloppe. Le gaz ainsi formé est ensuite brûlé dans le foyer.

1.2 Réduction des émissions de polluants

La réduction des émissions de polluants est l'un des enjeux majeurs de la conception de nouvelles turbomachines pour l'aéronautique. Dans ce but, l'amélioration des systèmes d'injection permet de concevoir des systèmes plus performants et moins polluants.

D'importantes recherches sont effectuées sur les systèmes d'injection pour obtenir une répartition optimale du carburant liquide dans la chambre, garante d'une bonne combustion. Le contrôle de l'injection permet en effet de gérer la richesse du mélange air-carburant. Une richesse trop faible empêche la combustion de démarrer par manque de combustible : il est donc nécessaire d'avoir au moins une poche de gaz dotée d'une richesse suffisante pour l'allumage. Néanmoins, il ne faut pas non plus que le mélange soit trop riche car on risque alors d'avoir des imbrûlés en sortie par manque de comburant. Les études en cours cherchent ainsi à développer un système d'injection qui procure une combustion de qualité tout au long du cycle de fonctionnement de la turbomachine et qui permette de réduire la quantité de polluants produits.

Deux des principales espèces polluantes qui se forment lors de la combustion sont le monoxyde de carbone CO et les oxydes d'azote NO_x. Le monoxyde de carbone apparaît lorsque la combustion n'est pas complète du fait d'un manque d'oxygène ou lorsque la température de combustion est trop basse. Les oxydes d'azote se forment quant à eux lorsque les températures de combustion sont trop élevées. On voit donc que la réduction des émissions de polluants n'est pas un problème facile car elle nécessite de conserver une bonne répartition du combustible et du comburant dans la chambre afin d'éviter les combustions incomplètes, et de garantir aussi une température de combustion qui ne soit ni trop basse ni trop élevée.

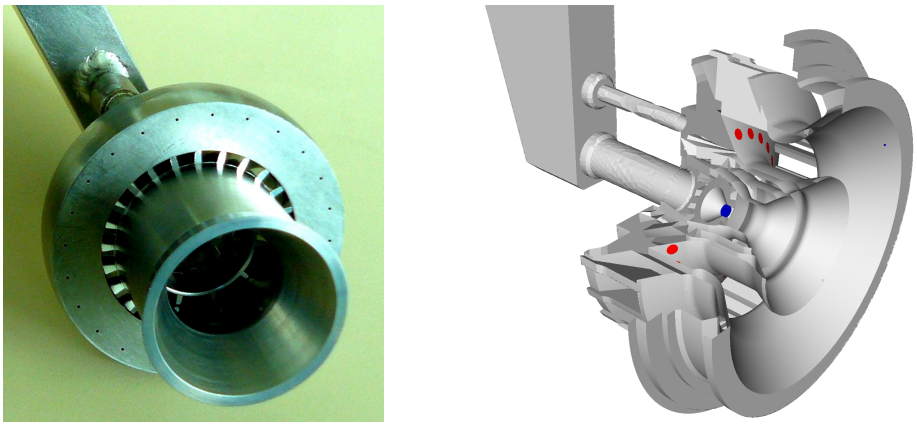


FIGURE 1.2 – Le système d'injection multipoints du dispositif expérimental à gauche, et une vue en coupe à droite. L'injecteur pilote est situé au centre (en bleu sur la vue en coupe), et l'injecteur principal est composé des différents points d'injection répartis sur la couronne (en rouge).

1.3 Injection multipoints

Auparavant, l'injection de carburant dans la chambre de combustion se faisait par le biais d'un seul injecteur central. On développe maintenant des systèmes d'injection multipoints dotés de plusieurs points d'injection différents. La multiplication des points d'injection permet de mieux répartir la dispersion du carburant dans la chambre et d'améliorer ainsi la qualité de la combustion, notamment pour les forts régimes moteur (décollage, montée, croisière) lorsqu'il est nécessaire d'avoir des débits de carburant élevés afin d'obtenir la poussée nécessaire. À ces régimes, la grande quantité de carburant supplémentaire est injectée de façon répartie dans l'écoulement d'air, ce qui permet de réduire les imbrûlés et autres polluants. Les régimes qui ne nécessitent pas toute la puissance du moteur (comme le ralenti par exemple) n'utilisent quant à eux que l'injecteur central classique.

Le système d'injection multipoints utilisé dans cette étude apparaît en détail sur la figure 1.2. Il est constitué d'un injecteur pilote situé sur l'axe de la turbomachine et d'un injecteur principal multipoints doté de 24 points d'injection répartis en cercle autour de l'axe. Ces points d'injection peuvent être aperçus sous la forme de petits points localisés sur la couronne sur la photographie du système d'injection. Les aubages situés entre l'injecteur pilote et les points d'injection principaux sont des entrées d'air. Une présentation plus détaillée du fonctionnement des injecteurs multipoints est donnée dans [Bar08].

La figure 1.3 présente une vue transversale du système d'injection où on retrouve en vert la pièce de la figure 1.2 fixée à l'entrée de la chambre de combustion (qui se situe à droite de la figure). On y retrouve les injecteurs pilote et principal. Le débit de carburant est réparti entre les différents points d'injection au travers des conduits qu'on voit sur la gauche de la figure.

Trois entrées d'air sont aussi indiquées sur la figure 1.3. L'air injecté dans la chambre de combustion passe dans une vrille qui l'anime d'un mouvement giratoire appelé « swirl ». Cet air chaud en rotation permet de focaliser et de stabiliser la zone de combustion au centre de la chambre dans une zone riche en carburant et bien alimentée en oxygène. La chambre possède donc trois entrées d'air swirlées. Deux autres entrées d'air non swirlées

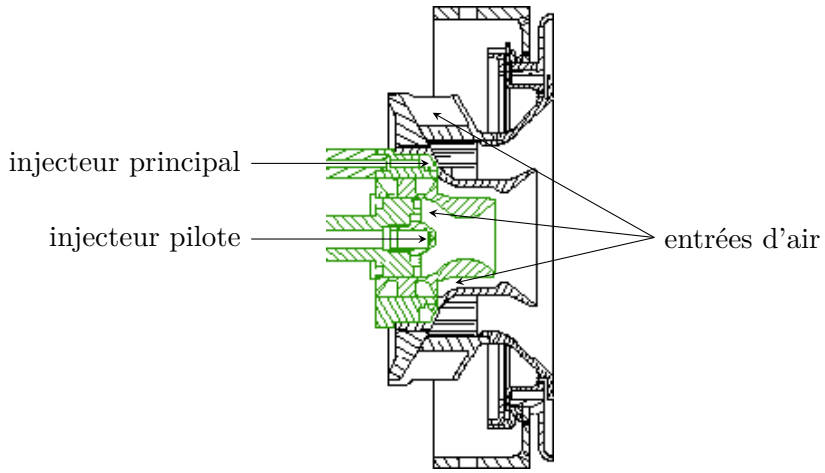


FIGURE 1.3 – Section du système d'injection avec visualisation des entrées d'air swirlées.

sont aussi présentes près de la paroi extérieure de la chambre (ces entrées supplémentaires ne sont pas visibles sur la figure 1.3). Tout comme le carburant, l'air pénètre dans la chambre avec un débit unique qui se répartit entre les différentes entrées.

1.4 Simulation numérique de la chambre de combustion

L'optimisation du système d'injection ne peut pas être réalisée directement sur le banc de test présenté ci-dessus, du fait du coût et des délais requis par les expérimentations. Le système est donc modélisé numériquement avec le code CEDRE (Onera) afin d'obtenir une simulation de son comportement qui permette de tester un plus grand nombre de configurations. De plus, la simulation numérique permet de mieux comprendre les phénomènes physiques qui se déroulent dans la chambre lors de son fonctionnement.

1.4.1 Simulation de la phase gazeuse

Simulation tridimensionnelle

Une première modélisation en trois dimensions de la chambre de combustion a été réalisée. Elle représente un canal de tranquillisation de l'air avec des vrilles en amont de la chambre, puis la chambre de combustion elle-même avec son système d'injection multipoints, et enfin un espace de dégagement en sortie (voir la figure 1.4). Cette représentation assez complète du système permet de simuler l'écoulement en amont et en aval sans avoir à spécifier les conditions aux limites de la chambre elle-même. L'air entrant se répartit de façon naturelle entre les différentes entrées avant de pénétrer dans la chambre par des vrilles qui lui donnent son mouvement giratoire. Les gaz d'échappement sont ensuite évacués dans l'espace de dégagement sans avoir à spécifier de pression en sortie. On a ainsi une représentation fidèle de la physique des écoulements dans le système. L'ensemble est maillé avec un maillage non structuré plus ou moins fin selon les régions modélisées. La zone centrale de la chambre de combustion se prête quant à elle à l'utilisation d'un maillage structuré. Plusieurs vues de ce maillage 3D sont représentées sur la figure 1.4.

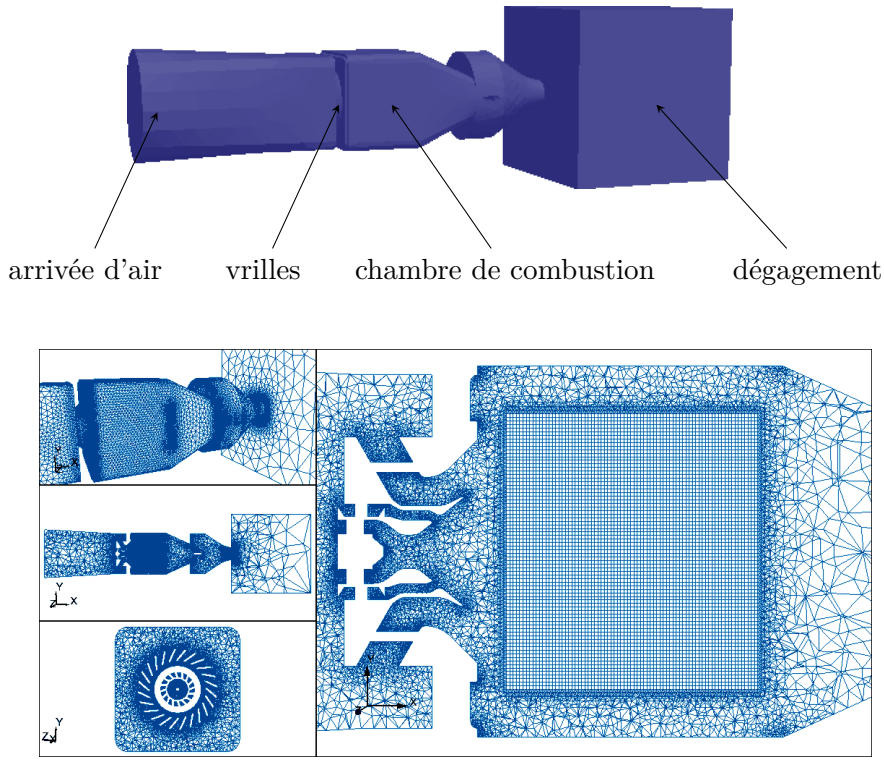


FIGURE 1.4 – Modélisation et maillage en trois dimensions de la chambre de combustion (source [BDMG08]).

Une simulation numérique LES (« Large Eddy Simulation ») en trois dimensions de la chambre à l'aide de ce maillage a été validée grâce à des points de mesure expérimentaux issus du banc d'essai. Ces mesures ont permis de vérifier le bon comportement de la simulation par rapport au système réel. Des résultats de simulations 3D LES sont présentés dans [Lav08] et [Jae09].

Le déroulement d'une simulation 3D LES est long (de l'ordre d'un mois de calcul). Des simulations 3D RANS (« Reynolds-averaged Navier-Stokes ») avec une modélisation KL de la turbulence ont donc aussi été mises en place et validées. Il s'agit de formulations sur les équations moyennées. Les simulations RANS sont donc moins précises que les calculs LES, mais elles permettent de réduire la durée d'un calcul à moins d'une semaine. Malgré cela, les modélisations en trois dimensions restent assez onéreuses en général. Si elles sont bien représentatives des phénomènes qui se déroulent à l'intérieur de la chambre de combustion et permettent une étude précise des spécificités d'une configuration donnée du système d'injection, leur utilisation dans le cadre d'une boucle d'optimisation semble difficile. C'est pourquoi nous nous sommes orientés vers des simulations de la chambre en deux dimensions, en faisant une hypothèse d'axisymétrie.

Simulation bidimensionnelle axisymétrique

L'optimisation d'un système oblige à tester plusieurs configurations différentes, et donc à lancer de nombreuses simulations avec des paramètres distincts. Afin de réduire le coût

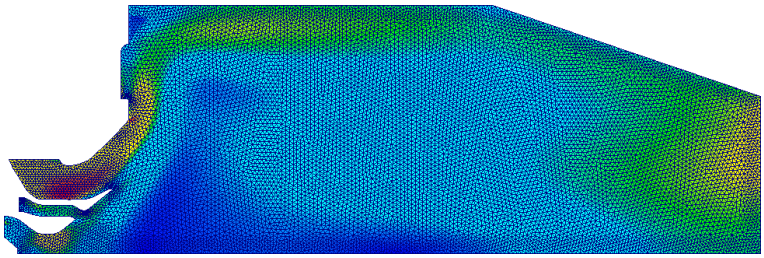


FIGURE 1.5 – Le maillage en deux dimensions de la chambre de combustion utilisé pour cette étude.

des simulations pour que l'optimisation puisse être réalisée dans un délai raisonnable, le système a été modélisé en deux dimensions en supposant une symétrie axiale de révolution. La chambre a en réalité une forme hexaédrique, mais le comportement axisymétrique de l'écoulement a été vérifié expérimentalement. On ne va donc simuler qu'une moitié de section plane de la chambre, la chambre complète pouvant être obtenue par révolution autour de l'axe central. Une étude comparative des méthodes de simulation 2D et 3D de la chambre est présentée dans [BDMG08].

Une simulation en deux dimensions est bien sûr moins précise qu'une simulation complète en trois dimensions, mais elle permet néanmoins d'avoir une bonne description du comportement du système et ce en un temps moindre (une simulation 2D RANS prend environ deux jours de calcul sur une machine standard). Cette simulation fidèle au comportement général de la chambre de combustion peut donc être utilisée pour réaliser l'optimisation qui nous intéresse. Une fois la configuration optimale trouvée, elle pourra si nécessaire être simulée en trois dimensions afin de vérifier la validité de sa simulation en 2D (la simulation 3D pouvant ensuite être validée à son tour par une expérimentation réelle sur le banc d'essai).

La modélisation 2D représente la chambre de combustion et le système d'injection uniquement. On n'a plus ici les modules situés en amont et en aval qui simulaient l'écoulement hors de la chambre au sein de la modélisation 3D. Au lieu de cela, des conditions d'entrée et de sortie d'écoulement subsonique ont été fixées : on impose ainsi la température et le débit en entrée de la chambre, et la pression à sa sortie. Le maillage non structuré utilisé est représenté sur la figure 1.5. On y retrouve notamment en bas à gauche les trois entrées d'air décrites sur la figure 1.3. Cette modélisation 2D axisymétrique a été calibrée et validée à partir de résultats des simulations 3D RANS.

On obtient donc une simulation simplifiée de la chambre de combustion qui rend possible l'optimisation du système d'injection. Cette modélisation en deux dimensions reste en effet assez fidèle au système réel, avec un temps de calcul raisonnable.

1.4.2 Simulation de la phase liquide

La simulation de l'injection du carburant dans une chambre de combustion requiert des techniques avancées. Elle est de plus critique pour notre application car nous allons rechercher la meilleure configuration d'injection possible pour la chambre étudiée. Il est donc primordial de représenter correctement le processus de dispersion du carburant dans l'air par les différents injecteurs.

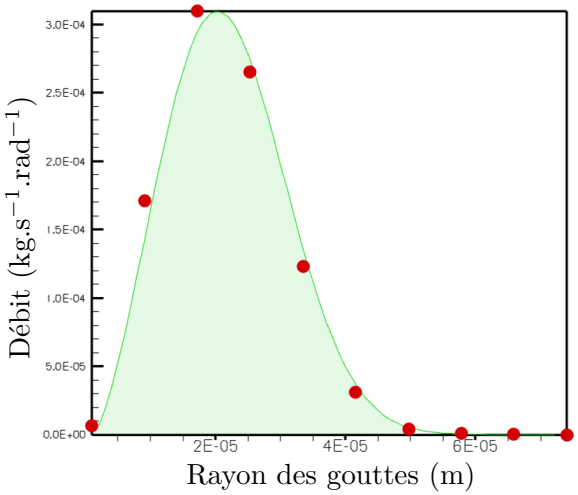


FIGURE 1.6 – Répartition du débit en fonction de la taille des gouttes selon une loi de Rosin-Rammler discrétisée en dix classes de gouttes.

Bénéficiant de la symétrie de révolution, la modélisation 2D de la chambre ne possède qu'un seul point d'injection principal pour représenter les 24 points d'injection répartis sur la couronne. Si on reconstitue la chambre en 3D par révolution de ce maillage, l'injecteur multipoints devient en fait une couronne d'injection continue le long d'un cercle faisant le tour de l'axe, au lieu d'être discrétisé en 24 points d'injection différents. Le comportement de cette couronne d'injection continue a été comparé et vérifié par rapport à celui de l'injecteur multipoints initial.

Dans les injecteurs réels, le carburant arrive en flux continu puis est pulvérisé dans l'air une fois dans la chambre. On se retrouve donc avec un brouillard de carburant à la sortie de chaque injecteur. Les injecteurs numériques modélisent les différentes gouttelettes de carburant qui constituent ce brouillard. Les gouttes sont en fait représentées par groupes : une goutte numérique correspond à un certain nombre de gouttes réelles. Le nombre de gouttes réelles représentées par une goutte numérique est fixé au départ de la simulation. On souhaite simuler un nombre de gouttes numériques assez important afin d'être le plus représentatif possible de la réalité, mais sans toutefois trop augmenter le coût du calcul (car celui-ci augmente avec le nombre de gouttes simulées). Chaque goutte numérique possède un diamètre, une température et une vitesse propres. Au cours d'une simulation, le diamètre d'une goutte peut évoluer par évaporation dans l'air. Les gouttes peuvent aussi se scinder en plusieurs gouttelettes (phénomène de fragmentation) ou bien se rassembler suite à des collisions (coalescence).

La distribution du diamètre des gouttes qui composent le spray de carburant est modélisée grâce à une loi de Rosin-Rammler. Elle définit le débit des gouttes envoyées en fonction de leur rayon (voir la figure 1.6). Ce rayon est de l'ordre de 10 µm. Les particules injectées sont discrétisées en dix classes de taille de goutte. Les coefficients de la loi de Rosin-Rammler ont été déterminés à partir de mesures expérimentales du DMS (diamètre moyen de Sauter) des gouttes effectuées sur la chambre (le banc LACOM permet en effet de réaliser des mesures de taille de gouttes par diffusion de la lumière d'un faisceau laser).

L'injecteur pilote sur l'axe de la chambre produit un brouillard conique de carburant. Il

s'agit d'un cône plein, qui est discrétisé en dix angles d'injection différents pour la simulation. Chaque angle possède un débit identique, leur somme étant égale au débit total fourni à l'injecteur. L'émission de gouttelettes selon ces dix angles reconstitue artificiellement une injection en cône plein.

Dans la simulation 2D de la chambre de combustion, l'injecteur principal est ainsi modélisé par dix entités numériques, chacune injectant une taille de gouttes prédéfinie avec un débit fonction de la répartition choisie sur les tailles de gouttes. L'injecteur pilote est quant à lui représenté par cent modules numériques, soit dix entités injectant une taille de goutte différente pour chacune des dix valeurs d'angle d'injection.

Nous avons vu comment simuler numériquement la chambre de combustion. Nous allons maintenant nous intéresser à son optimisation afin de trouver la meilleure configuration possible pour le système d'injection.

1.4.3 Déroulement d'une simulation numérique

Même en deux dimensions, la simulation numérique d'une chambre de combustion est complexe car elle met en jeu différentes phases (gazeuse pour l'air, liquide pour le carburant injecté) ainsi que des phénomènes chimiques réactifs (la combustion). La combustion se déroule de plus dans un environnement confiné, et les parois de la chambre de combustion viennent modifier le comportement des fluides présents en son sein. Le déroulement d'une simulation 2D RANS de la chambre de combustion que nous détaillons ici est présentée dans [MDGG08].

Une simulation complète diphasique réactive de l'ensemble de la chambre serait bien trop longue à converger. Afin de réduire son temps de calcul, cette simulation est réalisée en plusieurs étapes dont le but est de fournir un état initial plus avancé à la simulation complète pour la voir converger plus rapidement. On commence ainsi par simuler d'abord l'écoulement d'air à l'intérieur de la chambre de manière monophasique. Le carburant est ensuite injecté et on passe donc à une simulation diphasique avec interaction entre l'air et le carburant. Le combustible s'évapore dans le flux d'air chaud, puis vient enfin la simulation réactive complète de la combustion lorsqu'on allume le mélange de gaz formé précédemment. On simule alors les réactions chimiques qui entrent en jeu. La réaction de combustion est ici simulée par le modèle d'Eddy Break Up. Il s'agit d'un modèle simple de réaction qui ne permet pas le calcul de toutes les espèces chimiques créées lors d'une combustion réelle, mais qui modélise les réactions principales qui s'y déroulent.

Cette succession de simulations de plus en plus complexes permet de réduire le temps de simulation comparativement au lancement d'une simulation complète avec la physique la plus complexe dès l'état initial du système. En effet, il n'est pas utile de prendre en compte la chimie des éléments tant que la combustion n'a pas démarré par exemple. Trois simulations différentes se succèdent donc avant d'arriver à la description du comportement final de la chambre de combustion pour une configuration donnée du système d'injection : simulation d'écoulement monophasique, simulation d'écoulement diphasique, et enfin simulation d'écoulement diphasique réactif.

Ces simulations sont effectuées à l'aide du code CEDRE [CCD+05] de l'Onera, en utilisant le solveur RANS Charme pour la phase gazeuse et le solveur lagrangien Sparte pour la phase liquide avec couplage « two-way coupling » qui prend en compte la dispersion des gouttes par le gaz et l'effet rétroactif des gouttes sur le gaz, par évaporation et traînée. Le calcul d'une configuration donnée du système d'injection prend environ deux jours sur

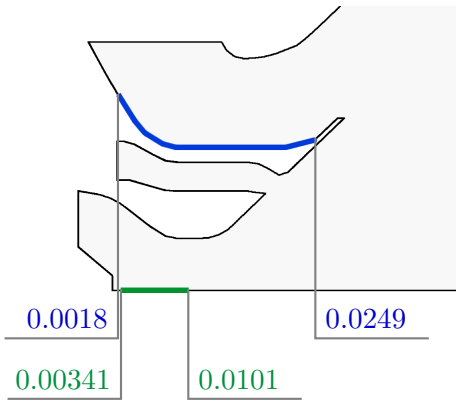


FIGURE 1.7 – Positions possibles des injecteurs pilote (en vert) et principal (en bleu).

une machine personnelle standard. La durée d'un calcul peut différer légèrement d'une simulation à l'autre en fonction de la convergence de la configuration testée, mais nous considèrerons ici que tous les calculs sont de même durée. Dans le cadre de cette étude, nous avons de plus la possibilité de lancer une dizaine de calculs en parallèle, ce qui permet de réduire encore le coût direct de ces simulations.

Le code de calcul qui permet de faire ces simulations n'est pas totalement déterministe. Le modèle intègre en effet des variables aléatoires nécessaires à la modélisation de la physique du système. Pour un même jeu de paramètres en entrée, les résultats peuvent donc différer d'une exécution à l'autre (tout en restant relativement proches). Ceci peut créer un léger « bruit » sur les données obtenues.

1.5 Conception d'un système d'injection optimal

Le but de cette étude est d'optimiser un système d'injection multipoints afin de minimiser les émissions polluantes en sortie de la chambre de combustion. Pour cela, nous allons pouvoir jouer sur trois paramètres influents du système d'injection. Ces paramètres sont d'une part le positionnement des injecteurs pilote et principal, et d'autre part l'angle de la vrille de l'une des entrées d'air. Tous trois sont des paramètres continus. Il ne s'agit pas de faire une optimisation de forme : la géométrie de la chambre de combustion restera quant à elle fixée.

Le cycle LTO (« landing – take-off ») d'un avion à proximité des aéroports se décompose en quatre phases : décollage, montée, approche et roulage au sol. C'est durant ce cycle que sont émis les polluants locaux à basse altitude qui nous intéressent. L'optimisation de l'injection doit normalement se faire sur l'ensemble de ces quatre phases. Néanmoins, les émissions de CO ont principalement lieu pendant le roulage, et les NO_x sont produits aux trois quarts durant la phase de montée. Nous nous placerons ici dans le cas d'un fonctionnement de la turbomachine en montée uniquement, car l'injecteur multipoints principal n'est pas utilisé lors du roulage. En phase de montée, l'air entre dans la chambre à une température de 728 K et avec un débit de 1.295 $kg.s^{-1}$. Le kérosène est injecté à température ambiante avec un débit de 38.1 $g.s^{-1}$. La pression en sortie est fixée à 19 bar.

L'injecteur pilote peut être déplacé le long de l'axe de la chambre de combustion. Il

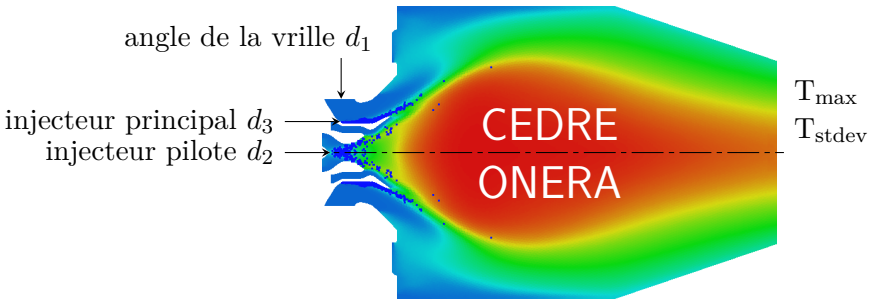


FIGURE 1.8 – Paramètres et objectifs du système d'injection à optimiser.

viendra ainsi projeter du carburant plus ou moins proche de l'entrée d'air qui l'entoure. Selon la position de cet injecteur, le carburant est emporté différemment dans le flux d'air entrant et donc dispersé autrement au sein de la chambre de combustion. La position de l'injecteur pilote est représentée par un réel correspondant à sa coordonnée sur l'axe de révolution de la chambre.

L'injecteur principal est quant à lui situé sur la paroi d'une veine d'air. Il peut être déplacé le long de cette paroi, ce qui provoquera là aussi un entraînement différent des gouttelettes de carburant dans l'air du conduit. Pour représenter la position de ce point d'injection, on utilise un réel qui indique sa coordonnée sur l'axe de la chambre. Son ordonnée est calculée en fonction du profil de la veine. La figure 1.7 présente la plage de variation de la position des deux injecteurs sur la modélisation 2D de la chambre.

Le troisième paramètre à optimiser est l'angle de la vrille qui donne le mouvement de swirl à l'air entrant dans la veine où est situé l'injecteur principal. Selon la valeur de cet angle, la vitesse tangentielle de l'air sera différente et le mélange de cet air avec le carburant en sera modifié. Plus l'angle de la vrille augmente, et plus le tourbillon créé par l'air sera puissant et confinera la zone de combustion vers le milieu de la chambre. Si l'angle est trop faible, le mélange entre l'air et le carburant ne pourra pas se faire correctement et la combustion ne sera pas bonne. Si l'angle de swirl est au contraire trop élevé, la zone de combustion sera plus réduite et la température des gaz en sortie sera plus élevée.

Les trois paramètres du système d'injection sont donc des réels pouvant varier dans un intervalle borné. La figure 1.8 présente la localisation de ces différents paramètres d'optimisation sur la modélisation 2D de la chambre, ainsi que les objectifs du problème que nous allons aborder maintenant.

L'objectif de cette optimisation est de réduire les émissions de polluants (CO et NO_x) en sortie de la chambre de combustion. Mais toutes les réactions chimiques ne sont pas simulées par le modèle 2D, et les quantités de monoxyde de carbone et d'oxydes d'azote produites ne sont donc pas directement accessibles [4]. Nous sommes alors obligés de passer par des mesures intermédiaires pour lesquelles des données sont fournies par les simulations.

Nous avons choisi de minimiser la température maximale T_{max} à l'intérieur de la chambre afin de réduire les émissions de NO_x, et de minimiser l'écart-type de la tem-

4. Un nouveau modèle permettant de simuler en 2D les émissions d'oxydes d'azote a depuis été développé au sein de la plateforme CEDRE. La quantité de NO_x peut ainsi être minimisée directement désormais.

Variables de décision			
d_1	Angle de la vrille	[30,60]	°
d_2	Position de l'injecteur pilote	[0.00341,0.0101]	m
d_3	Position de l'injecteur principal	[0.0018,0.0249]	m
Paramètre environnemental			
e_1	Répartition du carburant entre les deux injecteurs	0	
Incertitude			
χ_{e_1}	Répartition du carburant entre les deux injecteurs	$\mathcal{N}(0, 0.000576)$	
Objectifs			
f_1	Température maximale à l'intérieur de la chambre T_{max}	minimiser	K
f_2	Écart-type de la température en sortie T_{stdev}	minimiser	K

TABLE 1.1 – Variables et objectifs du problème d'optimisation d'un système d'injection.

pérature T_{stdev} en sortie de la chambre pour minimiser les émissions de CO. On sait en effet que les NO_x se forment au cours de combustions à température élevée : en réduisant la température maximale dans la chambre on diminuera aussi la création de ces oxydes d'azote. Concernant le second objectif, le fait de chercher à obtenir une température de sortie la plus homogène possible permet de s'assurer que le combustible était bien réparti à l'intérieur de la chambre. Une répartition non homogène du carburant entraîne des zones de forte température et des zones de température plus faible (selon qu'il y a beaucoup de carburant à brûler ou pas). Avec une variance réduite de la température en sortie de la chambre, on aura ainsi une combustion performante et complète qui réduira la production de CO.

Les deux objectifs doivent être minimisés simultanément, donnant un problème d'optimisation multiobjectifs. Le problème est sans contrainte, mis à part les contraintes de borne définies pour les paramètres d'optimisation. En effet, il a été vérifié auparavant que les configurations étudiées ont une richesse suffisante pour l'allumage du mélange et ne présentent pas de problème de retour de flamme ni de présence d'imbrûlés. Dans le cas contraire, il aurait fallu intégrer ces contraintes au problème.

Tous les paramètres et objectifs du problème sont récapitulés dans le tableau 1.1. Si on note $\boldsymbol{d}$ le vecteur des trois paramètres d'optimisation, le problème à résoudre est le suivant :

$$\begin{aligned} &\underset{\boldsymbol{d}}{\text{minimiser}} \ \{f_1(\boldsymbol{d}), f_2(\boldsymbol{d})\}, \\ &\text{s.c. } \boldsymbol{d}_{INF} \leq \boldsymbol{d} \leq \boldsymbol{d}_{SUP}, \end{aligned} \tag{1.1}$$

avec $\boldsymbol{d}_{INF}, \boldsymbol{d}_{SUP} \in \mathbb{R}^3$ les bornes inférieures et supérieures des paramètres de design.

Les deux objectifs sont calculés simultanément par le code de calcul qui réalise la simulation numérique de la chambre de combustion. Cette simulation sera vue comme une « boîte noire », c'est-à-dire une fonction inconnue dont on ne possède pas d'expression analytique et dont on ne connaît pas les propriétés. L'évaluation de cette fonction étant

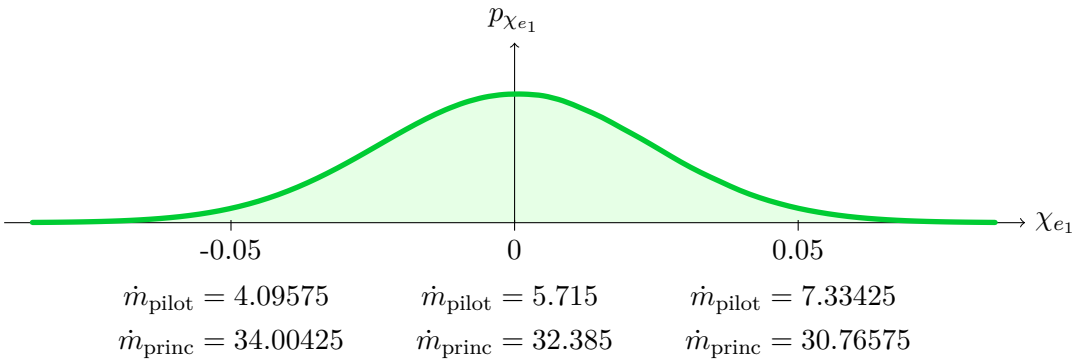


FIGURE 1.9 – Incertitude sur la répartition du carburant entre les injecteurs.

assez longue (deux jours environ), on souhaitera y faire appel le moins possible afin de réduire la durée du processus d'optimisation.

Pour déterminer le système d'injection optimal, le concepteur souhaite avoir accès à différentes solutions optimales possibles parmi lesquelles il fera son choix a posteriori.

1.6 Incertitude sur la répartition du carburant

Lorsque l'appareil est en phase de montée, 15% du débit total de kérosène injecté dans la chambre de combustion est affecté à l'injecteur pilote (ce qui représente un débit $\dot{m}_{\text{pilot}}$ de 5.715 g.s^{-1}), et le reste rejoint l'injecteur multipoints principal (soit un débit $\dot{m}_{\text{princ}}$ de 32.385 g.s^{-1}). L'injecteur principal bénéficie d'un débit de carburant plus important. Mais il peut arriver que certains des injecteurs se bouchent complètement ou partiellement durant le fonctionnement de la turbomachine, et ce de manière aléatoire. La répartition du carburant entre les injecteurs s'en trouve alors modifiée (le débit total de kérosène restant quand à lui constant), ce qui va affecter la dispersion du carburant dans la chambre et donc les performances du moteur.

Il est souhaitable de prendre en compte cette incertitude dans le problème d'optimisation du système d'injection. Nous l'avons modélisée en introduisant une variable e_1 qui vient modifier la répartition du carburant entre l'injecteur pilote et l'injecteur principal. La variation de débit sur un injecteur est définie comme un pourcentage de la valeur nominale du débit de l'injecteur principal. Le débit de chaque injecteur est donc exprimé comme suit :

$$\begin{aligned} \dot{m}_{\text{pilot}} &= 5.715 + 32.385 e_1, \\ \dot{m}_{\text{princ}} &= 32.385 - 32.385 e_1. \end{aligned} \tag{1.2}$$

Lorsque e_1 est nul, cela signifie que le carburant est normalement réparti entre les deux injecteurs pilote et principal. Lorsqu'il s'éloigne de cette valeur, la répartition est modifiée et l'un des injecteurs reçoit plus de carburant que prévu, l'autre étant considéré comme partiellement bouché. Une valeur positive de e_1 représente le fait que l'injecteur principal est bouché (et donc que le débit de l'injecteur pilote est augmenté), et à l'inverse e_1 devient négatif lorsque c'est l'injecteur pilote qui se bouche.

On considère que les injecteurs ne sont pas bouchés dans le cas général, et qu'il est plus probable qu'ils ne se bouchent que partiellement lorsque cela arrive. Nous avons donc remplacé le paramètre e_1 par une variable aléatoire χ_{e_1} doté d'une loi de probabilité $h_{\chi_{e_1}}$ gaussienne centrée pour représenter cette incertitude (voir la figure 1.9). Malgré le support infini de cette distribution, nous nous limiterons dans cette étude aux simulations avec $\chi_{e_1} \in [-0.05, 0.05]$ (ce qui correspond donc à une variation de débit de $\pm 5\%$ de la valeur nominale de $\dot{m}_{\text{princ}}$).

Le problème d'optimisation classique présenté plus haut doit maintenant intégrer cette donnée sur l'incertitude de la répartition du carburant entre les deux injecteurs. Il devient ainsi le problème stochastique suivant :

$$\begin{aligned} &\underset{\boldsymbol{d}}{\text{minimiser}}\ \{f_1(\boldsymbol{d}, \chi_{e_1}), f_2(\boldsymbol{d}, \chi_{e_1})\}, \\ &\text{s.c. } \boldsymbol{d}_{INF} \leq \boldsymbol{d} \leq \boldsymbol{d}_{SUP}. \end{aligned} \tag{1.3}$$

L'objectif est de trouver des solutions qui réduisent les émissions polluantes de la chambre tout en garantissant une certaine robustesse vis-à-vis de cette incertitude. On pourra par exemple souhaiter que les émissions de polluants ne dépassent pas un certain seuil quel que soit l'injecteur qui se bouche, ou bien que les émissions restent faibles en moyenne lors du fonctionnement du moteur.

Chapitre 2

Étude conceptuelle d'un design d'avion

Sommaire

2.1 La conception avion 22
2.2 Simulation multidisciplinaire de l'avion 23
2.2.1 Structure 24
2.2.2 Aérodynamique 25
2.2.3 Propulsion 25
2.2.4 Performances 25
2.2.5 Un système multidisciplinaire 26
2.3 Définition du problème d'optimisation du design d'avion 27
2.4 Implémentation de la méthodologie d'optimisation multidisciplinaire . . 28
2.5 Incertitude sur la surface de l'empennage vertical 31

La seconde application à laquelle nous allons nous intéresser au cours de cette étude concerne l'optimisation multidisciplinaire d'un design d'avion. Il s'agit de définir les dimensions générales d'un avion dans les premiers stades de sa conception. Le comportement de l'appareil est simulé grâce à un système multidisciplinaire faisant intervenir différents domaines d'expertise comme la mécanique des structures ou encore l'aérodynamique. Cette modélisation est décrite dans la première partie de ce chapitre.

La forme globale de l'avion doit être optimisée en vue de minimiser sa masse à vide ainsi que sa consommation sur une mission donnée. Dans la représentation utilisée, la surface de l'empennage vertical n'est pas connue avec précision. Cette donnée incertaine devra être prise en compte durant l'optimisation du design de l'appareil pour que ses performances ne soient pas dégradées lorsque la surface sera calculée avec plus de précision lors des étapes suivantes de la conception. Nous explicitons donc en fin de chapitre le problème d'optimisation multidisciplinaire sous incertitude posé par cette application.

2.1 La conception avion

La conception de nouveaux modèles d'avion est l'activité phare des constructeurs aéronautiques. Il leur faut en effet proposer en permanence des appareils toujours plus performants et qui restent en phase avec les demandes de leurs clients. Une présentation détaillée du processus général de conception d'un avion est donnée dans [Ray06].

La conception d'un nouvel avion est un processus complexe qui vise à définir des solutions de compromis entre les demandes des compagnies aériennes, les coûts de fabrication des appareils et les contraintes physiques imposées par les technologies qu'on maîtrise actuellement. Tout débute par la spécification d'un cahier des charges indiquant les objectifs du nouveau design : on souhaite par exemple pouvoir transporter un certain nombre de passagers sur une distance donnée tout en consommant le moins de carburant possible. A partir de ce cahier des charges, on peut ensuite commencer à dessiner les grandes lignes de l'avion.

Le choix du design d'un avion se déroule généralement en trois grandes phases, en commençant par une étude conceptuelle suivie d'une conception préliminaire de l'appareil puis de sa conception détaillée. Ces trois étapes sont décrites ci-dessous. Elles permettent de fixer progressivement la géométrie de l'appareil considéré en utilisant des descriptions de plus en plus détaillées.

Étude conceptuelle Cette première étape dans la définition d'un nouvel appareil vise à vérifier la viabilité du projet de conception pour voir si on peut construire à un prix abordable un avion qui réponde au cahier des charges. De nouvelles idées et de nouveaux concepts sont alors étudiés pour y parvenir. Le cahier des charges peut aussi être adapté en conséquence si nécessaire. Durant la phase d'étude conceptuelle, on détermine les grands choix de conception pour avoir une première idée des caractéristiques principales de l'appareil : le type d'avion, sa géométrie, ses dimensions globales, son nombre de moteurs, etc. On définit ce à quoi va ressembler l'appareil, quel sera son poids, son coût et ses performances générales. Si cela est possible, on se base sur des avions déjà existants pour avoir une base de données conséquente des performances offertes par ces appareils. On fait aussi appel à des codes de calcul simples pour vérifier la satisfaction des spécifications requises. Différents concepts peuvent être évalués et comparés afin de déterminer le design le plus prometteur en termes de performances et de coûts.

Conception préliminaire Dans cette phase, on précise le design choisi précédemment en affinant la définition des caractéristiques de l'avion. Différents spécialistes des domaines mis en jeu analysent la portion de l'avion qui les concerne. On calcule les charges appliquées à l'appareil pour déterminer sa structure et sa masse, on étudie plus précisément son aérodynamique pour identifier les zones qui nécessitent des raffinements, etc. Cela permet de mieux estimer les performances de l'avion et de pouvoir les améliorer en optimisant les différents paramètres du design. Cette étape peut faire appel à des codes de calculs plus complexes ainsi qu'à des essais en soufflerie. La configuration de l'appareil et le design de ses éléments principaux est alors fixée définitivement.

Conception détaillée Il s'agit de l'étape finale de la conception, au cours de laquelle chacune des parties de l'avion est décrite en détail dans le but de préparer sa fabrication. On réalise une conception détaillée de tous les mécanismes de l'avion et l'emplacement définitif des différents équipements et systèmes est fixé. Il faut donc

préciser les multiples paramètres de l'appareil, qui peuvent maintenant être optimisés finement afin d'aboutir à un design aux performances optimales. Si les grands éléments de l'avion étaient considérés comme un tout lors de l'étape précédente, on doit maintenant étudier et dessiner chacune des pièces qui les composent. Le processus de fabrication de ces pièces ainsi que leur assemblage sont aussi définis. Le développement des différentes parties est confié à des équipes spécialisées. Lors de la conception détaillée, il n'est plus possible de réaliser d'optimisation globale de l'appareil, sa modélisation dans son ensemble étant trop coûteuse. L'optimisation se fait uniquement à un niveau local à partir des résultats de la phase précédente. On finalise aussi l'évaluation précise des performances et des coûts du nouvel avion. Cette phase se termine avec la mise en production de l'appareil.

Le nombre de paramètres pris en compte augmente à chaque étape de la conception, et les modélisations de l'appareil deviennent aussi de plus en plus proches de la réalité. Cela permet d'affiner petit à petit la définition de l'avion. Si au cours des deux premières phases une optimisation multidisciplinaire de l'avion tout entier est possible, la complexité des simulations rend toute considération générale impossible lors de la conception détaillée.

Chacun des étages de la conception est en fait un processus itératif au cours duquel le design est amélioré dans le cadre défini lors des étapes antérieures. La remise en cause des choix réalisés auparavant est très coûteuse car elle nécessite de revenir à l'étape précédente et de recommencer tout son processus en quelque sorte. Il est donc primordial de faire les bons choix du premier coup, en s'aidant de techniques d'optimisation avancées et de modèles permettant de prédire le comportement de l'avion avec suffisamment de précision.

Dans le cadre de cette étude, nous nous situons dans la phase d'étude conceptuelle de l'avion, bien que le calcul des structures mis en œuvre soit un peu plus évolué qu'habituellement et se rapproche de celui utilisé en conception préliminaire. Le but sera donc de définir les caractéristiques principales d'un appareil répondant au cahier des charges à l'aide d'une simulation numérique simplifiée du système.

Les concepts standards d'avions arrivent aujourd'hui dans les limites de leurs possibilités en termes de performances, et c'est pourquoi l'étude conceptuelle de nouveaux aéronefs fait l'objet de nombreuses recherches. Lorsqu'il s'agit de développer un appareil de design plus classique, l'optimisation poussée de ses paramètres est ainsi essentielle si on souhaite avoir un engin compétitif.

2.2 Simulation multidisciplinaire de l'avion

L'optimisation du design d'un nouvel avion lors de la phase d'étude conceptuelle se base sur des simulations numériques qui permettent d'estimer les performances de l'appareil. La modélisation d'un avion fait intervenir plusieurs domaines différents tels que la mécanique des structures, l'aérodynamique, l'énergétique ou encore l'acoustique. Ces domaines d'expertise sont appelés des « disciplines ». Un avion est donc un système multidisciplinaire dans lequel différents domaines entrent en interaction.

La modélisation utilisée pour cette étude est une représentation simplifiée d'un avion complet qui met en jeu quatre disciplines distinctes : la structure, l'aérodynamique, la propulsion et les performances, comme présenté sur la figure 2.1. Ces différentes disciplines sont détaillées ci-après. Elles sont implémentées au sein d'un code interne Onera nommé ACODE (pour « Airline COnceptual DEsign »).

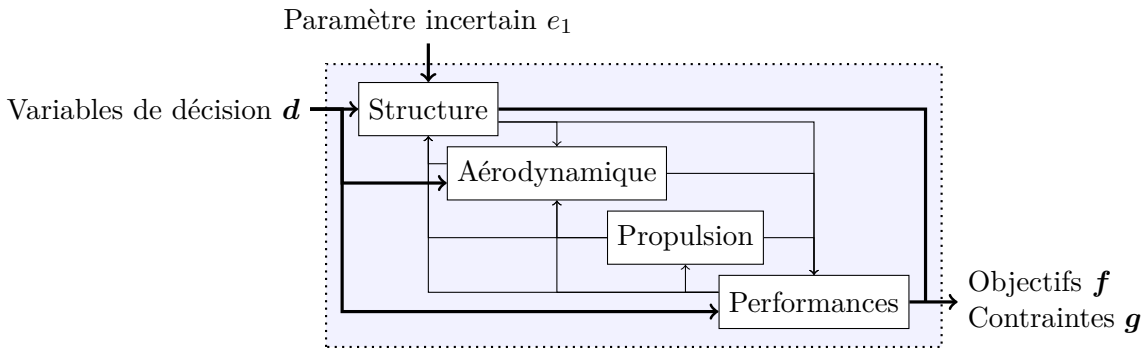


FIGURE 2.1 – Composition du système multidisciplinaire permettant de simuler un avion complet.

2.2.1 Structure

La première discipline modélisée concerne la structure de l'appareil. Les matériaux utilisés ainsi que leur assemblage doivent pouvoir résister à la pression de l'air et aux déformations produites par les différentes charges auxquelles l'avion sera soumis durant ses manœuvres en vol, quelles que soient les conditions atmosphériques rencontrées. Cette discipline permet de définir la structure de l'avion de manière à lui garantir une solidité suffisante. Son intérêt principal est de déterminer la masse de l'appareil.

La masse totale est égale à la somme de la masse des différents constituants élémentaires de l'avion. Elle est décomposée en 15 postes élémentaires suivant la norme 2001D. Les deux éléments principaux permettant d'évaluer la masse structurale sont la voilure et le fuselage. Ils bénéficient d'un calcul de masse avancé. La masse des autres éléments (moteurs, trains d'atterrissage, charge marchande, équipage, etc.) est estimée de manière statistique à partir de bases de données de devis de masse d'appareils existants.

La masse de la voilure est calculée grâce aux formulations de la mécanique des structures. Cette masse se décompose en deux parties. On a d'un côté la masse « travaillante », qui est dimensionnée pour faire face aux principales charges auxquelles l'avion sera soumis et qui doit respecter les normes de dimensionnement de la réglementation (cas de charges limites, résistance aux rafales verticales, etc.). Elle est déterminée par un calcul d'éléments finis permettant de fixer les épaisseurs des éléments de la structure. On calcule aussi d'autre part une masse « non travaillante ». Il s'agit de la masse d'éléments tels que les réservoirs, les becs, les volets, etc. Elle est déterminée en fonction de la topologie de l'aile à partir de lois statistiques issues d'une base de données de voilures.

La masse du fuselage est obtenue à l'aide d'une méthode mixte qui allie dimensionnements en pression et en flexion avec une méthodologie statistique pour ses autres composantes (aménagements intérieurs). Les masses de pression et de flexion dépendent notamment de la dimension du fuselage, de la masse du chargement et de l'altitude de croisière de l'avion.

Les différents modèles de calcul de masse utilisés font intervenir 53 paramètres, parmi lesquels 17 sont indépendants des variables de design. Ils ont été validés sur une base de données de devis de masse d'avions de transport civil.

La discipline structure permet aussi de déterminer la position du centre de gravité de l'avion pour réaliser son équilibrage. Le centre de gravité est défini à partir des positions des centres de gravité des différents éléments de l'appareil : fuselage, voilure, empennages, ensemble propulsif, carburant, passagers, équipage, systèmes, trains d'atterrissage, etc.

2.2.2 Aérodynamique

A partir de la géométrie de la voilure de l'appareil et des conditions de vol définies (altitude, nombre de Mach, etc.), cette discipline permet de déterminer les principales caractéristiques aérodynamiques de l'avion comme sa traînée ou portance maximale. Le comportement aérodynamique de l'appareil est simulé durant toutes les phases de vol, avec notamment les phases à basse vitesse hypersustentées (décollage et atterrissage) et la phase de croisière. Ces phases sont primordiales pour le dimensionnement de l'avion.

Les différents éléments de l'avion sont modélisés de façon simple afin de limiter le nombre de données nécessaires et de faciliter la mise en œuvre des calculs. Le modèle utilisé est basé sur un calcul 2D des performances aérodynamiques des profils de la voilure, avec une extension en trois dimensions pour prendre en compte la géométrie plane de l'aile.

Le calcul de la traînée totale de l'avion est décomposé en quatre contributions différentes qui ont chacune une signification physique propre : la traînée de frottement liée aux forces tangentielles entre l'air et la surface de l'avion, la traînée induite propre aux surfaces portantes, la traînée de pression visqueuse produite par l'accumulation et le décollement de la couche limite, et la traînée d'onde relative à l'existence de zones supersoniques à la surface de l'avion.

Le modèle aérodynamique fait appel à 52 paramètres parmi lesquels on retrouve toutes les variables de design. Il a été validé grâce à des données expérimentales issues d'essais en soufflerie sur des configurations lisses ou hypersustentées d'avions conventionnels.

2.2.3 Propulsion

La discipline propulsion permet de modéliser la puissance fournie par les moteurs de l'appareil ainsi que leur consommation dans des conditions de vol données. Les moteurs sont modélisés avec un point de vue macroscopique, car la simulation détaillée d'une turbomachine complète serait bien trop complexe. Les performances des moteurs sont estimées à partir de tables numériques qui indiquent la poussée et la consommation spécifique en fonction des différentes phases de vol (nombre de Mach, altitude, etc.).

2.2.4 Performances

Cette discipline permet de connaître les performances de l'avion au cours d'une mission donnée. Ces performances peuvent être par exemple le rayon d'action de l'appareil, sa longueur de décollage, sa vitesse d'approche avant l'atterrissage, sa consommation, etc. Les performances peuvent aussi être calculées pour différentes conditions atmosphériques et différentes conditions d'utilisation de l'appareil (cas de moteur en panne par exemple). Tous les calculs sont basés sur l'intégration des équations de la mécanique du vol, l'avion étant considéré comme un simple point matériel.

Le modèle permet de simuler le comportement de l'avion sur une mission complète. Il fait appel à 39 paramètres. Un exemple de mission est décrit sur la figure 2.2. Une mission comprend tout d'abord une phase de décollage puis une montée à vitesse conventionnelle

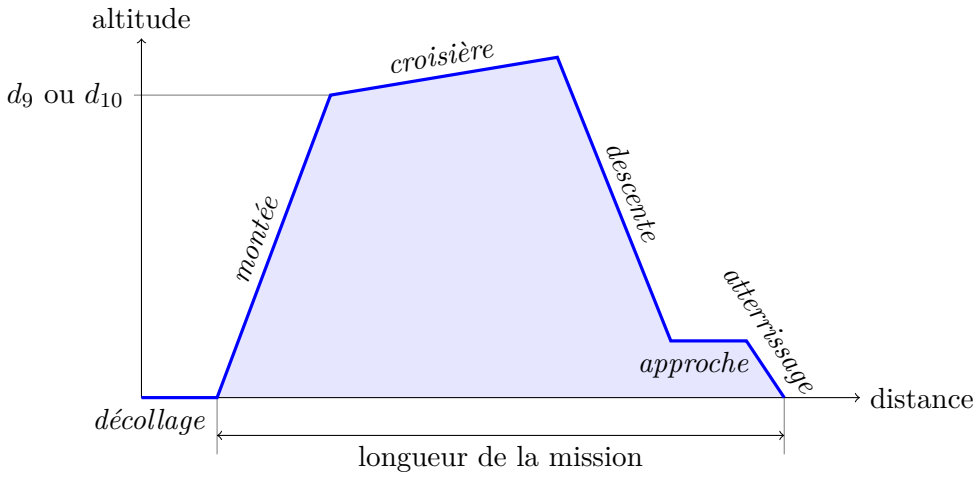


FIGURE 2.2 – Profil de mission de l'avion.

constante jusqu'à une altitude de croisière définie. Une fois cette altitude atteinte, on passe en phase de croisière où l'altitude de l'avion évolue à finesse constante (la croisière peut aussi être décomposée en plusieurs paliers successifs). On redescend ensuite à vitesse conventionnelle constante vers l'altitude d'approche, et vient enfin la phase d'approche puis l'atterrissage proprement dit.

Pour le calcul des performances au décollage, on simule tout d'abord un décollage classique dont les résultats sont utilisés pour la poursuite de la mission. La consommation, la durée et la vitesse de décollage sont déterminées. On simule ensuite un décollage avec une panne moteur au cours duquel sont calculées les distances de décollage poursuivi et d'accélération-arrêt qui permettent de déterminer la vitesse de décision et la longueur maximale de décollage de l'avion.

La consommation de l'appareil est déterminée à partir de la masse de carburant nécessaire à l'accomplissement de la mission, à laquelle il faut ajouter les réserves réglementaires obligatoires. La quantité de carburant de réserve à emporter est calculée en simulant successivement un déroutement de l'avion de 200 miles nautiques (NM), un prolongement de 10% de la phase de croisière ainsi qu'un « overshoot » (remise des gaz lors d'un atterrissage interrompu). Les phases de roulage, d'attente et d'atterrissage sont quant à elles forfaitaires en termes de durée et donc de consommation.

2.2.5 Un système multidisciplinaire

La conjonction des quatre disciplines présentées ci-dessus définit un système multidisciplinaire au sein duquel les disciplines sont en interaction permanente. Chaque discipline partage des variables avec les autres disciplines et dépend donc en partie de leurs résultats.

Pour donner un exemple, lorsque la masse de la structure de l'avion augmente, sa consommation augmente aussi et il faut emporter plus de carburant pour effectuer une mission donnée. Ce carburant supplémentaire vient se rajouter au poids de l'avion, ce qui nécessite de renforcer sa structure. La masse de la structure est alors augmentée, et la consommation de l'appareil à nouveau accrue. Les interactions entre disciplines font ainsi apparaître des « bouclages » qui rendent la simulation du système complexe. Le but final

FIGURE 2.3 – Un Airbus A320.

est de parvenir à stabiliser ces bouclages pour arriver à un équilibre entre les disciplines.

Une discipline peut en fait être elle-même un modèle plus ou moins pluridisciplinaire. La simulation du groupe propulsif de l'appareil fait par exemple intervenir les différents domaines liés aux turbomachines (de l'aérodynamique, de l'énergétique, de la thermodynamique, des matériaux, etc.). La notion de discipline est donc liée à une description donnée du système qu'on considère. Les modèles utilisés par chacune des disciplines peuvent aussi être de fidélités diverses. Comme nous l'avons remarqué auparavant, la simulation du groupe propulsif de l'appareil peut se faire d'un point de vue macroscopique ou bien avec des modèles plus détaillés. Dans le cadre d'une optimisation en phase d'étude conceptuelle, on utilise généralement des modélisations simplifiées afin de pouvoir effectuer un grand nombre de simulations en un temps raisonnable.

2.3 Définition du problème d'optimisation du design d'avion

L'avion étant modélisé par un système multidisciplinaire adapté pour une étude conceptuelle, nous pouvons maintenant envisager son optimisation. Les spécifications de l'avion que nous allons étudier sont proches de celles d'un A320 (voir la figure 2.3). Il s'agit d'un avion de design standard et de taille moyenne, pourvu de deux moteurs et prévu pour embarquer entre 150 et 180 passagers. L'avion va être optimisé pour effectuer une mission moyen-courrier de 3500 NM (soit 6482 km), qu'on appellera la « mission optimale ». Une seconde mission de référence de 500 NM (soit 926 km) sera aussi simulée afin de pouvoir comparer les performances du nouvel avion par rapport aux avions existants.

Dans ce problème, le design de l'avion est gouverné par 10 paramètres. Les huit premiers définissent la géométrie générale de la voilure (qui est représentée sur la figure 2.4), et les deux derniers sont des paramètres opérationnels (les altitudes de croisière durant les missions optimale et de référence).

L'objectif de cette étude est de minimiser la masse à vide équipée f_1 de l'avion ainsi que la masse de fuel f_2 nécessaire à la réalisation de la mission optimale (qui correspond à la masse de carburant brûlé au cours de la mission et à la réserve de carburant supplémentaire emporté).

Les solutions de ce problème d'optimisation sont limitées par six contraintes (en plus des contraintes de borne des différents paramètres de design). La moitié d'entre elles sont des contraintes opérationnelles qui proviennent de la règlementation aéronautique, et les autres des contraintes géométriques qui assurent la cohérence physique de la forme des

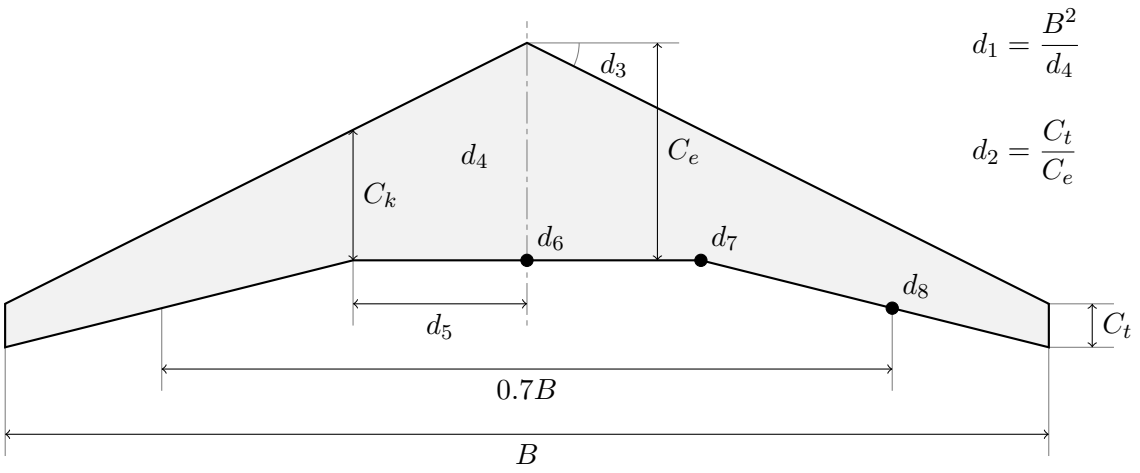


FIGURE 2.4 – Paramètres de la voilure de l'avion.

ailes de l'avion.

Un résumé de l'ensemble des paramètres, des objectifs et des contraintes du problème est donné dans le tableau 2.1. Si on note $\boldsymbol{d}$ le vecteur des 10 variables de design de l'avion et $\boldsymbol{g}$ le vecteur des 6 contraintes, le problème d'optimisation multiobjectifs et multidisciplinaire qu'on souhaite résoudre est donc le suivant :

$$\begin{aligned} &\underset{\boldsymbol{d}}{\text{minimiser}}\ \{f_1(\boldsymbol{d}), f_2(\boldsymbol{d})\}, \\ &\text{s.c.}\ \begin{cases} \boldsymbol{g}(\boldsymbol{d}) \leq \boldsymbol{0}, \\ \boldsymbol{d}_{INF} \leq \boldsymbol{d} \leq \boldsymbol{d}_{SUP}, \end{cases} \end{aligned} \tag{2.1}$$

avec $\boldsymbol{d}_{INF}, \boldsymbol{d}_{SUP} \in \mathbb{R}^{10}$ les bornes inférieures et supérieures des paramètres de design indiquées dans le tableau 2.1.

2.4 Implémentation de la méthodologie d'optimisation multidisciplinaire

Afin de pouvoir résoudre le problème présenté ci-dessus, les modèles des différentes disciplines ont été importés dans Model Center, un logiciel d'optimisation multidisciplinaire. Ce logiciel permet de gérer automatiquement les communications entre les codes des disciplines à partir des liens entre variables partagées qui lui ont été indiqués. Il offre aussi la possibilité de définir des boucles de convergence et d'optimisation.

Les liens à mettre en œuvre entre les disciplines dépendent de l'approche d'optimisation multidisciplinaire choisie. On peut en effet résoudre le problème par une formulation globale (c'est-à-dire en considérant le système multidisciplinaire comme une seule et même entité), ou bien par des approches multi-niveaux dans lesquelles on trouve des boucles d'optimisation locales aux disciplines et d'autres qui gèrent l'optimisation de manière plus globale.

La méthodologie qui a été employée ici est une approche globale MDF (pour « Multi Discipline Feasible », aussi appelée approche « all-in-one »). La simplicité et la rapidité des

Variables de décision			
d_1	Allongement λ	[5,15]	
d_2	Effilement ε	[0.1,0.5]	
d_3	Flèche φ	[0,30]	m
d_4	Surface de la voilure S	[90,200]	m^2
d_5	Position de la cassure Y_k	[5,10]	m
d_6	Épaisseur relative de la voilure à l'emplanture E_e	[0.12,0.16]	
d_7	Épaisseur relative de la voilure au niveau de la cassure E_k	[0.1,0.14]	
d_8	Épaisseur relative de la voilure à 70% de l'envergure E_r	[0.09,0.12]	
d_9	Altitude de croisière de la mission de référence $H_{\text{réf}}$	[9448,12450]	m
d_{10}	Altitude de croisière de la mission optimale H_{opt}	[9448,12450]	m
Paramètre environnemental			
e_1	Erreur sur la surface de l'empennage vertical	0	m^2
Sortie disciplinaire affectée par e_1			
$D_{1,1}$	Masse de l'empennage vertical W_{empV}		kg
Incertitude			
χ_{e_1}	Erreur sur la surface de l'empennage vertical	$U(-5,5)$	m^2
Objectifs			
f_1	Masse à vide équipée MTOW	minimiser	kg
f_2	Masse de fuel pour la mission optimale W_{fuel}	minimiser	kg
Contraintes			
g_1	Vitesse d'approche V_{app}	< 72	$m.s^{-1}$
g_2	Longueur de décollage $L_{\text{déco}}$	< 2200	m
g_3	Envergure B	< 35.95	m
g_4	Différence entre les cordes à la cassure et à l'extrémité $(C_k - C_t)$	> 0	m
g_5	Différence entre les épaisseurs relatives à l'emplanture et à la cassure $(d_6 - d_7)$	> 0	
g_6	Différence entre les épaisseurs relatives à la cassure et à 70% de l'envergure $(d_7 - d_8)$	> 0	

TABLE 2.1 – Variables, objectifs et contraintes du problème d'optimisation multidisciplinaire du design d'un avion.

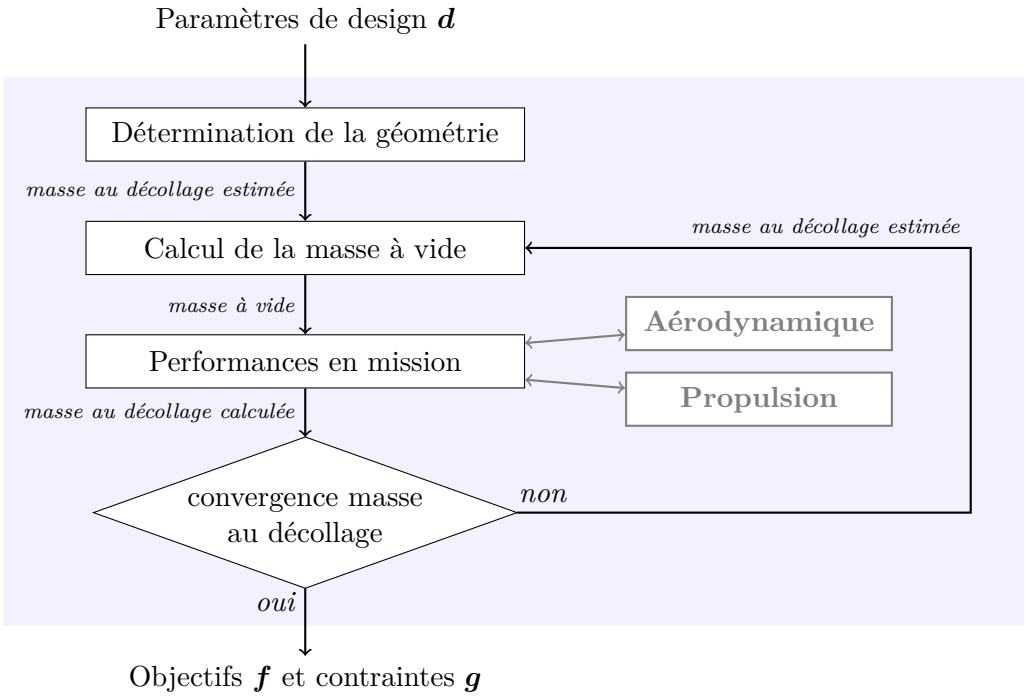


FIGURE 2.5 – Déroulement d'une simulation de l'avion.

modèles disciplinaires choisis dans le cadre d'une étude conceptuelle permettent en effet de considérer le système comme un tout et de l'évaluer dans son ensemble pour réaliser une optimisation globale. Le système n'en reste pas moins multidisciplinaire pour autant, et la recherche d'un équilibre entre toutes ses disciplines à chaque évaluation de configuration reste nécessaire.

Le déroulement général du calcul d'une configuration donnée est présenté sur la figure 2.5. A partir des paramètres de design $\boldsymbol{d}$, la géométrie de l'avion est tout d'abord déterminée. Le bouclage principal se déroule par la suite sur le calcul de la masse au décollage de l'avion. On affecte à cette masse une valeur initiale estimée quelconque, ce qui permet de définir la structure de l'appareil et de calculer sa masse à vide. Les performances de l'avion sont ensuite simulées au cours d'une mission pour obtenir la quantité de carburant qu'il est nécessaire d'emporter. Ajoutée à la masse à vide, la masse de carburant ainsi obtenue permet de déterminer la masse au décollage « calculée » de l'appareil. Tant que cette masse ne correspond pas à celle qui avait été estimée initialement, on recommence une itération en prenant comme initialisation la nouvelle valeur de la masse au décollage. Lorsque la convergence en masse est atteinte, les performances réelles de l'avion peuvent être calculées et les valeurs des objectifs et des contraintes sont fournies en sortie.

Model Center permet aussi de sélectionner différents algorithmes de résolution. C'est un algorithme génétique qui a été retenu ici. Ce type d'algorithme permet d'avoir une certaine souplesse d'utilisation en explorant l'espace des solutions sans information particulière sur les propriétés des fonctions objectifs (qui sont vues comme des « boîtes noires »), et en pouvant accepter des arrêts prématurés de calculs. Il est en effet possible qu'au cours de l'optimisation certaines configurations d'avion ne puissent pas être évaluées correctement

(si par exemple l'avion ne parvient pas à décoller), ce qui provoquera l'arrêt des calculs pour le design concerné. Ces configurations seront ignorées par l'algorithme génétique qui saura continuer ses recherches dans d'autres directions.

L'implémentation qu'on possède du système multidisciplinaire de simulation d'un avion complet permet donc de réaliser une optimisation afin de trouver la meilleure configuration possible au vu des objectifs fixés. Mais nous souhaiterions à présent pouvoir aussi tenir compte des incertitudes présentes dans cette modélisation. C'est l'objet de la section suivante.

2.5 Incertitude sur la surface de l'empennage vertical

Dans les premières phases de la conception d'un nouvel avion, les représentations utilisées sont des modèles simplifiés qui permettent d'évaluer de façon approchée les performances du nouvel appareil. Il est toutefois primordial pour les constructeurs de faire des choix réfléchis durant ces étapes préliminaires car elles déterminent le déroulement de la suite de la conception et tout retour en arrière est coûteux. Les approximations réalisées par les modèles peuvent donc avoir un fort impact si jamais le design sélectionné lors de l'étude conceptuelle se révèle trop sensible aux incertitudes. On observe généralement par exemple une dérive de la masse de l'appareil au fil des différentes phases de conception : l'avion final est souvent plus lourd que ce qui avait été prévu initialement car différents aménagements ont du être effectués entre son étude conceptuelle et sa conception finale afin de répondre aux exigences en termes de fiabilité et de respect des réglementations.

Les constructeurs aéronautiques souhaitent prendre un minimum de risques lors de la conception d'un nouvel appareil afin de s'assurer que sa conception pourra se dérouler comme prévu. La prise en compte des incertitudes dès la phase d'étude préliminaire est donc un sujet d'intérêt aujourd'hui. De plus, les performances des concepts standards d'avion arrivent désormais à leurs limites. Il devient donc nécessaire d'étudier de nouveaux concepts, assez proches des appareils existants ou bien plus éloignés. L'estimation des performances de ces concepts atypiques est difficile car on ne possède pas de base de référence pour valider les modèles physiques utilisés. Il est donc là aussi primordial de pouvoir prendre en compte les erreurs de modélisation afin de sélectionner un design relativement robuste pour lequel on aura plus de garanties de succès.

Dans le modèle de l'avion que nous utilisons, l'équilibrage est réalisé par des lois simplifiées. L'empennage est notamment dimensionné grâce à des formules statistiques alors qu'en réalité il faudrait tenir compte des contraintes opérationnelles. Dans le cas d'une panne d'un des moteurs de l'appareil par exemple, il faudrait définir la surface de l'empennage vertical de manière à ce que sa gouverne puisse contrer les effets du moment créé par l'utilisation d'un seul moteur, mais cela n'est pas considéré ici.

Les études de validation du modèle montrent qu'on observe une incertitude de 5 m^2 sur la surface de l'empennage vertical. Une variable e_1 est donc ajoutée à la surface calculée pour représenter l'erreur commise sur son estimation. Dans le cas nominal, on considère qu'il n'y a pas d'erreur et donc e_1 vaut 0. Mais il se peut aussi que e_1 varie dans l'intervalle $[-5, 5]$. Pour représenter cette incertitude, on remplace e_1 par une variable aléatoire χ_{e_1} dotée d'une loi de probabilité uniforme sur cet intervalle.

L'incertitude sur la surface de l'empennage vertical affecte le calcul de la masse de l'empennage vertical (notée $D_{1,1}$), et par là même la masse totale de l'avion ainsi que sa consommation. Il va donc falloir prendre en compte cette nouvelle variable incertaine dans

le problème :

$$\begin{aligned} &\underset{\boldsymbol{d}}{\text{minimiser}} \ \{f_1(\boldsymbol{d}, \chi_{e_1}), f_2(\boldsymbol{d}, \chi_{e_1})\}, \\ &\text{s.c.} \ \begin{cases} \boldsymbol{g}(\boldsymbol{d}, \chi_{e_1}) \leq \mathbf{0}, \\ \boldsymbol{d}_{INF} \leq \boldsymbol{d} \leq \boldsymbol{d}_{SUP}. \end{cases} \end{aligned} \tag{2.2}$$

Nous avons affaire à un problème d'optimisation multidisciplinaire stochastique. Le fait de prendre en compte cette variable incertaine dans l'optimisation permettra de trouver des designs dont les performances ne seront pas trop affectées par l'erreur de calcul de la surface de l'empennage vertical. Ainsi, si dans les phases suivantes de la conception de l'avion l'empennage vertical est dimensionné avec une surface légèrement différente, l'avion restera aussi performant qu'on l'attendait.

Deuxième partie

État de l'art

Chapitre 3

Modèles de substitution

Sommaire

3.1 Modèles de substitution et apprentissage 36
3.2 Polynômes 37
3.3 Réseaux de neurones 39
3.3.1 Neurone artificiel 40
3.3.2 Réseau de neurones 41
3.3.3 Mise en œuvre des réseaux de neurones 44
3.4 Krigeage 44
3.4.1 Modèle de krigeage 44
3.4.2 Erreur du modèle de krigeage 46
3.4.3 Simulation de processus gaussien 47
3.4.4 Modèle de krigeage régressant 48
3.4.5 Mise en œuvre du krigeage 50
3.5 Autres modèles de substitution 50
3.6 Comparaison des types de modèles de substitution 51
3.7 Erreur d'un modèle de substitution 52
3.7.1 Contrôle de l'erreur durant l'apprentissage 52
3.7.2 Estimation de l'erreur d'un modèle 54
3.8 Plans d'expériences 57
3.8.1 Plan d'expériences initial 58
3.8.2 Complétion adaptative du plan d'expériences initial 59
3.8.3 Simulation de l'ajout d'un nouvel échantillon d'apprentissage 61

Les simulations numériques complexes qui sont employées pour représenter ou concevoir des systèmes sont souvent extrêmement coûteuses. Leur utilisation directe pour des applications telles que l'optimisation, le contrôle ou la propagation d'incertitudes est problématique du fait du grand nombre d'évaluations à réaliser. C'est pourquoi on cherche à remplacer ces simulations par des modèles approchés plus rapides. Ces modèles autorisent un nombre d'appels plus important, rendant ainsi envisageables les différentes applications mentionnées. Plusieurs approches existent pour construire de tels modèles.

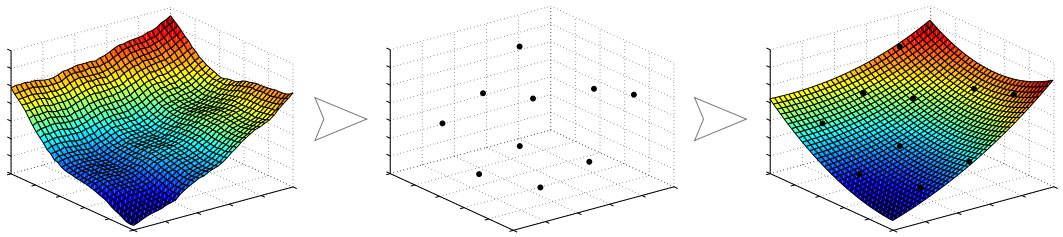


FIGURE 3.1 – Construction d'un modèle de substitution à partir d'échantillons d'un code de calcul coûteux.

Une première possibilité consiste à réduire lorsque cela est possible la complexité de la simulation grâce à des lois de la physique simplifiées ou bien une discrétisation numérique. Si les équations de la physique régissant le système sont accessibles, cela permet d'obtenir un modèle approché plus rapide et qui conserve son sens physique. Diverses fidélités de simulation peuvent ainsi être définies en fonctions des lois de la physique considérées ou bien de la finesse de la discrétisation numérique. Mais le gain en réactivité apporté par ces simplifications n'est pas toujours suffisant pour certaines applications, ou bien cela se fait aux dépens de la physique et le comportement du système n'est alors plus représenté avec suffisamment de précision.

Une seconde solution que nous présentons ici est d'établir un modèle de substitution à partir de quelques résultats numériques réellement calculés avec la simulation coûteuse. Ces modèles sont construits indépendamment de la physique du problème étudié, mais n'en restent pas moins fiables. Leur fidélité peut par ailleurs être adaptée à l'application envisagée, et ils présentent un temps d'évaluation négligeable qui permet de multiplier les évaluations. Ils fournissent aussi généralement une expression analytique de la fonction représentée, donnant accès à certaines informations utiles comme le gradient de cette fonction par exemple.

Nous présentons dans ce chapitre le principe général de l'apprentissage automatique, ainsi que différents types de modèles de substitution qui peuvent être utilisés : les approximations polynomiales, les réseaux de neurones et les modèles de krigeage. Nous nous intéressons aussi aux méthodes d'amélioration de la précision des modèles grâce à la construction de plans d'expériences adaptatifs.

3.1 Modèles de substitution et apprentissage

Un modèle de substitution (aussi appelé « métamodèle », « modèle réduit » ou encore parfois « surface de réponse ») est une approximation d'une fonction dont l'évaluation est coûteuse. L'évaluation du modèle de substitution est quant à elle très rapide, ce qui permet d'avoir en tout point de l'espace une estimation de la valeur de la fonction coûteuse sans avoir à l'évaluer réellement. Comme présenté sur la figure 3.1, la construction d'un modèle de substitution est réalisée à partir de quelques échantillons de la fonction initiale, sans aucun a priori sur sa physique ou ses particularités.

Le modèle de substitution, que nous noterons $\hat{f}$, doit réaliser un apprentissage supervisé

de la fonction coûteuse $f : \mathbb{R}^{n_x} \to \mathbb{R}$ sur un espace $\boldsymbol{X} \subseteq \mathbb{R}^{n_x}$ donné. Le principe est d'apprendre à prédire de manière automatique le comportement de cette fonction à partir des quelques exemples qui sont fournis. Il faut que le modèle soit capable de généraliser de manière raisonnable ce qu'il a appris grâce aux échantillons, afin de pouvoir estimer la valeur de la fonction pour n'importe quel point d'entrée.

L'apprentissage est réalisé à partir d'une base d'apprentissage constituée de n_s échantillons sous la forme de couples $(\boldsymbol{s}^i, u^i)$, avec $\boldsymbol{s}^i = (s_1^i, s_2^i, ..., s_{n_x}^i) \in \boldsymbol{X} \subseteq \mathbb{R}^{n_x}$ et $u^i \in \mathbb{R}$. L'ensemble des valeurs d'entrée des échantillons est noté $\boldsymbol{S} = (\boldsymbol{s}^1, \boldsymbol{s}^2, ..., \boldsymbol{s}^{n_s}) \in M_{n_s,n_x}(\mathbb{R})$, et l'ensemble de leurs images par f est noté $\boldsymbol{u} = (u^1, u^2, ..., u^{n_s}) \in \mathbb{R}^{n_s}$. Pour chaque échantillon i, on a donc $u^i = f(\boldsymbol{s}^i) \in \mathbb{R}$.

Avant de construire un modèle de substitution, il est préférable de normaliser les échantillons d'apprentissage. On obtient en effet de meilleurs résultats lorsque les données sont ramenées dans un intervalle restreint, aussi bien au niveau des entrées $\boldsymbol{S}$ que des sorties $\boldsymbol{u}$. Cela permet de s'assurer que les différentes variables auront la même influence sur le modèle (particulièrement dans le cas où l'un des domaines de variation est beaucoup plus étendu que les autres), et permet de plus au modèle de n'avoir à traiter que des valeurs « raisonnables » (ce qui évite de saturer ses paramètres). Deux types de normalisation existent : d'une part la normalisation vers l'intervalle $[-1, 1]$, et d'autre part la normalisation gaussienne visant à centrer et réduire les données. L'une ou l'autre peut être utilisée indifféremment dans la plupart des cas. Nous avons par exemple choisi pour cette étude d'effectuer une normalisation gaussienne à partir de l'estimation de la moyenne et de la variance des échantillons d'apprentissage. Nous supposerons donc par la suite que les données traitées par les modèles de substitution sont normalisées au préalable et dé-normalisées en sortie de manière transparente.

Pour représenter la fonction initiale à partir de la base d'apprentissage $(\boldsymbol{S}, \boldsymbol{u})$, il existe différentes familles de modèles de substitution. Elles se distinguent par les fonctions génératrices et les techniques d'apprentissage qu'elles utilisent. Selon le cas, elles permettent de réaliser une interpolation ou bien une régression sur les échantillons, et peuvent modéliser un degré plus ou moins grand de non linéarité. Nous décrivons dans la suite trois des principaux types de modèles de substitution utilisés actuellement : les approximations polynomiales, les réseaux de neurones ainsi que le krigeage.

3.2 Polynômes

Les modèles à base d'approximations polynomiales sont les modèles les plus simples. Ils sont parfois historiquement associés au terme de « surface de réponse » car ils étaient utilisés au départ comme aide à l'optimisation. Une approximation polynomiale $\hat{f}$ s'écrit sous la forme d'un polynôme multivarié à n_x variables et de degré fixé k :

$$\hat{f}(\boldsymbol{x}) = \sum_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_1 \leq k} c_{\boldsymbol{\alpha}} \boldsymbol{x}^{\boldsymbol{\alpha}}, \tag{3.1}$$

avec :

$$\begin{cases} \boldsymbol{x} = (x_1, x_2, ..., x_{n_x}) \in \mathbb{R}^{n_x}, \\ \boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_{n_x}) \in \mathbb{N}^{n_x}, \\ \|\boldsymbol{\alpha}\|_1 = \sum_{i=1}^{n_x} \alpha_i, \\ c_{\boldsymbol{\alpha}} \in \mathbb{R}, \\ \boldsymbol{x}^{\boldsymbol{\alpha}} = \prod_{i=1}^{n_x} x_i^{\alpha_i}. \end{cases}$$

Un polynôme est donc tout simplement une somme de monômes $\boldsymbol{x}^{\boldsymbol{\alpha}}$ pondérés par des coefficients $c_{\boldsymbol{\alpha}}$ à déterminer. Un polynôme de degré trois à deux variables peut par exemple s'écrire :

$$\hat{f}(x_1, x_2) = 3 + 2x_1 - x_2^2 + x_1^2 x_2.$$

La base polynomiale standard est composée des monômes $\boldsymbol{x}^{\boldsymbol{\alpha}}$, mais il existe aussi des modèles polynomiaux construits sur des bases différentes. On peut par exemple citer les polynômes de chaos qui utilisent notamment les bases d'Hermite ou de Legendre. Ces modèles polynomiaux particuliers sont présentés dans la section 5.8.2. Les bases polynomiales sont dans tous les cas uniquement composées de puissances de $\boldsymbol{x}$ et les modèles peuvent donc se réécrire sous la forme générale présentée ci-dessus.

Pour établir une approximation polynomiale d'une fonction, il faut déterminer $\dfrac{(k+n_x)!}{k!n_x!}$ coefficients $c_{\boldsymbol{\alpha}}$. On a donc besoin d'au moins autant d'échantillons de la fonction à approcher. Le calcul de ces coefficients peut se faire simplement par la résolution d'un système d'équations linéaires. Lorsque le nombre d'échantillons disponibles est égal au nombre de coefficients du modèle, on obtient un modèle interpolant. Mais on peut aussi prendre un nombre plus important d'échantillons afin de construire un modèle de régression. Les coefficients sont alors déterminés par la résolution d'un problème d'optimisation visant à minimiser la distance entre le modèle et les échantillons. On utilise généralement pour cela la méthode des moindres carrés, c'est-à-dire la minimisation du carré de la distance euclidienne entre les échantillons et leur valeur estimée par le modèle. Le difficulté des modèles polynomiaux est que le nombre de coefficients $c_{\boldsymbol{\alpha}}$ à évaluer croît très rapidement en fonction de la dimension de l'espace d'entrée et du degré du polynôme, complexifiant d'autant plus le problème de leur estimation.

L'établissement d'un modèle polynomial se fait donc par résolution d'un système d'équations linéaires possiblement surdéterminé (voire parfois sous-déterminé). Si on note $\boldsymbol{c}$ le vecteur des coefficients $c_{\boldsymbol{\alpha}}$ à déterminer, le système à résoudre est le suivant :

$$\boldsymbol{A}\boldsymbol{c} = \boldsymbol{u}, \tag{3.2}$$

avec :

$$\begin{cases} \boldsymbol{A} = \left[(s^i)^{\boldsymbol{\alpha}}\right]_{1 \leq i \leq n_s, \|\boldsymbol{\alpha}\|_1 \leq k}, \\ \boldsymbol{c} = (c_{\boldsymbol{\alpha}})_{\|\boldsymbol{\alpha}\|_1 \leq k}, \\ \boldsymbol{u} = (u^i)_{1 \leq i \leq n_s}. \end{cases}$$

Lorsque le système possède un nombre différent d'équations et d'inconnues, la résolution se fait par la méthode des moindres carrés. Pour cette étude, nous avons utilisé la bibliothèque Lapack [ABB+99] à cet effet.

Pour pallier l'augmentation du nombre de coefficients à déterminer lorsque l'ordre du modèle augmente, on peut aussi utiliser des polynômes « creux ». Il s'agit de modèles dont tous les monômes $\boldsymbol{x}^{\boldsymbol{\alpha}}$ ne sont pas nécessairement présents (certains coefficients $c_{\boldsymbol{\alpha}}$ restent

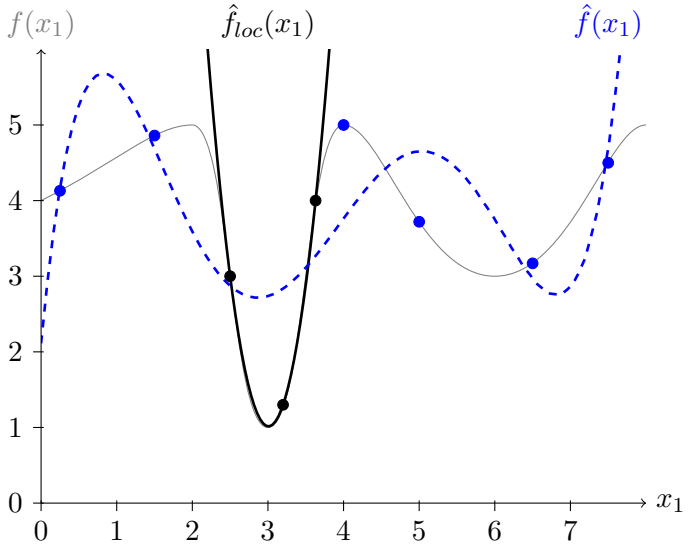


FIGURE 3.2 – Représentation d'une fonction f (en gris) par un polynôme local $\hat{f}_{loc}$ d'ordre 2 construit à partir de 3 échantillons d'apprentissage (points noirs), et par une régression polynomiale globale $\hat{f}$ d'ordre 5 construite à partir de 9 échantillons (points noirs et points bleus).

nuls). La détermination des coefficients en nombre restreint est donc plus simple, mais il faut déterminer intelligemment quels monômes faire apparaître ou pas dans la formulation afin de pouvoir représenter correctement la fonction à approcher.

L'intérêt des modèles polynomiaux réside dans leur simplicité, autant de construction que d'utilisation. Selon l'ordre du modèle, certains points particuliers peuvent aussi être calculés (comme par exemple le calcul de l'extremum d'un polynôme d'ordre 2). Les polynômes permettent de réaliser de bonnes approximations locales d'une fonction, mais sur des étendues plus grandes on se heurte rapidement à des problèmes de représentation des extrema locaux : un polynôme d'ordre 2 ne possède par exemple qu'un seul extremum, et ne pourra donc pas approcher correctement une fonction possédant deux minima. Des oscillations artificielles peuvent aussi apparaître lorsque le degré du modèle polynomial utilisé est trop important. On les utilisera donc principalement pour établir des modèles locaux simples et rapides sur une région de confiance donnée.

La figure 3.2 présente un exemple de polynômes local et global approchant une fonction donnée f. Si le polynôme local permet une bonne approximation du minimum de f, le modèle global est quant à lui bien loin de la réalité. Pour l'améliorer, il faudrait augmenter l'ordre du polynôme et ajouter d'autres échantillons d'apprentissage.

3.3 Réseaux de neurones

Les réseaux de neurones sont un second type de modèle de substitution qui peut être utilisé pour approcher une fonction. Un réseau de neurones artificiels est un modèle qui s'inspire schématiquement du fonctionnement des neurones biologiques. S'il est utilisé dans de nombreuses applications différentes, nous ne nous intéresserons ici qu'à ses ca-

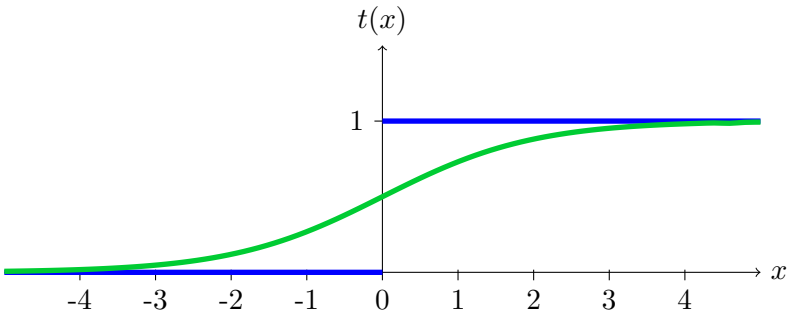


FIGURE 3.3 – Fonction de Heaviside (avec un seuil $k = 0$, en bleu), et sigmoïde (avec $\lambda = 1$, en vert).

pacités d'approximation de fonctions. Une présentation générale des réseaux de neurones est donnée dans [DSM+04]. Nous commencerons par décrire le fonctionnement d'un neurone artificiel avant de les assembler en réseau de plusieurs couches, appelé « Multi Layer Perceptron » (MLP).

3.3.1 Neurone artificiel

Un neurone est une entité qui possède plusieurs entrées $a_i \in \mathbb{R}$ (qu'on peut regrouper en un vecteur $\boldsymbol{a} = (a_1, ..., a_n) \in \mathbb{R}^n$) et une sortie réelle z qui dépend de ces entrées. Pour calculer la valeur de sa sortie, le neurone réalise une somme pondérée de ses entrées (chaque entrée étant affectée d'un poids $w_i \in \mathbb{R}$) ainsi que d'un biais w_0, avant d'y appliquer une fonction d'activation t :

$$z(\boldsymbol{a}) = t\left(w_0 + \sum_{i=1}^{n} w_i a_i\right). \tag{3.3}$$

Le comportement d'un neurone dépend donc du vecteur de réels $w = (w_0, w_1, ..., w_n)$ et de la fonction t. Les fonctions d'activation les plus utilisées sont les suivantes :

Fonction de Heaviside Aussi appelée fonction seuil, elle permet d'activer le neurone lorsqu'on dépasse une certaine valeur seuil $k \in \mathbb{R}$ fixée (voir l'illustration de la figure 3.3) :

$$t(x) = \begin{cases} 0 & \text{si } x < k, \\ 1 & \text{si } x \geq k. \end{cases}$$

La fonction de Heaviside est utilisée pour construire des réseaux de neurones à des fins de classification, car elle permet de faire un choix binaire « tout ou rien ».

Fonction sigmoïde Il s'agit d'une fonction différentiable en forme de « S » qui met en place une sorte de seuil comme la fonction de Heaviside mais de manière beaucoup moins brutale. Elle dépend d'un paramètre λ qui permet de régler l'intensité de la variation au niveau de ce seuil (voir l'exemple sur la figure 3.3) :

$$t(x) = \frac{1}{1 + e^{-\lambda x}}.$$

D'autres fonctions au comportement similaire sont aussi utilisées.

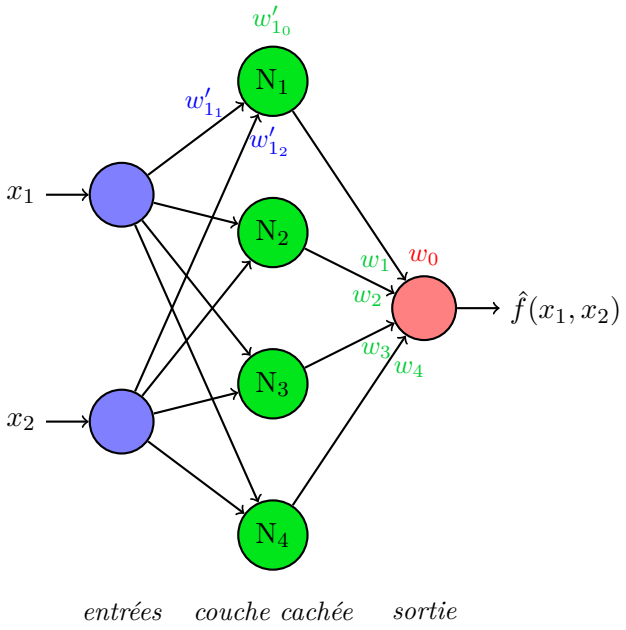


FIGURE 3.4 – Réseau de neurones possédant deux entrées, une couche cachée constituée de quatre neurones et une sortie.

Fonction linéaire Elle revient à ne pas utiliser de fonction d'activation et à transmettre directement la somme pondérée des entrées en sortie du neurone :

$$t(x) = x.$$

Un réseau de neurones qui serait uniquement basé sur cette fonction aurait une sortie totalement linéaire par rapport à ses entrées.

3.3.2 Réseau de neurones

Un réseau de neurones est une interconnexion de neurones artificiels répartis en différentes couches. Dans le cas qui nous concerne, la couche d'entrée est constituée des variables d'entrée $\boldsymbol{x} = (x_1, x_2, ..., x_{n_x})$ de la fonction à modéliser, et la couche de sortie ne possède qu'un seul neurone qui donne la valeur $\hat{f}(\boldsymbol{x})$ du modèle en sortie (on peut aussi construire des réseaux de neurones possédant plusieurs sorties, mais nous n'aborderons pas cette particularité ici). Les couches intermédiaires situées entre la couche d'entrée et celle de sortie sont appelées « couches cachées ». Chaque neurone N_i des couches cachées prend en entrée l'ensemble des sorties des neurones de la couche précédente. Un exemple de réseau de neurones possédant une seule couche cachée est donné sur la figure 3.4.

Régression

Toute fonction peut être approchée avec une précision arbitraire par un réseau de neurones à trois couches (c'est-à-dire possédant une unique couche cachée) utilisant des fonctions d'activation sigmoïdes et une sortie linéaire [Cyb89, HSW89, Fun89]. On utilise donc généralement un réseau de neurones à une couche cachée pour construire un modèle

de substitution d'une fonction. En notant n_n le nombre de neurones de cette couche cachée, le modèle $\hat{f}$ s'écrit alors :

$$\hat{f}(\boldsymbol{x}) = w_0 + \sum_{i=1}^{n_n} w_i t \left(w'_{i_0} + \sum_{j=1}^{n_x} w'_{i_j} x_j \right) = w_0 + \sum_{i=1}^{n_n} w_i t(w'_{i_0} + \boldsymbol{x}^T \boldsymbol{w}'_i), \tag{3.4}$$

où $(w_1, w_2, ..., w_{n_n}) \in \mathbb{R}^{n_n}$ représentent les poids des entrées du neurone de sortie et $\boldsymbol{w}'_i = (w'_{i_1}, w'_{i_2}, ..., w'_{i_{n_x}}) \in \mathbb{R}^{n_x}$ les poids des entrées du neurone N_i de la couche cachée. La fonction t est une fonction sigmoïde.

L'approximation d'une fonction f par un réseau de neurones se fait en déterminant les $(1+n_n(1+n_x))$ poids w, ainsi que le nombre n_n de neurones de sa couche cachée grâce aux échantillons d'apprentissage qu'on possède. L'apprentissage se fait donc en deux temps : il faut tout d'abord être capable de trouver les coefficients optimaux du réseau pour un nombre fixé de neurones sur la couche cachée, puis on peut ensuite optimiser ce nombre de neurones cachés afin que le réseau représente au mieux la fonction à approcher.

Rétro-propagation

Un réseau de neurones est un modèle non interpolant pour lequel on cherche à minimiser l'erreur commise sur un ensemble de n_s échantillons d'apprentissage $(\boldsymbol{s}^i, u^i)$ de la fonction. Il faut donc résoudre un problème de moindres carrés visant à minimiser une fonction coût globale c :

$$c(\boldsymbol{w}) = \sum_{i=1}^{n_s} c^i(\boldsymbol{w}) = \sum_{i=1}^{n_s} \|u^i - \hat{f}(\boldsymbol{s}^i)\|^2, \tag{3.5}$$

où $\boldsymbol{w} \in \mathbb{R}^{1+n_n(1+n_x)}$ représente l'ensemble des poids du réseau et c^i la fonction coût associée à un unique échantillon $(\boldsymbol{s}^i, u^i)$. Lorsque les fonctions d'activation t du réseau de neurones ne sont pas linéaires (comme dans le cas des sigmoïdes par exemple), on a affaire à un problème d'optimisation non linéaire. Les fonctions sigmoïdes étant par contre dérivables, il est possible de calculer analytiquement le gradient de c par rapport aux coefficients du modèle. La résolution du problème de moindres carrés peut donc se faire à l'aide de méthodes itératives de descente.

La principale technique utilisée dans l'apprentissage d'un réseau de neurones pour calculer le gradient de la fonction coût c est l'algorithme de rétro-propagation [RHW86]. Son nom vient du fait qu'elle vise à calculer ce gradient de manière récursive en commençant par la sortie du réseau et en remontant vers les entrées. Lorsqu'un échantillon d'apprentissage $\boldsymbol{s}^i$ est présenté à l'entrée du réseau, on peut comparer sa valeur « réelle » u^i sur la fonction initiale par rapport à la valeur $\hat{f}(\boldsymbol{s})$ calculée en sortie par le réseau de neurones. Le gradient de cette erreur vis-à-vis des poids des entrées du neurone de sortie peut alors être calculé. Le calcul des gradients par rapport aux poids des neurones de la couche cachée se fait ensuite à partir de l'erreur rétro-propagée venant du neurone de sortie. L'erreur remonte ainsi à travers le réseau pour obtenir les gradients de c^i relativement à tous les poids $\boldsymbol{w}$. Ces poids pourront alors être modifiés en fonction du gradient par la méthode d'optimisation retenue.

Selon les méthodes d'apprentissage, la mise à jour des poids du réseau se fait pour chaque nouvel échantillon à apprendre, ou bien une fois que tous les échantillons ont été présentés (on cumule alors les erreurs commises sur l'ensemble des échantillons avant de

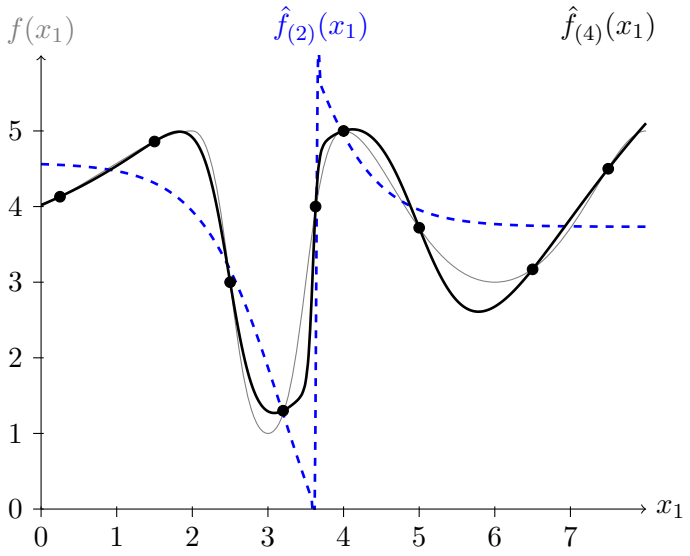


FIGURE 3.5 – Approximation d'une fonction f (en gris) à partir de 9 échantillons d'apprentissage (points noirs) par des réseaux de neurones $f_{(2)}$ et $f_{(4)}$ possédant respectivement deux et quatre neurones sur leur couche cachée.

modifier les poids des neurones). Dans tous les cas, les échantillons sont généralement pris en compte plusieurs fois afin d'atteindre une certaine convergence du réseau.

Nombre de neurones de la couche cachée

Sachant trouver les coefficients optimaux d'un réseau possédant un nombre donné de neurones cachés, on peut maintenant optimiser le nombre n_n de ces neurones. Ce paramètre influe en effet sur les capacités de représentation du réseau. Si le nombre de neurones de la couche cachée est trop petit, le réseau ne pourra pas approcher correctement la fonction à modéliser. Si au contraire n_n est trop grand, le nombre de coefficients dans le réseau sera important et il sera d'autant plus difficile de trouver les valeurs optimales de ces coefficients à partir du nombre limité d'échantillons d'apprentissage qu'on possède.

Deux exemples de réseaux de neurones sont présentés sur la figure 3.5. Ils ont été construits à partir des neuf points d'apprentissage utilisées précédemment pour la construction d'approximations polynomiales dans la figure 3.2. Le premier réseau possède deux neurones sur sa couche cachée, et le second en possède quatre. On voit que le réseau doté de deux neurones seulement a du mal à approcher la fonction f du fait de cette limitation, alors que le réseau avec quatre neurones y arrive bien mieux. Cela souligne l'importance de choisir un nombre de neurones cachés adapté à la fonction qu'on souhaite représenter.

Plusieurs stratégies existent pour déterminer le nombre optimal de neurones de la couche cachée. On peut tout d'abord construire plusieurs réseaux avec des nombres de neurones différents pour ne conserver que le meilleur d'entre eux. On peut aussi établir successivement des réseaux de neurones possédant de plus en plus de neurones, en arrêtant de rajouter des neurones dans la couche cachée lorsque l'erreur commise par le modèle ne diminue plus, ou bien a contrario partir d'un réseau doté d'un grand nombre de neurones et en enlever un à chaque itération (mais on aura alors plus de coefficients à optimiser).

3.3.3 Mise en œuvre des réseaux de neurones

Pour cette étude, nous nous sommes basés sur la bibliothèque FANN (« Fast Artificial Neural Network » [Nis03]) pour la création de réseaux de neurones. Il s'agit d'un code écrit en C qui permet de déterminer les poids d'un réseau relativement à un ensemble d'échantillons d'apprentissage. Plusieurs types de réseaux de neurones dotés de différentes fonctions d'activation peuvent ainsi être construits, en utilisant diverses méthodes d'apprentissage. Le nombre de neurones composant la couche cachée doit cependant être spécifié a priori par l'utilisateur. Il a donc été nécessaire de compléter cette bibliothèque pour pouvoir déterminer le nombre optimal de neurones et générer des modèles de substitution plus précis.

3.4 Krigeage

Le krigeage (ou « kriging » en anglais) doit son nom à son auteur D. G. Krige [Kri53]. Il s'agit d'une technique issue à l'origine de l'analyse minière et de la géostatistique [Mat63] et qui fait aujourd'hui partie des méthodes les plus populaires pour la construction de modèles de substitution. Au même titre que les réseaux de neurones, les modèles de krigeage permettent d'approcher des fonctions complexes à partir de la connaissance de quelques échantillons seulement. Le krigeage estime la valeur d'une fonction f en un point donné par une combinaison linéaire des échantillons d'apprentissage. Le poids associé à chaque échantillon dépend de la distance au point considéré, et on suppose que des points rapprochés dans l'espace tendent à posséder des caractéristiques similaires (l'échantillon le plus proche recevra donc le poids le plus important). Le krigeage a l'avantage de fournir en plus une estimation en tout point de l'erreur commise sur la prédiction de la valeur de la fonction f.

3.4.1 Modèle de krigeage

Le krigeage considère que la fonction $f : \mathbb{R}^{n_x} \to \mathbb{R}$ à modéliser est une réalisation d'un processus stochastique gaussien F composé d'une partie déterministe μ et d'une partie aléatoire ϵ :

$$F(\boldsymbol{x}) = \mu(\boldsymbol{x}) + \epsilon(\boldsymbol{x}), \tag{3.6}$$

avec $\boldsymbol{x} \in \mathbb{R}^{n_x}$. μ représente la tendance moyenne du processus, et ϵ est un processus gaussien stationnaire d'espérance nulle ($\mathrm{E}[\epsilon(\boldsymbol{x})] = 0$), de variance $\sigma_K^2 \in \mathbb{R}$ et de covariance :

$$\mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{x}')) = \sigma_K^2 R(\boldsymbol{x}, \boldsymbol{x}'). \tag{3.7}$$

La particularité de cette covariance est que R est une fonction de corrélation telle que la corrélation entre deux points ne dépend que de leur distance. Elle est de la forme :

$$R(\boldsymbol{x}, \boldsymbol{x}') = \prod_{i=1}^{n_x} R_i(|x_i - x_i'|), \tag{3.8}$$

avec $\boldsymbol{x} = (x_1, x_2, ..., x_{n_x}) \in \mathbb{R}^{n_x}$ et $\boldsymbol{x}' = (x_1', x_2', ..., x_{n_x}') \in \mathbb{R}^{n_x}$.

L'établissement d'un modèle de krigeage revient donc à approcher le processus stochastique F par un processus $\hat{F}$. On considère que l'espérance $\hat{f}$ des réalisations de ce processus est une approximation de la fonction f.

Types de krigeage

Plusieurs types de krigeage existent selon la forme donnée à $\mu(\boldsymbol{x})$. Les principaux sont les suivants (allant du plus simple au plus général) :

Krigeage simple $\mu(\boldsymbol{x})$ est une constante connue : $\mu(\boldsymbol{x}) = 0$ par exemple.

Krigeage ordinaire $\mu(\boldsymbol{x})$ est une constante inconnue : $\mu(\boldsymbol{x}) = \mu \in \mathbb{R}$.

Krigeage universel $\mu(\boldsymbol{x})$ est une combinaison linéaire de n fonctions t_i connues, pondérées par des coefficients $\beta_i \in \mathbb{R}$ à déterminer : $\mu(\boldsymbol{x}) = \sum_{i=1}^{n} \beta_i t_i(\boldsymbol{x})$ (on choisit généralement une base de fonctions t_i polynomiales).

Le krigeage universel réalise en fait une première régression grossière à partir des échantillons d'apprentissage grâce à la fonction μ, puis affine le modèle via le second terme ϵ. Si certains préfèrent employer ce type de krigeage, d'autres argumentent qu'il est tout aussi correct d'utiliser une fonction μ plus simple et de laisser ϵ approcher au mieux la fonction. Le krigeage universel serait plus approprié dans le cas où les données présentent une tendance générale qui peut être modélisée par une fonction simple (comme un polynôme par exemple). Dans la pratique, les deux approches donnent des résultats satisfaisants. Nous avons choisi ici de nous concentrer sur le krigeage ordinaire du fait de la plus grande simplicité de son écriture qui facilitera les calculs développés par la suite.

Fonctions de corrélation

Différentes fonctions peuvent être utilisées pour exprimer la corrélation R de la fonction aléatoire ϵ [KO96]. R introduit une hypothèse de régularité de la fonction à modéliser : on suppose que plus les points $\boldsymbol{x}$ et $\boldsymbol{x'}$ sont proches, plus il y a de chances que leurs images respectives par f soient proches elles aussi. Nous avons choisi une fonction gaussienne pour chaque dimension $i \in \{1, 2, ..., n_x\}$ de l'espace :

$$R_i(|x_i - x'_i|) = e^{-\theta_i (x_i - x'_i)^2}, \tag{3.9}$$

où $\theta_i \in \mathbb{R}^+$ est un paramètre à régler. C'est la fonction la plus couramment employée dans les modèles de krigeage.

Le paramètre θ_i traduit l'idée de régularité de la fonction le long de la dimension i : une valeur élevée imposera un comportement plutôt lisse du modèle alors qu'une valeur plus faible provoquera des variations plus importantes. Il est à noter qu'on pourrait aussi considérer des paramètres θ_i qui dépendent de $\boldsymbol{x}$ pour pouvoir représenter des fonctions dont la régularité varie selon les zones de l'espace, mais le modèle serait alors beaucoup plus complexe à établir.

Modèle de krigeage

Nous nous plaçons donc dans le cadre des modèles de krigeage ordinaire à fonction de corrélation gaussienne. Le modèle de krigeage $\hat{F}$ défait de sa valeur moyenne μ est défini comme une combinaison linéaire des échantillons d'apprentissage u^i, dont les poids λ_i dépendent de $\boldsymbol{x}$:

$$\hat{F}(\boldsymbol{x}) = \mu + \sum_{i=1}^{n_s} \lambda_i(\boldsymbol{x})(u^i - \mu). \tag{3.10}$$

Le krigeage cherche à minimiser en chaque point $\boldsymbol{x}$ l'erreur quadratique moyenne du modèle : $\mathrm{E}\left[(\hat{F}(\boldsymbol{x}) - F(\boldsymbol{x}))^2\right]$. En ajoutant une contrainte de non-biais ($\mathrm{E}[\hat{F}(\boldsymbol{x})] = \mathrm{E}[F(\boldsymbol{x})]$), la résolution de ce problème permet d'obtenir le meilleur prédicteur linéaire sans biais de F. L'espérance de ce prédicteur, considérée comme une approximation de f, vaut :

$$\hat{f}(\boldsymbol{x}) = \mathrm{E}[\hat{F}(\boldsymbol{x})] = \hat{\mu} + \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{C}^{-1}(\boldsymbol{u} - \hat{\mu}), \tag{3.11}$$

où $\hat{\mu} \in \mathbb{R}$ est une estimation de μ et :

$$\begin{cases} \boldsymbol{c}(\boldsymbol{x}) = \left(\mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{s}^1)), \mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{s}^2)), ..., \mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{s}^{n_s}))\right) \in \mathbb{R}^{n_s}, \\ \boldsymbol{C} = \left[\mathrm{Cov}(\epsilon(\boldsymbol{s}^i), \epsilon(\boldsymbol{s}^j))\right]_{1 \leq i,j \leq n_s} \in M_{n_s}(\mathbb{R}), \\ \boldsymbol{u} = (u^1, u^2, ..., u^{n_s}) \in \mathbb{R}^{n_s}. \end{cases}$$

Cette équation permet de calculer une valeur approchée de $f(\boldsymbol{x})$. Elle dépend de (n_x+2) paramètres qu'il faut déterminer : σ_K, $\hat{\mu}$ et les différents θ_i. Ils peuvent par exemple être obtenus via une recherche du maximum de vraisemblance de $\hat{F}$ à partir de l'ensemble des échantillons d'apprentissage (se référer à [SWN03] pour plus de détails). On remarque que seul le premier facteur du second terme de l'équation de $\hat{f}$ dépend de $\boldsymbol{x}$, les autres facteurs pouvant être calculés a priori une fois pour toutes. L'équation (3.11) se réécrit alors sous la forme :

$$\begin{aligned} \hat{f}(\boldsymbol{x}) = \mathrm{E}[\hat{F}(\boldsymbol{x})] &= \hat{\mu} + \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\gamma} \\ &= \hat{\mu} + \sum_{i=1}^{n_s} \gamma_i \,\mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{s}^i)), \end{aligned} \tag{3.12}$$

avec $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, ..., \gamma_{n_s}) = \boldsymbol{C}^{-1}(\boldsymbol{u} - \hat{\mu}) \in \mathbb{R}^{n_s}$.

L'évaluation du modèle de krigeage nécessite d'inverser la matrice de covariance $\boldsymbol{C}$ des échantillons d'apprentissage. Des erreurs numériques liées à son conditionnement peuvent parfois apparaître dans certains cas (lorsque certains échantillons d'apprentissage sont trop proches les uns des autres par exemple). Il est donc important de prendre en compte ce fait lors de la mise en œuvre d'un tel modèle.

Le modèle de krigeage réalise une interpolation des échantillons d'apprentissage : on peut vérifier que $\forall i \in \{1, 2, ..., n_s\}, \hat{f}(\boldsymbol{s}^i) = u^i$. Nous verrons dans la section 3.4.4 qu'il est aussi possible de construire des modèles de krigeage régressants.

3.4.2 Erreur du modèle de krigeage

La particularité du modèle de krigeage est qu'il fournit aussi une estimation de son erreur de prédiction en tout point de l'espace. Cette erreur de prédiction est accessible à partir de l'estimation de la variance du processus stochastique $\hat{F}$. Tout comme on a déterminé son espérance, on peut en effet calculer sa covariance et sa variance comme suit :

$$\begin{aligned} \mathrm{Cov}(\hat{F}(\boldsymbol{x}), \hat{F}(\boldsymbol{x}')) &= \mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{x}')) - \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{C}^{-1} \boldsymbol{c}(\boldsymbol{x}') \\ &= \mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{x}')) - \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} \,\mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{s}^i)) \,\mathrm{Cov}(\epsilon(\boldsymbol{x}'), \epsilon(\boldsymbol{s}^j)), \end{aligned} \tag{3.13}$$

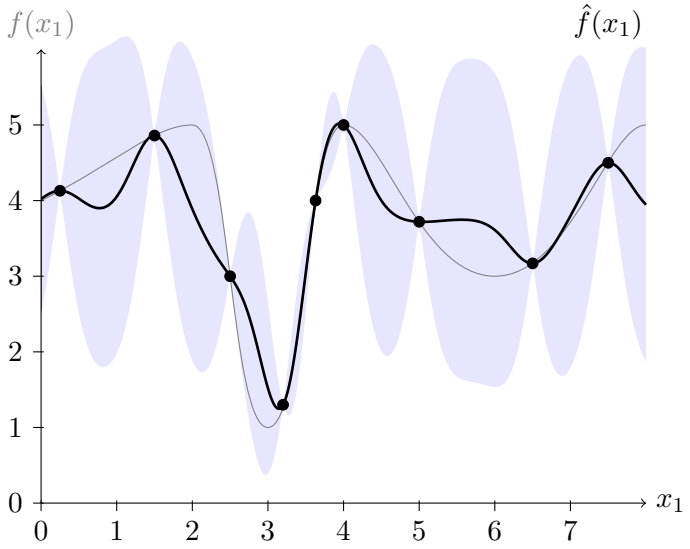


FIGURE 3.6 – Approximation d'une fonction f (en gris) par un modèle de krigeage $\hat{f}$ (en noir) construit à partir de 9 échantillons d'apprentissage (points noirs), avec son intervalle de confiance à 95% (en bleu).

$$\begin{aligned}\mathrm{Var}(\hat{F}(\boldsymbol{x})) &= \sigma_K^2 - \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{C}^{-1} \boldsymbol{c}(\boldsymbol{x}) \\ &= \sigma_K^2 - \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} \,\mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{s}^i))\,\mathrm{Cov}(\epsilon(\boldsymbol{x}), \epsilon(\boldsymbol{s}^j)),\end{aligned} \tag{3.14}$$

avec $\boldsymbol{C}^{-1} = [C_{ij}^{-1}]_{1 \leq i,j \leq n_s}$.

Un exemple de modèle de krigeage est donné sur la figure 3.6. On peut y voir tracés la moyenne $\hat{f}$ du processus ainsi que son intervalle de confiance à 95% (noté IC) calculé à partir de la variance de $\hat{F}$ en chaque point :

$$\begin{aligned}\mathrm{IC}(\hat{F}(\boldsymbol{x})) &= [\mathrm{IC}_{INF}(\hat{F}(\boldsymbol{x})), \mathrm{IC}_{SUP}(\hat{F}(\boldsymbol{x}))] \\ &\simeq \left[\hat{f}(\boldsymbol{x}) - 1.96\sqrt{\mathrm{Var}(\hat{F}(\boldsymbol{x}))}, \hat{f}(\boldsymbol{x}) + 1.96\sqrt{\mathrm{Var}(\hat{F}(\boldsymbol{x}))}\right].\end{aligned} \tag{3.15}$$

On observe que la variance du modèle de krigeage augmente lorsque le point qu'on souhaite prédire s'éloigne des échantillons d'apprentissage, et que la variance est nulle en ces points car le modèle est interpolant.

Comme pour tout autre modèle de substitution, les techniques standard d'estimation de l'erreur de prédiction peuvent aussi être utilisées sur un modèle de krigeage (ces techniques sont abordées dans la section 3.7). L'avantage de l'erreur fournie par le krigeage est qu'elle peut être directement calculée de manière analytique à partir des coefficients du modèle.

3.4.3 Simulation de processus gaussien

Nous avons vu que le modèle de krigeage est un processus stochastique gaussien dont l'espérance et la covariance peuvent être estimées. Une fois le modèle établi, on peut

facilement en simuler des réalisations.

Pour simuler une réalisation de l'estimateur de krigeage en un ensemble donné de points $\{\boldsymbol{p}^1, \boldsymbol{p}^2, ..., \boldsymbol{p}^{n_p}\}$ (avec $\boldsymbol{p}^i \in \mathbb{R}^{n_x}$), il faut générer un tirage de variables aléatoires qui respectent l'espérance et la covariance définies par le modèle en ces points. Pour cela, on calcule tout d'abord la matrice de covariance $\boldsymbol{K}$ des points via l'expression (3.13) :

$$\boldsymbol{K} = \Big[\mathrm{Cov}(\hat{F}(\boldsymbol{p}^i), \hat{F}(\boldsymbol{p}^j))\Big]_{1 \leq i,j \leq n_p} \in M_{n_p}(\mathbb{R}). \tag{3.16}$$

On réalise ensuite une décomposition de Cholesky de cette matrice symétrique définie positive, qu'on notera $\boldsymbol{L}$ (on a donc : $\boldsymbol{K} = \boldsymbol{L}^T\boldsymbol{L}$). Il faut enfin calculer l'espérance du modèle de krigeage en ces points grâce à l'expression (3.12), qu'on stocke dans un vecteur $\boldsymbol{m} = \Big(\mathrm{E}[\hat{F}(\boldsymbol{p}^1)], \mathrm{E}[\hat{F}(\boldsymbol{p}^2)], ..., \mathrm{E}[\hat{F}(\boldsymbol{p}^{n_p})]\Big) \in \mathbb{R}^{n_p}$.

La simulation d'une réalisation du processus gaussien se calcule alors simplement à partir de la génération d'un vecteur $\boldsymbol{r} \in \mathbb{R}^{n_p}$ de variables aléatoires indépendantes gaussiennes centrées réduites : il suffit de faire autant de tirages d'une loi normale centrée réduite $\mathcal{N}(0,1)$ que de points qu'on veut évaluer. La valeur $\boldsymbol{v} \in \mathbb{R}^{n_p}$ de la réalisation du processus gaussien $\hat{F}$ en ces points s'obtient alors par le calcul suivant :

$$\boldsymbol{v} = \boldsymbol{m} + \boldsymbol{r}\boldsymbol{L}. \tag{3.17}$$

Quelques réalisations du processus gaussien défini par le modèle de krigeage de la figure 3.6 sont représentées sur la figure 3.7. Le modèle de krigeage étant interpolant, les réalisations passent toutes par les échantillons d'apprentissage au niveau desquels la variance du modèle est nulle. Entre les échantillons d'apprentissage, les réalisations restent globalement au sein de l'intervalle de confiance défini par le modèle, mais elles peuvent aussi parfois en sortir car il ne s'agit que d'un intervalle de confiance à 95%.

Dans le cadre de cette étude, nous avons utilisé cette génération de réalisations de processus gaussien pour vérifier de manière statistique certains calculs analytiques complexes menés sur les modèles de krigeage (ces calculs seront présentés ultérieurement).

3.4.4 Modèle de krigeage régressant

Si le krigeage est initialement un modèle interpolant, il est aussi possible de le rendre régressant de manière assez simple. Il suffit en effet de rajouter un terme σ_S^2 dans la diagonale de sa matrice de covariance $\boldsymbol{C}$ (introduite dans l'équation (3.11)) pour créer une nouvelle matrice $\boldsymbol{C}_2$:

$$\boldsymbol{C}_2 = \boldsymbol{C} + \sigma_S^2 \boldsymbol{I}_{n_s},$$

avec $\boldsymbol{I}_{n_s}$ la matrice unité d'ordre n_s.

Cela revient à augmenter la variance du processus $\hat{F}$ au niveau des échantillons d'apprentissage, qui n'auront donc plus une erreur nulle. Selon la quantité σ_S^2 rajoutée sur la diagonale, on laisse plus ou moins de latitude au modèle par rapport aux valeurs préconisées par les échantillons d'apprentissage. Il est d'ailleurs assez fréquent de rajouter une très faible quantité sur la diagonale de la matrice de covariance des modèles de krigeage interpolants afin d'éviter les problèmes de conditionnement lors de son inversion.

Un exemple de modèle de krigeage régressant est présenté sur la figure 3.8. On voit d'une part que l'intervalle de confiance du modèle est plus important qu'auparavant au niveau des échantillons d'apprentissage, et que le modèle ne passe pas nécessairement par chacun de ces points.

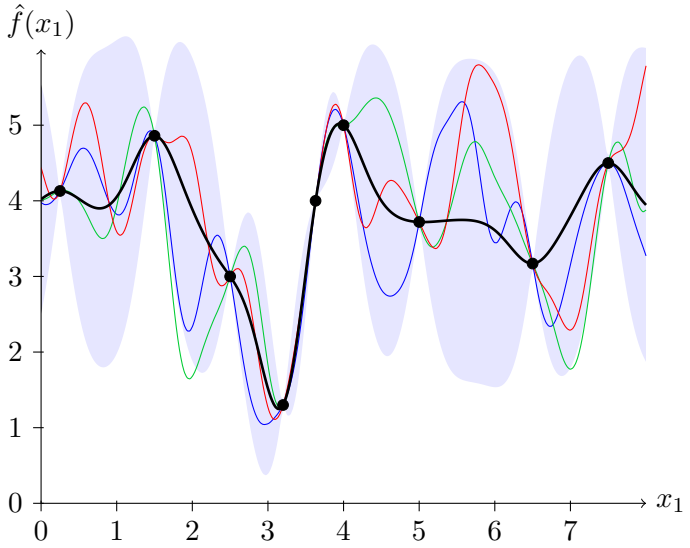


FIGURE 3.7 – Trois réalisations (en bleu, vert et rouge) du processus gaussien défini par le modèle de krigeage de la figure 3.6 (représenté en noir avec son intervalle de confiance à 95% en bleu clair).

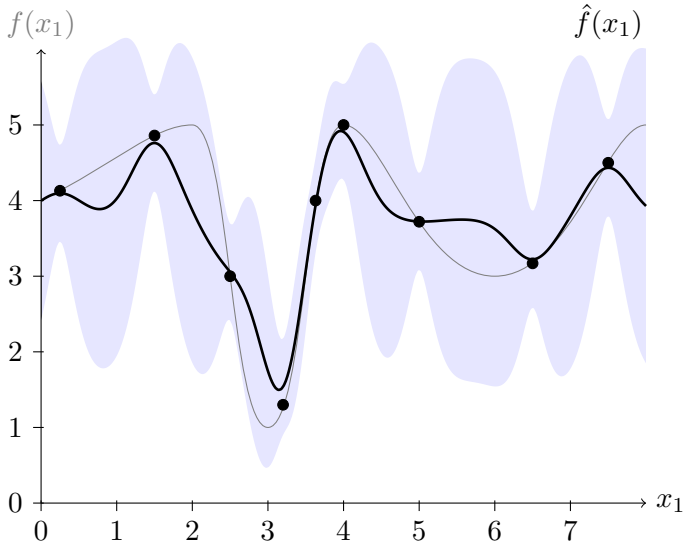


FIGURE 3.8 – Approximation d'une fonction f (en gris) par un modèle de krigeage régressant $\hat{f}$ (en noir) pour $\sigma_S^2 = 0.09$, avec son intervalle de confiance à 95% (en bleu).

L'utilisation d'un modèle régressant plutôt qu'interpolant est intéressante lorsque les données d'apprentissage sont entachées d'un certain bruit. Cela peut être le cas lorsqu'on souhaite modéliser la sortie d'un code de calcul numérique complexe faisant appel à des simulations aléatoires par exemple.

3.4.5 Mise en œuvre du krigeage

Pour cette étude, nous avons utilisé la bibliothèque GPML [RW06] pour créer des modèles de krigeage. Il s'agit d'un code développé sous Matlab. Il permet de construire des modèles régressants en déterminant automatiquement la quantité σ_S^2 ajoutée à la diagonale de la matrice de covariance à partir d'une estimation du bruit sur les données d'apprentissage. Cela lui permet de s'adapter aussi bien aux fonctions déterministes qu'aux fonctions bruitées. Les différents types de krigeage sont accessibles au sein de cette bibliothèque, ainsi que diverses fonctions de corrélation.

3.5 Autres modèles de substitution

Nous avons présenté ci-dessus trois des principaux types de modèles de substitution utilisés pour représenter des fonctions à partir d'un ensemble d'échantillons d'apprentissage. Il en existe bien sûr de nombreux autres, parmi lesquels on peut citer les RBF (« Radial Basis Function Network » [BL88]), les SVM (« Support Vector Machines » [Vap00]), les MLS (« Moving Least Squares » [Lev98]) ou encore les modèles à base de splines (MARS, « Multivariate Adaptive Regression Splines » [Fri91]). Ces autres techniques de modélisation ne seront pas détaillées ici.

Il existe aussi des modèles dont les fonctions de base ne sont pas fixées à priori (contrairement aux modèles qui ont été décrits ici). Ce type de modèles est étudié dans le cadre de la régression symbolique. On a alors affaire à des modèles beaucoup plus souples et qui peuvent théoriquement s'adapter plus facilement à la représentation de fonctions très différentes. Mais la mise en œuvre de tels modèles reste généralement assez lourde par rapport aux modèles à base fixée.

En dehors des développements qui visent à améliorer la construction de ces différents types de modèles de substitution, les recherches se dirigent aujourd'hui vers l'élaboration de modèles prenant en compte des informations supplémentaires sur la fonction à représenter, comme par exemple la dérivée de cette fonction aux points d'apprentissage ou encore le fait qu'elle soit strictement positive. On cherche aussi à construire des modèles basés sur des échantillons de différentes fidélités, c'est-à-dire des échantillons pour lesquels la valeur de la fonction est connue avec une précision variable et qui peuvent par exemple être issus de codes de calculs différents voire d'expérimentations réelles sur des prototypes.

On peut enfin souligner que la modélisation d'une fonction peut se faire par l'intermédiaire de plusieurs modèles agrégés selon un découpage de l'espace d'entrée. On parle alors de mélange d'experts [BBG+11]. L'utilisation de plusieurs modèles peut être très utile dans le cas de fonctions discontinues par exemple, et elle permet une meilleure approximation des particularités de la fonction initiale (au prix bien sûr de traitements plus complexes).

3.6 Comparaison des types de modèles de substitution

Il existe un grand nombre de modèles de substitution différents. Nous nous sommes concentrés dans cette étude sur les trois types les plus répandus aujourd'hui : les approximations polynomiales, les réseaux de neurones et le krigeage. Chacun possède ses avantages et ses points faibles, qu'il est important de connaître afin de pouvoir sélectionner un modèle adapté au problème à traiter.

Plusieurs études comparatives concernant les différents types de modèles de substitution ont été publiées. On peut notamment citer [CB93] qui compare les approximations polynomiales et les réseaux de neurones, [GW98] qui étudie les approximations polynomiales et le krigeage, ou encore [SPKA01] qui confronte les trois modèles. [JCS01] présente aussi une étude assez complète comparant plusieurs types de modèles (dont les polynômes et le krigeage) sur une large gamme de fonctions à approcher. Ces fonctions possèdent diverses caractéristiques auxquelles on peut être confronté sur des cas réels : des fonctions dont la dimension d'entrée est plus ou moins importante, des fonctions hautement non linéaires ou plus lisses, des fonctions déterministes ou bien bruitées, etc. On peut de plus avoir accès à un nombre plus ou moins grand d'échantillons de ces fonctions pour construire les modèles.

Il ressort de ces études que les approximations polynomiales sont les modèles les mieux maîtrisés et les plus simples à mettre en œuvre. Elles fournissent une expression analytique simple à comprendre, l'importance de chacune des variables d'entrée étant clairement mise en évidence. Mais elles sont plutôt limitées à l'approximation de fonctions en dimension réduite, car le nombre d'échantillons d'apprentissage nécessaire à l'établissement d'un modèle de bonne qualité croît très rapidement avec la dimension de l'espace. Le nombre d'échantillons est aussi affecté par l'ordre du polynôme utilisé, et il faudra donc se restreindre à des fonctions relativement lisses ou bien à des représentations localisées afin de bénéficier d'un degré de non linéarité moins important.

Les réseaux de neurones et le krigeage permettent quant à eux d'obtenir de bonnes représentations globales de fonctions non linéaires, au prix d'une mise en œuvre plus délicate. La construction de réseaux de neurones est généralement plus coûteuse que celle de modèles de krigeage, mais ils s'accommodent bien des espaces de grande dimension car leur complexité est uniquement limitée par leur nombre de neurones. La complexité d'un modèle de krigeage augmente au contraire avec le nombre d'échantillons d'apprentissage : ce type de modèle sera donc plus approprié pour des problèmes de dimension raisonnable et qui requièrent un nombre moins important d'échantillons. On peut aussi noter que les réseaux de neurones ne permettent pas de faire d'interpolation alors que les deux autres types de modèle peuvent être interpolants ou bien régressants. Diverses améliorations existent toutefois pour chacun des trois modèles afin d'atténuer les inconvénients que nous venons de présenter.

En conclusion, on peut dire que ces différents modèles de substitution sont largement employés aujourd'hui et fonctionnent relativement bien. L'utilisation d'un type de modèle ou d'un autre se fera en fonction de leurs qualités propres vis-à-vis de la nature des applications traitées, mais aussi de la maîtrise qu'on a d'eux. Il est en effet souvent plus efficace d'approcher une fonction avec un modèle dont on connaît le comportement et qu'on sait facilement manipuler et optimiser, plutôt que d'utiliser un modèle mal contrôlé et dont on ne saura pas tirer complètement parti.

3.7 Erreur d'un modèle de substitution

Un modèle de substitution est une approximation d'une fonction construite à partir d'un certain nombre d'échantillons d'apprentissage. Lors de l'utilisation d'un tel modèle, il est important de garder en tête que les valeurs qu'il fournit ne sont que des estimations des valeurs réelles de cette fonction de référence. Il est ainsi intéressant de pouvoir estimer l'erreur de prédiction d'un modèle. Quel que soit le type de modèle utilisé, cette estimation n'est pas aisée car l'erreur d'un modèle est liée par définition à la valeur réelle de la fonction qu'on est en train de modéliser. Or cette fonction n'est connue qu'en un nombre limité de points seulement. Il est donc impossible de connaître la véritable erreur d'un modèle en tout point, mais on peut par contre en faire des estimations à partir des données qu'on possède.

Les buts et les méthodes d'estimation de l'erreur d'un modèle sont multiples. Premièrement, on souhaite construire un modèle de substitution possédant la plus petite erreur possible : celle-ci peut tout d'abord être contrôlée au cours de l'établissement du modèle par des méthodes de « validation ». D'autre part, une fois le modèle construit, on peut estimer son erreur globale afin d'avoir une idée de sa qualité. On peut enfin chercher à obtenir un intervalle de confiance pour chacune des prédictions du modèle, en déterminant une erreur plus locale par des techniques de bootstrap par exemple. Ces différentes méthodes d'estimation d'erreur et leur utilisation sont présentées dans cette section.

3.7.1 Contrôle de l'erreur durant l'apprentissage

Les paramètres d'un modèle de substitution sont optimisés dans le but d'approcher au mieux les échantillons de la fonction qui lui sont fournis. Lors de la mise en place d'un modèle, on peut évaluer son erreur d'apprentissage en comparant les valeurs prédites au niveau de ces échantillons par rapport à leur valeur réelle. Cette erreur est bien évidemment nulle dans le cas d'un modèle interpolant, mais pour un modèle régressant elle décroît généralement au fur et à mesure des itérations d'apprentissage.

Lorsque l'erreur d'apprentissage devient trop faible, on peut assister à un phénomène de « sur-apprentissage ». Ce phénomène correspond à l'obtention d'un modèle qui approche de très près les échantillons d'apprentissage mais qui a perdu toute capacité de généralisation pour des prédictions en dehors de ces points. On peut par exemple avoir un modèle qui oscille de manière importante afin de passer précisément par chaque échantillon d'apprentissage, alors que la fonction de référence est elle-même relativement lisse. Ce phénomène est d'autant plus problématique lorsque les données d'apprentissage sont bruitées, car on cherche alors à se rapprocher des échantillons d'apprentissage mais sans pour autant les prédire avec exactitude. La considération de l'erreur d'apprentissage seule n'est donc pas suffisante pour obtenir un modèle de qualité. Lors de l'établissement de modèles régressants, on peut contrôler leur erreur de généralisation pour éviter le sur-apprentissage. On utilise pour cela des méthodes de validation (voir [KS00]).

Validation

L'erreur d'un modèle peut être estimée au cours de l'apprentissage grâce à des échantillons dits de « validation ». Il s'agit d'échantillons issus de la fonction de référence qui servent à déterminer la qualité du modèle à chaque itération. Ces échantillons sont prélevés

dans la base d'apprentissage fournie au départ pour construire le modèle : l'apprentissage est donc réalisé sur un nombre d'échantillons plus restreint.

Durant l'optimisation des coefficients du modèle avec la base d'apprentissage réduite, on évalue une erreur moyenne sur la base de validation en comparant les valeurs prédites aux valeurs réelles en ces points. Tant que l'erreur de validation baisse, on continue à faire des itérations d'apprentissage pour améliorer le modèle. Par contre, lorsque cette erreur remonte c'est signe que la capacité de généralisation du modèle commence à se dégrader, et on peut donc arrêter l'apprentissage. Le sur-apprentissage intervient en effet lorsque l'erreur sur la base d'apprentissage diminue (le modèle devient plus précis par rapport aux échantillons d'apprentissage) mais que l'erreur de validation se dégrade (le modèle perd sa capacité à approcher des zones absentes de la base d'apprentissage). L'utilisation d'échantillons de validation permet ainsi de contrôler l'évolution de l'apprentissage et de stopper le processus au moment opportun.

Le nombre d'échantillons de validation utilisés est le plus souvent fixé en fonction du nombre total d'échantillons d'apprentissage disponibles (on prend typiquement 20% des échantillons). Ils sont généralement choisis au hasard parmi la base d'apprentissage initiale, mais on peut aussi imaginer des méthodes de sélection plus évoluées afin de limiter le biais d'estimation de l'erreur du modèle (en garantissant par exemple une bonne répartition des échantillons de validation dans l'espace d'entrée).

Validation croisée

Lorsque le nombre d'échantillons disponibles pour la construction du modèle est limité, il peut être préjudiciable de devoir se séparer d'un certain nombre d'entre eux à des fins de validation. On peut alors employer une stratégie de validation croisée (voir [HTF09] par exemple).

La validation croisée repose sur le principe de validation présenté ci-dessus, mais appliqué de manière répétée avec différentes bases de points. L'ensemble des échantillons d'apprentissage disponibles est ainsi séparé en différents groupes. Tour à tour, chacun de ces groupes est utilisé comme base de validation durant la construction d'un modèle prenant comme base d'apprentissage l'ensemble des échantillons des groupes restants. On obtient au final autant de modèles que de groupes d'échantillons constitués. Chacun de ces modèles est une approximation de la fonction de référence, basée sur des données d'apprentissage en partie différentes des autres. Ces modèles peuvent ensuite être agrégés afin de rassembler toutes les informations apprises, en considérant par exemple la valeur moyenne des sorties des modèles pour la prédiction d'un point.

La validation croisée, bien que coûteuse par le fait qu'elle nécessite d'établir un nombre plus important de modèles, permet de prendre en compte l'ensemble des échantillons d'apprentissage disponibles tout en évitant le sur-apprentissage (car on contrôle l'erreur de généralisation lors de la construction de chaque modèle). On utilise typiquement cinq groupes d'échantillons, ce qui représente une base de 20% d'échantillons de validation pour chaque modèle. Ici aussi, des techniques avancées de constitution des groupes peuvent être mises en place.

Les techniques de validation et de validation croisée permettent d'établir des modèles de substitution qui possèdent de bonnes capacités de généralisation, grâce la mise en place d'un contrôle de leur erreur au cours de l'apprentissage. Elles peuvent être appliquées à tout type de modèle.

3.7.2 Estimation de l'erreur d'un modèle

L'erreur d'un modèle de substitution peut être appréciée de manière globale afin de déterminer sa qualité d'approximation générale, ou bien de manière plus locale pour connaître la précision du modèle en un point donné.

L'erreur locale $\mathrm{err}_{\hat{f}}(\boldsymbol{x})$ d'un modèle $\hat{f}$ en un point $\boldsymbol{x}$ peut être exprimée comme une erreur absolue ou bien relative par rapport à la valeur réelle de la fonction de référence f. L'erreur absolue s'écrit ainsi :

$$\mathrm{err}_{\hat{f}}(\boldsymbol{x}) = |\hat{f}(\boldsymbol{x}) - f(\boldsymbol{x})|, \tag{3.18}$$

et l'erreur relative :

$$\mathrm{err}_{\hat{f}}(\boldsymbol{x}) = \left| \frac{\hat{f}(\boldsymbol{x}) - f(\boldsymbol{x})}{f(\boldsymbol{x})} \right|. \tag{3.19}$$

On considère aussi parfois le carré de ces erreurs.

L'erreur globale d'un modèle s'exprime généralement comme une erreur moyenne ou bien une erreur maximale, calculées à partir des erreurs locales (relatives ou absolues) des points de l'espace $\boldsymbol{X}$ de définition du modèle :

$$\mathrm{ErrMax}(\hat{f}) = \max_{\boldsymbol{x} \in \boldsymbol{X}} \mathrm{err}_{\hat{f}}(\boldsymbol{x}), \tag{3.20}$$

et :

$$\mathrm{ErrMoy}(\hat{f}) = \frac{1}{\lambda(\boldsymbol{X})} \int_{\boldsymbol{X}} \mathrm{err}_{\hat{f}}(\boldsymbol{x}) d\boldsymbol{x}, \tag{3.21}$$

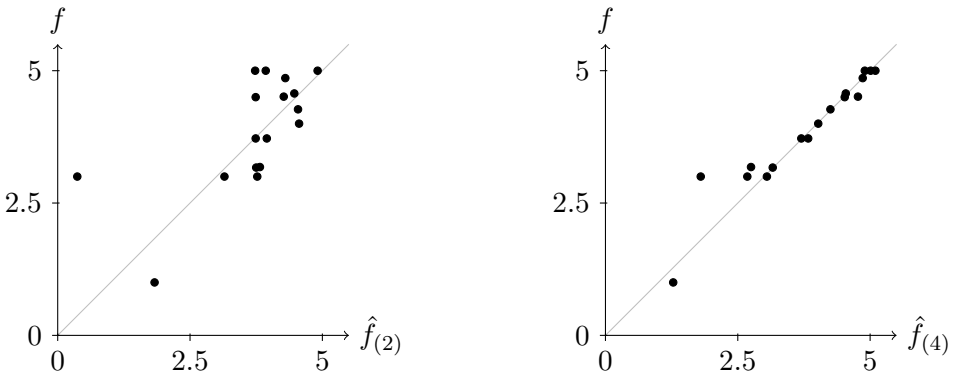
avec $\lambda(\boldsymbol{X})$ la mesure de Lebesgue de $\boldsymbol{X}$. La connaissance de l'erreur globale permet ainsi d'affirmer par exemple que le modèle possède une erreur de 1 mm au maximum (en considérant ErrMax avec une erreur absolue), ou bien qu'il fournit des prédictions à 1% près en moyenne (pour ErrMoy avec une erreur relative). Une mesure d'erreur souvent employée est l'erreur quadratique moyenne notée MSE (pour « Mean Squared Error » en anglais), parfois appelée « risque quadratique » et qui correspond à ErrMoy avec le carré de l'erreur absolue :

$$\mathrm{MSE}(\hat{f}) = \frac{1}{\lambda(\boldsymbol{X})} \int_{\boldsymbol{X}} \mathrm{err}^2_{\hat{f}}(\boldsymbol{x}) d\boldsymbol{x}. \tag{3.22}$$

L'erreur d'un modèle étant définie à partir des valeurs de la fonction de référence f qui n'est connue qu'en un nombre limité de points, il n'est pas possible de la calculer directement. On utilise donc des méthodes d'estimation basées sur les informations apportées par les échantillons de f qu'on possède. Deux possibilités sont abordées ci-dessous : la première utilise une base d'échantillons de test, et la deuxième repose sur une méthode statistique nommée « bootstrap ».

Base de test

L'erreur globale d'un modèle de substitution peut être estimée à partir de l'erreur commise sur les échantillons d'apprentissage ou bien de validation s'il y en a. Mais cette estimation d'erreur est biaisée car tous ces échantillons ont participé à l'établissement du modèle. Les échantillons d'apprentissage sont en effet directement utilisés dans le processus, et les échantillons de validation entrent eux aussi en jeu car l'apprentissage est arrêté lorsque l'erreur sur ces échantillons est minimale. On ne peut donc pas réutiliser ces échantillons pour estimer a posteriori l'erreur d'un modèle.



ErrMax = 2.63
ErrMoy = 0.63
MSE = 0.77

ErrMax = 1.20
ErrMoy = 0.18
MSE = 0.11

FIGURE 3.9 – Erreur des réseaux de neurones de la figure 3.5 sur une base de test de 17 échantillons : réseau à 2 neurones cachés $\hat{f}_{(2)}$ (à gauche) et réseau à 4 neurones cachés $\hat{f}_{(4)}$ (à droite).

Pour évaluer cette erreur de manière plus fiable, il est nécessaire d'utiliser d'autres échantillons qui n'ont pas servi à la construction du modèle. On parle d'échantillons de « test », qu'on notera $\boldsymbol{t}_i \in \boldsymbol{X} \subseteq \mathbb{R}^{n_x}$ avec $i \in \{1, 2, ..., n_t\}$. Ces échantillons sont généralement extraits de la base d'apprentissage initiale. Dans le cadre d'une expérimentation, on peut aussi avoir à disposition une base de test spécifique pour contrôler la qualité des modèles. Comme ils ne participent pas à la construction du modèle de substitution (ni pour l'apprentissage, ni pour la validation), les échantillons de test peuvent servir à obtenir une estimation non biaisée de l'erreur de généralisation du modèle.

Les erreurs globales maximale et moyenne sont alors estimées par :

$$\widehat{\text{ErrMax}} = \max_{i \in \{1,2,...,n_t\}} \text{err}_{\hat{f}}(\boldsymbol{t}^i), \tag{3.23}$$

et :

$$\widehat{\text{ErrMoy}} = \frac{1}{n_t} \sum_{i=1}^{n_t} \text{err}_{\hat{f}}(\boldsymbol{t}^i). \tag{3.24}$$

Plus le nombre n_t d'échantillons de test est grand, et plus ces estimations sont précises. En pratique, ce nombre est généralement déterminé à partir de la quantité initiale d'échantillons d'apprentissage (on choisit typiquement 10 à 20% de ces échantillons). L'utilisation d'échantillons de test vient donc réduire le nombre d'échantillons effectivement utilisés pour l'apprentissage du modèle. Le fait de vouloir estimer l'erreur du modèle va ainsi affecter sa qualité, car le modèle sera établi à partir d'un nombre d'échantillons plus faible.

La figure 3.9 présente un exemple d'estimation de l'erreur globale des deux réseaux de neurones de la figure 3.5. Ces estimations sont basées sur $n_t = 17$ échantillons de test uniformément répartis sur l'axe des x_1. Les valeurs de f et celles prédites par chaque modèle $\hat{f}_{(2)}$ et $\hat{f}_{(4)}$ en ces points sont représentés sur cette figure. La mise en concurrence des valeurs prédites par rapport aux valeurs réelles permet d'avoir visuellement une idée de l'erreur d'un modèle : plus les points sont proches de l'axe diagonal $f(x_1) = \hat{f}(x_1)$, plus le

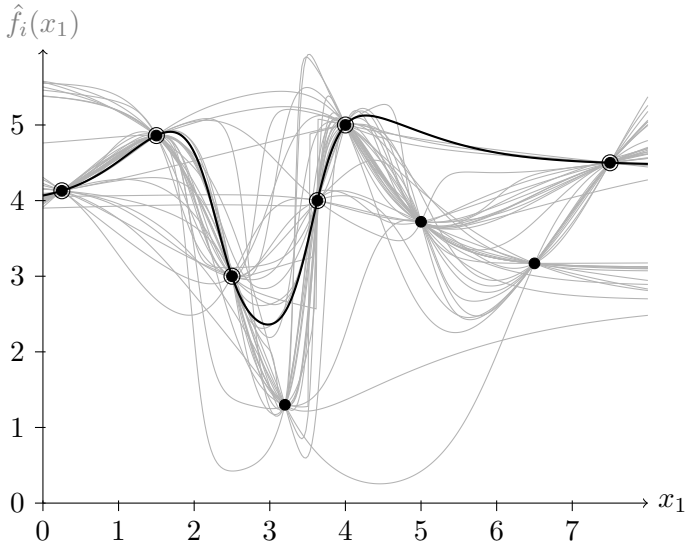


FIGURE 3.10 – Exemple de bootstrap : 40 modèles (en gris) ont été générés par bootstrap à partir de 9 échantillons d'apprentissage (points noirs). L'un des modèles est représenté en noir et ses échantillons d'apprentissages entourés. Les trois échantillons restants n'ont ainsi pas été pris en compte dans la construction de ce modèle particulier.

modèle est de qualité. On voit ainsi que le modèle possédant quatre neurones cachés fournit de meilleures prédictions que celui avec deux neurones cachés (ce qu'on avait déjà souligné auparavant). Les mesures d'erreur permettent de quantifier la qualité de ces modèles et de vérifier qu'elle satisfait aux exigences des applications pour lesquelles les modèles seront utilisés.

L'utilisation d'une base de test permet de déterminer l'erreur globale d'un modèle, mais pas son erreur locale de prédiction en un point donné (en dehors des points de test). Pour cela, on peut mettre en œuvre des méthodes statistiques comme le bootstrap.

Bootstrap

Le bootstrap [Efr81] est une technique statistique de rééchantillonnage qui permet d'évaluer l'erreur commise sur une mesure calculée à partir d'un ensemble d'échantillons. On peut ainsi la mettre en œuvre dans le cadre des modèles de substitution, où on cherche à estimer la valeur de la fonction de référence en un point donné à partir d'un ensemble d'apprentissage, afin de connaître l'erreur de prédiction en ce point.

Le bootstrap consiste à créer un certain nombre n_B de bases d'apprentissage à partir de l'ensemble initial par tirages aléatoires avec remise. On obtient ainsi une multitude de bases différentes de même taille, qui peuvent contenir plusieurs exemplaires du même échantillon initial. Grâce à ces ensembles d'apprentissage, on construit n_B modèles de substitution $\hat{f}_i$ distincts. Chacun de ces modèles fournit une valeur approchée de f en tout point. La figure 3.10 présente des modèles obtenus par bootstrap sur la base d'apprentissage utilisée dans les exemples précédents.

En un point $\boldsymbol{x}$ donné, on a donc un ensemble de valeurs $\hat{f}_i(\boldsymbol{x})$ qui forment une distribution empirique des valeurs possible de $f(\boldsymbol{x})$. Plus le nombre de modèles générés par

boostrap est grand, plus cette distribution sera connue avec précision. Comme dans le cas du krigeage (voir la section 3.4), on est alors en présence d'un modèle de processus stochastique $\hat{F}$. Celui-ci n'est par contre pas nécessairement gaussien car aucune hypothèse n'a été faite ici. On peut notamment exprimer en tout point $\boldsymbol{x}$ sa moyenne :

$$\mathrm{E}[\hat{F}(\boldsymbol{x})] = \frac{1}{n_B} \sum_{i=1}^{n_B} \hat{f}_i(\boldsymbol{x}), \tag{3.25}$$

et sa variance :

$$\mathrm{Var}(\hat{F}(\boldsymbol{x})) = \frac{1}{n_B - 1} \sum_{i=1}^{n_B} \left(\hat{f}_i(\boldsymbol{x}) - \mathrm{E}[\hat{F}(\boldsymbol{x})] \right)^2. \tag{3.26}$$

On obtient donc une estimation de $f(\boldsymbol{x})$ grâce à la moyenne des modèles bootstrap, ainsi qu'une indication sur la précision de ce calcul avec leur variance (plus celle-ci est grande, plus la mesure est imprécise). Même dans le cas de modèles interpolants, la variance ne sera pas forcément nulle au niveau des points d'apprentissage car les modèles bootstrap ne sont construits que sur une partie de ces échantillons.

On peut alors déterminer un intervalle de confiance $\mathrm{IC} = [\mathrm{IC}_{INF}, \mathrm{IC}_{SUP}]$ pour la prédiction de $f(\boldsymbol{x})$, soit à partir de la moyenne et de la variance en faisant une approximation gaussienne comme dans le cas du krigeage (voir l'équation (3.15)), soit en calculant directement cet intervalle à partir de la distribution empirique des échantillons bootstrap (qui n'est généralement pas gaussienne).

Le bootstrap ne fait appel à aucune hypothèse particulière, ce qui en fait un outil très général et puissant. Il permet d'estimer des intervalles de confiance pour les prédictions de n'importe quel type de modèle de substitution. L'inconvénient de cette méthode est bien entendu son coût, car elle nécessite de construire un nombre important de modèles pour avoir des estimations fiables.

La connaissance de l'erreur des modèles de substitution permet de déterminer les zones de l'espace qui sont plus ou moins bien décrites, et peut être utilisée pour choisir intelligemment la position de nouveaux échantillons à évaluer avec la fonction de référence dans le but d'améliorer la qualité du modèle. On parle alors de stratégies de « planification d'expériences », qui sont présentées dans la section suivante.

3.8 Plans d'expériences

Pour approcher une fonction, on souhaite construire un modèle de substitution le plus précis possible à partir d'un minimum d'échantillons d'apprentissage. L'évaluation de ces échantillons est en effet généralement coûteuse, et il faut donc les utiliser avec parcimonie. L'ensemble des échantillons destinés à l'établissement d'un modèle est appelé un « plan d'expériences » (« design of experiments » en anglais). Le but recherché est de planifier intelligemment ces expériences afin d'en tirer profit au maximum.

Dans un premier temps, on établit un plan d'expériences initial dont les échantillons recouvrent au mieux l'ensemble de l'espace de définition du modèle. On peut ensuite compléter ce plan d'expériences de manière adaptative afin de cerner au mieux les particularités de la fonction à représenter.

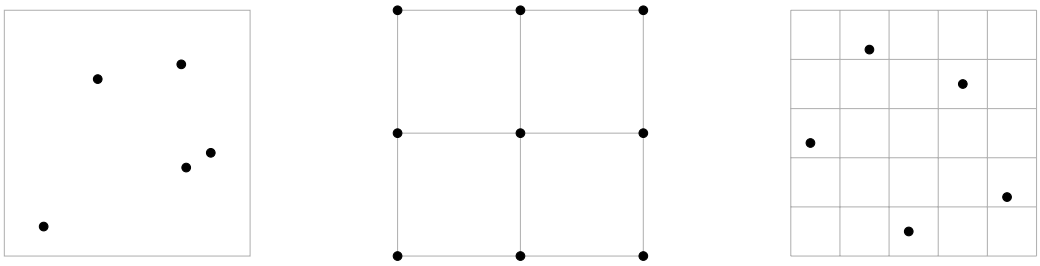


FIGURE 3.11 – Exemples de plans d'expériences.

3.8.1 Plan d'expériences initial

La manière la plus simple de choisir des échantillons d'apprentissage pour construire un modèle de substitution est de les prendre au hasard dans l'espace d'entrée. Cependant, cette méthode ne garantit pas un bon recouvrement de l'ensemble de l'espace et le modèle pourra alors se révéler de piètre qualité dans certaines régions mal définies. C'est pourquoi on recherche d'autres moyens de générer des plans d'expériences plus représentatifs de la totalité de l'espace considéré. Le but de ces méthodes est de fournir un plan d'expériences optimal au sens d'une certaine mesure de la bonne répartition des points dans l'espace. Des exemples de plans d'expériences en deux dimensions sont présentés sur la figure 3.11. Le principe de chacun d'entre eux est détaillé ci-dessous.

Le premier type de plan d'expériences qu'on rencontre est le plan factoriel. Il consiste à utiliser une grille qui découpe le domaine de variation de chacun des n_x paramètres en k parties égales, et de disposer $(k+1)^{n_x}$ points sur les nœuds de cette grille. Les échantillons sont ainsi bien répartis dans l'espace. L'inconvénient est qu'on a besoin d'un très grand nombre de points lorsque le nombre de paramètres ou la finesse de la grille augmente. Ce n'est donc pas une solution adaptée aux problèmes qui nous concernent.

Les plans d'expériences les plus répandus aujourd'hui sont les hypercubes latins (ou LHS, pour « Latin Hypercube Sampling »). Dans un espace en deux dimensions, ils reviennent à choisir les échantillons au sein d'une grille de manière à n'avoir qu'un seul point sur chaque ligne et chaque colonne. Chacune des parties du domaine de variation d'un paramètre est ainsi décrite par un unique échantillon. Les LHS permettent de recouvrir convenablement l'espace en utilisant un nombre restreint de points par rapport aux plans factoriels (il y a ici autant d'échantillons que de lignes ou de colonnes). La grille peut aussi être adaptée afin que l'échantillonnage suive une certaine distribution, en définissant des colonnes équiprobables (plus resserrées dans les zones de forte probabilité et plus larges au niveau des zones de faible probabilité). Des recherches sont encore effectuées de nos jours afin de générer les meilleurs LHS possibles parmi les différentes alternatives existantes au sein d'une grille donnée (voir [BST04] par exemple).

Un plan d'expérience doit aussi être adapté au modèle de substitution utilisé. Si les réseaux de neurones ou le krigeage laissent une certaine liberté quant à ce choix, d'autres modèles comme les approximations polynomiales par exemple bénéficient de plans d'expériences optimaux spécifiques qui permettent d'établir des modèles de meilleure qualité.

Ce point ne sera pas abordé plus en détail car nous allons plutôt nous intéresser ici à la construction adaptative de plans d'expériences.

Les plans d'expériences que nous venons de présenter cherchent à répartir au mieux les échantillons d'apprentissage dans l'espace de recherche, dans le but de pouvoir créer un bon modèle global. Si cette stratégie de recouvrement de l'espace est payante lorsqu'on ne connaît pas du tout la fonction à représenter, ce n'est plus forcément le cas quand ses caractéristiques commencent à se dessiner. Une fonction relativement plane et qui ne présente des variations importantes que dans une région restreinte de l'espace sera par exemple modélisée plus efficacement avec un plan d'expériences possédant une répartition plus dense de points dans la région de variation. On cherche alors à construire les plans d'expériences de manière adaptative, c'est-à-dire en rajoutant au fur et à mesure des points dans des zones d'intérêt grâce aux informations apportées par les échantillons évalués précédemment.

3.8.2 Complétion adaptative du plan d'expériences initial

Plutôt que d'établir directement un plan d'expériences avec le nombre maximal d'échantillons qu'il est permis d'évaluer sur la fonction de référence, on peut démarrer par un plan plus limité et utiliser les informations apportées par les échantillons initiaux sur le comportement de la fonction pour décider des endroits les plus propices où placer les échantillons restants. La construction adaptative de plans d'expériences vise ainsi à choisir la position de ces échantillons de manière réfléchie pour s'adapter aux spécificités de la fonction à modéliser.

Le principe de l'établissement d'un plan d'expériences adaptatif est résumé dans l'algorithme 3.1. À partir d'un plan d'expériences initial $\boldsymbol{PDEinitial}$, on construit un premier modèle de substitution $\hat{f}$ à l'aide duquel on détermine la position d'un ou plusieurs échantillons p à calculer sur la fonction de référence. Une fois évalués, ces échantillons sont rajoutés au plan d'expériences $\boldsymbol{PDE}$ et on recommence l'établissement d'un nouveau modèle. Selon le type de modèle utilisé, le modèle peut parfois être simplement ajusté localement pour prendre en compte les nouveaux points, un procédé plus léger que la reconstruction complète du modèle ([GRMS06] présente ainsi une étude pour déterminer si les coefficients d'un modèle de krigeage doivent être recalculés ou pas après ajout d'un échantillon d'apprentissage). Le plan d'expériences est raffiné de la sorte jusqu'à ce que le nombre maximum d'évaluations de la fonction de référence soit atteint, ou bien jusqu'à ce que le modèle possède une certaine qualité.

Selon les cas, on peut souhaiter rajouter un ou plusieurs points à la fois au plan d'expériences à chaque itération. Si dans la plupart des cas les points sont rajoutés l'un après l'autre, on a parfois la possibilité d'évaluer la fonction de référence en plusieurs points en parallèle. Les méthodes de construction de plans d'expériences adaptatifs cherchent alors à localiser le placement idéal de ces multiples points simultanément (ce qui amène généralement à un problème plus complexe que la recherche d'un seul point améliorant).

Le but recherché ici est d'améliorer globalement le modèle de substitution afin d'obtenir une représentation fidèle de la fonction de référence sur l'ensemble de l'espace (la planification adaptative d'expériences peut aussi être employée dans le cadre de l'optimisation où on cherche à améliorer certaines zones uniquement, ce qui sera présenté dans la section 4.6.3). Le modèle peut alors être utilisé pour remplacer la fonction initiale : il pourra ainsi se substituer par exemple à une discipline trop coûteuse au sein d'un système

Algorithme 3.1 : Construction d'un plan d'expériences adaptatif pour l'amélioration globale d'un modèle

```
1 PDE ← PDEinitial
2 Évaluer_avec_fonction_de_référence(PDE)
3 tant que le critère d'arrêt n'est pas vérifié faire
4     f̂ ← Construire_modèle(PDE)
5     p ← Chercher_point_à_ajouter(f̂)
6     Évaluer_avec_fonction_de_référence(p)
7     PDE ← PDE ∪ {p}
8 fin
```

multidisciplinaire. Comme on ne sait pas à l'avance quelle zone de l'espace sera utilisée par le système (ce sera possiblement l'ensemble de l'espace d'entrée qui sera en fait parcouru), il faut nécessairement que le modèle soit précis sur la totalité de son domaine de définition. On raffine donc le plan d'expériences initial de manière adaptative afin de rajouter des échantillons dans les zones où la fonction est plus difficile à modéliser. La fonction de référence n'étant connue qu'au niveau des échantillons qu'on possède déjà, tout l'art de la construction adaptative réside dans la localisation de ces zones d'intérêt. Différents critères de sélection existent pour choisir le prochain point à calculer avec la fonction de référence, et sont généralement définis à partir d'une estimation de l'erreur du modèle (par exemple la variance de ses prédictions $\mathrm{Var}(\hat{F}(\boldsymbol{x}))$). Les deux critères principaux pour l'amélioration globale de modèles sont présentés ci-dessous : la recherche du point d'erreur maximale et la recherche du point améliorant l'erreur moyenne.

Les études traitant de l'amélioration simultanée de plusieurs modèles de substitution (lorsqu'un code de calcul possède par exemple plusieurs sorties qu'on souhaite modéliser à partir d'un même plan d'expériences) sont assez rares. On peut toutefois citer le papier [LAFMD06], dans lequel est présentée une méthodologie de modélisation simultanée de plusieurs fonctions avec ajout itératif d'échantillons au plan d'expériences. Cette méthode est appliquée sur deux modèles de krigeage interdépendants pour lesquels le critère d'entropie présenté ci-dessous peut être étendu. La méthode permet donc de sélectionner des points qui améliorent simultanément les deux modèles. Elle reste par contre restreinte à des modèles interdépendants et ne s'applique donc pas aux cas où les modèles sont complètement séparés.

Point d'erreur maximale

Un premier critère pour l'amélioration générale d'un modèle consiste à rajouter au plan d'expériences le point qui présente la plus grande variance de prédiction. Il s'agit du critère MMSE (pour « Maximum Mean Squared Error »). Ce nom vient du fait que la variance du modèle peut être considérée comme un risque quadratique. Le problème à résoudre pour trouver le point qu'on va évaluer sur la fonction de référence est donc le suivant :

$$\mathrm{MMSE}(\hat{F}) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmax}} \ \mathrm{Var}(\hat{F}(\boldsymbol{x})). \tag{3.27}$$

Comme décrit dans [JCS02], cette méthode est un cas particulier d'un critère basé sur l'entropie présenté dans [CMMY91] dans le cadre du krigeage. L'entropie définit la quantité

d'information apportée au modèle par un point donné, et on cherche donc à rajouter le point qui va donner un maximum d'information. Le critère d'entropie peut aussi servir à définir les positions de plusieurs échantillons à calculer simultanément. Dans le cas où on ne rajoute qu'un seul point à la fois, cela revient à sélectionner celui qui possède la plus grande variance.

Le critère MMSE est par exemple utilisé dans [MS02] où les performances d'un véhicule sous-marin sont approchées par un modèle de krigeage construit de manière adaptative.

Point améliorant l'erreur moyenne

Le principe du deuxième critère d'amélioration globale de modèle est d'ajouter le point qui va faire diminuer le plus possible l'erreur moyenne du modèle. Il s'agit du critère IMSE (pour « Integrated Mean Squared Error »), introduit par [SWMW89]. Le calcul de l'erreur moyenne passe en effet par une intégration du risque quadratique sur le domaine de définition du modèle. On cherche donc à minimiser la variance moyenne du modèle en fonction du point rajouté. Si on note $\hat{F}^{+\boldsymbol{x}}$ le modèle $\hat{F}$ auquel on a ajouté l'échantillon d'apprentissage $\boldsymbol{x}$, ce critère s'écrit :

$$\mathrm{IMSE}(\hat{F}) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmin}} \int_{\boldsymbol{X}} \mathrm{Var}(\hat{F}^{+\boldsymbol{x}}(\boldsymbol{x}'))d\boldsymbol{x}'. \tag{3.28}$$

L'optimisation de ce critère est plus complexe que celle du précédent car elle nécessite de réaliser des intégrations sur l'espace d'entrée du modèle. Celles-ci sont généralement évaluées par discrétisation sur un ensemble de points donné. Le critère IMSE nécessite de plus d'estimer la variance du modèle après l'ajout d'un nouveau point d'apprentissage. Cet échantillon n'a en réalité pas encore été évalué avec la fonction de référence, et nous parlerons donc d'ajout « virtuel », dont la méthode est présentée dans la section suivante.

3.8.3 Simulation de l'ajout d'un nouvel échantillon d'apprentissage

Lors de l'utilisation d'un modèle de substitution, il peut être intéressant d'étudier l'effet de l'ajout d'un nouvel échantillon à la base d'apprentissage, notamment en termes d'impact sur l'erreur du modèle. L'idée est ici de simuler l'ajout de ce nouvel échantillon sans connaître son image par f, dans le but de déterminer s'il serait effectivement intéressant de l'évaluer avec la fonction de référence. Nous distinguons le cas du krigeage où l'erreur est estimée analytiquement à partir des coefficients du modèle, et le cas plus général de l'utilisation d'un modèle de substitution quelconque avec évaluation de son erreur par bootstrap.

Cas du krigeage

Pour le modèle de krigeage, le calcul de la variance des prédictions $\mathrm{Var}(\hat{F}(\boldsymbol{x}))$ est basé sur la matrice de covariance $\boldsymbol{C}$ des échantillons d'apprentissage (voir l'équation (3.14)). L'erreur fournie par le krigeage dépend donc de la position des échantillons d'apprentissage dans l'espace d'entrée ainsi que des paramètres du modèle, mais pas directement de la valeur $\boldsymbol{u}$ de l'image de ces échantillons par la fonction f ($\boldsymbol{u}$ est uniquement pris en compte lors de l'optimisation des paramètres du modèle). On peut alors étudier l'effet sur l'erreur du modèle de l'ajout « virtuel » d'un nouvel échantillon d'apprentissage en se basant uniquement sur sa position $\boldsymbol{p}$ dans l'espace d'entrée. On va intégrer ce nouvel échantillon

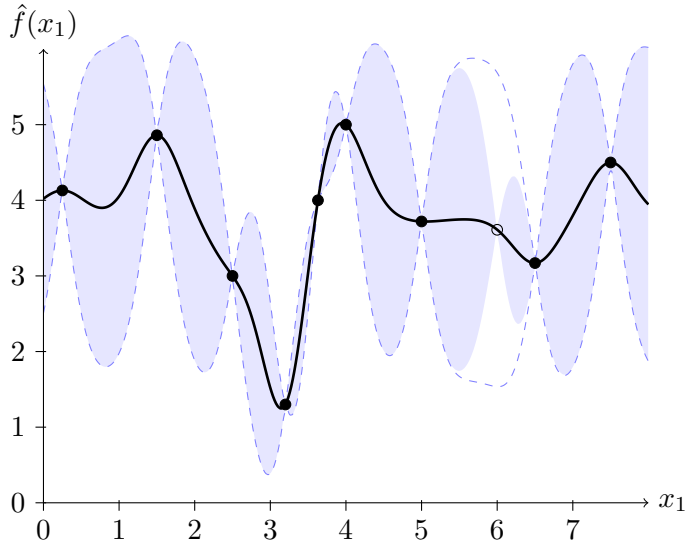


FIGURE 3.12 – Simulation de l'ajout du point $x_1 = 6$ au modèle de krigeage de la figure 3.6 par modification de sa matrice de covariance (l'intervalle de confiance à 95% initial du modèle est représentée en pointillés, et le nouvel intervalle obtenu en bleu).

dans le calcul de la variance du modèle de krigeage, sans reconstruire le modèle et sans modifier ses paramètres ni son espérance.

Pour intégrer un nouveau point $\boldsymbol{p}$ dans le calcul d'erreur du krigeage, il suffit de modifier la matrice de covariance $\boldsymbol{C}$ pour créer une nouvelle matrice $\boldsymbol{C}_2$:

$$\boldsymbol{C}_2 = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{c}(\boldsymbol{p}) \\ \boldsymbol{c}^T(\boldsymbol{p}) & \sigma_K^2 \end{bmatrix}. \tag{3.29}$$

La variance du modèle de krigeage est exprimée à partir de l'inverse de cette matrice. Pour le calculer, on peut utiliser une décomposition de Cholesky (car les matrices de covariance sont symétriques définies positives). Cette décomposition consiste à trouver la matrice triangulaire supérieure $\boldsymbol{D}_2$ telle que $\boldsymbol{C}_2 = \boldsymbol{D}_2^T \boldsymbol{D}_2$. Comme $\boldsymbol{D}_2$ est triangulaire, son inverse est facile à calculer et on a $\boldsymbol{C}_2^{-1} = \boldsymbol{D}_2^{-1}(\boldsymbol{D}_2^T)^{-1}$.

Si la décomposition de Cholesky $\boldsymbol{C} = \boldsymbol{D}^T\boldsymbol{D}$ du modèle de krigeage initial est déjà connue, une décomposition par bloc permet d'exprimer $\boldsymbol{D}_2$ à partir de la matrice $\boldsymbol{D}$:

$$\boldsymbol{D}_2 = \begin{bmatrix} \boldsymbol{D} & (\boldsymbol{D}^T)^{-1}\boldsymbol{c}(\boldsymbol{p}) \\ 0 & \sqrt{\sigma_K^2 - \boldsymbol{c}^T(\boldsymbol{p})\boldsymbol{D}^{-1}(\boldsymbol{D}^T)^{-1}\boldsymbol{c}(\boldsymbol{p})} \end{bmatrix}.$$

On obtient ainsi la nouvelle décomposition de Cholesky de la matrice de covariance $\boldsymbol{C}_2$, ce qui permet de calculer son inverse et d'exprimer la variance du modèle incluant le nouvel échantillon d'apprentissage $\boldsymbol{p}$.

On peut ainsi très facilement connaître l'effet sur l'erreur du modèle de l'ajout d'un nouveau point à la base d'apprentissage. Il ne s'agit bien sûr que d'une estimation car l'ajout effectif d'un nouvel échantillon d'apprentissage nécessiterait de reconstruire le modèle en réévaluant ses paramètres, ce qui n'est pas réalisé ici. Un exemple d'ajout « virtuel »

de nouveau point est présenté sur la figure 3.12. On remarque que l'intervalle de confiance des prédictions du modèle est modifié autour du nouvel échantillon, et que la variance s'annule en ce point (car on a affaire à un modèle interpolant). On peut alors estimer l'erreur globale du modèle afin de décider s'il serait effectivement intéressant d'évaluer ce point avec la fonction de référence en vue d'améliorer sa représentation.

Cas général

Dans le cadre plus général de l'utilisation d'un modèle de substitution quelconque, l'estimation de son erreur de prédiction peut être réalisée par bootstrap. Si pour le krigeage on pouvait se passer de la valeur de f au niveau du nouvel échantillon d'apprentissage, il est ici nécessaire de l'estimer afin que les modèles puissent être construits au cours du bootstrap.

Plusieurs stratégies existent pour fixer cette valeur. [GLRC10] propose par exemple de faire confiance au modèle initial et d'affecter au nouvel échantillon la valeur prédite par ce modèle, ou bien de lui donner une valeur fixée définie a priori à partir des échantillons d'apprentissage déjà évalués. On choisira par exemple la plus petite valeur de f connue, ce qui peut se justifier dans le cadre d'une procédure d'optimisation de la fonction. La valeur du nouvel échantillon sera donc estimée en fonction du but recherché dans la construction du plan d'expériences adaptatif.

Une fois que le nouveau point est totalement défini, on peut le prendre en compte comme un échantillon d'apprentissage classique au sein d'une procédure de bootstrap et évaluer ainsi l'erreur du nouveau modèle obtenu. Cette erreur reste bien sûr une simple estimation car l'image du nouvel échantillon par f n'est en réalité pas connue. Cette estimation permettra néanmoins de déterminer s'il peut être intéressant d'évaluer effectivement le point avec la fonction de référence.

Chapitre 4

Optimisation multiobjectifs

Sommaire

4.1 Optimisation 65
4.2 Optimisation multiobjectifs 67
4.3 Algorithmes d'optimisation multiobjectifs 69
4.3.1 Trouver une unique solution optimale 70
4.3.2 Trouver l'ensemble du front de Pareto 71
4.3.3 NSGA-II 72
4.4 Élagage du front de Pareto par classification 75
4.5 Test des algorithmes d'optimisation 77
4.5.1 Problèmes de test 78
4.5.2 Mesures de performance 80
4.6 Optimisation par modèles de substitution 83
4.6.1 Méthodes locales 83
4.6.2 Modèles globaux 84
4.6.3 Plans d'expériences adaptatifs pour l'optimisation 85

Dans cette étude, nous allons être amenés à résoudre des problèmes d'optimisation multiobjectifs, c'est-à-dire des problèmes faisant intervenir plusieurs objectifs (souvent contradictoires) à optimiser simultanément. Nous présentons dans cette section quelques rappels sur ce domaine, en détaillant le principe de front de Pareto ainsi que des algorithmes génétiques qui peuvent être utilisés pour résoudre ce type de problèmes. Nous parlerons aussi des techniques de classification qui permettent d'organiser les solutions optimales obtenues, ainsi que des méthodes de test de ces algorithmes d'optimisation multiobjectifs. Nous aborderons enfin l'utilisation de modèles de substitution pour l'optimisation de fonctions coûteuses.

4.1 Optimisation

L'optimisation est un domaine de l'aide à la décision où on cherche à déterminer le minimum (ou le maximum) d'une fonction afin de guider le décideur vers un choix

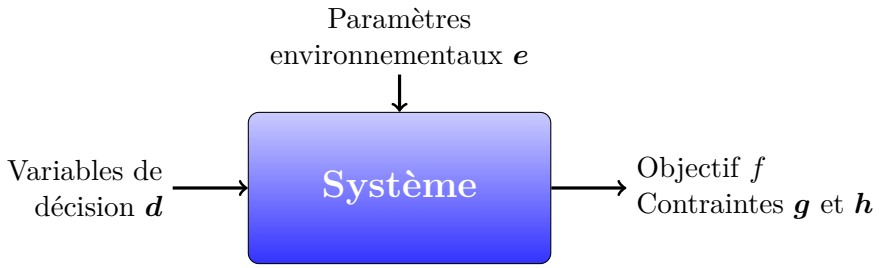


FIGURE 4.1 – Entrées et sorties pour l'optimisation d'un système.

profitable. Nous parlerons ici de minimisation uniquement sans perte de généralité. Nous effectuons ce genre de choix quotidiennement sans même en avoir conscience, comme par exemple pour décider quel type de transport nous allons utiliser pour nous rendre à une destination donnée. Les problèmes rencontrés dans les applications industrielles pour l'optimisation de systèmes physiques sont bien souvent plus complexes et nécessitent l'utilisation de techniques d'optimisation avancées pour les résoudre. Nous allons nous intéresser ici au champ de l'optimisation dite continue, c'est-à-dire faisant intervenir des variables continues (par opposition à l'optimisation discrète qui fait intervenir des variables discrètes, ou à l'optimisation mixte qui met en jeu les deux types de variables).

Nous cherchons à minimiser une fonction f appelée « fonction objectif ». Elle correspond à une représentation de la performance du système à optimiser. Cette fonction dépend d'entrées $\boldsymbol{x} = (x_1, x_2, ..., x_{n_x}) \in \mathbb{R}^{n_x}$, qui se répartissent en variables de décision $\boldsymbol{d} = (d_1, d_2, ..., d_{n_d}) \in \mathbb{R}^{n_d}$ (encore appelées variables d'optimisation ou de conception) pour lesquelles on recherche la valeur optimale, et en paramètres environnementaux $\boldsymbol{e} = (e_1, e_2, ..., e_{n_e}) \in \mathbb{R}^{n_e}$, des variables non contrôlables dont la valeur est fixée. On a donc $\boldsymbol{x} = (\boldsymbol{d}, \boldsymbol{e})$ et $n_x = n_d + n_e$. Chaque variable de décision d_i peut varier dans un certain intervalle $[d_{INF_i}, d_{SUP_i}] \subseteq \mathbb{R}$, définissant ainsi un espace de recherche global $\boldsymbol{D} \subseteq \mathbb{R}^{n_d}$. C'est au sein de cet espace qu'on cherche la solution optimale permettant de minimiser la valeur de l'objectif $f(\boldsymbol{d}, \boldsymbol{e})$ (la valeur de $\boldsymbol{e}$ étant fixée). Des contraintes peuvent aussi faire partie du problème afin de limiter l'espace de recherche. Ces contraintes sont des contraintes d'égalité notées $\boldsymbol{h}$ ou bien d'inégalité notées $\boldsymbol{g}$ (on considèrera parfois des contraintes d'inégalité uniquement sans perte de généralité). Seules les solutions qui vérifient ces contraintes sont considérées comme admissibles pour le problème. La figure 4.1 résume les différentes entrées et sorties présentes dans un problème d'optimisation.

Un problème d'optimisation peut donc s'écrire :

$$\begin{aligned} &\underset{\boldsymbol{d} \in \boldsymbol{D}}{\text{minimiser}}\ f(\boldsymbol{d}, \boldsymbol{e}), \\ &\text{s.c.} \begin{cases} \boldsymbol{g}(\boldsymbol{d}, \boldsymbol{e}) \leq \boldsymbol{0}, \\ \boldsymbol{h}(\boldsymbol{d}, \boldsymbol{e}) = \boldsymbol{0}, \end{cases} \end{aligned} \tag{4.1}$$

avec :

$$\begin{cases} f : \mathbb{R}^{n_x} \to \mathbb{R} \text{ la fonction objectif,} \\ \boldsymbol{d} = (d_1, d_2, ..., d_{n_d}) \in \mathbb{R}^{n_d} \text{ les variables de décision,} \\ \boldsymbol{e} = (e_1, e_2, ..., e_{n_e}) \in \mathbb{R}^{n_e} \text{ les paramètres environnementaux fixés,} \\ \boldsymbol{D} \subseteq \mathbb{R}^{n_d} \text{ l'espace de recherche,} \\ \boldsymbol{g}(\boldsymbol{d}, \boldsymbol{e}) = (g_1(\boldsymbol{d}, \boldsymbol{e}), g_2(\boldsymbol{d}, \boldsymbol{e}), ..., g_{n_g}(\boldsymbol{d}, \boldsymbol{e})) \in \mathbb{R}^{n_g} \text{ les contraintes d'inégalité,} \\ \boldsymbol{h}(\boldsymbol{d}, \boldsymbol{e}) = (h_1(\boldsymbol{d}, \boldsymbol{e}), h_2(\boldsymbol{d}, \boldsymbol{e}), ..., h_{n_h}(\boldsymbol{d}, \boldsymbol{e})) \in \mathbb{R}^{n_h} \text{ les contraintes d'égalité.} \end{cases}$$

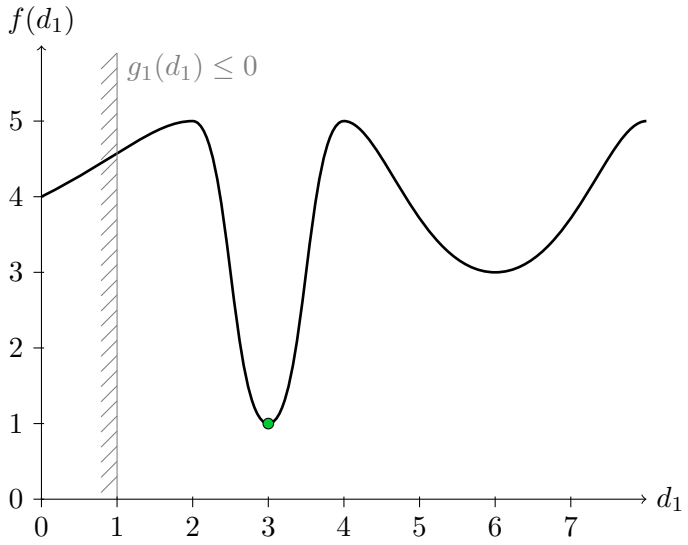


FIGURE 4.2 – Exemple de fonction objectif. L'optimum global est atteint pour $d_1 = 3$.

La figure 4.2 présente un exemple simple de fonction objectif f qui dépend d'une seule variable de design d_1 ($n_x = n_d = 1$). Une contrainte d'inégalité g_1 vient restreindre l'espace des solutions admissibles. Le problème a résoudre est le suivant :

$$\begin{aligned} &\underset{d_1 \in [0,8]}{\text{minimiser}}\ f(d_1), \\ &\text{s.c. } g_1(d_1) = 1 - d_1 \leq 0. \end{aligned}$$

Le minimum se situe ici pour $d_1 = 3$. Le système aura donc une performance optimale lorsque le paramètre de conception d_1 sera fixé à cette valeur. Il s'agit du minimum global de f, mais on observe aussi la présence d'un minimum local pour $d_1 = 6$.

Dans cette étude, nous traiterons de problèmes dont le nombre de variables de décision est relativement limité (de l'ordre de la dizaine au maximum). Lorsqu'on se trouve face à un problème possédant une plus grande dimension d'entrée, des techniques comme l'ACP (« Analyse en Composantes Principales », voir [AW10]) par exemple peuvent être employées pour réduire le nombre de variables et ne conserver que celles qui jouent un rôle prépondérant.

4.2 Optimisation multiobjectifs

Les problèmes d'optimisation qu'on rencontre sont en fait très souvent des problèmes multiobjectifs (même si tous les objectifs ne sont pas directement exprimés ou exprimables), c'est-à-dire qu'ils font intervenir plusieurs fonctions objectifs différentes que nous regrouperons en un vecteur $\boldsymbol{f} = (f_1, f_2, ..., f_{n_f})$. Dans le cadre du choix d'un moyen de transport pour se rendre à une destination donnée, on souhaitera par exemple emprunter un moyen de transport à la fois rapide et peu onéreux. On se retrouve ainsi face à un problème qui possède deux fonctions objectifs. Les méthodes d'optimisation multiobjectifs pour l'aide à la décision ont aujourd'hui largement dépassé les portes des laboratoires de recherche et sont utilisées au sein des entreprises. Une très bonne introduction à l'optimi-

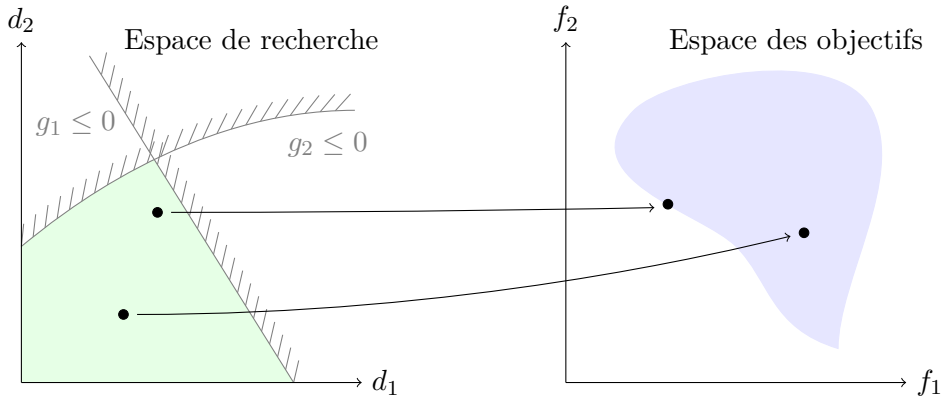


FIGURE 4.3 – De l'espace de recherche (à gauche) vers l'espace des objectifs (à droite). Deux solutions admissibles sont représentées par des points dans les deux espaces.

sation multiobjectifs dans son ensemble avec des méthodes de résolution de problèmes est donnée dans [CS03].

Dans le cadre de l'optimisation multiobjectifs, plusieurs objectifs souvent contradictoires doivent donc être minimisés de façon simultanée. Le problème devient alors :

$$\begin{aligned} &\underset{\boldsymbol{d}\in\boldsymbol{D}}{\text{minimiser}}\ \boldsymbol{f}(\boldsymbol{d},\boldsymbol{e}),\\ &\text{s.c.}\ \begin{cases} \boldsymbol{g}(\boldsymbol{d},\boldsymbol{e}) \leq \boldsymbol{0}, \\ \boldsymbol{h}(\boldsymbol{d},\boldsymbol{e}) = \boldsymbol{0}, \end{cases} \end{aligned} \tag{4.2}$$

avec $\boldsymbol{f}(\boldsymbol{d},\boldsymbol{e}) = (f_1(\boldsymbol{d},\boldsymbol{e}), f_2(\boldsymbol{d},\boldsymbol{e}), ..., f_{n_f}(\boldsymbol{d},\boldsymbol{e})) \in \mathbb{R}^{n_f}$ le vecteur des fonctions objectifs. $\boldsymbol{f}$ est maintenant une application définie depuis l'espace de recherche vers un espace des objectifs de dimension n_f (alors qu'auparavant il était de dimension 1). Le schéma 4.3 présente un exemple à deux dimensions ($n_d = 2$ et $n_f = 2$), avec un espace de recherche limité par deux contraintes inégalités g_1 et g_2.

L'opérateur de minimisation du problème 4.2 devrait en fait être écrit entre guillemets car la notion de minimum d'un ensemble de fonctions n'a pas encore été définie. La solution d'un problème d'optimisation multiobjectifs n'est en effet généralement pas unique : il peut exister plusieurs solutions de compromis entre les différents objectifs. Nous considèrerons ici la notion d'optimalité au sens de Pareto.

Les compromis optimaux d'un problème d'optimisation multiobjectifs au sens de Pareto constituent ce qu'on appelle un « front de Pareto » dans l'espace des objectifs. La définition de ce front est basée sur la notion de « dominance ». La dominance de Pareto définit une relation entre deux solutions candidates permettant de déterminer si l'une est meilleure que l'autre ou bien si les deux solutions sont équivalentes. On dit ainsi qu'un point $\boldsymbol{a}$ domine un point $\boldsymbol{b}$ si toutes les valeurs de ses objectifs sont plus petites que celles de $\boldsymbol{b}$. La solution $\boldsymbol{a}$ sera alors considérée plus intéressante que $\boldsymbol{b}$ car on souhaite minimiser la valeur des objectifs. La dominance (stricte) de Pareto, notée $\prec$, est définie comme suit :

$$\text{« } \boldsymbol{a} \text{ domine } \boldsymbol{b} \text{ »} \Leftrightarrow \boldsymbol{a} \prec \boldsymbol{b} \Leftrightarrow \begin{cases} \forall i \in \{1,2,...,n_f\}, f_i(\boldsymbol{a}) \leq f_i(\boldsymbol{b}), \\ \exists j \in \{1,2,...,n_f\} / f_j(\boldsymbol{a}) < f_j(\boldsymbol{b}). \end{cases} \tag{4.3}$$

Deux solutions peuvent être équivalentes au sens de Pareto, dans le cas où certains objectifs ont des valeurs inférieures et d'autres des valeurs supérieures. La dominance de Pareto

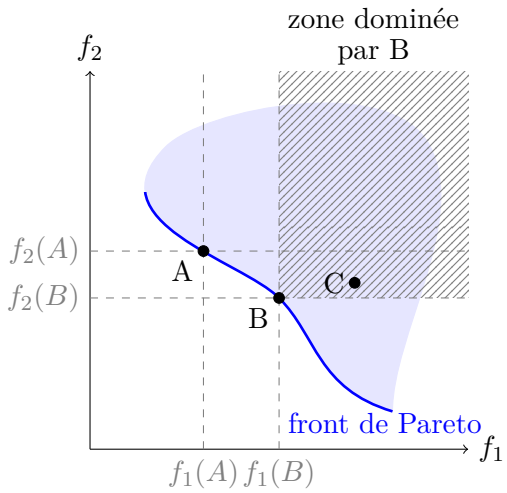


FIGURE 4.4 – Un exemple de front de Pareto (ligne bleue) pour un problème à deux objectifs. Les solutions A et B sont équivalentes. La solution C est dominée par B.

définit donc une relation d'ordre partiel parmi les points de l'espace des objectifs, ce qui permet de préciser la notion de minimisation d'un ensemble de fonctions.

Une solution dans l'espace des objectifs appartient au front de Pareto si elle n'est dominée par aucune autre solution de cet espace. Le front de Pareto contient ainsi l'ensemble des optima d'un problème d'optimisation multiobjectifs. Les points du front de Pareto sont optimaux dans le sens où si on essaye d'améliorer un des objectifs, on va nécessairement altérer un autre objectif. Le schéma 4.4 présente un exemple de front de Pareto pour le problème de minimisation de deux fonctions objectif f_1 et f_2 présenté précédemment sur la figure 4.3. Les points A et B sont situés sur le front de Pareto et sont des solutions optimales équivalentes, alors que le point C n'est pas optimal car il est dominé par B. Par contre, on peut remarquer que les points A et C sont équivalents au sens de la dominance de Pareto (car A$\not\prec$C et C$\not\prec$A).

Le front de Pareto est donc composé de l'ensemble des solutions optimales du problème d'optimisation, c'est-à-dire des solutions équivalentes entre elles et qui rassemblées dominent toutes les autres. Ce front peut être fini ou non, continu ou présenter des discontinuités, être convexe ou non, etc. Le décideur pourra ensuite choisir parmi ces différentes solutions celle qui lui convient le mieux. La figure 4.5 présente un exemple de problème d'optimisation qui met en jeu deux fonctions objectifs f_1 et f_2. L'ensemble des solutions optimales au sens de Pareto est l'intervalle $[2, 3]$ tout entier.

4.3 Algorithmes d'optimisation multiobjectifs

Le but des algorithmes de résolution de problèmes d'optimisation multiobjectifs est de trouver une des solutions optimales du problème ou bien d'approcher au mieux l'ensemble du front de Pareto. Nous détaillons ci-dessous différentes approches pour y parvenir, allant des méthodes agrégatives qui se ramènent à un problème mono-objectif aux algorithmes stochastiques évolutionnaires.

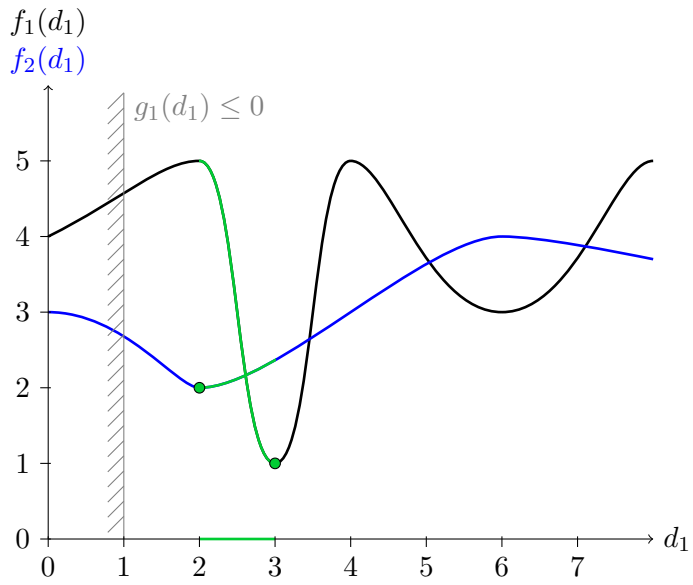


FIGURE 4.5 – Exemple de problème d'optimisation à deux objectifs. Les solutions optimales au sens de Pareto forment l'intervalle $[2, 3]$.

4.3.1 Trouver une unique solution optimale

La première possibilité pour résoudre un problème d'optimisation multiobjectifs est de se ramener à un problème mono-objectif qui pourra être résolu avec une méthode d'optimisation « classique ». Ces méthodes retournent une seule et unique solution optimale. Comme le décideur ne retiendra au final qu'une solution parmi l'ensemble du front de Pareto, les choix menant à sa sélection peuvent en effet être réalisés a priori.

L'approche la plus basique pour obtenir une solution du front de Pareto est d'agréger les objectifs en une somme pondérée :

$$f(\boldsymbol{x}) = \sum_{i=1}^{n_f} w_i f_i(\boldsymbol{x}). \tag{4.4}$$

Les poids de l'agrégation $w_i \in \mathbb{R}$ sont définis relativement à l'importance de chaque objectif par rapport aux autres. On obtient au final une unique solution qui dépend du choix de ces poids. Le décideur doit donc exprimer a priori ses préférences sur les différents objectifs. Cette manière de faire ne permet toutefois pas d'accéder aux points situés dans la partie concave d'un front de Pareto.

Le décideur peut aussi définir une solution idéale qu'il souhaiterait atteindre, c'est-à-dire des valeurs « utopiques » pour chacun des objectifs. On va alors chercher à se rapprocher au plus près de ce point idéal, au sens d'une norme définie. On parle de méthodes de but. Elles se ramènent à la résolution d'un problème mono-objectif de minimisation de distance.

La méthode ε-contrainte consiste quant à elle à minimiser l'un des objectifs (généralement celui qui est considéré comme le plus important) en imposant des contraintes de valeur maximale sur les autres. Ces contraintes doivent être définies a priori selon les connaissances et les attentes du décideur. Les contraintes peuvent aussi être adaptées au fur et à mesure de l'optimisation avec des méthodes dites « interactives ».

On peut enfin citer les approches qui se ramènent à la résolution successive de plusieurs problèmes d'optimisation mono-objectif. Pour cela, on traite généralement les objectifs un par un. Dans la méthode lexicographique par exemple, les objectifs sont classés par ordre d'importance et l'optimisation est d'abord réalisée uniquement sur l'objectif le plus important. Le second objectif est ensuite optimisé à son tour mais avec une contrainte d'égalité sur la valeur du premier objectif (afin que celui-ci conserve sa valeur optimale). Les optimisations successives se continuent ainsi pour les objectifs suivants jusqu'à ce qu'on ait traité tous les objectifs. On aura alors résolu autant de problèmes d'optimisation mono-objectif qu'il y avait d'objectifs dans le problème.

Les méthodes qui retournent une unique solution ont l'avantage d'être rapides (car on résout un problème plus simple que le problème de départ), mais elles nécessitent que le décideur définisse a priori certaines préférences sur les objectifs. Cela n'est pas toujours facile, et celui-ci souhaite parfois pouvoir faire ce choix a posteriori parmi les solutions du front de Pareto. La connaissance de l'ensemble du front donne en effet une meilleure vision des différentes possibilités offertes pour chacun des objectifs et des compromis qui doivent être réalisés. D'autres méthodes ont donc été développées afin de trouver un ensemble de points bien répartis sur le front de Pareto.

4.3.2 Trouver l'ensemble du front de Pareto

Les méthodes qui permettent d'obtenir un ensemble de solutions optimales sont appelées « méthodes a posteriori » car le choix d'une solution se fait après avoir résolu le problème d'optimisation (par opposition aux méthodes précédentes pour lesquelles les choix sont faits a priori).

Les premières méthodes proposées pour retourner des solutions du front de Pareto sont inspirées des algorithmes précédents. Il s'agit en fait de résoudre successivement plusieurs problèmes d'optimisation mono-objectif avec différents paramètres pour obtenir des solutions Pareto-optimales différentes. On peut par exemple faire varier les pondérations des objectifs dans la méthode agrégative afin de reconstituer peu à peu le front de Pareto (voir [KdW06] et [JOS01]). Mais le problème est qu'on n'est pas garanti d'obtenir une bonne répartition des points sur le front à partir d'une répartition régulière des poids sélectionnés.

D'autres algorithmes ont donc été élaborés pour que les solutions soient mieux réparties le long du front de Pareto. C'est notamment le cas des méthodes NBI (« Normal Bound Intersection », voir [DD98]) et DSD (« Directed Search Domain », voir [EU10]). Dans ces approches, la recherche de points du front de Pareto est menée à partir d'un hyperplan défini par les points optimaux extrêmes, c'est-à-dire les points obtenus en optimisant un unique objectif (on obtient ainsi un point optimal par objectif). En partant d'un point de cet hyperplan, la méthode NBI réalise une optimisation mono-objectif le long de la droite normale à l'hyperplan. On trouve alors le point du front de Pareto situé sur cette droite (selon le cas ce point peut parfois ne pas être Pareto-optimal). L'algorithme DSD recherche quant à lui le point optimal à l'intérieur d'un cône issu d'un point de l'hyperplan. Là aussi l'optimisation est mono-objectif grâce à une agrégation des objectifs, le cône qui définit l'espace de recherche étant modélisé par un ensemble de contraintes ajoutées au problème. Lorsqu'on effectue des recherches d'optima avec ces méthodes à partir de différents points équitablement répartis sur l'hyperplan, on obtient des points relativement bien répartis le long du front de Pareto (en dehors de cas particuliers). Elles permettent donc d'obtenir

une bonne représentation du front optimal, mais peuvent parfois perdre de leur efficacité selon la forme du front de Pareto recherché.

Des algorithmes stochastiques ont aussi été développés spécifiquement pour l'optimisation multiobjectifs. Il s'agit d'heuristiques, c'est-à-dire des méthodes de résolution approchée mais qui permettent de trouver de bonnes solutions en un temps raisonnable. On peut notamment citer la recherche tabou ou bien le recuit simulé, mais les méthodes stochastiques les plus étudiées aujourd'hui sont les algorithmes dits évolutionnaires. Ils ont été inspirés par la biologie. On retrouve notamment parmi eux les algorithmes génétiques, les colonies de fourmis ainsi que les essaims particulaires. Une présentation générale des méthodes évolutionnaires est donnée dans [FF95].

Nous nous sommes intéressés plus particulièrement aux algorithmes génétiques. Les algorithmes génétiques sont basés sur les notions d'évolution et de sélection naturelle. Ils utilisent une population d'individus représentant chacun une possible solution au problème, et dont les meilleurs éléments sont sélectionnés à chaque itération pour donner naissance à de nouveaux individus par croisement et mutation. Les croisements et les mutations sont des échanges ou de légères modifications des valeurs des variables de décision des individus, ce qui permet d'obtenir de nouveaux individus « proches » de leurs parents (les individus qui les ont engendrés). Les parents ayant été sélectionnés parmi les bons candidats, on espère ainsi que les enfants qu'ils auront produits seront aussi des solutions intéressantes et possiblement meilleures que leurs parents. Génération après génération, la population de solutions se rapproche donc du front de Pareto du problème. De nombreux algorithmes génétiques différents existent selon les techniques choisies pour l'évaluation de la qualité des solutions, la sélection des parents et les opérations de croisement et de mutation. L'un d'eux, nommé NSGA-II, est présenté plus en détail dans la section suivante.

L'inconvénient des algorithmes génétiques est qu'il faut régler certains de leurs paramètres pour un fonctionnement optimal (notamment la taille de la population simulée, le nombre de générations, les probabilités de croisement et de mutation, etc.). Ils sont aussi coûteux en termes de nombre d'évaluations des fonctions objectifs et ne retournent que des solutions approchées. Ces algorithmes sont par contre assez robustes et peuvent s'adapter à de nombreux problèmes différents sans limitation vis-à-vis des propriétés des fonctions objectifs ni de la forme du front de Pareto recherché. Dans cette étude, les applications que nous allons traiter sont basées sur des simulations numériques et des codes de calcul complexes. Les objectifs à optimiser sont considérés comme des boîtes noires pour lesquelles on ne dispose d'aucune information a priori (convexité, dérivée, etc.). Les algorithmes génétiques sont donc tout à fait adaptés à ce type de problèmes, et c'est pourquoi nous avons choisi cette technique d'optimisation.

4.3.3 NSGA-II

L'algorithme génétique que nous avons sélectionné pour résoudre les problèmes d'optimisation rencontrés est l'algorithme NSGA-II (« Nondominated Sorting Genetic Algorithm II », voir [DPAM02]). Il s'agit d'un des algorithmes génétiques les plus répandus et les plus éprouvés aujourd'hui. Il s'adapte généralement bien à différents environnements et arrive à résoudre une large palette de problèmes différents. C'est donc sa robustesse et sa relative efficacité qui a guidé ce choix. Le principe de cet algorithme est présenté ci-dessous.

Comme tout algorithme génétique, NSGA-II est basé sur l'évolution d'une population

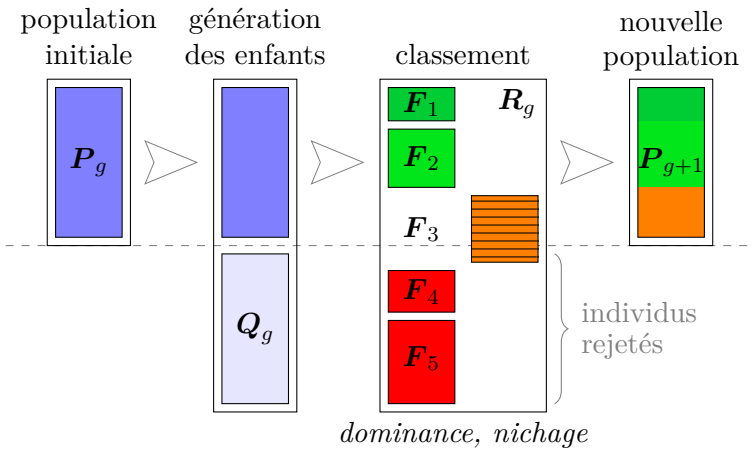


FIGURE 4.6 – Obtention de la nouvelle population par croisement et mutation, puis classement par dominance et nichage (d'après [DPAM02]).

d'individus $\boldsymbol{P}_g$ (voir la figure 4.6). Des « enfants » $\boldsymbol{Q}_g$ sont tout d'abord générés par croisement et mutation des individus de la population d'origine. Il existe de nombreuses méthodes de croisement et mutations qui ne seront pas détaillées ici. Nous pouvons par exemple citer les techniques standard que nous avons utilisées pour cette étude : la sélection par tournoi binaire et croisement binaire simulé (« SBX », voir [DA94]), ainsi que la mutation polynomiale [DG96].

Après les étapes de croisement et de mutation, on effectue un classement de la population $\boldsymbol{R}_g$ qui regroupe parents et enfants pour ne conserver que les meilleurs d'entre eux et créer la nouvelle population $\boldsymbol{P}_{g+1}$. La particularité de NSGA-II se situe au niveau de sa façon d'effectuer le classement des individus. Ce classement s'effectue en deux temps, en commençant par un classement par dominance de Pareto. Le rang 1 est ainsi attribué à tous les individus non dominés de la population courante $\boldsymbol{R}_g$ et qui forment un front de Pareto. Ces individus sont ensuite retirés de la population et on procède de même pour déterminer les individus de rang 2. Ce traitement est effectué récursivement en attribuant le rang k aux individus non dominés de la population initiale à laquelle ont été retirés les individus des rangs 1 à $(k-1)$. Le processus s'arrête lorsqu'un rang unique a été associé à tous les individus de la population courante. Un exemple de classement pour un problème bi-objectifs est présenté sur la figure 4.7.

La seconde étape du classement repose sur une technique dite de « nichage ». Les individus des premiers rangs sont directement sélectionnés pour faire partie de la nouvelle population $\boldsymbol{P}_{g+1}$ comme indiqué sur la figure 4.6. Mais cette population a une taille limitée, et il peut arriver que tous les individus du dernier rang sélectionné ne puissent pas en faire partie (le rang $\boldsymbol{F}_3$ dans notre exemple). On utilise donc la méthode de nichage pour différencier les solutions d'un même rang et choisir celles qui pourront intégrer la nouvelle population. Le principe de cette technique est d'éviter de sélectionner des individus trop proches les uns des autres afin que les solutions finales soient suffisamment diversifiées et bien réparties sur le front de Pareto. Sur la figure 4.7, on voit ainsi que seuls certains points du front $\boldsymbol{F}_3$ sont conservés, en prenant soin de choisir des points bien répartis sur le front.

Au sein de NSGA-II, l'évaluation de la densité de population présente autour d'un

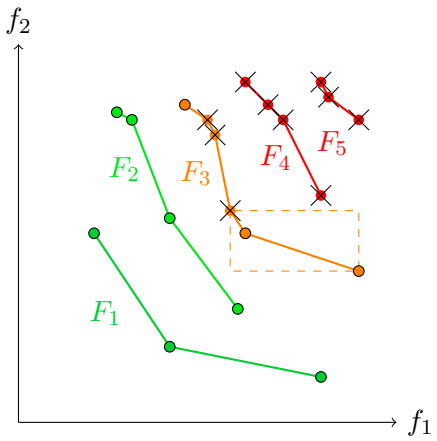


FIGURE 4.7 – Classement des solutions dans l'algorithme NSGA-II par dominance puis par nichage pour $\boldsymbol{F}_3$. Les 10 solutions retenues sont entourées en noir, et celles qui sont rejetées sont marquées d'une croix.

Algorithme 4.1 : Classement_nichage($\boldsymbol{F}_i$)

$NombreSolutions \leftarrow |\boldsymbol{F}_i|$
pour $j \in \{1, 2, ..., NombreSolutions\}$ **faire**
    $\boldsymbol{DistancesNichage}[j] \leftarrow 0$
fin
pour $j \in \{1, 2, ..., n_f\}$ **faire**
    $\boldsymbol{T} \leftarrow$ Trier_selon_valeur_objectif($\boldsymbol{F}_i, f_j$)
    $\boldsymbol{DistancesNichage}[\boldsymbol{T}_1] \leftarrow +\infty$
    $\boldsymbol{DistancesNichage}[\boldsymbol{T}_{NombreSolutions}] \leftarrow +\infty$
    $VariationGlobalef \leftarrow f_j(\boldsymbol{F}_i[\boldsymbol{T}_{NombreSolutions}]) - f_j(\boldsymbol{F}_i[\boldsymbol{T}_1])$
    pour $k \in \{2, 3, ..., NombreSolutions - 1\}$ **faire**
        $\Delta f \leftarrow f_j(\boldsymbol{F}_i[\boldsymbol{T}_{k+1}]) - f_j(\boldsymbol{F}_i[\boldsymbol{T}_{k-1}])$
        $\boldsymbol{DistancesNichage}[\boldsymbol{T}_k] \leftarrow \boldsymbol{DistancesNichage}[\boldsymbol{T}_k] + \frac{\Delta f}{VariationGlobalef}$
    fin
fin
Trier_selon_distance($\boldsymbol{F}_i, \boldsymbol{DistancesNichage}$)

point donné est relativement simple. Le déroulement du calcul de la distance de nichage pour classer un front $\boldsymbol{F}_i$ contenant $NombreSolutions$ individus est présenté dans l'algorithme 4.1. La distance de nichage affectée à un point donné correspond en fait à une distance de type Manhattan entre ses deux plus proches voisins dans le front. Les distances sont normalisées sur chacun des objectifs par rapport à la plage de variation maximale de l'objectif. Les points extrêmaux du front sont systématiquement sélectionnés car ils bénéficient d'une distance de nichage infinie. Finalement, les solutions dont la distance de nichage est la plus grande sont classées en premier et elles seront donc sélectionnées en priorité pour faire partie de la nouvelle population.

Le déroulement général des opérations effectuées lors d'une itération de NSGA-II est résumé dans l'algorithme 4.2. On choisit typiquement une taille de population de l'ordre de

Algorithme 4.2 : Principe de NSGA-II

pour *chaque génération g de population* $\boldsymbol{P}_g$ **faire**
  $\boldsymbol{Q}_g \leftarrow$ Générer_fils($\boldsymbol{P}_g$)
  Évaluer_objectifs($\boldsymbol{Q}_g$)
  $\boldsymbol{R}_g \leftarrow \boldsymbol{P}_g \cup \boldsymbol{Q}_g$
  $\boldsymbol{F} \leftarrow$ Classement_dominance($\boldsymbol{R}_g$)
  $\boldsymbol{P}_{g+1} \leftarrow \emptyset$
  $i \leftarrow 1$
  tant que $|\boldsymbol{P}_{g+1}| + |\boldsymbol{F}_i| \leq TaillePopulation$ **faire**
    $\boldsymbol{P}_{g+1} \leftarrow \boldsymbol{P}_{g+1} \cup \boldsymbol{F}_i$
    $i \leftarrow i + 1$
  fin
  Classement_nichage($\boldsymbol{F}_i$)
  $\boldsymbol{P}_{g+1} \leftarrow \boldsymbol{P}_{g+1} \cup \boldsymbol{F}_i[1 : TaillePopulation - |\boldsymbol{P}_{g+1}|]$
fin

$TaillePopulation \in [20, 100]$ et un nombre de générations $NombreGénérations \in [200, 500]$ (ces chiffres sont donnés à simple titre indicatif et doivent être adaptés en fonction de la difficulté du problème traité). Le nombre d'évaluations des objectifs au cours d'une optimisation est ainsi de l'ordre de $(TaillePopulation \times NombreGénérations)$.

L'algorithme NSGA-II standard ne s'intéresse qu'aux objectifs du problème et ne permet donc pas de prendre en compte de contraintes. Des techniques similaires de classement par front ont néanmoins été développées pour gérer la présence de contraintes. Elles se basent notamment sur une mesure de la distance d'une solution à l'espace admissible (voir par exemple [OSF05] pour plus d'informations à ce sujet).

Pour garder en mémoire l'ensemble des solutions optimales visitées lors des itérations de l'algorithme, on peut avoir recours à une archive dans laquelle on stocke les meilleurs individus obtenus à chaque génération. Seules les solutions optimales au sens de Pareto y sont conservées, et un classement par dominance de Pareto est donc effectué à chaque nouvel ajout pour éliminer les solutions dominées. L'archive ayant une capacité limitée, un tri par nichage peut aussi être effectué. NSGA-II prévoit initialement de retourner uniquement les individus optimaux de la dernière génération, mais l'utilisation d'une archive permet de s'assurer qu'aucune des solutions visitées ne sera perdue, au prix d'un léger alourdissement de l'algorithme pour la gestion de cette archive.

NSGA-II permet finalement d'obtenir un ensemble de solutions qui forment une approximation du front de Pareto optimal du problème. Le décideur pourra ensuite choisir parmi ces solutions le compromis qui lui convient le mieux.

4.4 Élagage du front de Pareto par classification

Le front de Pareto retourné par les algorithmes de résolution de problèmes d'optimisation multiobjectifs est souvent constitué d'un grand nombre de solutions. Le choix parmi cet ensemble est alors parfois difficile pour le décideur. Diverses méthodes ont été développées pour l'assister dans ce choix. Il est par exemple possible de faire un élagage (« pruning » en anglais) du front de Pareto pour ne conserver qu'un nombre réduit de

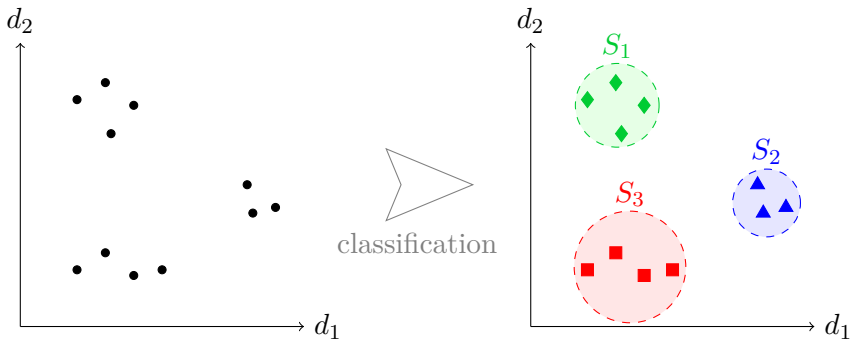


FIGURE 4.8 – Classification d'un ensemble de solutions en trois groupes S_1, S_2 et S_3.

solutions représentatives. L'élagage se base sur une classification des solutions visant à les regrouper en un certain nombre d'ensembles contenant des solutions similaires (voir la figure 4.8). Un exemple d'élagage du front de Pareto par classification est présenté dans [TBCW07]. Si le partitionnement de données en deux dimensions peut sembler assez évident, cela peut devenir beaucoup plus complexe lorsque la dimension augmente.

La classification non supervisée a pour but de regrouper un ensemble de données en différents paquets homogènes, de façon à ce que les données de chaque sous-ensemble partagent des caractéristiques communes. Ces caractéristiques correspondent le plus souvent à des critères de proximité qu'on définit en introduisant une mesure de distance entre deux données. La mesure de distance peut par exemple être simplement définie comme la distance euclidienne entre deux points dans un espace normalisé. Le problème de la classification est un problème complexe et les méthodes proposées pour le résoudre sont le plus souvent des heuristiques, c'est-à-dire des méthodes de résolution approchée.

L'algorithme de classification le plus simple et le plus connu est l'algorithme des « k-moyennes » (« k-means » en anglais [HW79]). Il consiste à faire évoluer dans l'espace un ensemble de points représentant les centres $\boldsymbol{c}_i$ des classes S_i. La position de ces points est choisie aléatoirement au départ, en prenant autant de centres qu'on souhaite obtenir de classes. Chaque solution à classer est ensuite assignée à la classe dont le centre est le plus proche (au sens de la mesure de distance définie). Dans un deuxième temps, les centres $\boldsymbol{c}_i$ sont recalculés comme étant le barycentre de leur classe. On peut alors réassigner les données aux classes, et l'algorithme continue ainsi jusqu'à ce que la composition des classes ne soit plus modifiée entre deux itérations. Le principe de l'algorithme des k-moyennes est résumé dans l'algorithme 4.3. Il est ici nécessaire de définir à l'avance le nombre k de classes recherchées, mais il existe aussi des méthodes pour calculer automatiquement un nombre de classes optimal en fonction des données à classer.

Comme les centres des classes $\boldsymbol{c}_i$ sont définis au hasard lors de l'initialisation, on peut effectuer une classification k-moyennes plusieurs fois d'affilée avec des initialisations différentes et calculer ainsi des classes « moyennées » afin d'éviter de se retrouver avec une classification sous-optimale que peut parfois retourner l'heuristique du fait d'une mauvaise initialisation.

Les centres des classes $\boldsymbol{c}_i$ définis comme des barycentres sont des points quelconques de l'espace comme présenté ci-dessus, ce qui convient tout à fait lorsqu'on souhaite simplement regrouper les solutions par paquets. Mais on peut aussi imposer que les centres soient situés sur des points solution ($\boldsymbol{c}_i$ est alors la solution la plus proche du barycentre de

Algorithme 4.3 : Classification par k-moyennes

Choisir au hasard les centres initiaux des classes $\boldsymbol{c}_1^0, \boldsymbol{c}_2^0, ..., \boldsymbol{c}_k^0$
$t \leftarrow 0$
tant que *la composition des classes change* **faire**
    pour *chaque classe* S_i **faire**
        $\boldsymbol{S}_i^{t+1} \leftarrow \emptyset$
    fin
    pour *chaque point* $\boldsymbol{x}$ *à classer* **faire**
        Trouver i tel que $\forall j \in \{1, 2, ..., k\}, ||\boldsymbol{x} - \boldsymbol{c}_i^t|| \leq ||\boldsymbol{x} - \boldsymbol{c}_j^t||$
        $\boldsymbol{S}_i^{t+1} \leftarrow \boldsymbol{S}_i^{t+1} \cup \{\boldsymbol{x}\}$
    fin
    pour *chaque classe* $\boldsymbol{S}_i$ **faire**
        $\boldsymbol{c}_i^{t+1} \leftarrow \frac{1}{|\boldsymbol{S}_i^{t+1}|} \sum_{\boldsymbol{x} \in \boldsymbol{S}_i^{t+1}} \boldsymbol{x}$
    fin
    $t \leftarrow t + 1$
fin

la classe). Le centre de la classe est alors considéré comme une solution représentative du groupe, ce qui peut être intéressant si on souhaite présenter au décideur une seule solution par classe.

L'inconvénient des k-moyennes est qu'il s'agit d'un algorithme très simple qui vise à constituer des classes sphériques de tailles similaires. Selon les jeux de données à trier, il pourra donc se révéler plus ou moins efficace. Cet algorithme de classification est par contre très rapide et c'est ce qui fait son intérêt.

Il existe d'autres méthodes plus intelligentes de classification, comme par exemple l'algorithme EM (« Expectation-Maximization », voir [DLR77]). Ces méthodes n'ont pas été retenues ici du fait de leur coût plus élevé en temps de calcul. La classification n'étant pas le cœur de cette étude, nous avons préféré utiliser une méthode simple et rapide qui pourra être améliorée plus tard.

L'élagage par classification des solutions du front de Pareto permet ainsi de présenter un nombre restreint de solutions optimales parmi lesquelles le décideur pourra faire son choix plus aisément.

4.5 Test des algorithmes d'optimisation

Nous avons vu que de nombreux algorithmes d'optimisation différents existent. Pour les tester et comparer leurs performances, des problèmes analytiques spécifiques ont été définis. Nous présentons quelques exemples choisis ci-dessous. Les résultats obtenus par chaque algorithme sur ces problèmes de test peuvent ensuite être dépouillés grâce à des mesures de performance qui permettent de comparer les algorithmes et de déterminer les points forts et les faiblesses de chacun.

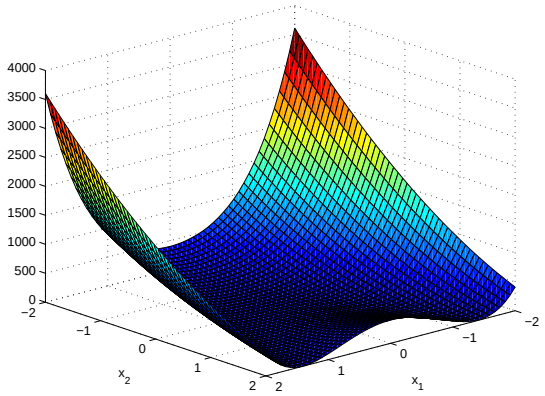


FIGURE 4.9 – Fonction de Rosenbrock.

4.5.1 Problèmes de test

Des fonctions plus ou moins complexes ont été proposées pour mettre à l'épreuve les algorithmes d'optimisation. Le minimum de ces fonctions est généralement connu de manière analytique et on peut donc comparer la solution obtenue avec un algorithme donné par rapport à cet optimum idéal. Ces fonctions de test ont été créées pour l'évaluation des algorithmes d'optimisation mono-objectif, mais on peut aussi les combiner pour définir des problèmes multiobjectifs. Des problèmes spécifiquement multiobjectifs ont aussi été proposés. Nous présentons ci-dessous des fonctions de test standard parmi les plus courantes.

Fonction de Rosenbrock

La fonction de Rosenbrock [Ros60], encore appelée fonction « banane », est une fonction de deux variables qui présente une large vallée dans laquelle se cache le minimum global (elle est représentée sur la figure 4.9). Trouver la vallée de cette fonction est donc très facile, mais il est plus compliqué de converger rapidement et précisément vers l'optimum global.

L'expression analytique de la fonction de Rosenbrock est la suivante :

$$f(x_1, x_2) = (1 - x_1)^2 + 100(x_1^2 - x_2)^2. \tag{4.5}$$

On prend typiquement $\boldsymbol{x} \in [-2, 2]^2$ comme espace de recherche. Le minimum se situe en $(1, 1)$.

Une extension à un nombre quelconque de dimensions n_x a aussi été définie par la suite :

$$f(\boldsymbol{x}) = \sum_{i=1}^{n_x - 1} (1 - x_i)^2 + 100(x_i^2 - x_{i+1})^2. \tag{4.6}$$

Contrairement à la version initiale, cette nouvelle fonction peut posséder plusieurs minima locaux selon la dimension d'entrée.

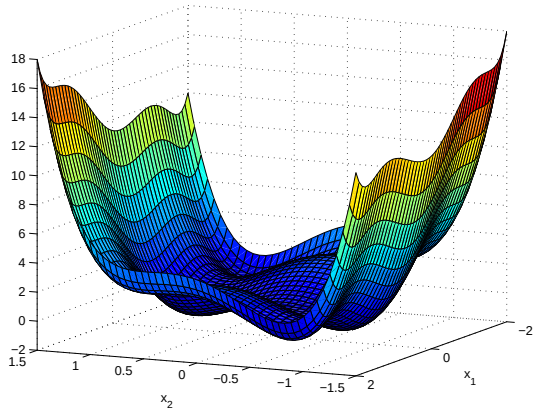


FIGURE 4.10 – Fonction « chameau ».

Fonction « chameau »

La fonction chameau (ou « six-hump camel back function » en anglais) a été introduite dans [DS78]. Il s'agit d'une fonction de deux variables qui présente six minima locaux dont deux minima globaux. Cette fonction est représentée sur la figure 4.10.

L'expression de la fonction chameau est :

$$f(x_1, x_2) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1 x_2 + 4x_2^2(x_2^2 - 1). \tag{4.7}$$

On prend généralement $\boldsymbol{x} \in [-2, 2]^2$. Les minima globaux se trouvent approximativement en $(-0.0898, 0.7126)$ et $(0.0898, -0.7126)$.

Fonction de Michalewicz

La fonction de Michalewicz [Mic96] est une fonction multimodale constituée de nombreuses vallées de différentes profondeurs au croisement desquelles se situent des minima locaux. Son expression est la suivante :

$$f(\boldsymbol{x}) = -\sum_{i=1}^{n_x} \sin(x_i) \sin^{2m}\left(\frac{i x_i^2}{\pi}\right). \tag{4.8}$$

On prend généralement $\boldsymbol{x} \in [0, \pi]^{n_x}$, mais un espace de recherche plus large peut aussi être défini si on souhaite augmenter le nombre de vallées présentes.

Le paramètre $m \in \mathbb{R}$ permet de définir la largeur des vallées. Plus m est grand, plus le problème est complexe car les vallées deviennent des entailles abruptes au sein d'un vaste plateau, rendant l'optimum beaucoup plus difficile à trouver. La figure 4.11 présente deux exemples pour un espace d'entrée de dimension $n_x = 2$. Le minimum global est situé approximativement en $(2.203, 1.571)$.

Problèmes multiobjectifs

Le domaine de l'optimisation multiobjectifs est bien fourni en problèmes de test pour évaluer la qualité des algorithmes de résolution. Les cas tests les plus employés aujour-

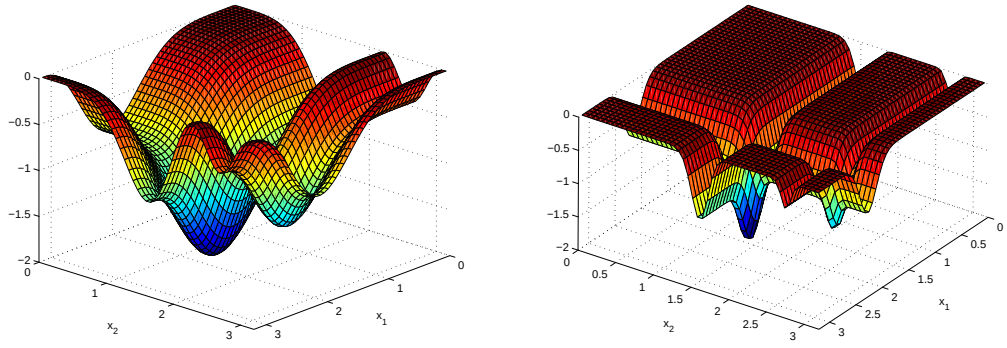


FIGURE 4.11 – Fonction de Michalewicz pour $m = 1$ (à gauche) et $m = 10$ (à droite).

d'hui sont les suites ZDT [ZDT00] et DTLZ [DTLZ01]. Elles sont par exemple analysées dans [HHBW06]. L'objectif de ces suites de test est d'offrir un ensemble de problèmes dont on peut faire varier la taille (en termes de dimension d'entrée ou de nombre d'objectifs). Ils se veulent représentatifs de la diversité des problèmes réels auxquels on peut avoir à faire face.

Nous présentons ici l'exemple du problème ZDT6 qui fait intervenir deux objectifs et un nombre quelconque de variables regroupées en un vecteur $\boldsymbol{x} \in [0,1]^{n_x}$. Le but est de minimiser deux objectifs f_1 et f_2 définis comme suit :

$$\begin{cases} f_1(\boldsymbol{x}) = 1 - e^{-4x_1} sin^6(6\pi x_1), \\ f_2(\boldsymbol{x}) = g(\boldsymbol{x}) h(f_1(\boldsymbol{x}), g(\boldsymbol{x})), \end{cases} \tag{4.9}$$

$$\text{avec} \quad g(\boldsymbol{x}) = 1 + 9\sqrt[4]{\frac{\sum_{i=2}^{n_x} x_i}{n_x - 1}},$$
$$h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^2.$$

Le front de Pareto de ce problème multiobjectifs peut être calculé analytiquement. Il est obtenu en posant $g(\boldsymbol{x}) = 1$, ce qui correspond à l'équation $f_2 = 1 - f_1^2$ pour $f_1 \in [0.281, 1]$ environ. Ce front est tracé sur la figure 4.12, ainsi qu'un ensemble de points choisis au hasard dans l'espace de recherche. On remarque que ces points sont assez éloignés du front optimal, et que leur densité diminue très fortement à son approche. Les solutions optimales sont donc difficiles à trouver et une recherche purement aléatoire est vaine. Les solutions ne sont pas non plus réparties de façon homogène le long du front de Pareto, la densité étant plus forte lorsque f_1 est proche de 1.

4.5.2 Mesures de performance

Les problèmes présentés ci-dessus permettent de tester différents algorithmes de résolution. Pour pouvoir les comparer de manière quantitative, plusieurs mesures de performances ont été proposées.

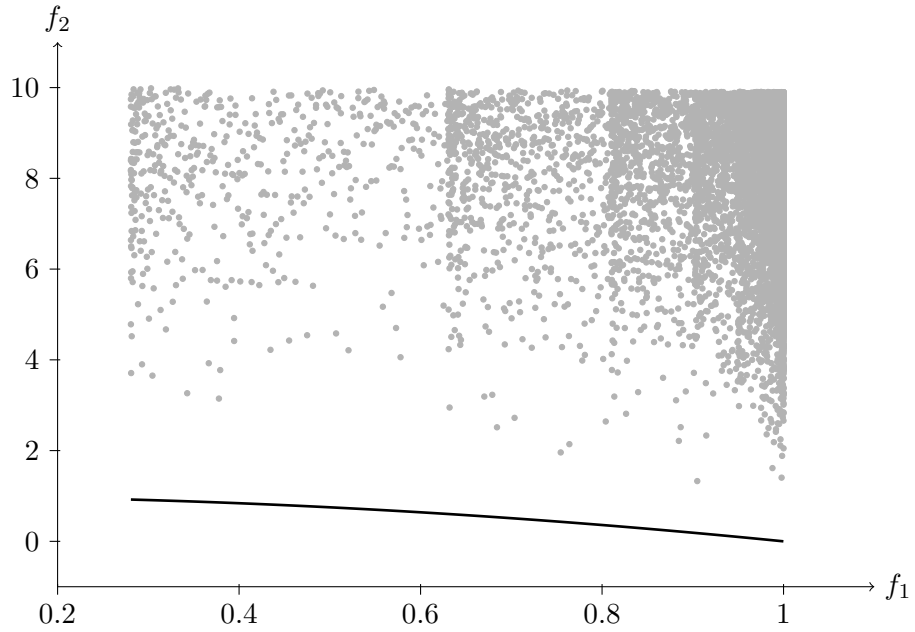


FIGURE 4.12 – Problème ZDT6 : visualisation dans l'espace des objectifs d'un LHS de 10000 points (en gris) tirés dans un espace d'entrée de dimension $n_x = 2$, ainsi que du front de Pareto analytique du problème (courbe noire).

La mesure de la performance d'un algorithme d'optimisation mono-objectif reste assez simple. Il suffit par exemple de déterminer la distance de la solution obtenue par rapport à l'optimum réel pour obtenir une mesure qui permet de comparer l'efficacité de différents algorithmes. Dans le cadre de l'optimisation multiobjectifs, le problème se complexifie car le résultat n'est plus une unique solution mais un ensemble de points constituant un front de Pareto. Il faut alors utiliser des indicateurs de performance adaptés afin de pouvoir analyser la convergence de l'algorithme et la diversité des solutions. Un tour d'horizon de ces indicateurs est présenté dans [ZKT08]. Nous introduisons trois d'entre eux ci-dessous : les indicateurs de dispersion et de contribution, ainsi que la mesure de l'hypervolume.

Dispersion

L'indicateur de dispersion (« spread » [DPAM02]) permet de déterminer si les solutions retournées par un algorithme d'optimisation multiobjectifs sont bien réparties sur le front de Pareto et le recouvrent complètement. On souhaite en effet que les solutions optimales représentent le front de Pareto dans son ensemble afin que tous les compromis possibles soient proposés au décideur. Un algorithme dont les solutions sont bien dispersées le long du front de Pareto sera ainsi préféré à un autre dont les solutions sont regroupées sur une simple portion de ce front.

Contribution

L'indicateur de contribution [MTR00] permet de comparer la convergence de deux algorithmes en déterminant la proportion de points optimaux apportés par chacun d'eux

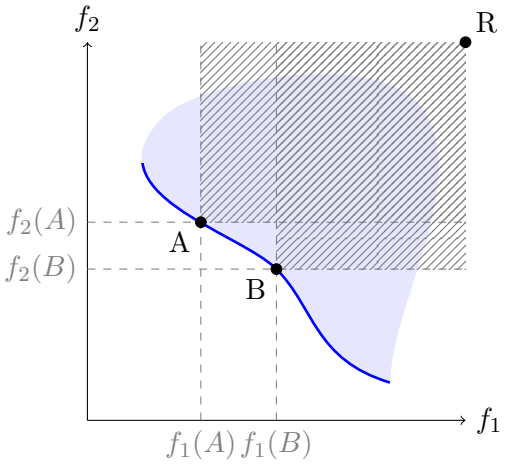


FIGURE 4.13 – Hypervolume $\mathcal{H}$ (surface grisée) défini par un front de Pareto composé de deux solutions A et B par rapport à un point de référence R.

quand on réunit leurs solutions en un même ensemble. On peut ainsi déterminer lequel des deux a convergé plus rapidement vers des solutions intéressantes.

Hypervolume

L'hypervolume $\mathcal{H}$ [ZT99] est un indicateur qui permet de mesurer à la fois la diversité et la convergence des solutions retournées par un algorithme d'optimisation multiobjectifs. Son principe est basé sur la mesure dans l'espace des objectifs de l'aire dominée par les solutions du front de Pareto fourni par l'algorithme. Lorsque le nombre d'objectifs est supérieur à 2, cette l'aire correspond en fait à un hypervolume (d'où le nom de cette mesure).

L'hypervolume est calculé à partir d'un point maximal de référence R. Ce point peut par exemple être obtenu en prenant les plus mauvaises valeurs observées pour chaque objectif. La mesure de l'hypervolume est évaluée en construisant pour chacune des solutions du front de Pareto un hypercube entre la solution et ce point de référence R. Le regroupement de tous les hypercubes donne alors l'hypervolume du front. Un exemple est présenté sur la figure 4.13 pour un front de Pareto constitué de deux points A et B seulement. Il est à noter que les objectifs doivent être normalisés au préalable afin que la mesure reste équitable.

La mesure de l'hypervolume permet de visualiser l'avancement d'un algorithme au fur et à mesure de ses itérations, ou bien de comparer les résultats obtenus par différents algorithmes. On souhaite en effet que le front de Pareto trouvé définisse un hypervolume le plus grand possible, car on aura alors une large portion de l'espace des objectifs dominée par ses solutions. L'hypervolume est un bon indicateur de convergence et de dispersion, mais il est toutefois assez long à évaluer car la complexité de son calcul est exponentielle par rapport au nombre d'objectifs.

4.6 Optimisation par modèles de substitution

La construction de modèles de substitution (voir le chapitre 3) dont l'évaluation est bien plus rapide que celle de la fonction qu'ils approchent permet de réaliser des opérations qui étaient jusqu'alors inaccessibles car bien trop coûteuses. L'optimisation fait partie de celles-ci. La minimisation d'une fonction demande en effet un certain nombre d'évaluations qui peut devenir important selon la complexité de cette fonction. Avec l'aide d'un modèle de substitution, le nombre d'appels n'est plus un problème car l'évaluation devient quasi instantanée (mais on n'a en contrepartie accès qu'à une approximation de la valeur réelle de la fonction). Les modèles de substitution permettent ainsi d'optimiser des fonctions complexes en un temps raisonnable.

Dans cette section, nous passons en revue différents travaux réalisés dans le cadre de l'optimisation par modèles de substitution. Pour plus de détails, [Jon01] et [WS07] proposent une vue d'ensemble du domaine et des méthodes afférentes. On peut distinguer d'une part les algorithmes d'optimisation qui utilisent des modèles de substitution locaux pour se déplacer sur la fonction coûteuse, et d'autre part les méthodes qui appliquent des algorithmes classiques d'optimisation sur un modèle global de la fonction.

4.6.1 Méthodes locales

Une première utilisation de modèles de substitution en optimisation concerne les méthodes dites locales. L'emploi de modèles locaux au sein d'une boucle d'optimisation a notamment été mis en œuvre dans les méthodes de « région de confiance ». Leur principe est de considérer une zone réduite de l'espace (la région de confiance) sur laquelle on réalise une optimisation locale. En fonction de la position de l'optimum local, on choisit ensuite de déplacer la région de confiance dans la direction de cet optimum, ou bien d'en modifier l'étendue. La région de confiance se stabilise ainsi au bout d'un certain temps au niveau d'un point optimal de la fonction (sous certaines hypothèses). Pour réaliser l'optimisation locale, il est possible d'utiliser un modèle de substitution défini sur la région de confiance. L'article [GE00] met ainsi en œuvre des modèles polynomiaux locaux dans sa stratégie d'optimisation par région de confiance, et [OKN03] un modèle de krigeage local pour optimiser le design d'une aile d'avion avec la même méthode. D'autres approches existent aussi. On peut par exemple citer les travaux de [RS04] où les objectifs sont approchés grâce à des approximations polynomiales ou des RBF locaux construits à partir des échantillons les plus proches du point à évaluer, dans une stratégie d'optimisation par algorithme évolutionnaire.

Les méthodes utilisant des modèles de substitution locaux peuvent être intéressantes dans le cadre d'une optimisation mono-objectif où la position de l'optimum correspond à une seule région de l'espace. Mais si on se place dans un contexte multiobjectifs où les solutions optimales sont multiples, leur utilisation devient plus complexe.

Les modèles locaux peuvent par contre être utilisés dans les cas où les contraintes du problème définissent des petites régions admissibles dans l'espace de recherche, éloignées les unes des autres. Il n'est alors pas forcément judicieux de construire un modèle de substitution global qui va tenter de modéliser inutilement des régions inadmissibles. De plus, si la fonction objectif est raisonnablement lisse dans une zone locale donnée, on peut employer des modèles plus simples et qui permettent d'obtenir facilement le ou les optima de cette région (comme dans le cas de modèles polynomiaux de faible ordre par exemple).

4.6.2 Modèles globaux

Dans cette étude, nous avons affaire à des problèmes d'optimisation multiobjectifs où les contraintes ne sont pas prépondérantes. Les zones admissibles de l'espace de recherche sont donc assez larges et les multiples optima y seront possiblement dispersés. Dans ce cadre, les méthodes locales peuvent se révéler plus coûteuses en termes de nombre d'évaluations des fonctions objectifs par rapport à l'établissement d'un modèle de substitution global. En effet, les modèles locaux nécessitent d'avoir un nombre suffisant d'échantillons d'apprentissage à disposition dans chacune des régions de l'espace étudiées. Quand ces zones deviennent trop nombreuses, il est préférable de modéliser l'ensemble de l'espace d'un seul tenant pour avoir une meilleure idée du comportement global des objectifs du problème.

Le principe de l'utilisation de modèles de substitution globaux est relativement simple : chaque objectif et chaque contrainte du problème sont remplacés par un modèle global construit a priori, et l'optimisation est ensuite réalisée directement sur ces modèles avec n'importe quelle technique classique d'optimisation. On obtient ainsi des solutions optimales approchées (aux erreurs de modélisation près).

Les modèles de substitution globaux sont utilisés dans de très nombreuses études. On les retrouve tout d'abord en optimisation mono-objectif : [Sak03] présente ainsi plusieurs optimisations par descente de gradient dans le domaine des structures avec l'aide de modèles de krigeage et de réseaux de neurones, et [OBLP96] réalise une optimisation d'aile d'avion grâce à un modèle de krigeage construit sur des données expérimentales. Il y a aussi de nombreux exemples en optimisation multiobjectifs : [MGSH07] présente l'optimisation multiobjectifs d'une turbine de fusée par algorithme génétique en se basant sur une approximation polynomiale des objectifs, et [WCSF01] réalise l'optimisation multiobjectifs d'un actionneur piézo-électrique avec des modèles polynomiaux ainsi que des modèles de krigeage, par recherche extensive des solutions sur une grille.

En optimisation multiobjectifs, on construit généralement un modèle de substitution par objectif (et contrainte) du problème. Certains modèles de substitution comme les réseaux de neurones peuvent aussi prendre en compte plusieurs sorties, ce qui permet de représenter l'ensemble des objectifs avec un seul modèle. Mais cela revient quelque part à considérer que les objectifs présentent des similarités pour pouvoir être modélisés par une structure unique. Nous nous sommes contentés ici de construire un modèle de substitution différent pour chacun des objectifs. On bénéficie alors d'un nombre plus important de degrés de liberté pour représenter chacun des objectifs que si on établissait un seul modèle général. D'autres stratégies originales n'utilisant qu'un unique modèle pour guider une optimisation multiobjectifs ont aussi été proposées (voir [LSS10] par exemple), mais nous ne nous y sommes pas intéressés ici.

Si les modèles de substitution sont largement utilisés pour mener à bien des optimisations de fonctions boîte noire coûteuses, les recherches s'orientent aujourd'hui vers une meilleure prise en compte de leur erreur d'approximation. En effet, lorsqu'on trouve une solution optimale grâce à un modèle de substitution, sa valeur prédite est nécessairement entachée d'une certaine erreur. Il faut donc évaluer ce point solution sur la fonction objectif initiale afin de valider son estimation. Si le modèle est assez précis, la valeur réelle ne sera pas très éloignée de la valeur prédite et la solution obtenue pourra donc être considérée comme optimale pour le problème. Mais bien souvent l'erreur du modèle est plus importante car on essaie toujours de modéliser les fonctions objectifs avec le minimum

d'échantillons d'apprentissage possible. Il y a alors un certain écart entre le modèle et la fonction de référence. Il est possible qu'une solution qui paraissait optimale sur le modèle de substitution se révèle de piètre performance sur la fonction initiale. L'erreur du modèle de substitution empêche donc parfois les algorithmes d'optimisation de converger vers les solutions optimales du problème réel.

Il est primordial de contrôler la qualité d'un modèle de substitution utilisé au sein d'un processus d'optimisation. Son erreur globale peut être estimée (voire améliorée) a priori pour fournir à l'optimiseur des modèles de qualité garantie, ou bien on peut aussi s'orienter vers des méthodes de construction adaptative de plans d'expériences dédiées à l'optimisation, afin d'améliorer le modèle dans les zones où se situent les optima. Ces méthodes sont présentées dans la partie suivante.

4.6.3 Plans d'expériences adaptatifs pour l'optimisation

Lorsque les modèles de substitution sont utilisés dans le cadre d'une optimisation, il est possible d'adapter spécifiquement leur erreur au problème à résoudre. On souhaite en effet que les solutions optimales soient correctement décrites par les modèles, alors que les autres zones de l'espace peuvent être approchées de manière plus grossière. Il est donc inutile d'évaluer la fonction de référence pour des échantillons situés dans des zones clairement non optimales. Le but de la construction de plans d'expériences adaptatifs pour l'optimisation consiste donc à cerner les zones où se situent possiblement les optima afin d'y ajouter de nouveaux échantillons d'apprentissage.

Le modèle de substitution va ainsi évoluer au fur et à mesure de l'optimisation : plutôt que de lancer a priori un grand nombre d'évaluations de la fonction de référence, on construit un premier modèle à partir d'un nombre plus restreint de points de calcul puis on rajoute des échantillons « intéressants » au cours de l'optimisation afin d'obtenir un modèle plus précis dans les zones qui semblent prometteuses. Le but n'est pas d'avoir un modèle le plus représentatif possible de la fonction (comme c'était le cas pour l'amélioration globale de modèles, voir la section 3.8.2), mais plutôt un modèle qui permette une bonne approximation de ses optima. Cette stratégie adaptative pour l'optimisation est schématisée sur la figure 4.14. Nous présentons dans la suite les principales méthodes d'amélioration qui ont été proposées en optimisation mono-objectif puis en optimisation multiobjectifs.

Critères d'amélioration pour l'optimisation mono-objectif

En optimisation mono-objectif, les critères visant à déterminer la position du prochain point à évaluer dans les stratégies de plans d'expériences adaptatifs reposent sur la notion d'amélioration I (pour « Improvement » en anglais, voir [EGN06] par exemple). On cherche en effet un point pour lequel la valeur de l'objectif sera inférieure au minimum actuellement obtenu (noté $f_{\min}$). L'amélioration apportée par un point $\boldsymbol{x}$ d'image $y = f(\boldsymbol{x})$ est exprimée comme :

$$\mathrm{I}(y) = \begin{cases} 0 & \text{si } y > f_{\min}, \\ f_{\min} - y & \text{sinon} \end{cases} = \max(0, f_{\min} - y). \tag{4.10}$$

La valeur de y n'est bien entendu pas connue directement en réalité mais approchée grâce à un modèle $\hat{f}$.

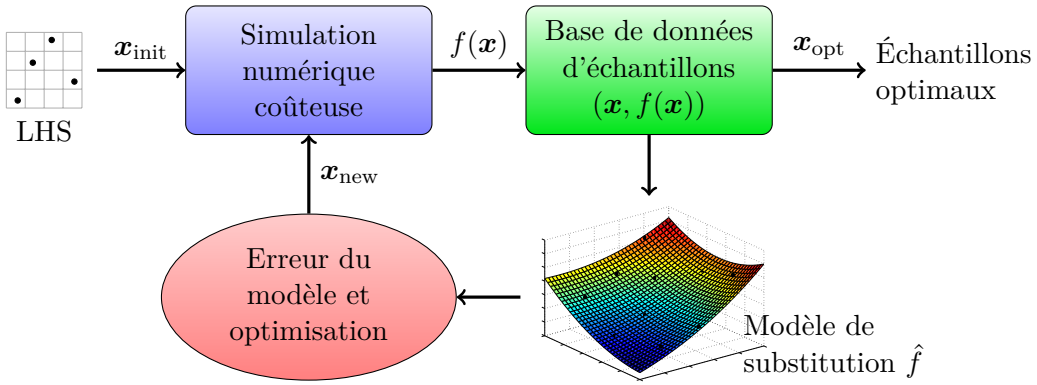


FIGURE 4.14 – Optimisation par construction adaptative d'un modèle de substitution : le modèle est établi à partir d'un ensemble initial d'échantillons calculés sur la fonction de référence (placés sur un LHS par exemple), puis de nouveaux échantillons sont évalués et ajoutés au modèle en fonction de son erreur et des résultats de l'optimisation.

La première idée qu'on peut avoir pour améliorer la connaissance du minimum d'une fonction est de rajouter au plan d'expériences le point qui maximise l'amélioration I (calculée grâce au modèle $\hat{f}$). Ce critère revient en fait à rechercher le point optimal du modèle de substitution. Comme le modèle indique la valeur la plus probable de ce point optimal, on parle de « Most Likely Improvement », noté MI :

$$\mathrm{MI}(\hat{f}) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmax}}\ \mathrm{I}(\hat{f}(\boldsymbol{x})) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmin}}\ \hat{f}(\boldsymbol{x}). \tag{4.11}$$

Le problème est que ce critère de choix ne tient pas compte de l'erreur du modèle, et il peut rester bloqué dans des minima locaux de la fonction objectif.

Puisqu'il est possible d'avoir une estimation de l'erreur du modèle de substitution, on peut la prendre en compte lors de la recherche du nouvel échantillon à évaluer. Afin de favoriser les points qui pourraient être optimaux du fait de cette erreur, on peut ainsi considérer la borne inférieure IC_{INF} de l'intervalle de confiance des prédictions à la place du modèle lui même. Le point présentant la meilleure possibilité d'amélioration sera alors le point qui possède la plus petite valeur de borne inférieure. Il s'agit du critère LB (pour « Lower Confidence Bound ») :

$$\mathrm{LB}(\hat{F}) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmax}}\ \mathrm{I}(\mathrm{IC}_{INF}(\hat{F}(\boldsymbol{x}))) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmin}}\ \mathrm{IC}_{INF}(\hat{F}(\boldsymbol{x})). \tag{4.12}$$

Ce critère est par exemple utilisé dans [EGO+02] au sein d'une optimisation par algorithme génétique sur un modèle de krigeage construit de manière adaptative.

Connaissant la distribution de l'erreur du modèle, on peut aussi définir une probabilité d'amélioration qui correspond à la probabilité qu'un point donné apporte une amélioration. Il s'agit du critère PI (pour « Probability of Improvement »), qui propose de sélectionner le point possédant la plus grande probabilité d'amélioration. Cela revient à choisir le point qui a le plus de chances d'être au-dessous de $f_{\min}$:

$$\mathrm{PI}(\hat{F}) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmax}}\ P(\mathrm{I}(\hat{F}(\boldsymbol{x})) > 0) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmax}}\ P(\hat{F}(\boldsymbol{x}) \leq f_{\min}). \tag{4.13}$$

Dans un cadre gaussien, cette probabilité peut-être exprimée en fonction de la moyenne et de la variance du modèle. Le critère devient alors :

$$\mathrm{PI}(\hat{F}) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmax}}\ \Phi\left(\frac{f_{\min} - \hat{f}(\boldsymbol{x})}{\mathrm{Var}(\hat{F}(\boldsymbol{x}))}\right), \tag{4.14}$$

avec Φ la fonction de répartition de la loi normale centrée réduite.

Le quatrième critère d'amélioration pour l'optimisation mono-objectifs que nous présentons ici est le plus répandu aujourd'hui. Il s'agit du critère d'amélioration espérée, ou « Expected Improvement » (EI). Au delà du calcul de la probabilité d'amélioration d'un point, on peut en effet prendre aussi en compte l'ampleur de cette amélioration. EI calcule ainsi l'espérance de l'amélioration apportée par un point :

$$\mathrm{EI}(\hat{F}) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\operatorname{argmax}}\ \mathrm{E}[\mathrm{I}(\hat{F}(\boldsymbol{x}))]. \tag{4.15}$$

L'algorithme EGO (« Efficient Global Optimization » [JSW98]) utilise ce critère pour réaliser une optimisation par modèle de krigeage. [SWJ98] introduit aussi le critère GEI (pour « Generalized Expected Improvement ») défini comme l'espérance d'une puissance de l'amélioration $\mathrm{E}[\mathrm{I}^g]$. De faibles valeurs de g rendent la recherche plus locale en favorisant les zones optimales, alors que des valeurs élevées élargissent la recherche vers les zones de forte incertitude (la valeur de g peut ainsi être réduite au fur et à mesure de l'optimisation sur le principe d'un recuit simulé).

Un critère différent pour la recherche de point améliorant est utilisé dans l'algorithme IAGO [VVW08]. Les critères précédents tentaient de déterminer la position du minimum de f afin de l'ajouter au plan d'expériences. Baptisé ECM (pour « Entropie Conditionnelle des Minimiseurs »), ce dernier critère vise quant à lui à trouver le point qui permettra d'améliorer la connaissance de la localisation du minimum de f. Bien que plus appropriée, cette méthode reste toutefois plus lourde que EGO car elle nécessite de simuler des réalisations de processus gaussiens pour l'évaluation du critère.

On a parfois la possibilité d'évaluer plusieurs échantillons en parallèle avec la fonction de référence. Dans ce cas, deux possibilités sont présentées dans [BKGB10] : les points peuvent être ajoutés un à un de manière « virtuelle » aux modèles de substitution (voir la section 3.8.3) pour que les points déjà sélectionnés soient pris en compte lors de la recherche d'un nouveau point, ou bien on peut utiliser différentes versions du critère de choix si celui-ci possède un paramètre variable. Les critère PI et EI dépendent par exemple de la valeur choisie pour le seuil $f_{\min}$, et on peut donc utiliser plusieurs valeurs différentes pour obtenir différents points améliorants. [SLK04] proposent aussi d'utiliser les optima locaux de EI pour sélectionner plusieurs points à évaluer en parallèle. Une version multipoints de EI est enfin définie dans [GLRC10]. Elle permet de rechercher simultanément la position de plusieurs points améliorant en prenant en compte leurs interactions (mais les calculs deviennent alors beaucoup plus lourds).

Critères d'amélioration pour l'optimisation multiobjectifs

Si l'utilisation de modèles de substitution en optimisation mono-objectif est relativement bien maîtrisée aujourd'hui, l'application des méthodes de construction adaptative de plans d'expériences à l'optimisation multiobjectifs est plus récente. A titre général, l'article [EGN06] détaille des critères d'amélioration en optimisation mono-objectif et multiobjectifs et les compare sur plusieurs cas, et [WEDP11] recense et étudie diverses adaptations

de l'algorithme EGO aux problèmes multiobjectifs. Plusieurs stratégies se dessinent : on peut d'une part se ramener à des problèmes mono-objectifs afin de réutiliser les critères présentés précédemment, ou bien on peut redéfinir ces critères pour traiter les problèmes multiobjectifs directement.

Une première possibilité est donc de conserver les critères définis pour l'optimisation mono-objectif et de les appliquer de manière adaptée au problème. [JO05] suggère par exemple de calculer un critère d'amélioration pour chaque objectif, puis de résoudre un problème multiobjectifs visant à améliorer l'ensemble de ces critères. Dans le cas du critère EI qui est employé dans cet article, le problème à résoudre devient donc :

$$\underset{\boldsymbol{x}\in\boldsymbol{X}}{\text{minimiser}}\ \{\mathrm{EI}(\hat{F}_1), \mathrm{EI}(\hat{F}_2), ..., \mathrm{EI}(\hat{F}_{n_f})\}. \tag{4.16}$$

On obtient ainsi un front de Pareto dans lequel des points peuvent être sélectionnés pour être évalués avec la fonction de référence. Toujours dans l'idée de se ramener à des problèmes d'optimisation mono-objectifs, l'algorithme ParEGO [Kno06] consiste à agréger les objectifs initiaux en une somme pondérée pour obtenir un unique objectif sur lequel est calculé le critère d'amélioration (EI dans le cas présent). Le front de Pareto est alors obtenu en faisant varier de manière aléatoire les paramètres de l'agrégation. L'avantage de ces méthodes est qu'elles sont relativement simples, mais par contre elles ne considèrent pas vraiment les spécificités des problèmes multiobjectifs. C'est pourquoi des critères d'améliorations adaptés à l'optimisation multiobjectifs ont aussi été proposés.

Les critères MI et LB se ramènent en optimisation mono-objectif à la minimisation de la valeur prédite par le modèle $\hat{f}$ ou bien de la borne inférieure IC_{INF} de l'intervalle de confiance de ses prédictions. Comme évoqué dans [EGN06], ces deux critères peuvent facilement être étendus au cas multiobjectifs. L'équivalent de MI peut simplement consister à choisir des points Pareto-optimaux sur les modèles de substitution pour les ajouter au plan d'expériences (mais comme auparavant cette stratégie n'est pas garantie de converger correctement car on ne tient pas compte de l'erreur des modèles). La considération des erreurs de modélisation nécessite d'étendre la notion d'intervalle de confiance. La réunion des intervalles de confiance des prédictions de chaque modèle forme en effet une région de confiance $\mathbf{RC} \subseteq \mathbb{R}^{n_f}$ définie comme le produit de ces intervalles :

$$\mathbf{RC}(\hat{\boldsymbol{F}}(\boldsymbol{x})) = \mathrm{IC}(\hat{F}_1(\boldsymbol{x})) \times \mathrm{IC}(\hat{F}_2(\boldsymbol{x})) \times ... \times \mathrm{IC}(\hat{F}_{n_f}(\boldsymbol{x})). \tag{4.17}$$

Un exemple de région de confiance est présenté sur la figure 4.15. L'intervalle de confiance à une dimension est donc devenu un hypercube à n_f dimensions. On note $\mathbf{RC}_{\boldsymbol{INF}} \in \mathbb{R}^{n_f}$ le point « minimal » de cet hypercube, qui correspond aux bornes inférieures de tous les intervalles de confiance :

$$\mathbf{RC}_{\boldsymbol{INF}}(\hat{\boldsymbol{F}}(\boldsymbol{x})) = (\mathrm{IC}_{INF}(\hat{F}_1(\boldsymbol{x})), \mathrm{IC}_{INF}(\hat{F}_2(\boldsymbol{x})), ..., \mathrm{IC}_{INF}(\hat{F}_{n_f}(\boldsymbol{x}))). \tag{4.18}$$

On peut définir de la même manière le point « maximal » $\mathbf{RC}_{\boldsymbol{SUP}}$. En optimisation mono-objectif, LB revient à sélectionner le point qui possède la plus petite borne inférieure pour l'intégrer au plan d'expérience. On va donc rechercher ici les échantillons dont le point minimal de la région de confiance $\mathbf{RC}_{\boldsymbol{INF}}$ est Pareto-optimal. Le critère LB en optimisation multiobjectifs consiste alors à résoudre le problème suivant :

$$\underset{\boldsymbol{x}\in\boldsymbol{X}}{\text{minimiser}}\ \mathbf{RC}_{\boldsymbol{INF}}(\hat{\boldsymbol{F}}(\boldsymbol{x})). \tag{4.19}$$

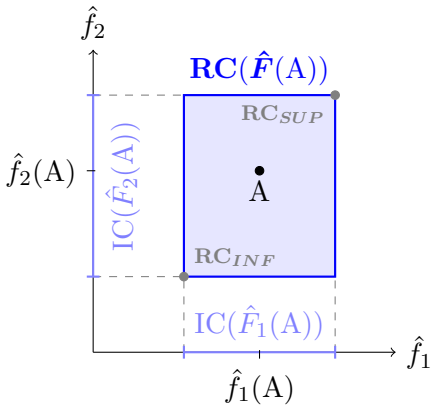


FIGURE 4.15 – Région de confiance **RC** (en bleu) autour d'un point A (point noir) définie à partir des intervalles de confiance IC sur les prédictions des modèles $\hat{f}_1$ et $\hat{f}_2$.

On obtient un front de Pareto constitué de points possiblement optimaux du fait de leur région de confiance, parmi lesquels on peut sélectionner ceux qui seront évalués avec la fonction de référence. Cette stratégie va favoriser d'une part les solutions les plus optimales, et d'autre part des solutions qui présentent un fort degré d'erreur (car leur point minimal $\mathbf{RC}_{\boldsymbol{INF}}$ aura alors plus de chances d'être optimal). Dans la même veine, [EN04] présentent différentes méthodes basées sur les régions de confiance pour la prise en compte de l'erreur des modèles dans l'optimisation multiobjectifs. Selon les cas, le point minimal, la valeur moyenne ou bien le point maximal de la région de confiance peut être considéré pour déterminer si une solution est meilleure qu'une autre.

Pour redéfinir les critères de recherche de nouveaux points, on peut aussi se baser sur le principe d'amélioration qui est le point de départ des critères en optimisation mono-objectif. La différence est qu'en optimisation multiobjectifs il n'y a plus un seul optimum mais possiblement plusieurs solutions de compromis : l'amélioration ne peut donc pas être définie par rapport à un unique seuil $f_{\min}$. Il faut prendre en considération l'ensemble du front de Pareto $\boldsymbol{P}_{\min}$ actuellement trouvé pour le problème. [Kea06] suggère par exemple que l'amélioration apportée par un point soit fonction de sa distance à la plus proche solution du front de Pareto :

$$\mathrm{I}(\boldsymbol{y}) = \begin{cases} 0 & \text{si } \boldsymbol{y} \text{ est dominé par } \boldsymbol{P}_{\min}, \\ \mathrm{dist}(\boldsymbol{y}, \boldsymbol{P}_{\min}) & \text{sinon.} \end{cases} \tag{4.20}$$

Dans les travaux de [HS07], l'amélioration d'un point est définie en fonction du nombre de points k du front de Pareto actuel qu'il domine. Les auteurs utilisent cette mesure pour définir une probabilité d'amélioration PI. On aura par exemple :

$$\mathrm{I}(\boldsymbol{y}) = \begin{cases} 1 & \text{si } \boldsymbol{y} \text{ domine au moins } k \text{ solutions de } \boldsymbol{P}_{\min}, \\ 0 & \text{sinon.} \end{cases} \tag{4.21}$$

L'amélioration peut enfin être caractérisée grâce à la mesure de l'hypervolume $\mathcal{H}$ (présentée dans la section 4.5.2). Un point qui domine des solutions du front de Pareto actuel (ou qui leur est équivalent) va en effet permettre d'augmenter l'hypervolume du front. On peut

donc évaluer l'amélioration apportée grâce à la différence entre l'hypervolume actuel et l'hypervolume obtenu une fois ce point ajouté :

$$\mathrm{I}(\boldsymbol{y}) = \max(0, \mathcal{H}(\boldsymbol{P}_{\min} \cup \{\boldsymbol{y}\}) - \mathcal{H}(\boldsymbol{P}_{\min})). \tag{4.22}$$

Cette mesure est par exemple utilisée dans l'article [EDK08]. Les définitions de l'amélioration I présentées ici permettent d'exprimer l'ensemble des critères pour les plans d'expériences adaptatifs de la même manière qu'en optimisation mono-objectif.

La construction de plans d'expériences adaptatifs permet ainsi de sélectionner les échantillons évalués sur les fonctions coûteuses de manière réfléchie et d'adapter les modèles de substitution au problème à traiter. Les problèmes pourront être résolus de manière plus précise ou à l'aide de moins d'échantillons que si les modèles avaient simplement été établis a priori sans aucune autre considération. Ces méthodes sont par contre plus complexes à mettre en œuvre et requièrent la possibilité de réaliser des évaluations des fonctions de référence au cours du processus d'optimisation.

Chapitre 5

Optimisation sous incertitude

Sommaire

5.1 Sources d'incertitude 93
5.2 Analyse de sensibilité 94
5.3 Modélisation de l'incertitude 95
5.4 Mesures de robustesse 97
5.4.1 Mesures basées sur les espérances 97
5.4.2 Mesures basées sur les quantiles 99
5.4.3 En conclusion sur les mesures de robustesse 103
5.5 Formulation d'un problème d'optimisation robuste 104
5.5.1 Remplacement d'objectifs 105
5.5.2 Ajout de nouveaux objectifs 106
5.5.3 Ajout de nouvelles contraintes 107
5.6 Problèmes de test pour l'optimisation robuste 108
5.7 Résolution d'un problème d'optimisation robuste 112
5.8 Calcul des mesures de robustesse 113
5.8.1 Monte-Carlo 113
5.8.2 Chaos polynomial 114
5.8.3 Calcul analytique 116
5.8.4 Autres méthodes de calcul des mesures de robustesse 119
5.8.5 Erreur sur l'estimation des mesures de robustesse 122
5.9 Plans d'expériences adaptatifs pour l'optimisation robuste 124

Les solutions optimales trouvées lors de la résolution d'un problème d'optimisation déterministe classique ne sont peut-être pas les meilleures solutions à utiliser en pratique. En effet, elles ne tiennent pas compte des incertitudes qui affectent le système réel. La définition du problème d'optimisation avec des fonctions objectifs et des contraintes représente déjà en elle-même un modèle, une approximation de la réalité. Le résultat obtenu peut donc être différent de l'optimum réel du fait de ces erreurs de modélisation. Il est de plus impossible de réaliser exactement le système décrit par une solution car la précision de fabrication est toujours limitée. Les systèmes conçus avec une approche déterministe

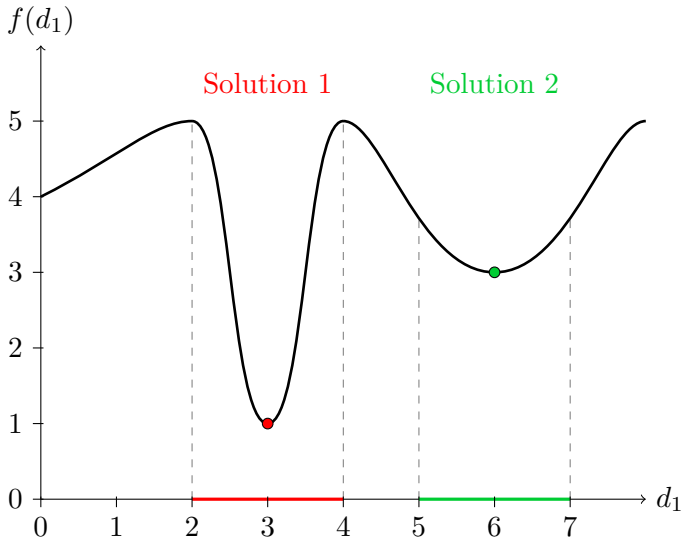


FIGURE 5.1 – Minimisation d'une fonction objectif f avec une incertitude sur la variable de décision d_1 (représentée par les intervalles en gras). La solution 2 est moins performante mais plus robuste que la solution 1.

peuvent être très sensibles à de petites variations sur certains de leurs paramètres qui vont fortement dégrader leur performance. La figure 5.1 présente un exemple de fonction objectif f dépendant d'un unique paramètre d_1. Si des incertitudes viennent perturber la valeur nominale de ce paramètre autour des deux solutions indiquées, on observe que la performance de la solution 1 est grandement altérée alors que l'impact sur la solution 2 est beaucoup moins notable. Il pourra ainsi être judicieux de choisir cette seconde solution plutôt que la première même si sa performance nominale est moindre. C'est donc avec un intérêt tout particulier que de nombreuses études se penchent aujourd'hui sur la gestion des incertitudes en optimisation.

Dans les problèmes d'optimisation déterministe, les incertitudes ne sont pas du tout considérées. Il est néanmoins possible de quantifier a posteriori la robustesse des solutions optimales obtenues, c'est-à-dire d'évaluer leur degré de sensibilité aux paramètres incertains du problème. Cela permet de comparer les différentes solutions afin de sélectionner celle qui convient le mieux dans ce contexte incertain. L'étape suivante est d'intégrer directement la notion de robustesse au problème d'optimisation lui-même afin de trouver les solutions les plus robustes possibles (voir la figure 5.2). Il est en effet probable que certaines solutions robustes ne fassent pas partie des solutions optimales obtenues lors d'une optimisation déterministe classique.

Nous présentons tout d'abord dans ce chapitre les différentes sources d'incertitude qu'on peut rencontrer dans un problème d'optimisation. Une fois identifiées, les incertitudes doivent être modélisées et quantifiées afin de pouvoir évaluer la robustesse des solutions. Nous détaillons ainsi diverses mesures de robustesse utilisées dans ce but. Le choix d'une mesure particulière permet ensuite de formuler un problème d'optimisation robuste prenant en compte les incertitudes définies auparavant, et dont la résolution fournit des solutions robustes. Nous nous attardons aussi sur les méthodes de calcul des mesures de robustesse, en s'appuyant notamment sur des modèles de substitution.

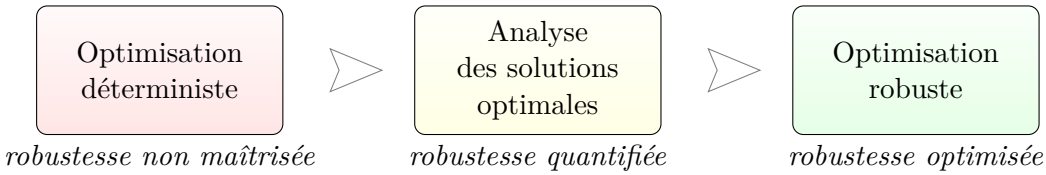


FIGURE 5.2 – Vers l'optimisation de la robustesse des solutions.

5.1 Sources d'incertitude

Les incertitudes qui peuvent affecter un problème d'optimisation sont d'origines multiples. Un article de revue sur l'optimisation robuste [BS07] les classe en quatre sources principales, dont voici les trois premières :

Classe A Cette classe rassemble les incertitudes dues aux conditions environnementales, qui peuvent être représentées comme des paramètres non contrôlables $\boldsymbol{e}$ du modèle. On peut ainsi modéliser chaque fonction objectif du système par $f(\boldsymbol{d}, \boldsymbol{e})$, avec $\boldsymbol{d}$ le vecteur des variables de décision et $\boldsymbol{e}$ les paramètres environnementaux. Les paramètres $\boldsymbol{e}$ peuvent varier d'une évaluation de la fonction à l'autre, donnant des résultats différents pour un même jeu de variables $\boldsymbol{d}$.

Classe B Classe des imprécisions sur les variables de décision. Il est impossible de fabriquer un système réel avec une précision infinie. Généralement, les systèmes sont construits avec une certaine tolérance sur les variables de décision $\boldsymbol{d}$. On considère donc $f(\boldsymbol{d} + \boldsymbol{\delta}, \boldsymbol{e})$, avec $\boldsymbol{\delta}$ la perturbation due à la fabrication. L'incertitude de fabrication peut parfois être aussi exprimée sous la forme d'un pourcentage de la valeur de $\boldsymbol{d}$: $f(\delta\boldsymbol{d}, \boldsymbol{e})$.

Classe C Cette classe représente les incertitudes de mesure sur la sortie du système. Il est en effet impossible de mesurer avec exactitude les valeurs de sortie et la performance du système. On n'a accès qu'à des mesures de $f(\boldsymbol{d}+\boldsymbol{\delta}, \boldsymbol{e})+\xi$, avec ξ l'erreur de mesure. Cette classe d'incertitudes inclut aussi bien les erreurs de mesure que les différentes approximations faites avec l'utilisation d'un modèle numérique pour représenter le système physique réel. Dans la pratique, on représente généralement ces incertitudes de manière approchée par un bruit blanc.

La quatrième et dernière classe, la classe D, concerne les incertitudes de faisabilité. Pour que le système soit réalisable, il est nécessaire de s'assurer que les contraintes du problème sont satisfaites. De même qu'il est difficile d'évaluer précisément les fonctions objectifs $\boldsymbol{f}$ comme on l'a vu dans les classes précédentes, on trouve des incertitudes sur les contraintes $\boldsymbol{g}$ et $\boldsymbol{h}$ qui limitent l'espace des paramètres. Une contrainte peut être vue sous la forme $g(\boldsymbol{d} + \boldsymbol{\delta}', \boldsymbol{e}) + \xi' \leq 0$ (de même pour les contraintes d'égalité $\boldsymbol{h}$). On aura donc une incertitude sur le fait que la solution qu'on est en train d'examiner appartient bien à l'espace des solutions admissibles. S'il est valable de distinguer les études s'intéressant aux impacts des incertitudes sur les objectifs de celles qui s'intéressent aux contraintes, cette quatrième classe d'incertitude ne semble pas vraiment justifiée. Il ne s'agit pas vraiment d'une quatrième source d'incertitude mais plutôt de l'effet des trois sources précédentes sur les contraintes du problème en lieu et place des objectifs.

Les entrées et sorties du système à optimiser sont schématisées sur la figure 5.3. Nous distinguerons donc ici les incertitudes provenant des paramètres environnementaux

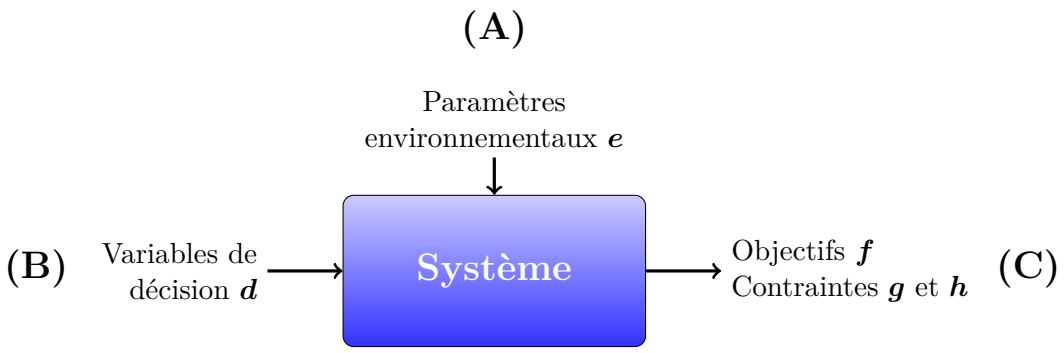


FIGURE 5.3 – Les entrées et sorties d'un système à optimiser peuvent être soumises à des incertitudes de différentes classes (A, B et C).

(classe A), des variables de décision (classe B) et des incertitudes de mesure (classe C). S'il est vrai que les variables de décision $\boldsymbol{d}$ et les paramètres environnementaux $\boldsymbol{e}$ jouent un rôle différent dans l'optimisation du système, on peut aussi les rassembler sous l'écriture $\boldsymbol{x} = (\boldsymbol{d}, \boldsymbol{e})$ afin de représenter d'un seul tenant les variables d'entrée du système et l'incertitude qui peut leur être associée. L'incertitude peut ainsi être présente sur les entrées $\boldsymbol{x}$ du système (classes A et B) ou bien sur la mesure de ses sorties (classe C). Lorsque cela sera utile, nous pourrons toujours distinguer les variables de décision dont la valeur peut évoluer au cours de l'optimisation des paramètres environnementaux dont la valeur est fixée (et ne pourra être modifiée que par des incertitudes).

5.2 Analyse de sensibilité

Un des premiers moyens de prendre en compte les incertitudes de classe A et B dans les problèmes d'optimisation est de réaliser une analyse de sensibilité sur les solutions optimales obtenues par des méthodes classiques d'optimisation déterministe. L'analyse de sensibilité [SRA+08] permet d'évaluer la sensibilité d'un point vis-à-vis de certains paramètres par étude de la dérivée des fonctions objectifs par rapport à ces paramètres. Elle détermine ainsi l'influence de chacun des paramètres d'entrée sur les variations des sorties du système. Il s'agit là d'un premier pas dans la prise en compte des incertitudes. L'analyse de sensibilité ne vise pas à quantifier la robustesse des solutions, mais plutôt à analyser les effets des incertitudes en mettant en évidence les liens entre les entrées et les sorties du système. À titre d'exemple, [SGCD11] réalise ainsi une analyse de sensibilité basée sur des modèles de substitution dans le contexte de la mécanique des fluides.

L'analyse de sensibilité permet d'analyser a posteriori la stabilité des solutions optimales, ainsi que de déterminer les variables d'entrées dont l'incertitude aura une grande influence sur les objectifs. Cette analyse n'est pas basée sur une quantification des incertitudes présentes dans le problème : on considère simplement que certaines variables sont incertaines et vont donc pouvoir varier de manière incontrôlée. Mais il peut aussi être intéressant de prendre en compte des informations supplémentaires sur ces incertitudes si elles sont disponibles. En effet, une solution sensible aux variations d'un paramètre incertain donné ne verra pas sa performance se dégrader beaucoup si ce paramètre ne varie que très peu. Afin de mieux gérer les incertitudes au delà de l'identification de leurs sources, il est nécessaire de les modéliser pour pouvoir quantifier leur impact sur le système à optimiser.

5.3 Modélisation de l'incertitude

Pour quantifier la robustesse des solutions d'un problème, il faut dans un premier temps quantifier et modéliser l'incertitude présente sur les différents paramètres d'entrée. Les incertitudes peuvent être modélisées des trois manières suivantes (de la plus spécifique à la plus générale) :

Modélisation déterministe On définit l'intervalle de variation d'un paramètre (la probabilité de répartition sur cet intervalle est inconnue). Par exemple, on aura : $x_1 \in [-1.5, 3.2]$.

Modélisation probabiliste L'incertitude est modélisée par une distribution de probabilité (le plus souvent une loi normale ou bien uniforme). Par exemple : $x_2 \sim \mathcal{N}(0, 1.2)$.

Modélisation possibiliste L'incertitude est décrite à l'aide de la logique floue (voir [DLP+08]). On peut ainsi définir par exemple un intervalle de variation sur lequel la distribution de probabilité n'est pas précisée mais dont on connaît la moyenne.

La représentation possibiliste permet de spécifier des incertitudes probabilistes, qui elles mêmes peuvent aussi exprimer des incertitudes décrites de façon déterministe. La représentation possibiliste est donc la plus générale mais aussi la plus complexe à manipuler, alors que la représentation déterministe est la plus simple mais aussi la plus limitée.

Les incertitudes peuvent être données de manière absolue (par exemple la longueur d'une pièce est précise au millimètre près) ou bien de manière relative (par exemple il y a une erreur de 2% sur la longueur de la pièce). Dans ce dernier cas, l'étendue de l'incertitude variera en fonction du point considéré.

Les différentes descriptions de l'incertitude se retrouvent dans la littérature. [Soa08] utilise ainsi une description par intervalles pour réaliser des optimisations robustes, et [VH99] effectuent leur optimisation avec une description possibiliste des incertitudes. Les études qui utilisent des probabilités sont nombreuses. C'est ce sur quoi nous allons nous concentrer ici. Faisant un compromis entre simplicité et représentabilité, nous considèrerons en effet des incertitudes données sous la forme de distributions de probabilité.

Pour représenter ces incertitudes, nous ajoutons un vecteur de variables aléatoires $\boldsymbol{\chi}_x$ aux paramètres d'entrée $\boldsymbol{x}$. La densité $p_{\chi_{x_i}}$ de chacune de ces variables aléatoires sera définie en fonction de l'incertitude sur le paramètre d'entrée x_i. Si certaines entrées ne sont pas incertaines, les variables χ_{x_i} correspondantes seront des constantes nulles. Nous considèrerons de plus que ces différentes incertitudes sont indépendantes (si cela n'est pas le cas, il est parfois possible de reformuler le problème pour satisfaire cette hypothèse). La densité de probabilité multivariée des entrées du système sera donc égale au produit des densités de chaque variable :

$$p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) = \prod_{i=1}^{n_x} p_{x_i+\chi_{x_i}}(z_i). \tag{5.1}$$

Les distributions de probabilité des incertitudes définies sur les entrées engendrent des distributions de probabilité sur les sorties. Ces distributions de sortie sont inconnues a priori et dépendent du système qui calcule ces sorties (voir l'illustration de la figure 5.4).

L'exemple de la figure 5.5 présente une fonction objectif f à une dimension. Le minimum classique est le point 1. On suppose qu'il y a une incertitude gaussienne sur l'unique variable d'entrée x_1, qui est représentée au niveau des trois points étudiés. Les distributions résultantes sur la valeur de f pour chacun de ces points sont tracées sur la droite

FIGURE 5.4 – Les incertitudes définies sur les entrées du système engendrent des incertitudes sur ses sorties.

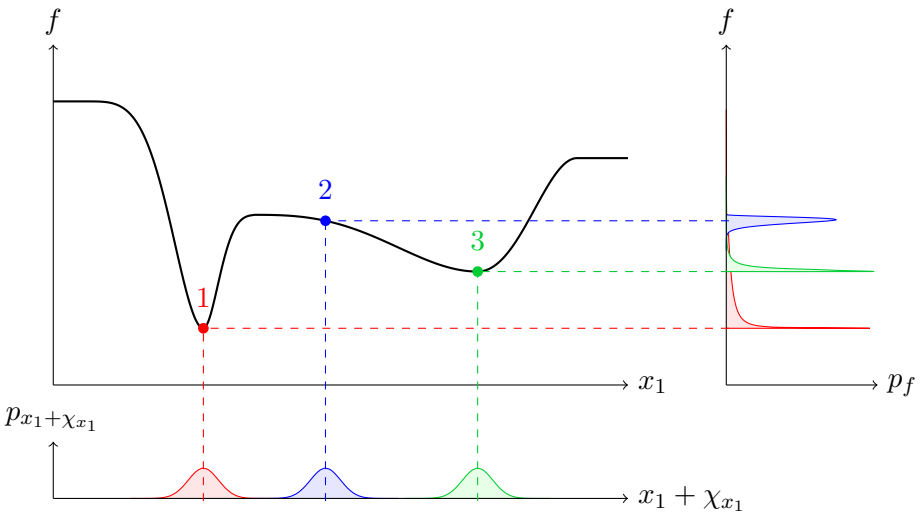


FIGURE 5.5 – Trois exemples de densités de probabilité obtenues en sortie à partir d'une incertitude gaussienne sur x_1.

(elles ont été renversées pour conserver le même axe des ordonnées : plus on va vers la droite et plus la valeur correspondante de l'objectif f est probable). On observe que les valeurs les plus probables correspondent aux centres des gaussiennes définies sur x_1. On peut aussi remarquer que l'optimum classique (le point 1) possède une distribution très dispersée, ce qui n'est pas forcément souhaitable. La fonction f varie en effet de façon assez importante dans le voisinage de ce point, alors qu'on peut trouver d'autres points où ses variations sont plus réduites. Le point 3 semble ainsi être une solution assez robuste pour ce problème. Cette fonction exemple nous permettra d'illustrer les différentes mesures de robustesse présentées par la suite.

La modélisation des incertitudes d'un problème doit toujours se faire en fonction des informations qu'on possède. Lorsque la seule donnée accessible est un intervalle de variation, le fait de supposer une distribution uniforme sur cet intervalle peut se révéler être un mauvais choix selon la distribution réelle de l'incertitude (même si la distribution uniforme reste la distribution la plus adaptée pour représenter ce type d'information). [Che00] compare ainsi les descriptions probabilistes et possibilistes, et conclut qu'une description probabiliste ne doit être utilisée que si suffisamment d'information est disponible. Supposer une distribution pour une incertitude qu'on ne connaît pas peut mener à des résultats erronés. Il vaut mieux dans ce cas utiliser par exemple une description possibiliste qui permet d'exprimer l'absence d'information. Les travaux de [LSN04] s'intéressent

aussi à ce problème de l'influence de la modélisation des incertitudes. Ils proposent de définir une notion de robustesse non pas par rapport aux incertitudes elles-mêmes mais par rapport à l'incertitude qu'il peut y avoir sur leur modélisation. En supposant que la distribution p_{χ_x} de l'incertitude sur $\boldsymbol{x}$ appartient à une famille $\mathcal{G}$ de distributions, ils définissent la $\mathcal{G}$-robustesse et la Π-robustesse comme étant respectivement le maximum et la moyenne de l'espérance des objectifs en considérant tous les cas possibles de distributions dans $\mathcal{G}$ (et en se donnant une densité de probabilité Π sur $\mathcal{G}$ pour le calcul de la moyenne). Ces deux formulations nécessitent toutefois des calculs assez lourds. L'incertitude sur la quantification des incertitudes pourrait plus naturellement être traitée là encore par une modélisation possibiliste. Nous supposerons dans cette étude que les distributions affectées aux variables incertaines sont connues avec précision.

5.4 Mesures de robustesse

Comme nous venons de le voir, la modélisation des incertitudes présentes en entrée d'un problème par des distributions de probabilité sur $\boldsymbol{x}$ définit des distributions de probabilité résultantes sur les sorties $\boldsymbol{f}(\boldsymbol{x})$ (ainsi que sur les contraintes de manière tout à fait similaire). Afin de départager différentes solutions possibles pour déterminer laquelle est la plus robuste, on peut comparer ces distributions de sortie. On cherche donc à quantifier la robustesse d'une solution $\boldsymbol{x}$ relativement à un objectif f donné en se basant sur sa distribution d'incertitude de sortie. On introduit une mesure de la robustesse $\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x))$ du point $\boldsymbol{x}$ doté d'une incertitude $\boldsymbol{\chi}_x$ par rapport à l'objectif f. Les mesures de robustesse visent à caractériser la distribution de $f(\boldsymbol{x}+\boldsymbol{\chi}_x)$ en calculant certains de ses paramètres. Grâce à ces mesures, on sera à même de quantifier la robustesse de différentes solutions, ce qui permettra de déterminer laquelle est optimale.

De nombreuses formulations de la robustesse ont été proposées (voir [BS07]). Les principales sont détaillées ci-dessous. Toutes les mesures peuvent être utilisées de manière équivalente, tant que leur emploi se justifie dans le cadre d'un problème donné. Le choix d'une mesure de robustesse particulière se fera en fonction des propriétés qu'on souhaite obtenir pour les solutions, ainsi que de la modélisation des incertitudes des paramètres. Nous parlerons ici principalement des mesures dédiées aux descriptions probabilistes, mais certaines s'adaptent directement aussi aux descriptions par intervalles. Nous n'aborderons pas le cas des mesures possibilistes.

On distingue notamment les mesures de robustesse basées sur les espérances (avec par exemple la mesure de l'espérance en elle-même ou de la variance), qui permettent d'évaluer la valeur moyenne ou l'amplitude des variations de la fonction dans le voisinage de la solution étudiée, ainsi que les mesures basées sur les quantiles qui permettent de garantir une certaine performance dans le voisinage (voir la figure 5.6).

5.4.1 Mesures basées sur les espérances

Le premier type de mesures de robustesse qu'on peut utiliser est celui des mesures basées sur le calcul d'espérances de fonctions de la distribution de sortie. Les deux premiers moments de cette distribution (l'espérance et la variance) sont les mesures les plus répandues.

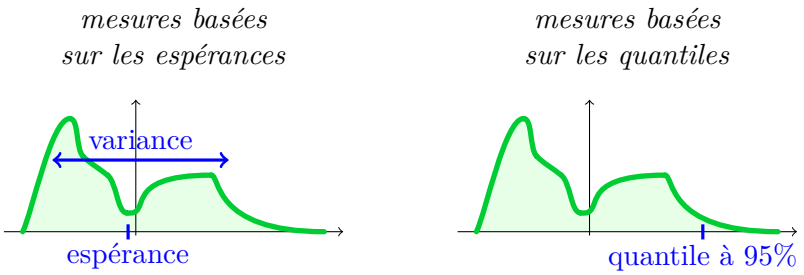


FIGURE 5.6 – On distingue deux grands types de mesures de robustesse sur la distribution de l'incertitude en sortie : d'un côté les mesures basées sur les espérances, et de l'autre celles basées sur les quantiles.

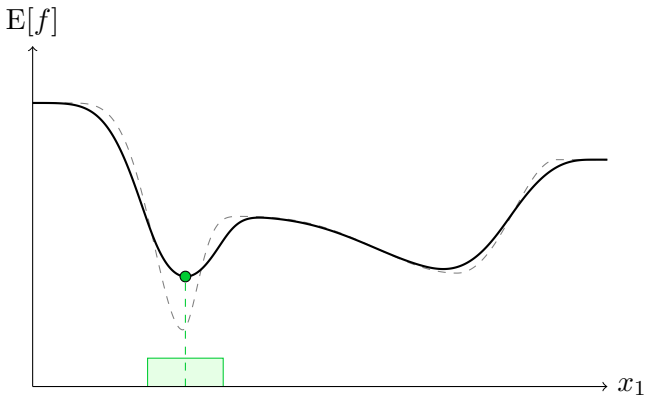


FIGURE 5.7 – Espérance (trait noir) de f (en pointillés) pour une incertitude uniforme sur x_1. Le point optimal ainsi que son incertitude sont représentés en vert.

Mesure d'espérance

La mesure de l'espérance considère la valeur moyenne de l'objectif dans le voisinage du point $\boldsymbol{x}$ étudié. La robustesse s'écrit alors comme l'espérance de f [Tro97] :

$$\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \mathrm{E}[f(\boldsymbol{x}+\boldsymbol{\chi}_x)] = \int_{\mathbb{R}^{n_x}} f(\boldsymbol{x}+\boldsymbol{z}) p_{\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z}, \tag{5.2}$$

avec $p_{\boldsymbol{\chi}_x}$ la distribution de probabilité de $\boldsymbol{\chi}_x$. La mesure de l'espérance réalise en quelque sorte un « lissage » de la fonction objectif initiale. Les vallées trop abruptes de f ne vont ainsi plus apparaître et ne seront donc pas sélectionnées comme des zones optimales robustes.

La figure 5.7 montre l'espérance de la fonction exemple présentée plus haut. On voit ici que l'optimum robuste (la solution qui possède l'espérance la plus faible) se situe dans la même zone que l'optimum déterministe classique, même s'il est légèrement décalé par rapport à celui-ci. Cela s'explique par la forme de la fonction f qui possède des valeurs moins importantes sur la partie droite de ce point par rapport à la partie gauche. La valeur moyenne sera donc légèrement inférieure à droite, d'où ce décalage.

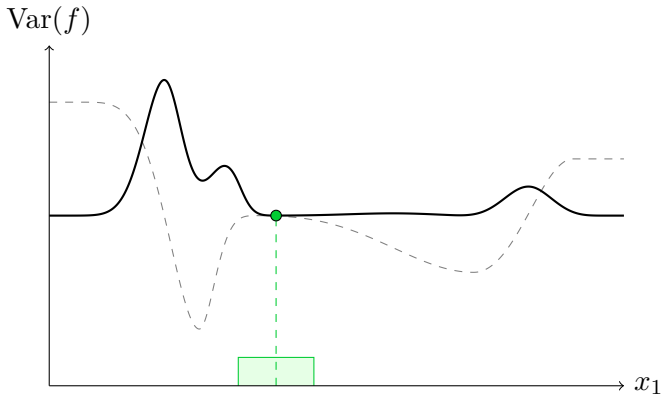


FIGURE 5.8 – Variance (trait noir) de f (en pointillés) pour une incertitude uniforme sur x_1. Le point optimal ainsi que son incertitude sont représentés en vert.

Mesures de variance

Pour quantifier la robustesse d'une solution, on peut aussi estimer la dispersion de la distribution de sortie en calculant par exemple la variance de la fonction objectif dans le voisinage du point considéré [PBJ06] :

$$\begin{aligned}\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) &= \mathrm{Var}(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) \\ &= \mathrm{E}[f^2(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \mathrm{E}^2[f(\boldsymbol{x}+\boldsymbol{\chi}_x)] \\ &= \mathrm{E}\left[(f(\boldsymbol{x}+\boldsymbol{\chi}_x) - \mathrm{E}[f(\boldsymbol{x}+\boldsymbol{\chi}_x)])^2\right].\end{aligned} \tag{5.3}$$

La mesure de variance va tout simplement permettre de quantifier l'importance des variations de la fonction objectif dans le voisinage des solutions considérées. Plus la variance est élevée, plus il y aura de chances que la performance de la solution se dégrade fortement du fait des incertitudes.

La figure 5.8 présente la variance de la fonction exemple. Si on omet les minima situés sur les bords gauche et droit, l'optimum robuste se trouve maintenant au niveau d'un maximum en optimisation déterministe classique. Il s'agit du point au niveau duquel la fonction objectif est la plus stable.

On peut utiliser de la même manière le moment d'ordre 2 de la distribution de sortie :

$$\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \mathrm{E}[f^2(\boldsymbol{x}+\boldsymbol{\chi}_x)], \tag{5.4}$$

plutôt que la variance (qui correspond quant à elle au moment centré d'ordre 2). D'autres mesures similaires de dispersion basées sur une espérance ont aussi été proposées, comme par exemple [Das00] :

$$\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \mathrm{E}[(f(\boldsymbol{x}+\boldsymbol{\chi}_x) - f(\boldsymbol{x}))^2]. \tag{5.5}$$

5.4.2 Mesures basées sur les quantiles

Au delà des mesures basées sur les espérances qui décrivent le comportement général de l'objectif dans le voisinage de la solution considérée, on peut souhaiter garantir un certain niveau de performance pour les solutions robustes. On utilisera alors une mesure basée sur les quantiles. La plus utilisée est la mesure du pire cas, mais on peut aussi employer des mesures de quantile moins conservatives.

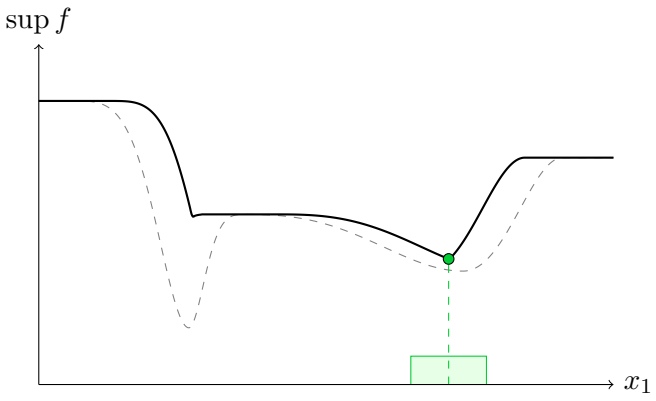


FIGURE 5.9 – Pire cas (trait noir) de f (en pointillés) pour une incertitude uniforme sur x_1. Le point optimal ainsi que son incertitude sont représentés en vert.

Pire cas

La mesure du pire cas considère la plus mauvaise valeur de la fonction objectif dans le voisinage du point $\boldsymbol{x}$. Cette approche est parfois appelée « approche pessimiste » ou « approche minimax ». La robustesse est évaluée par la fonction [Tro97] :

$$\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \sup\ \operatorname{supp}(p_{f(\boldsymbol{x}+\boldsymbol{\chi}_x)}) = \sup_{\boldsymbol{z}\in\operatorname{supp}(p_{\boldsymbol{x}+\boldsymbol{\chi}_x})} f(\boldsymbol{z}), \tag{5.6}$$

avec $\operatorname{supp}(p_w)$ le support de la densité de probabilité de w. Il faut bien sûr que la distribution de l'incertitude en entrée ne soit pas de support infini pour que la mesure ait un sens. La minimisation du pire cas permet d'obtenir une solution robuste pour laquelle la valeur maximale de la fonction dans le voisinage est plus petite que dans n'importe quel autre voisinage. On garantit donc que tout point du voisinage de la solution robuste aura au pire une performance égale à ce maximum. Cette mesure est souvent utilisée dans le cadre d'incertitudes décrites par de simples intervalles de variation (modélisation déterministe).

L'utilisation de la fonction « sup » va éventuellement entraîner des discontinuités dans la dérivée de cette mesure de robustesse, ce qui peut poser problème lors de l'optimisation de la robustesse selon l'algorithme utilisé. Il existe toutefois des algorithmes d'optimisation qui ne nécessitent pas d'information sur la dérivée ou bien qui peuvent s'appliquer à des fonctions dont la dérivée n'est pas totalement définie.

La figure 5.9 présente le tracé du pire cas sur notre fonction exemple. On voit que l'optimum robuste se situe désormais près d'un minimum local de la fonction, qui présente de meilleures caractéristiques dans son voisinage que le minimum classique. L'optimum robuste n'est cependant pas localisé à l'endroit exact du minimum local, du fait de l'asymétrie de la fonction dans le voisinage de ce point. Le minimum robuste préfère s'éloigner de la partie droite de la fonction qui présente un plus fort gradient.

Quantiles

La mesure du pire cas décrite ci-dessus correspond en fait à un quantile. En effet, en recherchant la pire valeur de la fonction dans la zone de variation des paramètres incertains on garantit que 100% des points de cette zone seront plus performants. On vient donc de calculer le quantile d'ordre 1 de la distribution de sortie.

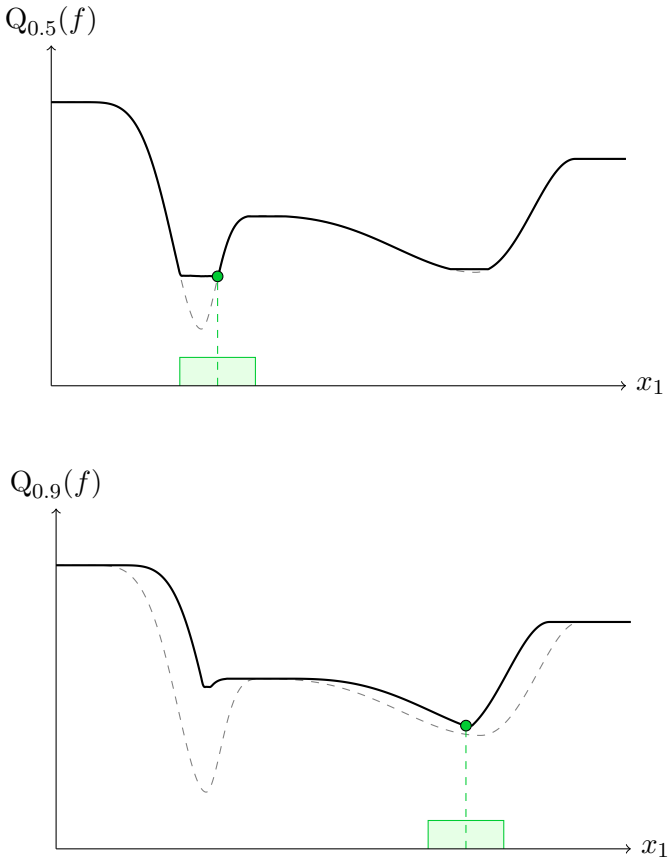


FIGURE 5.10 – Médiane (trait noir en haut) et quantile d'ordre 0.9 (trait noir en bas) de f (en pointillés) pour une incertitude uniforme sur x_1. Les points optimaux ainsi que leur incertitude sont représentés en vert.

Le pire cas est une mesure de robustesse très conservative : il suffit qu'un seul et unique point du voisinage soit très mauvais pour que la valeur du pire cas soit très mauvaise elle aussi. On peut cependant considérer des mesures de robustesse un peu plus souples en prenant un ordre de quantile inférieur. On garantira ainsi par exemple que 90% des points du voisinage seront plus performants que la valeur annoncée. La mesure du quantile d'ordre $k \in [0, 1]$ (que nous noterons Q_k) s'écrit :

$$\rho(f(\boldsymbol{x} + \boldsymbol{\chi}_x)) = Q_k(f(\boldsymbol{x} + \boldsymbol{\chi}_x)) = \inf\{q \in \mathbb{R} : P(f(\boldsymbol{x} + \boldsymbol{\chi}_x) \leq q) \geq k\}. \tag{5.7}$$

En prenant $k = 1$, on retrouve la mesure du pire des cas vue précédemment, et pour $k = 0.5$ on a la valeur médiane de l'objectif.

La figure 5.10 présente justement la valeur médiane de la fonction objectif évoquée précédemment. La médiane agit comme si elle tronquait les zones trop escarpées et donc non robustes de la fonction. On y trouve aussi le tracé du quantile à 90% de f. Le point optimal est le même que pour le pire cas, mais on remarque ici que la zone de l'optimum déterministe devient légèrement plus intéressante car on considère un quantile moins restrictif.

On voit que la mesure du quantile permet d'avoir un certain contrôle sur le « degré » de robustesse qu'on souhaite obtenir par l'intermédiaire du paramètre k. Par contre, les

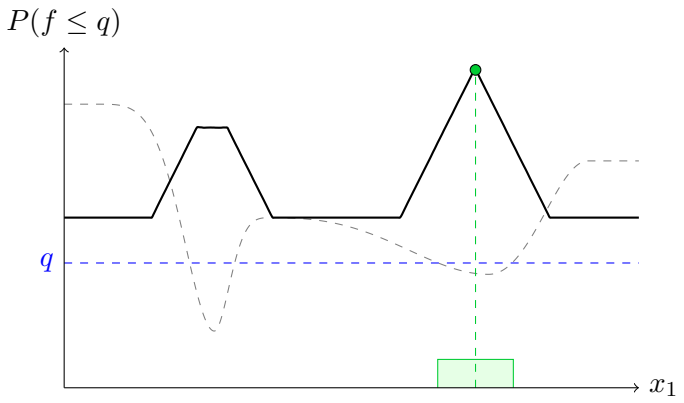


FIGURE 5.11 – Seuil probabiliste (trait noir) de f (en pointillés) pour une incertitude uniforme sur x_1, avec un seuil q représenté en bleu. Le point optimal ainsi que son incertitude sont représentés en vert.

fonctions quantiles présentent certaines caractéristiques particulières comme des zones plates ou des discontinuités dans la dérivée qui les rendent possiblement plus complexes à optimiser. En comparaison, les mesures basées sur les espérances sont beaucoup plus lisses.

Les mesures de quantiles sont le plus souvent employées en optimisation fiabiliste (c'est-à-dire lorsqu'on considère la robustesse par rapport aux contraintes du problème) car elles permettent de garantir une certaine probabilité d'amissibilité des solutions, et donc un certain niveau de faisabilité du système.

La définition du quantile de l'équation (5.7) fait apparaître d'un côté l'ordre k du quantile et de l'autre la valeur q du quantile lui-même. Dans la mesure précédente, nous avons spécifié l'ordre k mais on peut aussi définir une mesure de « seuil probabiliste » en recherchant la probabilité k qui correspond à un seuil q fixé. Ce seuil q représente une valeur de l'objectif qu'on aimerait atteindre, et on va chercher à maximiser la probabilité que les points dans le voisinage soient inférieurs à ce seuil [BS07] :

$$\rho(f(\boldsymbol{x} + \boldsymbol{\chi}_x)) = P(f(\boldsymbol{x} + \boldsymbol{\chi}_x) \leq q). \tag{5.8}$$

On peut, si on le souhaite, prendre l'opposé de cette valeur pour rester dans un contexte de minimisation.

La mesure du seuil probabiliste est présentée sur la figure 5.11. Cette mesure dépend bien évidemment de la valeur de seuil q qu'on a fixée. Si q est trop grand, la probabilité d'être inférieur sera égale à 1 dans tout l'espace et on ne pourra distinguer aucune solution. De même, si q est choisi plus petit que le minimum de la fonction, la probabilité sera toujours de 0. Il faut donc une certaine connaissance a priori de la fonction pour pouvoir fixer correctement ce paramètre (ou alors il faut faire plusieurs essais pour l'adapter).

Différence de quantiles

La dispersion de la distribution de sortie peut aussi être évaluée en calculant une différence de quantiles d'ordres k_1 et k_2 [DSC04] :

$$\rho(f(\boldsymbol{x} + \boldsymbol{\chi}_x)) = \mathrm{Q}_{k_1}(f(\boldsymbol{x} + \boldsymbol{\chi}_x)) - \mathrm{Q}_{k_2}(f(\boldsymbol{x} + \boldsymbol{\chi}_x)). \tag{5.9}$$

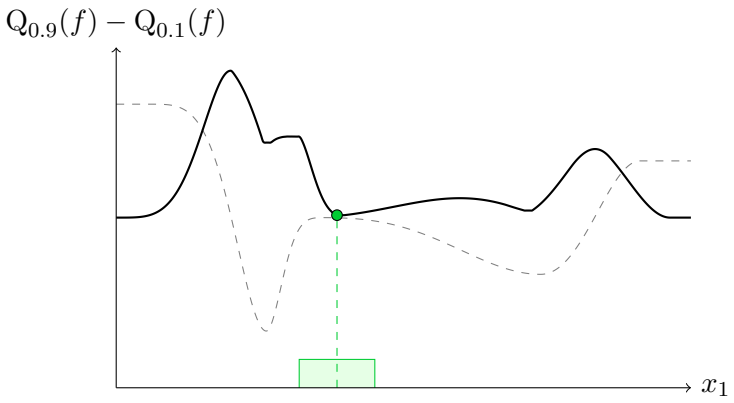


FIGURE 5.12 – Différence de quantiles d'ordre 0.9 et 0.1 (trait noir) de f (en pointillés) pour une incertitude uniforme sur x_1. Le point optimal ainsi que son incertitude sont représentés en vert.

Dans l'illustration de la figure 5.12, nous avons par exemple choisi $k_1 = 0.9$ et $k_2 = 0.1$. On remarque des similarités avec la mesure de variance présentée précédemment, qui a pour but d'estimer aussi la dispersion de la distribution de sortie. En oubliant les parties extrêmes à gauche et à droite de l'espace de recherche, l'optimum se situe ici encore au niveau d'un maximum local déterministe. C'est en effet à cet endroit que la fonction varie le moins.

Cette mesure peut être utilisée dans le cas d'une modélisation déterministe des incertitudes. Pour des intervalles de variation, on ne peut en effet pas calculer de variance mais, par contre, la taille de l'intervalle en sortie correspond à la différence des quantiles d'ordre 1 et 0. Cette mesure est par exemple employée dans ce cadre dans [Li07].

5.4.3 En conclusion sur les mesures de robustesse

Finalement, on voit qu'il y a de nombreuses expressions possibles de la robustesse d'une solution. Selon la formulation de la robustesse qu'on considère, les solutions d'un même problème peuvent différer. Il n'existe pas de critère universel pour la robustesse : chacun des critères présentés ci-dessus peut s'appliquer et se justifier pour un cas précis d'optimisation. La mesure de la robustesse à utiliser devra donc être choisie en fonction des informations qu'on possède sur les incertitudes du problème ainsi que des propriétés de robustesse qu'on souhaite obtenir pour les solutions optimales.

Le tableau 5.1 récapitule les quatre principales mesures de robustesse que nous venons de présenter. Certaines mesures prennent en compte la performance du système en plus de la robustesse des solutions (ce que nous avons appelé « performance robuste »), et d'autres sont des mesures de robustesse pure qui sont totalement indépendantes de la performance (il s'agit des mesures de dispersion). Pour illustration, on peut remarquer que notre fonction exemple a le même comportement dans le voisinage des bornes de son espace de définition (où la fonction est relativement plane). Ces deux bornes ont des variances égales (mesure de robustesse pure) mais des espérances différentes (mesure de performance robuste). L'espérance tient en effet compte de la valeur de la fonction objectif en plus de son comportement dans le voisinage des solutions, alors que la variance considère uniquement les variations de cette fonction.

	Mesures basées sur les espérances	*Mesures basées sur les quantiles*
Performance robuste	espérance	quantiles
Dispersion	variance	différence de quantiles

TABLE 5.1 – Les quatre principales mesures de robustesse.

Nous avons exprimé ces mesures de robustesse en considérant un objectif f, mais elles pourraient être écrites de la même manière pour une contrainte g ou h. Deux notions différentes de robustesse cohabitent : d'un côté la robustesse liée aux objectifs (on ne souhaite pas que les objectifs se dégradent trop fortement près de l'optimum), et de l'autre la robustesse liée à la satisfaction des contraintes (on ne souhaite pas que l'optimum se situe trop près des limites imposées par les contraintes). Si on regarde rapidement dans la littérature, les mesures d'espérance sont plutôt utilisées pour l'optimisation robuste (aussi appelée RO, pour « Robust Optimization » en anglais) alors que les mesures à base de quantiles sont employées en optimisation fiabiliste (RBO pour « Reliability-Based Optimization » [YCD05]). Nous nous concentrons plutôt ici sur l'optimisation robuste.

L'espérance et la variance sont les mesures les plus utilisées actuellement en optimisation robuste, ainsi que le pire cas. Le quantile nous semble être une autre mesure intéressante du fait de sa signification concrète qui pourrait intéresser les industriels (on garantit par exemple que 95% des systèmes réalisés auront une performance supérieure ou égale à une valeur donnée). Il permet d'avoir une version « relaxée » de la mesure du pire cas qui est souvent trop contraignante.

5.5 Formulation d'un problème d'optimisation robuste

Les mesures de robustesse permettent de quantifier la robustesse de différentes solutions afin de les comparer. On peut ainsi analyser par exemple les solutions optimales fournies par une optimisation déterministe classique afin de déterminer laquelle est la plus robuste. Mais on peut aussi chercher à optimiser cette robustesse de façon à obtenir les solutions les plus robustes possibles. Les optima robustes ne sont en effet pas forcément situés au même endroit que les optima classiques (même locaux). C'est pourquoi la prise en compte de la robustesse au sein même du processus d'optimisation se justifie vis-à-vis d'une simple analyse de la robustesse des solutions issues d'une optimisation déterministe.

L'optimisation robuste est un pan de la recherche opérationnelle qui cherche à optimiser un système en prenant en compte les différentes incertitudes du problème. La conception de systèmes robustes est née sous l'impulsion de Taguchi [Tag89], même si on utilise aujourd'hui des méthodes différentes de celles qu'il avait proposées. Le but reste néanmoins le même : trouver des solutions optimales peu sensibles aux variations qui pourraient apparaître du fait des incertitudes du problème.

L'explicitation des incertitudes présentes au sein d'un problème d'optimisation déterministe initial amène à la définition d'un problème stochastique où les objectifs et les contraintes possèdent une certaine distribution de probabilité. Les mesures de robustesse permettent de quantifier certaines caractéristiques de ces distributions, et peuvent donc être intégrées au problème stochastique afin de le ramener à un problème déterministe

robuste qu'on saura résoudre. Les mesures de robustesse peuvent être introduites de plusieurs manières différentes : elles peuvent remplacer certains des objectifs initiaux, ou bien être ajoutées en tant que nouveaux objectifs ou nouvelles contraintes.

5.5.1 Remplacement d'objectifs

Un problème d'optimisation robuste peut être formulé en remplaçant chaque objectif du problème déterministe initial par une mesure de robustesse. Le problème stochastique :

$$\text{minimiser } \boldsymbol{f}(\boldsymbol{x} + \boldsymbol{\chi}_x) \tag{5.10}$$

devient alors le problème déterministe robuste suivant :

$$\text{minimiser } \boldsymbol{\rho}(\boldsymbol{f}(\boldsymbol{x} + \boldsymbol{\chi}_x)). \tag{5.11}$$

On peut donc par exemple simplement remplacer les objectifs initiaux par la mesure de leur espérance. Le problème devient ainsi un problème d'optimisation robuste dans lequel les incertitudes sont prises en compte en considérant une performance moyenne plutôt que des valeurs « ponctuelles » des objectifs qui n'ont plus vraiment de signification dans le cadre d'un environnement incertain. La formulation d'un problème d'optimisation robuste par remplacement d'objectifs ne modifie pas le nombre d'objectifs du problème initial, ce qui a l'avantage de ne pas augmenter sa complexité.

Les mesures de robustesse utilisées peuvent être différentes pour chaque objectif, mais dans tous les cas il est préférable qu'elles combinent à la fois robustesse et performance (comme par exemple les mesures de l'espérance et du quantile, que nous avons appelées mesures de performance robuste). En effet, une mesure de robustesse pure ne tiendra pas compte de la performance effective des solutions et l'objectif concerné ne sera alors pas optimisé au sens où on l'attendait initialement. On peut aussi définir de nouvelles mesures de performance robuste en combinant une mesure de dispersion avec l'objectif initial ou bien avec une autre mesure de robustesse. Un cas très répandu dans la littérature est d'optimiser l'espérance et la variance agrégées avec un facteur de pondération α (voir [Jur07] par exemple) :

$$\rho(f(\boldsymbol{x} + \boldsymbol{\chi}_x)) = \mathrm{E}[f(\boldsymbol{x} + \boldsymbol{\chi}_x)] + \alpha\sqrt{\mathrm{Var}(f(\boldsymbol{x} + \boldsymbol{\chi}_x))}. \tag{5.12}$$

Une agrégation espérance-variance est illustrée sur la figure 5.13, et cette méthode est utilisée dans [ALC06] pour réaliser une optimisation robuste. Cette nouvelle mesure permet d'éviter les zones de trop forte variance tout en tenant compte de la performance moyenne des solutions. Il faut par contre pouvoir fixer correctement la valeur de α car les résultats obtenus en dépendent. De nombreuses autres mesures similaires peuvent être définies par agrégation de mesures de robustesse et/ou de performance.

Dans le cadre d'une optimisation fiabiliste, la même méthode peut être appliquée sur les contraintes. On utilise généralement pour cela une mesure de quantile pour définir une garantie probabiliste de faisabilité (voir par exemple [DC99]). Un seuil de probabilité k est alors choisi pour chacune des contraintes g et h du problème, et ces contraintes sont modifiées pour accepter une probabilité de non-faisabilité de $(1-k)$ dans le voisinage d'un point $\boldsymbol{x}$ considéré. On peut distinguer les contraintes « dures », c'est-à-dire celles qui ne doivent absolument pas être violées et pour lesquelles on prendra $k = 1$ (par exemple pour une contrainte exprimant le fait qu'une masse doit rester positive) et les contraintes

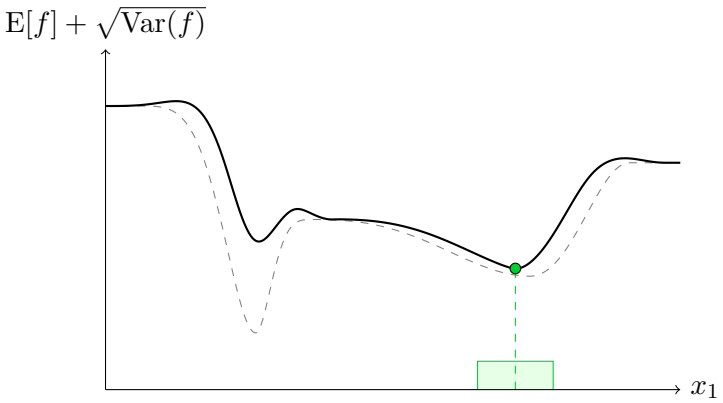


FIGURE 5.13 – Agrégation de l'espérance et de la variance (trait noir) de f (en pointillés) pour une incertitude uniforme sur x_1. Le point optimal ainsi que son incertitude sont représentés en vert.

« souples » pour lesquelles on accepte un certain degré de violation ($k < 1$). Dans certains cas, il est aussi possible de transformer ces contraintes souples en objectifs à optimiser : on va alors chercher à minimiser la violation des contraintes sans définir de seuils de probabilité k acceptables.

5.5.2 Ajout de nouveaux objectifs

On peut aussi voir le problème d'optimisation robuste sous un angle multiobjectifs en considérant que la robustesse est un objectif qui entre en contradiction avec l'objectif initial de performance. Il est donc possible de conserver les objectifs initiaux et d'ajouter des objectifs de robustesse au problème, pour obtenir des solutions de compromis sur un front de Pareto performance-robustesse (voir la figure 5.14). Le problème (5.10) devient ainsi :

$$\text{minimiser } \{\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{\rho}(\boldsymbol{f}(\boldsymbol{x} + \boldsymbol{\chi}_x))\}. \tag{5.13}$$

On choisit généralement dans ce cadre des mesures de robustesse pure (comme la variance ou la différence de quantiles), c'est-à-dire des mesures qui ne tiennent pas compte de la performance car celle-ci est déjà présente dans les objectifs initiaux. Cette méthode va donc doubler le nombre d'objectifs. Si le nombre d'objectifs du problème initial est déjà assez important, on peut aussi ne rajouter qu'un seul objectif de robustesse en agrégeant les valeurs de robustesse des différents objectifs (en considérant par exemple une robustesse moyenne on bien la pire robustesse obtenue). Il faudra alors que ces mesures soient normalisées avant d'être agrégées.

Cette formulation fait intervenir les objectifs de performance initiaux qui ne prennent pas en compte les incertitudes du problème. Dans un contexte incertain, cette valeur de performance ponctuelle n'a pas vraiment de sens, mais cela permet néanmoins d'obtenir au final un ensemble de solutions qui contient les solutions déterministes classiques déterminées par les objectifs initiaux uniquement. On pourra ainsi décider a posteriori de prendre en compte ou pas les incertitudes spécifiées, en fonction des résultats obtenus. Si on veut véritablement se placer dans un contexte d'optimisation sous incertitude, on peut

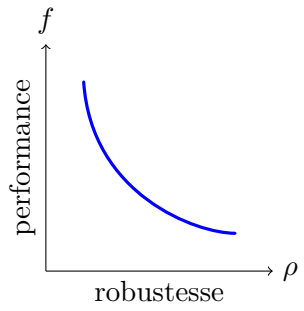


FIGURE 5.14 – Solutions de compromis entre performance et robustesse.

aussi choisir de remplacer chaque objectif par deux expressions différentes de la robustesse, l'une de robustesse pure et l'autre de performance robuste (comme la variance et l'espérance par exemple) :

$$\text{minimiser } \{\boldsymbol{\rho}_1(\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)), \boldsymbol{\rho}_2(\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{\chi}_x))\}. \tag{5.14}$$

Dans la section précédente, la définition d'une agrégation de ces deux mesures sous-entendait en effet implicitement qu'on souhaitait les optimiser simultanément. En séparant les deux objectifs, on obtient ainsi des compromis entre performance robuste et robustesse pure et on ne considère alors plus la performance ponctuelle d'une solution.

L'approche par ajout d'objectifs permet de choisir a posteriori le degré de robustesse qu'on souhaite avoir vis-à-vis des performances qui en découlent, alors que dans l'approche de remplacement d'objectifs le degré de robustesse est fixé a priori (donné au départ ou induit par l'agrégation des mesures de robustesse et de performance). Cependant, elle implique de démultiplier le nombre d'objectifs à prendre en considération. Cela peut être problématique pour résoudre le problème ainsi que pour effectuer un choix final de solution, la recherche et le parcours d'un front de Pareto en grande dimension n'étant pas aisés.

5.5.3 Ajout de nouvelles contraintes

La troisième et dernière manière de formuler un problème d'optimisation robuste est d'intégrer la robustesse sous la forme de contraintes ajoutées au problème initial. On souhaite en effet que les solutions optimales ne soient pas trop sensibles aux incertitudes. En introduisant de nouvelles contraintes, on peut ainsi quantifier le seuil de sensibilité qu'on est prêt à accepter. Des contraintes sur la variance des objectifs peuvent être ajoutées au problème par exemple. Le problème (5.10) devient alors :

$$\begin{array}{l} \text{minimiser } \ \boldsymbol{f}(\boldsymbol{x}) \\ \text{s.c. } \boldsymbol{\rho}(\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)) \leq \boldsymbol{\lambda}. \end{array} \tag{5.15}$$

Comme dans le cas précédent, le problème fait apparaître les objectifs initiaux qui ne prennent pas en compte les incertitudes. Il est aussi possible de modifier en plus ces objectifs avec l'ajout de nouvelles contraintes. L'article [LSN04] définit ainsi la M-robustesse comme étant le fait de minimiser l'espérance d'un objectif avec une contrainte sur sa variance, et la V-robustesse comme le fait de minimiser la variance avec une contrainte sur l'espérance.

Une méthode assez répandue appelée méthode « six-sigma » vise à trouver des solutions pour lesquelles la performance reste acceptable dans un rayon de six fois l'écart-type autour de la moyenne des objectifs. Le but est d'obtenir des points avec une variance réduite dans une plage de performance donnée. Le problème d'optimisation s'écrit comme une minimisation des objectifs initiaux (ou bien de leur espérance et de leur variance [SOF05]) sous les contraintes suivantes :

$$\begin{cases} \mathrm{E}[f(\boldsymbol{x}+\boldsymbol{\chi}_x)] + n\sqrt{\mathrm{Var}(f(\boldsymbol{x}+\boldsymbol{\chi}_x))} \leq U, \\ \mathrm{E}[f(\boldsymbol{x}+\boldsymbol{\chi}_x)] - n\sqrt{\mathrm{Var}(f(\boldsymbol{x}+\boldsymbol{\chi}_x))} \geq L, \end{cases} \tag{5.16}$$

avec $U, L \in \mathbb{R}$ des constantes fixées qui définissent la plage de performance acceptable et n le « niveau sigma » à atteindre (on prend par exemple $n = 6$ pour un niveau « six-sigma »). Cette méthode permet d'obtenir des solutions dont la performance reste correcte même si des variations surviennent.

L'utilisation de contraintes sur la robustesse des solutions permet de ne pas augmenter le nombre d'objectifs du problème initial, mais sa complexité est tout de même accrue car on augmente le nombre de contraintes.

5.6 Problèmes de test pour l'optimisation robuste

Alors que le domaine de l'optimisation multiobjectifs classique est relativement bien fourni en problèmes de test, il n'existe à notre connaissance que très peu de cas dédiés à l'optimisation robuste.

Le but d'une suite de tests est de définir des problèmes qui présentent les diverses difficultés rencontrées dans les problèmes réels et dont les solutions optimales sont connues. Il semble difficile de proposer un tel ensemble de cas tests pour l'optimisation robuste du fait des nombreuses approches existantes pour la notion de robustesse. Aucune mesure de robustesse ni formulation de problème n'est meilleure que les autres, tout dépend du contexte de son utilisation. Différentes mesures ou différentes formulations peuvent conduire à des solutions optimales complètement distinctes pour un même problème initial. Les solutions dépendent aussi des distributions de probabilité définies sur les paramètres incertains considérés. Une suite de problèmes de test en optimisation robuste devrait donc fournir l'ensemble des solutions optimales pour différentes mesures de robustesse, différentes formulations de problème et différentes distributions d'incertitude. Ceci explique qu'aucune suite de ce type ne soit connue à ce jour.

Les tests des méthodes de résolution de problèmes d'optimisation robuste se font alors généralement sur des problèmes simples mono-objectifs, ou bien se basent sur des problèmes issus de l'optimisation déterministe classique (comme dans [Soa08] par exemple). Quelques problèmes spécifiques d'optimisation robuste existent tout de même dans la littérature. Nous en présentons deux ci-dessous.

Cas test analytique (problème DG2)

L'article [DG06] évoque les difficultés pouvant être rencontrées lors de la recherche de solutions robustes et présente plusieurs exemples de problèmes analytiques d'optimisation robuste en lien avec ces difficultés. Dans ces problèmes, on cherche à minimiser la valeur

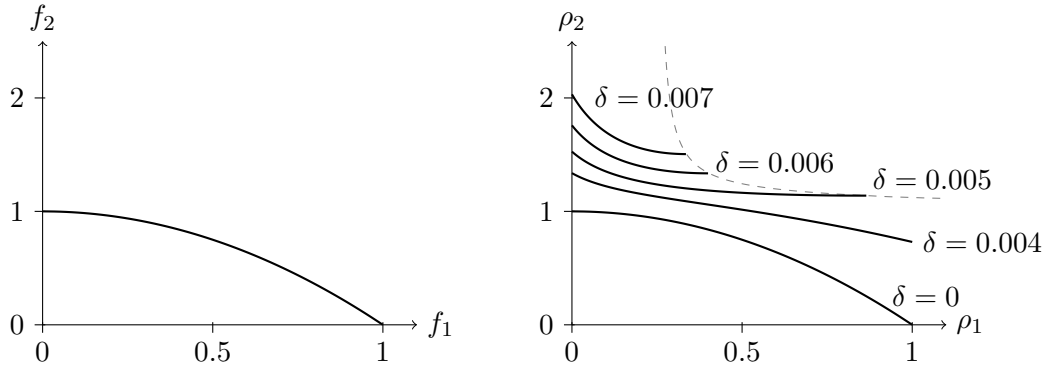


FIGURE 5.15 – Fronts de Pareto théoriques pour le problème DG2 : à gauche le front de Pareto déterministe, et à droite des fronts de Pareto robustes pour différentes valeurs de δ.

moyenne des objectifs en considérant une incertitude uniforme sur les paramètres d'entrée. Nous présentons ici l'énoncé du « problème 2 » de cet article (que nous appellerons problème DG2).

Il s'agit d'un problème à deux objectifs f_1 et f_2 avec un nombre quelconque n_x de variables d'entrées $\boldsymbol{x}$. Le problème déterministe initial est le suivant :

$$\underset{\boldsymbol{x}}{\text{minimiser}}\ \{f_1(\boldsymbol{x}), f_2(\boldsymbol{x})\}, \tag{5.17}$$

avec :

$$\begin{aligned} &f_1(\boldsymbol{x}) = x_1, \\ &f_2(\boldsymbol{x}) = 1 - x_1^2 + \left(\frac{1}{0.2 + x_1} + 10x_1^2\right) \sum_{i=2}^{n_x} \left(10 + x_i^2 - 10\cos(4\pi x_i)\right). \end{aligned}$$

Les intervalles de variations pour les paramètres sont $x_1 \in [0, 1]$ et $x_i \in [-1, 1]$ pour $2 \leq i \leq n_x$. Le front de Pareto déterministe est donné par la relation : $f_2 = 1 - f_1^2$ (il est représenté sur la figure 5.15).

Dans ce problème, les variables $\boldsymbol{x}$ sont affectées d'une incertitude uniforme. On considère par exemple $\chi_{x_1} \sim U(-\delta, \delta)$ et $\chi_{x_i} \sim U(-2\delta, 2\delta)$ pour $2 \leq i \leq n_x$, avec $\delta \in \mathbb{R}^+$ comme proposé dans l'article. On souhaite minimiser l'espérance résultante sur chacun des objectifs. Le problème d'optimisation robuste à résoudre est donc le suivant :

$$\underset{\boldsymbol{x}}{\text{minimiser}}\ \{\rho_1(f_1(\boldsymbol{x} + \boldsymbol{\chi}_x)), \rho_2(f_2(\boldsymbol{x} + \boldsymbol{\chi}_x))\}, \tag{5.18}$$

avec $\rho_i(f_i(\boldsymbol{x} + \boldsymbol{\chi}_x)) = \mathrm{E}[f_i(\boldsymbol{x} + \boldsymbol{\chi}_x)]$. Ces espérances peuvent ici être calculées analytiquement afin de trouver le front de Pareto robuste théorique :

$$\begin{aligned} &\rho_1(f_1(\boldsymbol{x} + \boldsymbol{\chi}_x)) = x_1, \\ &\rho_2(f_2(\boldsymbol{x} + \boldsymbol{\chi}_x)) = 1 - x_1^2 - \frac{\delta^2}{3} + \left(\frac{1}{2\delta}\log\left(\frac{0.2 + x_1 + \delta}{0.2 + x_1 - \delta}\right) + 10x_1^2 + \frac{10\delta^2}{3}\right) \\ &\qquad \times \sum_{i=2}^{n_x} \left(10 + \frac{\delta^2}{3} - \frac{10}{4\pi\delta}\sin(4\pi\delta)\right). \end{aligned} \tag{5.19}$$

L'équation du front de Pareto robuste correspondant peut être obtenue en remplaçant x_1 par ρ_1 dans l'équation de ρ_2 (le front robuste effectif ne constitue le plus souvent qu'une partie de la courbe décrite par cette équation). Plus la valeur de δ augmente, et plus le nombre de solutions robustes est restreint. Ces fronts théoriques sont représentés sur la figure 5.15.

Cas test des deux barres

Un second problème d'optimisation robuste issu d'un cas physique simple est présenté dans [JDC03]. Il est repris dans [PLRBR09, PLRRB09] sous la forme d'un problème d'optimisation à la fois robuste et fiabiliste.

Ce problème décrit le comportement d'une structure composée de deux tubes creux de même dimension, fixés au sol et reliés entre eux à leur sommet (voir la figure 5.16). Une force $\vec{F}$ dirigée vers le sol est appliquée au sommet de la structure. On souhaite minimiser le volume total V de matière utilisée en optimisant la longueur L et le diamètre d des tubes, tout en garantissant la bonne tenue de la structure : la contrainte normale s doit rester inférieure à la limite d'élasticité $s_{\max}$ ainsi qu'à la contrainte critique de flambage s_{crit}. Le problème déterministe s'écrit donc :

$$\begin{array}{l} \min\limits_{d,L} \; V(T,d,L) \\ s.c. \left\{ \begin{array}{l} s(F,L,T,d,B) \leq s_{\max}, \\ s(F,L,T,d,B) \leq s_{\text{crit}}(E,T,d,L). \end{array} \right. \end{array} \tag{5.20}$$

Le modèle est décrit par les équations suivantes :

$$\begin{aligned} V &= 2\pi T d L \times 10^{-6}, \\ s &= \frac{FL}{2\pi T d \sqrt{L^2 - B^2}}, \\ s_{\text{crit}} &= \frac{\pi^2 E (T^2 + d^2)}{8L^2}. \end{aligned}$$

Les variables F, E, B, T et $s_{\max}$ sont des paramètres environnementaux fixés. L'ensemble des variables du problème est détaillé dans le tableau 5.2. La résolution de ce problème déterministe donne comme point optimal la solution $(d, L) = (38, 964)$ pour laquelle $V = 0.5754$ L, $s = 399.9931$ N.mm^{-2} et $s_{\text{crit}} = 404.3131$ N.mm^{-2}.

Une étude de cette solution optimale montre qu'elle est très peu fiable lorsqu'on prend en compte des incertitudes gaussiennes sur certains paramètres du problème (décrites dans le tableau 5.2) : les contraintes sont alors violées une fois sur deux. Il est donc nécessaire d'intégrer ces incertitudes au problème pour le transformer en problème d'optimisation robuste. On cherche à la fois à maximiser le taux de fiabilité (c'est-à-dire la probabilité pour que les contraintes soient respectées) et à minimiser le quantile à 90% du volume. Le problème d'optimisation robuste multiobjectifs à résoudre est donc le suivant (en omettant les variables dont dépendent les fonctions s, s_{crit} et V) :

$$\underset{d,L}{\text{minimiser}} \; \{-P((s \leq s_{\max}) \cap (s \leq s_{\text{crit}})), \mathrm{Q}_{0.9}(V)\}. \tag{5.21}$$

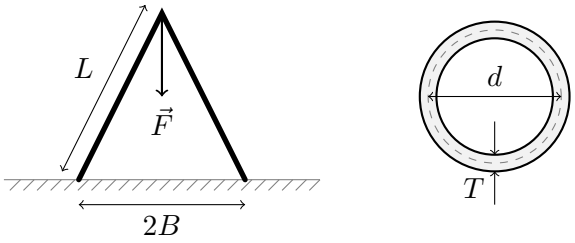


FIGURE 5.16 – Représentation schématique des deux barres à gauche, avec une vue de leur section à droite.

Variables de décision			
d_1	Diamètre de la section des tubes d	[20,80]	mm
d_2	Longueur des tubes L	[800,1200]	mm
Paramètres environnementaux			
e_1	Force appliquée F	150000	N
e_2	Module d'Young E	210000	N.mm^{-2}
e_3	Demi-largeur de la structure B	750	mm
e_4	Épaisseur de la section des tubes T	2.5	mm
e_5	Limite d'élasticité $s_{\max}$	400	N.mm^{-2}
Incertitudes			
χ_{d_1}	Diamètre de la section des tubes d	$\mathcal{N}(0,1)$	mm
χ_{d_2}	Longueur des tubes L	$\mathcal{N}(0,25)$	mm
χ_{e_1}	Force appliquée F	$\mathcal{N}(0,9\times10^8)$	N
χ_{e_2}	Module d'Young E	$\mathcal{N}(0,4.41\times10^8)$	N.mm^{-2}
χ_{e_3}	Demi-largeur de la structure B	$\mathcal{N}(0,25)$	mm
χ_{e_4}	Épaisseur de la section des tubes T	$\mathcal{N}(0,0.01)$	mm
Objectif			
f_1	Volume de matière utilisée V	minimiser	L
Contraintes			
g_1	Contrainte normale s	$\leq s_{\max}$	N.mm^{-2}
g_2	Contrainte normale s	$\leq s_{\text{crit}}$	N.mm^{-2}

TABLE 5.2 – Paramètres, objectifs et contraintes du cas test des deux barres.

5.7 Résolution d'un problème d'optimisation robuste

La résolution d'un problème d'optimisation robuste est relativement simple. Nous venons en effet de voir comment les incertitudes sur les variables d'entrée (classes A et B) peuvent être intégrées au problème initial par le biais de mesures de robustesse. On obtient ainsi une formulation totalement déterministe du problème robuste, qui peut être résolue par les algorithmes classiques d'optimisation. Ce problème robuste est parfois légèrement plus complexe que le problème initial du fait de l'ajout d'objectifs ou de contraintes. La seule difficulté réside dans le fait qu'il faut évaluer des mesures de robustesse au lieu des objectifs initiaux (ou en plus de ceux-ci). Les méthodes de calcul de robustesse sont détaillées dans la section suivante.

Nous n'avons par contre pas abordé le cas des incertitudes de classe C qui affectent la mesure des sorties du système. On peut rencontrer ce type d'incertitudes lors de l'utilisation de simulations numériques complexes visant à reproduire la physique du système réel. Outre le fait que ces simulations sont des modèles approchés de la réalité, elles peuvent être basées sur la description de phénomènes aléatoires et leurs sorties possèdent donc une certaine variabilité (qui reste tout de même relativement restreinte en général). Des algorithmes spécifiques ont été développés pour traiter ce genre de problèmes dans le cadre de « l'optimisation bruitée » (« noisy optimization » en anglais). Les algorithmes d'optimisation bruitée cherchent à déterminer la position du minimum d'une fonction entachée d'un certain bruit d'évaluation. Ce bruit ne peut pas être traité comme les incertitudes d'entrée, car on n'en a aucun contrôle : les incertitudes de classe C sont internes au système et leur source est le plus souvent non maîtrisable. La seule manière de les estimer est de réaliser plusieurs simulations du système pour un même jeu de paramètres d'entrée. L'algorithme d'optimisation pourra donc décider d'évaluer plusieurs fois une même solution afin de déterminer la distribution du bruit en ce point (qui est généralement supposée gaussienne). Le domaine de l'optimisation bruitée ne sera pas abordé plus en détails ici. Lorsque les incertitudes de mesure sont de faible amplitude, elles peuvent en effet être prises en compte par la construction de modèles de substitution régressants des fonctions objectifs. Ces modèles réalisent un lissage des données d'apprentissage et permettent d'approcher la fonction initiale défaite de son bruit. Nous nous placerons dans ce cadre pour notre étude.

Des méthodes issues de l'optimisation bruitée ont aussi été transposées à l'optimisation robuste. On peut en effet considérer que les incertitudes sur les variables d'entrée produisent un certain bruit supplémentaire sur les objectifs. Dans les travaux de [TG97], la valeur de l'objectif est ainsi déterminée en évaluant un point choisi aléatoirement dans le voisinage de la solution considérée. Si on réévalue cette même solution, on obtient plusieurs valeurs différentes qui correspondent aux variations de l'objectif dans le voisinage. Cette méthode permet donc de trouver des solutions robustes à l'aide d'algorithmes d'optimisation bruitée. Néanmoins, la maîtrise des incertitudes d'entrée est beaucoup moins fine car celles-ci sont masquées et englobées dans l'évaluation des objectifs. L'estimation de la robustesse d'un point ne peut alors être réalisée qu'à partir d'échantillonnages. Nous allons voir qu'il existe en fait d'autres méthodes de calcul de la robustesse à partir des distributions d'incertitude des variables d'entrée.

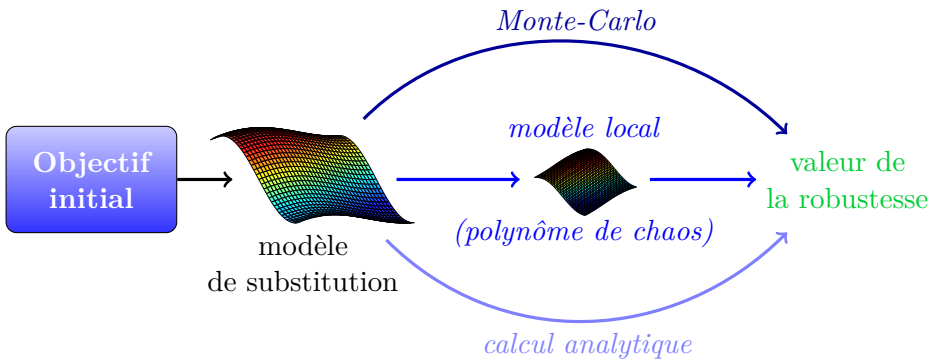


FIGURE 5.17 – Méthodes de calcul des mesures de robustesse à partir d'un modèle de substitution de l'objectif initial.

5.8 Calcul des mesures de robustesse

La résolution d'un problème d'optimisation robuste nécessite d'évaluer les mesures de robustesse employées dans sa formulation. Le calcul direct de la robustesse sur l'objectif f initial est généralement impossible car il demande d'effectuer un grand nombre d'évaluations de cette fonction (souvent coûteuses) afin d'approcher les caractéristiques de sa distribution pour un point donné. C'est donc tout naturellement qu'on préfère construire un modèle de substitution $\hat{f}$ de l'objectif sur lequel seront basés les calculs de robustesse (l'utilisation d'un modèle régressant permet de plus de s'affranchir des incertitudes de classe C).

Les mesures de robustesse peuvent être calculées de différentes manières sur ce modèle de substitution. Nous en présentons trois parmi les plus répandues (voir la figure 5.17) : l'estimation par la méthode de Monte-Carlo, l'utilisation de modèles locaux comme les polynômes de chaos, et le calcul analytique sur certains modèles particuliers. Dans un second temps, nous nous intéresserons aussi à l'estimation de l'erreur commise lors de l'évaluation de la robustesse sur un modèle de substitution.

5.8.1 Monte-Carlo

Le moyen le plus simple et le plus évident pour évaluer la robustesse d'une solution $\boldsymbol{x}$ est la méthode des tirages de Monte-Carlo. Il s'agit d'une technique assez universelle qui permet d'estimer l'ensemble des mesures de robustesse présentées plus haut. Elle repose sur le calcul avec le modèle $\hat{f}$ d'un ensemble de n échantillons $(\boldsymbol{x}^i, y^i)$ $(1 \leq i \leq n$, avec $y^i = \hat{f}(\boldsymbol{x}^i))$ choisis dans le voisinage de $\boldsymbol{x}$. Ces échantillons sont tirés au hasard en suivant la densité de probabilité des incertitudes $\boldsymbol{\chi}_x$, et représentent donc des points qui pourraient être obtenus du fait des incertitudes du problème si on sélectionne la solution $\boldsymbol{x}$. Plus le nombre d'échantillons n est grand, plus on a une représentation fidèle du voisinage probabiliste de $\boldsymbol{x}$ et donc plus l'estimation de la robustesse de cette solution sera précise. Afin d'assurer une bonne représentativité du voisinage par les échantillons, ceux-ci peuvent aussi être choisis sur un LHS (voir la section 3.8.1) qui garantit une bonne répartition des points dans l'espace. Cette méthode est par exemple utilisée dans [MS06].

À partir des échantillons de Monte-Carlo, l'espérance et la variance peuvent être esti-

mées comme suit :

$$\mathrm{E}[\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)] \simeq \frac{1}{n}\sum_{i=1}^{n} y^i = \bar{y}, \tag{5.22}$$

$$\mathrm{Var}(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)) \simeq \frac{1}{n-1}\sum_{i=1}^{n}(y^i-\bar{y})^2. \tag{5.23}$$

Le calcul du quantile d'ordre k nécessite dans un premier temps de classer les échantillons par ordre croissant des y^i. Le quantile est ensuite estimé par la valeur du $\lceil kn \rceil$-ième échantillon :

$$\mathrm{Q}_k(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)) \simeq y^{\lceil kn \rceil}. \tag{5.24}$$

Plusieurs autres méthodes proposent d'améliorer cette estimation en prenant par exemple une valeur comprise entre $y^{\lfloor kn \rfloor}$ et $y^{\lceil kn \rceil}$. Des techniques d'échantillonnage adaptatif permettent aussi d'obtenir une meilleure précision. Il n'existe par contre, à notre connaissance, aucune autre méthode efficace pour calculer la mesure de quantile sans effectuer des tirages de Monte-Carlo.

Pour réduire la variance des estimations, on peut utiliser la stratégie CRN (« Common Random Numbers » [YN91]). Elle consiste à conserver la même répartition d'échantillons dans le voisinage de chaque solution pour laquelle on évalue la robustesse. On peut par exemple construire un LHS qui respecte la distribution de probabilité de $\boldsymbol{\chi}_x$ puis le translater autour de chaque point $\boldsymbol{x}$ considéré. Cela introduit un certain biais sur le calcul de robustesse, mais les estimations seront moins bruitées (ce qui peut être intéressant dans un contexte d'optimisation).

La méthode de Monte-Carlo est une technique d'estimation de la robustesse très répandue du fait de sa simplicité et de sa généralité. Elle nécessite par contre de réaliser un grand nombre d'évaluations du modèle $\hat{f}$ pour obtenir des estimations fiables. Même si ce modèle de substitution est très réactif, le calcul répété de la robustesse de différents points peut alors devenir relativement coûteux. C'est pourquoi nous nous sommes penchés sur d'autres méthodes d'évaluation des mesures de robustesse.

Il est à noter que des stratégies avancées d'échantillonnage adaptatif ont aussi été développées afin de limiter le nombre de tirages nécessaires. On peut notamment citer les méthodes de stratification ou d'échantillonnage préférentiel (voir [Caf98, EL07] par exemple). Nous ne nous sommes pas intéressés à ce type d'approches ici.

5.8.2 Chaos polynomial

La méthode du chaos polynomial [Wie38] non intrusif est couramment utilisée en analyse de sensibilité et en propagation d'incertitudes afin d'évaluer les distributions de probabilité des sorties d'un système. Elle consiste à établir un modèle polynomial local particulier $\hat{f}_{pc}$ autour d'un point $\boldsymbol{x}$ dont on souhaite évaluer la robustesse. Ce modèle est construit grâce à une base polynomiale spécifique qui facilite l'estimation de l'espérance et de la variance de la distribution de $\hat{f}_{pc}(\boldsymbol{x}+\boldsymbol{\chi}_x)$.

Tout comme les données des modèles de substitution classiques doivent être normalisées au préalable, les polynômes de chaos sont généralement définis pour des incertitudes normalisées (loi normale réduite centrée, loi uniforme sur $[0,1]$, etc.). On supposera donc ici que l'espace est transformé a priori de manière à ce que les incertitudes définies sur chaque dimension soient normalisées. Les résultats pourront ensuite être dé-normalisés pour retrouver leur valeur dans l'espace initial.

La base polynomiale utilisée dans un chaos polynomial dépend de la distribution des incertitudes présentes en entrée. Dans le cas d'une loi normale centrée réduite sur la variable aléatoire χ, on choisit par exemple une base $\boldsymbol{H}$ de polynômes d'Hermite normalisés composée de polynômes d'ordre croissant et définis de manière récursive :

$$\begin{aligned} H^0(\chi) &= 1, \\ H^1(\chi) &= \chi, \\ H^2(\chi) &= \frac{\chi^2}{\sqrt{2}} - \sqrt{2}, \\ H^3(\chi) &= \frac{\chi^3}{\sqrt{6}} - 2\sqrt{\frac{2}{3}}\chi, \\ &\vdots \\ H^n(\chi) &= \frac{\chi}{\sqrt{n}} H^{n-1}(\chi) - \sqrt{\frac{n-1}{n}} H^{n-2}(\chi). \end{aligned} \tag{5.25}$$

D'autres bases polynomiales ont été établies pour des distributions de probabilité classiques : polynômes de Legendre pour les lois uniformes, polynômes de Laguerre pour les lois Gamma et polynômes de Jacobi pour les lois Bêta. Chaque variable $(x_i + \chi_{x_i})$ du problème peut ainsi être décrite avec la base qui correspond à son incertitude (nous garderons ici la notation « $\boldsymbol{H}$ » même s'il ne s'agit pas forcément de polynômes d'Hermite).

Lorsque la base $\boldsymbol{H}_i$ de chaque variable $(x_i+\chi_{x_i})$ a été déterminée, le polynôme de chaos multivarié $\hat{f}_{pc}$ s'écrit comme une combinaison des polynômes de chacune de ces bases :

$$\hat{f}_{pc}(\boldsymbol{x} + \boldsymbol{\chi}_x) = \sum_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_1 \leq k} c_{\boldsymbol{\alpha}} \prod_{i=1}^{n_x} H_i^{\alpha_i}(x_i + \chi_{x_i}), \tag{5.26}$$

avec :

$$\begin{cases} \boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_{n_x}) \in \mathbb{N}^{n_x} \text{ les ordres des polynômes monovariés,} \\ \|\boldsymbol{\alpha}\|_1 = \sum_{i=1}^{n_x} \alpha_i, \\ k \in \mathbb{N} \text{ l'ordre du polynôme de chaos,} \\ c_{\boldsymbol{\alpha}} \in \mathbb{R} \text{ les coefficients du modèle.} \end{cases}$$

Les coefficients $c_{\boldsymbol{\alpha}}$ peuvent être déterminés comme ceux d'un polynôme classique (voir la section 3.2) à partir d'une base locale d'échantillons d'apprentissage issus de la fonction objectif f ou bien de son modèle $\hat{f}$. Différentes méthodes existent pour choisir la position de ces échantillons et déterminer les coefficients correspondants dans le cadre des polynômes de chaos, mais nous ne nous y attarderons pas ici.

Les bases polynomiales servant à construire les polynômes de chaos sont définies de manière à avoir la propriété d'orthogonalité suivante (qui est liée à la densité p_χ de la variable aléatoire χ) :

$$\int_{-\infty}^{\infty} H^a(z) H^b(z) p_\chi(z) dz = \delta_{ab} = \begin{cases} 1 & \text{si } a = b, \\ 0 & \text{sinon.} \end{cases} \tag{5.27}$$

Pour les polynômes d'Hermite normalisés relatifs à la loi normale centrée réduite, on a ainsi :

$$\int_{-\infty}^{\infty} H^a(z) H^b(z) \frac{e^{-\frac{z^2}{2}}}{\sqrt{2\pi}} dz = \delta_{ab}. \tag{5.28}$$

Cette propriété permet de calculer très facilement l'espérance du polynôme :

$$\begin{aligned}
\mathrm{E}[\hat{f}_{pc}(\boldsymbol{x}+\boldsymbol{\chi}_x)] &= \int_{\mathbb{R}^{n_x}} \hat{f}_{pc}(\boldsymbol{z}) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} \\
&= \int_{\mathbb{R}^{n_x}} \left(\sum_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_1 \leq k} c_{\boldsymbol{\alpha}} \prod_{i=1}^{n_x} H_i^{\alpha_i}(z_i) \right) \prod_{i=1}^{n_x} p_{x_i+\chi_{x_i}}(z_i) dz_i \\
&= \sum_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_1 \leq k} c_{\boldsymbol{\alpha}} \prod_{i=1}^{n_x} \int_{-\infty}^{\infty} H_i^{\alpha_i}(z_i) p_{x_i+\chi_{x_i}}(z_i) dz_i \\
&= \sum_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_1 \leq k} c_{\boldsymbol{\alpha}} \prod_{i=1}^{n_x} \int_{-\infty}^{\infty} H_i^{\alpha_i}(z_i) H_i^0(z_i) p_{x_i+\chi_{x_i}}(z_i) dz_i \\
&= \sum_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_1 \leq k} c_{\boldsymbol{\alpha}} \prod_{i=1}^{n_x} \delta_{\alpha_i 0} \\
&= c_{\mathbf{0}}.
\end{aligned} \tag{5.29}$$

On obtient de manière similaire sa variance :

$$\mathrm{Var}(\hat{f}_{pc}(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \sum_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_1 \leq k} c_{\boldsymbol{\alpha}}^2. \tag{5.30}$$

L'espérance et la variance s'expriment donc très simplement à partir des coefficients du polynôme de chaos (comme nous le verrons dans la section suivante, on peut en fait calculer ces valeurs sur n'importe quel polynôme mais de manière plus lourde).

Le chaos polynomial est une méthode assez simple à mettre en œuvre, même s'il faut reconstruire un nouveau modèle local autour de chaque solution dont on souhaite évaluer la robustesse. Son inconvénient est que le nombre d'échantillons d'apprentissage nécessaires à la construction d'un polynôme croît exponentiellement avec la dimension de l'espace et l'ordre du polynôme. Cette méthode est donc adaptée aux cas où le nombre de paramètres incertains est relativement faible (le polynôme de chaos pouvant en effet être construit uniquement sur les variables incertaines du vecteur d'entrée $\boldsymbol{x}$, même si nous l'avons décrit ici dans un cadre général), et lorsque la fonction peut être correctement approchée par un polynôme local d'ordre restreint.

Pour éviter d'avoir à reconstruire complètement un modèle $\hat{f}_{pc}$ à chaque évaluation de robustesse, certains auteurs ont imaginé une stratégie de modélisation à deux niveaux [FCLB11]. Il s'agit en fait de construire des polynômes de chaos autour d'un certain nombre de points répartis dans l'espace de recherche, puis de représenter les variations de chaque coefficient $c_{\boldsymbol{\alpha}}$ en fonction de la position du point $\boldsymbol{x}$ par un modèle de substitution global. Lorsqu'on souhaite calculer la robustesse d'une nouvelle solution, on obtient ainsi les coefficients du polynôme de chaos local correspondant à l'aide des modèles de substitution globaux. Dans notre étude, nous ne nous sommes pas penchés sur ces méthodes évoluées.

5.8.3 Calcul analytique

Au delà des polynômes de chaos, l'espérance et la variance peuvent être calculées analytiquement sur certains modèles de substitution à partir de leurs coefficients [Jin04].

Cela permet d'obtenir une estimation plus précise et parfois plus rapide qu'en passant par un échantillonnage de Monte-Carlo. Ce calcul est possible sur les modèles de la forme :

$$\hat{f}(\boldsymbol{x}) = a_0 + \hat{f}_0(\boldsymbol{x}) = a_0 + \sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} g_{ij}(x_j), \tag{5.31}$$

où les $a_i \in \mathbb{R}$ sont les paramètres du modèle et les $g_{ij} : \mathbb{R} \to \mathbb{R}$ des fonctions monovariées. Les polynômes, le krigeage, les RBF et les MARS rentrent par exemple dans ce cadre, mais pas les réseaux de neurones.

On pose :

$$\begin{aligned} I_{ij} &= \int_{-\infty}^{\infty} g_{ij}(z_j) p_{x_j+\chi_{x_j}}(z_j) dz_j, \\ I_{i_1 i_2 j} &= \int_{-\infty}^{\infty} g_{i_1 j}(z_j) g_{i_2 j}(z_j) p_{x_j+\chi_{x_j}}(z_j) dz_j. \end{aligned} \tag{5.32}$$

En un point $\boldsymbol{x}$ dont on veut calculer la robustesse, on a alors :

$$\begin{aligned} \mathrm{E}[\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)] &= a_0 + \mathrm{E}[\hat{f}_0(\boldsymbol{x}+\boldsymbol{\chi}_x)] \\ &= a_0 + \int_{\mathbb{R}^{n_x}} \hat{f}_0(\boldsymbol{z}) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} \\ &= a_0 + \int_{\mathbb{R}^{n_x}} \left(\sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} g_{ij}(z_j) \right) \prod_{j=1}^{n_x} p_{x_j+\chi_{x_j}}(z_j) dz_j \\ &= a_0 + \sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} \int_{-\infty}^{\infty} g_{ij}(z_j) p_{x_j+\chi_{x_j}}(z_j) dz_j \\ &= a_0 + \sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} I_{ij}, \end{aligned} \tag{5.33}$$

et :

$$\begin{aligned} \mathrm{Var}(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)) &= \mathrm{Var}(\hat{f}_0(\boldsymbol{x}+\boldsymbol{\chi}_x)) \\ &= \mathrm{E}[\hat{f}_0^2(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \mathrm{E}^2[\hat{f}_0(\boldsymbol{x}+\boldsymbol{\chi}_x)] \\ &= \int_{\mathbb{R}^{n_x}} \left(\sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} g_{ij}(z_j) \right)^2 p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} - \mathrm{E}^2[\hat{f}_0(\boldsymbol{x}+\boldsymbol{\chi}_x)] \\ &= \int_{\mathbb{R}^{n_x}} \left(\sum_{i_1=1}^{m} \sum_{i_2=1}^{m} a_{i_1} a_{i_2} \prod_{j=1}^{n_x} g_{i_1 j}(z_j) g_{i_2 j}(z_j) \right) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} - \mathrm{E}^2[\hat{f}_0(\boldsymbol{x}+\boldsymbol{\chi}_x)] \\ &= \left(\sum_{i_1=1}^{m} \sum_{i_2=1}^{m} a_{i_1} a_{i_2} \prod_{j=1}^{n_x} \int_{-\infty}^{\infty} g_{i_1 j}(z_j) g_{i_2 j}(z_j) p_{x_j+\chi_{x_j}}(z_j) dz_j \right) - \left(\sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} I_{ij} \right)^2 \\ &= \left(\sum_{i_1=1}^{m} \sum_{i_2=1}^{m} a_{i_1} a_{i_2} \prod_{j=1}^{n_x} I_{i_1 i_2 j} \right) - \sum_{i_1=1}^{m} \sum_{i_2=1}^{m} a_{i_1} a_{i_2} \prod_{j=1}^{n_x} I_{i_1 j} I_{i_2 j} \\ &= \sum_{i_1=1}^{m} \sum_{i_2=1}^{m} a_{i_1} a_{i_2} \left(\left(\prod_{j=1}^{n_x} I_{i_1 i_2 j} \right) - \prod_{j=1}^{n_x} I_{i_1 j} I_{i_2 j} \right). \end{aligned} \tag{5.34}$$

Les calculs de l'espérance et de la variance du modèle se ramènent ainsi à une évaluation d'intégrales simples I_{ij} et $I_{i_1 i_2 j}$ à la place d'intégrales multiples. Selon la distribution de l'incertitude sur $(x_i + \chi_{x_i})$, ces intégrales peuvent être calculées analytiquement ou bien approchées numériquement. Nous présentons ci-après des exemples de calcul pour les polynômes et les modèles de krigeage.

La complexité du calcul de l'espérance est liée au nombre m de coefficients a_i du modèle. Pour la variance, la complexité évolue avec le carré de ce nombre de coefficients car on a affaire à une double somme sur m. Il est donc plus coûteux d'estimer une variance qu'une espérance.

Espérance et variance d'un polynôme

Un polynôme s'écrit sous la forme d'une combinaison de puissances de x (voir l'équation (3.1)) :

$$\hat{f}(\boldsymbol{x}) = a_0 + \sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} x_j^{\alpha_{ij}}, \tag{5.35}$$

avec $\alpha_{ij} \in \mathbb{N}$. Il rentre donc bien dans le cadre défini ci-dessus en posant $g_{ij}(x_j) = x_j^{\alpha_{ij}}$. Les intégrales I_{ij} et $I_{i_1 i_2 j}$ correspondent alors respectivement au moment $m_{\alpha_{ij} j}$ d'ordre k_{ij} et au moment $m_{(\alpha_{i_1 j} + \alpha_{i_2 j}) j}$ d'ordre $(\alpha_{i_1 j} + \alpha_{i_2 j})$ de la distribution sur la variable $(x_j + \chi_{x_j})$.

Si on note k le degré maximal des variables du polynôme, on a les inéquations suivantes : $\alpha_{ij} \leq k$ et $\alpha_{i_1 j} + \alpha_{i_2 j} \leq 2k$. La moyenne d'un polynôme est donc définie à partir des k premiers moments des distributions de probabilité sur $\boldsymbol{x}$, et sa variance à partir des $2k$ premiers moments.

Pour un polynôme $\hat{f}$, on a donc finalement :

$$\mathrm{E}[\hat{f}(\boldsymbol{x} + \boldsymbol{\chi}_x)] = a_0 + \sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} m_{\alpha_{ij} j}, \tag{5.36}$$

$$\mathrm{Var}(\hat{f}(\boldsymbol{x} + \boldsymbol{\chi}_x)) = \sum_{i_1=1}^{m} \sum_{i_2=1}^{m} a_{i_1} a_{i_2} \left(\left(\prod_{j=1}^{n_x} m_{(\alpha_{i_1 j} + \alpha_{i_2 j}) j} \right) - \prod_{j=1}^{n_x} m_{\alpha_{i_1 j} j} m_{\alpha_{i_2 j} j} \right). \tag{5.37}$$

Espérance et variance d'un krigeage ordinaire

Un modèle de krigeage ordinaire doté d'une fonction de corrélation gaussienne peut s'écrire sous la forme suivante (voir l'équation (3.12)) :

$$\hat{f}(\boldsymbol{x}) = \hat{\mu} + \sum_{i=1}^{n_s} \gamma_i \sigma_K^2 \prod_{j=1}^{n_x} e^{-\theta_j (x_j - s_j^i)^2} = a_0 + \sum_{i=1}^{m} a_i \prod_{j=1}^{n_x} e^{-\theta_j (x_j - s_j^i)^2}. \tag{5.38}$$

Le krigeage ordinaire rentre donc dans le cadre défini ci-dessus, avec $g_{ij}(x_j) = e^{-\theta_j (x_j - s_j^i)^2}$.

Si la variable $(x_j + \chi_{x_j})$ est dotée d'une distribution uniforme entre α et β, on aura par exemple :

$$\begin{aligned} I_{ij} &= \frac{1}{\beta - \alpha} \int_{\alpha}^{\beta} e^{-\theta_j (z_j - s_j^i)^2} dz_j \\ &= \frac{\sqrt{\pi} \left(\mathrm{erf}\left((\beta - s_j^i) \sqrt{\theta_j} \right) - \mathrm{erf}\left((\alpha - s_j^i) \sqrt{\theta_j} \right) \right)}{2(\beta - \alpha) \sqrt{\theta_j}}, \end{aligned} \tag{5.39}$$

et :

$$I_{i_1 i_2 j} = \frac{1}{\beta-\alpha}\int_{\alpha}^{\beta} e^{-\theta_j(z_j-s_j^{i_1})^2-\theta_j(z_j-s_j^{i_2})^2}dz_j$$
$$= \frac{\sqrt{\pi}\left(\operatorname{erf}\left(\left(\beta-\frac{s_j^{i_1}+s_j^{i_2}}{2}\right)\sqrt{2\theta_j}\right)-\operatorname{erf}\left(\left(\alpha-\frac{s_j^{i_1}+s_j^{i_2}}{2}\right)\sqrt{2\theta_j}\right)\right)e^{-\frac{\theta_j\left(s_j^{i_1}-s_j^{i_2}\right)^2}{2}}}{2(\beta-\alpha)\sqrt{2\theta_j}}. \tag{5.40}$$

On obtient ainsi une expression quasi analytique de la moyenne et de la variance. Elle fait appel à la fonction « erf » qui est directement implémentée numériquement dans les bibliothèques mathématiques.

Des résultats similaires peuvent être exprimés lorsque l'incertitude d'entrée est gaussienne (un calcul proche sur un kriging simple est par exemple présenté dans [Gir04]). Ce cas est détaillé en annexe. La figure 5.18 donne un exemple de modèle de krigeage ordinaire à deux dimensions pour lequel on a défini une incertitude gaussienne de variance 0.01 sur chacune des variables d'entrée. Les résultats obtenus par calcul analytique sont comparés à des évaluations par tirages de Monte-Carlo effectuées avec 1000 échantillons. On observe des résultats similaires mais légèrement bruités dans le cas de l'estimation par Monte-Carlo (alors que le calcul analytique donne des valeurs exactes).

Nous avons détaillé ici le calcul de l'espérance et de variance sur la valeur moyenne fournie par le modèle de krigeage $\hat{F}$. Nous verrons dans la section 5.8.5 qu'il est possible de donner une meilleure approximation de ces mesures de robustesse en prenant en compte le cadre bayésien défini par le krigeage.

Espérance et variance d'un krigeage universel

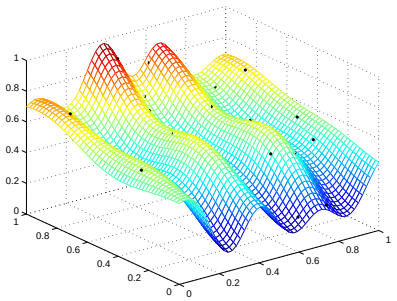
L'espérance et la variance peuvent aussi être exprimées analytiquement sur un modèle de krigeage universel doté d'une fonction de corrélation gaussienne et d'une base $\boldsymbol{\mu}$ polynomiale (voir [ALC06] par exemple). Le modèle s'écrit alors comme la somme d'un polynôme et d'un krigeage simple de moyenne nulle, et entre aussi dans le cadre de la formulation (5.31) avec des fonctions $g_{ij}(x_j)$ définies comme pour les polynômes jusqu'à un certain rang puis comme pour le krigeage ordinaire ensuite.

5.8.4 Autres méthodes de calcul des mesures de robustesse

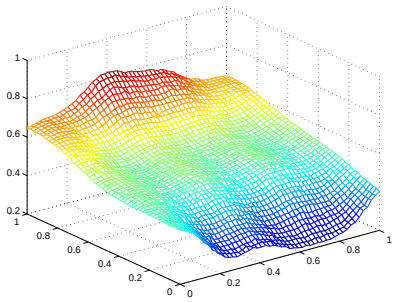
Nous avons abordé ci-dessus les trois principales méthodes de calcul de mesures de robustesse. Il en existe bien évidemment d'autres, qui sont notamment dédiées à des cas particuliers. Nous présentons ainsi ci-dessous des approches pour la mesure du pire cas et pour les mesures liées à une description des incertitudes par intervalle, ainsi qu'une méthode utilisée en optimisation fiabiliste.

Mesure du pire cas

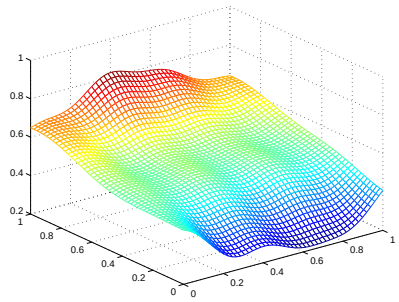
La mesure du pire cas correspond à un quantile particulier (le quantile d'ordre 1), mais néanmoins son évaluation peut être réalisée avec des méthodes différentes de celles utilisées pour les quantiles classiques. Les quantiles sont généralement estimés grâce à des tirages de Monte-Carlo, mais ce n'est pas vraiment la meilleure approche à suivre pour



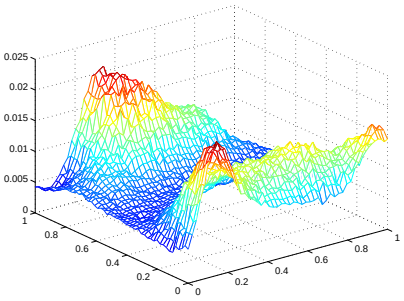
(a) Modèle de krigeage ordinaire



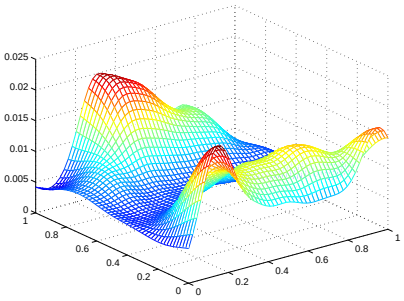
(b) Espérance par Monte-Carlo



(c) Espérance analytique



(d) Variance par Monte-Carlo



(e) Variance analytique

FIGURE 5.18 – Calcul de l'espérance et de la variance sur un modèle de krigeage ordinaire (a) avec une incertitude gaussienne sur les entrées par la méthode de Monte-Carlo (b) (d) et par calcul analytique (c) (e).

la mesure du pire cas. Le calcul de cette mesure correspond en effet à la résolution d'un problème d'optimisation (voir l'équation (5.6)) : on cherche à trouver la valeur maximale de l'objectif dans une certaine région de l'espace, qui est définie par le support de la densité de probabilité des variables incertaines. Il paraît donc évident que la résolution de ce problème par tirages aléatoires n'est pas la méthode la plus efficace. Il est généralement préférable d'utiliser un algorithme d'optimisation classique, quel qu'il soit. On peut de plus bénéficier d'informations sur la dérivée de l'objectif lorsque cette recherche de maximum est effectuée sur un modèle de substitution, ce qui peut considérablement accélérer la résolution du problème.

Analyse par intervalles

Lorsque les incertitudes sont modélisées sous la forme d'intervalles de variation (modélisation déterministe), les seules mesures de robustesse utilisables sont la mesure du pire cas ainsi que la différence des quantiles d'ordre 1 et 0. Comme dans le cas du paragraphe précédent, il s'agit donc de calculer des quantiles d'ordre extrême et la méthode de Monte-Carlo n'est pas la plus efficace. L'évaluation des mesures de robustesse peut ici aussi être effectuée grâce à un algorithme d'optimisation visant à trouver le minimum ou le maximum de l'objectif dans une région donnée.

La description déterministe des incertitudes se prête aussi à une autre approche : l'analyse par intervalles. Elle est par exemple mise en œuvre dans [Soa08]. Il s'agit en fait de propager les intervalles de variation définis sur les entrées afin d'obtenir les intervalles résultants sur les objectifs du problème. On définit pour cela une arithmétique des intervalles dotée de différentes opérations (comme l'addition : $[a, b] + [c, d] = [a + b, c + d]$, ou encore la fonction exponentielle : $e^{[a,b]} = [e^a, e^b]$ pour ne citer que les plus simples). Cette arithmétique sert à redéfinir chaque fonction objectif f ou bien leur modèle $\hat{f}$ si aucune expression analytique de la fonction n'est connue. L'analyse par intervalles permet donc de calculer l'image d'un intervalle par f, image à partir de laquelle l'estimation de la robustesse est aisée. Elle reste toutefois relativement coûteuse même si des méthodes approchées plus efficaces ont aussi été développées. Un autre désavantage de cette approche est qu'on ne prend généralement pas en considération la dépendance entre les variables d'une expression. Ainsi, si une même variable apparaît plusieurs fois dans l'expression de la fonction f elle sera considérée comme autant de variables différentes pouvant varier librement. L'intervalle obtenu en sortie sera alors bien plus grand que l'intervalle minimal réel, et la robustesse de la solution sera sous-évaluée.

Probabilité de défaillance

L'optimisation fiabiliste cherche à prendre en compte les effets des incertitudes sur les contraintes du problème. On souhaite maximiser les chances que celles-ci restent satisfaites lorsque les paramètres incertains varient. Il faut pour cela pouvoir évaluer des probabilités de défaillance, c'est-à-dire la probabilité pour qu'une contrainte g soit violée :

$$P(g(\boldsymbol{x} + \boldsymbol{\chi}_x) > 0). \tag{5.41}$$

On est alors dans le cadre de calcul de mesures de type quantile (sous la forme d'un seuil probabiliste plus précisément).

Cette probabilité peut être estimée grâce à des tirages de Monte-Carlo, mais d'autres méthodes approchées plus rapides ont été développées pour ce type d'applications. Il s'agit

des méthodes FORM (« First Order Reliability Method ») et SORM (« Second Order Reliability Method »), qui sont par exemple décrites dans [HM00]. Ces méthodes estiment la probabilité qu'une solution $\boldsymbol{x}$ ne satisfasse pas la contrainte g à partir de la distance entre la solution et la contrainte (dans un espace normalisé). En effet, plus la contrainte est éloignée de la solution considérée et plus la probabilité de défaillance est faible. Cette distance est obtenue en recherchant le point de défaillance le plus probable $\boldsymbol{x}^*$, qui correspond au point le plus proche de $\boldsymbol{x}$ et qui soit situé sur la limite imposée par la contrainte g. Ce point est généralement trouvé à l'aide d'un algorithme de descente. La probabilité de défaillance pour la solution $\boldsymbol{x}$ est ensuite estimée en faisant une approximation linéaire (dans le cas de la méthode FORM) ou bien du second ordre (pour la méthode SORM) de la contrainte g en $\boldsymbol{x}^*$.

Il s'agit donc de méthodes relativement simples qui permettent d'estimer une mesure de seuil probabiliste. Leur approximation n'est toutefois valable que sous certaines hypothèses : les contraintes doivent par exemple être suffisamment lisses pour pouvoir être approchées par un modèle du premier ou second ordre, on considère qu'il n'y a qu'un seul point de défaillance le plus probable, etc. De nombreuses variantes de ces méthodes ont été proposées pour affiner ces résultats, mais les calculs se complexifient alors avec la généralité de l'approche.

5.8.5 Erreur sur l'estimation des mesures de robustesse

Nous venons de présenter plusieurs méthodes utilisées pour calculer des mesures de robustesse. L'estimation de la robustesse d'une solution se base généralement sur un modèle de substitution de la fonction objectif initiale, car un calcul direct serait bien trop coûteux. Comme tout modèle, cette représentation est entachée d'une certaine erreur : on peut donc se demander quel est l'impact de l'erreur de modélisation de l'objectif sur les valeurs de robustesse obtenues. La connaissance de cette erreur sur l'estimation des mesures de robustesse permettra par la suite de mettre en place des techniques de plans d'expériences adaptatifs dédiées à l'optimisation robuste comme nous le verrons dans la section suivante.

Qu'elles fassent appel ou non à un modèle de substitution, la plupart des méthodes de calcul ne fournissent en fait que des estimations des mesures de robustesse. L'erreur d'estimation en elle-même pourrait donc aussi être prise en compte. Nous n'allons pas nous y intéresser ici, car l'objectif est plutôt d'étudier les effets de l'utilisation d'un modèle de substitution pour les calculs de robustesse. De plus, cette erreur d'estimation est nulle lorsque la robustesse est évaluée de manière analytique : la seule erreur à considérer dans ce cas est l'erreur liée au modèle. C'est notamment sur cette approche que nous allons nous pencher.

Nous détaillons ci-dessous deux méthodes permettant d'estimer l'erreur due au modèle de substitution lors de l'évaluation d'une mesure de robustesse ρ. Le but est de calculer la variance de l'estimateur de cette mesure pour donner un intervalle de confiance à ses estimations. Les deux approches se basent sur une description du modèle $\hat{f}$ sous la forme d'un processus stochastique $\hat{F}$ qui fournit une estimation de la variance de ses prédictions. Il peut être obtenu directement dans le cas d'un modèle de krigeage ou bien par bootstrap (voir la section 3.7.2). Une première possibilité est d'appliquer la méthode générale de Monte-Carlo, alors que la seconde est plus spécifiquement liée au calcul analytique de la robustesse sur un modèle de krigeage.

Estimation de l'erreur par Monte-Carlo

L'approche la plus simple et la plus évidente pour évaluer l'erreur commise sur l'estimation de la robustesse d'une solution $\boldsymbol{x}$ du fait de l'utilisation d'un modèle de substitution $\hat{f}$ est d'effectuer des tirages de Monte-Carlo.

Le processus stochastique $\hat{F}$ qui décrit les incertitudes du modèle $\hat{f}$ est en effet un modèle aléatoire dont on peut tirer différentes réalisations $\hat{f}_i$. Dans le cas d'un modèle de krigeage, ces réalisations sont obtenues en faisant des simulations de processus gaussien à partir des paramètres du modèle (voir la section 3.4.3). Si par contre l'erreur du modèle est évaluée par bootstrap, les modèles $\hat{f}_i$ obtenus avec cette méthode peuvent être réutilisés directement (cela revient alors à estimer l'erreur sur la mesure de robustesse par bootstrap). L'évaluation de la robustesse de $\boldsymbol{x}$ sur chacun des modèles $\hat{f}_i$ obtenus donne un ensemble d'échantillons de Monte-Carlo $\rho(\hat{f}_i(\boldsymbol{x}+\boldsymbol{\chi}_x))$ de la mesure de robustesse. Ces échantillons permettent d'approcher la distribution de l'estimateur de $\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x))$ et donc de déterminer sa variance et son intervalle de confiance.

Le tirage de réalisations possibles de $\hat{f}$ en fonction de son incertitude permet ainsi d'évaluer l'erreur commise sur une mesure de robustesse basée sur ce modèle. L'inconvénient de cette méthode est qu'il est nécessaire effectuer un grand nombre de tirages pour obtenir une estimation fiable, et comme les échantillons sont ici des modèles de substitution complets, cela peut avoir un coût relativement important.

Calcul analytique de l'erreur

Lorsque la robustesse est évaluée de manière analytique sur un modèle de krigeage, il est aussi possible de calculer analytiquement l'erreur induite par le modèle sur la mesure de robustesse. Le krigeage est en effet défini comme un processus gaussien $\hat{F}$ doté d'une estimation de son incertitude. Nous avons présenté plus haut le calcul des mesures d'espérance et de variance sur un modèle de krigeage. La robustesse ρ y était évaluée à partir de la valeur moyenne $\hat{f}$ du processus $\hat{F}$. En se replaçant véritablement dans le cadre bayésien défini par le krigeage, on peut aussi calculer ces mesures directement sur $\hat{F}$. Ce sont alors des variables aléatoires $\rho(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))$ dont on peut estimer la moyenne et la variance. Le détail des calculs abordés ici est présenté en annexe.

Pour plus de clarté, nous noterons « E_χ » l'espérance par rapport aux incertitudes sur les variables d'entrée (utilisée dans les calculs de robustesse), et « E » représentera l'espérance relative à la variabilité du modèle stochastique $\hat{F}$. L'opérateur de variance sera indicé de la même manière.

Pour la mesure de l'espérance, l'équation (5.33) correspond ainsi au calcul de :

$$\mathrm{E}_\chi[\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)] = \mathrm{E}_\chi[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] = \mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]], \tag{5.42}$$

car les deux opérateurs d'espérance peuvent être intervertis. Le calcul de la valeur moyenne de la mesure d'espérance sur $\hat{F}$ donne donc le même résultat que le calcul de l'espérance sur $\hat{f}$. On peut aussi exprimer la variance $\mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)])$ de cette mesure de robustesse.

Pour la mesure de la variance (équation (5.34)), on a par contre :

$$\begin{aligned}\mathrm{Var}_\chi(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)) &= \mathrm{Var}_\chi(\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) \\ &= \mathrm{E}[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] + \mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) - \mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))].\end{aligned} \tag{5.43}$$

Sauf cas particulier, la valeur moyenne de la variance sur $\hat{F}$ n'est donc pas égale à la valeur obtenue avec $\hat{f}$ car les opérateurs de variance ne sont pas interchangeables. Le calcul de cette mesure de robustesse sur $\hat{F}$ demande un peu plus de travail. On peut aussi évaluer sa variance $\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))$.

Les calculs analytiques pour les mesures de l'espérance, de la variance ainsi que de l'agrégation espérance/écart-type sur un modèle de krigeage ordinaire doté d'une fonction de corrélation gaussienne sont détaillés en annexe pour des incertitudes d'entrée gaussiennes. Nous nous sommes basés ici sur les résultats présentés dans [ALC06], mais d'autres calculs similaires sont donnés dans [OO04, Vil08, JLR12]. À partir de l'estimation d'erreur fournie par le modèle de krigeage, on peut donc évaluer analytiquement l'erreur commise sur l'évaluation des mesures de robustesse et calculer leur intervalle de confiance.

La complexité des calculs d'espérance et de variance est liée au nombre m de coefficients du modèle de substitution. Le coût de l'évaluation de $\mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]$ varie ainsi en fonction de m, alors que celui de $\mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)])$ et de $\mathrm{E}[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))]$ varie avec m^2. Le calcul de $\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))$ est quant à lui plus complexe car son coût est fonction de m^4.

5.9 Plans d'expériences adaptatifs pour l'optimisation robuste

La construction adaptative de plans d'expériences est employée en optimisation déterministe classique afin d'améliorer les modèles de substitution dans les zones d'intérêt (voir la section 4.6.3). Des techniques similaires peuvent être mises en œuvre dans le cadre de l'optimisation robuste grâce aux estimations d'erreur sur les mesures de robustesse présentées ci-dessus. La connaissance de ces erreurs permet en effet de guider la sélection de nouveaux échantillons à ajouter au plan d'expériences dans le but de mieux prédire la robustesse des solutions. [ALC06] utilisent par exemple cette information pour réduire l'espace de recherche dans lequel les échantillons sont choisis. Le domaine de l'optimisation robuste n'étant encore pas totalement mature, les études décrivant des méthodes de plans d'expériences adaptatifs dédiées sont relativement rares et récentes.

La méthode la plus populaire de construction adaptative de plans d'expériences pour l'optimisation déterministe mono-objectif est l'algorithme EGO [JSW98], basé sur le critère EI d'amélioration espérée. Des auteurs ont donc proposé d'étendre ce critère à l'optimisation robuste :

$$\mathrm{EI}(\rho(\hat{F})) = \underset{\boldsymbol{x}\in\boldsymbol{X}}{\mathrm{argmax}}\ \mathrm{E}[\max(0, \rho_{\min} - \rho(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))]. \tag{5.44}$$

L'article [LSN04] développe par exemple un algorithme basé sur ce critère pour résoudre un problème d'optimisation robuste où l'objectif est de minimiser l'espérance ou la variance d'une fonction représentée à l'aide d'un modèle de krigeage (cet algorithme peut aussi prendre en compte une contrainte en multipliant EI par la probabilité que la contrainte soit respectée au point considéré, comme suggéré dans [Sch97]).

L'utilisation du critère EI nécessite de définir la valeur de référence $\rho_{\min}$ par rapport à laquelle on évalue l'amélioration. En optimisation déterministe classique, on prend généralement la plus petite valeur connue de l'objectif, c'est-à-dire celle du meilleur point évalué jusqu'alors sur la fonction de référence. La différence est qu'ici l'objectif est une mesure

de robustesse estimée à partir d'un modèle de cette fonction de référence. Si en optimisation classique il suffit d'évaluer un échantillon sur la fonction de référence pour connaître parfaitement la performance d'une solution, pour connaître la robustesse d'une solution il n'est pas suffisant d'avoir la valeur de sa performance ponctuelle : il faut aussi connaître la performance des points dans le voisinage défini par les incertitudes. L'objectif de robustesse ne peut donc pas être connu avec certitude. Comme le font remarquer [WSN00, Jur07], $\rho_{\min}$ peut donc être considérée comme une variable aléatoire (sa moyenne et sa variance étant fournies par le modèle). Le point minimal qui sert à déterminer la valeur de $\rho_{\min}$ est tout de même généralement choisi parmi les points du plan d'expériences.

Au delà de la nouvelle définition de l'amélioration espérée que nous venons de décrire, il existe une autre différence par rapport à l'optimisation déterministe classique : on ne peut pas directement ajouter au plan d'expériences une solution intéressante localisée à l'aide de ce critère. EI fournit en effet un point repéré par des valeurs nominales uniquement, sans considération des paramètres incertains. Les incertitudes sont masquées par la mesure de robustesse ρ qui ne prend en entrée que des valeurs nominales $\boldsymbol{x}$ pour analyser ensuite en interne leur voisinage défini par les incertitudes $\boldsymbol{\chi}_x$. Si on ajoute au plan d'expériences la solution trouvée grâce à EI, on n'évaluera donc jamais la fonction de référence en dehors de la valeur nominale des paramètres environnementaux par exemple. La recherche de points améliorants se fait donc généralement en deux temps : on recherche tout d'abord dans l'espace des valeurs nominales le point qui présente la meilleure espérance d'amélioration de la robustesse (critère EI), puis on cherche ensuite dans l'espace des incertitudes le point qui va permettre d'améliorer l'estimation de la robustesse du point précédent. Plusieurs stratégies existent pour ce second choix. [LSN04] proposent ainsi de prendre le point le plus éloigné possible des autres échantillons d'apprentissage, [Jur07] choisit le point du voisinage qui possède la plus grande erreur de modélisation (en la pondérant par la probabilité du point vis-à-vis des incertitudes du problème), et [WSN00, JLR12] recherchent le point qui, une fois ajouté au plan d'expériences, va minimiser l'erreur sur la robustesse du point nominal sélectionné dans la première étape.

L'extension d'algorithmes de type EGO à l'optimisation robuste permet donc de trouver efficacement des solutions optimales en choisissant intelligemment les points à évaluer sur la fonction de référence. Ces méthodes sont par contre généralement restreintes à l'utilisation de modèles de krigeage et aux mesures de l'espérance et de la variance, du fait des facilités de calcul que cela apporte. Le cas des plans d'expériences adaptatifs pour l'optimisation robuste multiobjectifs est à notre connaissance pratiquement inexistant dans la littérature.

Chapitre 6

Optimisation multidisciplinaire

Sommaire

6.1 Optimisation multidisciplinaire 127
6.2 Méthodes d'optimisation multidisciplinaire 129
6.2.1 Méthodes mononiveau 129
6.2.2 Méthodes multiniveaux 130
6.3 Modèles de substitution en optimisation multidisciplinaire 132
6.4 Incertitudes en optimisation multidisciplinaire 133
6.4.1 Propagation d'incertitudes 133
6.4.2 Optimisation multidisciplinaire robuste 135

Le deuxième cas d'application que nous allons traiter dans cette étude (voir le chapitre 2) est un problème d'optimisation multidisciplinaire. Ce domaine étudie l'optimisation de systèmes qui font intervenir plusieurs disciplines interdépendantes. Il s'agit généralement de systèmes complexes pour lesquels les approches classiques sont inapplicables. Elles se heurtent en effet à la grande taille de ces systèmes, dont l'évaluation globale est le plus souvent extrêmement coûteuse (voire parfois impossible directement). Le fait de les décomposer en différents sous-systèmes permet donc de rendre leur optimisation réalisable. Nous présentons dans ce chapitre diverses méthodes existantes d'optimisation multidisciplinaire, ainsi que l'utilisation de modèles de substitution pour résoudre ce type de problèmes. Comme dans le cas de l'optimisation « monodisciplinaire », nous allons aussi nous intéresser à la prise en compte d'incertitudes dans le problème. La propagation d'incertitudes au sein des systèmes multidisciplinaires ainsi que l'optimisation multidisciplinaire robuste sont décrites dans la dernière partie de ce chapitre.

6.1 Optimisation multidisciplinaire

L'optimisation multidisciplinaire (OMD) est un pan de l'optimisation qui s'intéresse aux systèmes complexes constitués de plusieurs sous-systèmes interdépendants. En effet, nous nous sommes situés jusqu'à maintenant dans le cadre de l'optimisation monodisciplinaire où l'ensemble du système à optimiser est considéré comme une unique boîte

noire. Mais certains systèmes sont souvent bien trop importants pour pouvoir être traités ainsi. Sur ces systèmes, toute approche directe est difficile voire impossible, et il devient nécessaire de les voir comme des « boîtes grises », c'est-à-dire de décomposer la boite noire en différents constituants. On décrit donc un système comme un ensemble de sous-systèmes (aussi appelés « disciplines ») interconnectés.

Les sous-systèmes peuvent être distribués dans différents lieux, ou gérés par différents départements d'une entreprise par exemple, ce qui rend alors les approches classiques inapplicables. Des méthodes spécifiques ont donc été développées pour traiter ce type de problèmes. La difficulté de l'optimisation multidisciplinaire réside dans la prise en compte des différentes relations entre les sous-systèmes : on ne souhaite pas en effet optimiser chacune des disciplines une par une, mais bien l'ensemble du système global, c'est-à-dire les disciplines avec leurs interdépendances. Les systèmes complexes impliquent de plus des domaines d'expertise variés (comme la mécanique des structures, la mécanique des fluides, l'énergétique, la thermodynamique, etc.) avec une culture et un vocabulaire différents pour les spécialistes de chaque discipline, qui n'ont qu'une connaissance très partielle des autres domaines et des problèmes et contraintes rencontrés. Le but recherché en optimisation multidisciplinaire est donc de faciliter les échanges entre les disciplines afin qu'elles convergent ensemble vers une solution optimale globale.

Les problématiques multidisciplinaires sont très répandues dans l'industrie, notamment dans le secteur aéronautique et spatial. [SSH97] présente une vue d'ensemble du domaine et de ses récents progrès, et [Bar98] détaille plusieurs exemples de problèmes concrets qu'on peut rencontrer.

Un exemple simple de système multidisciplinaire comportant deux disciplines seulement est représenté sur la figure 6.1. Le système possède différentes variables d'entrées, qui peuvent être des variables de décision $\boldsymbol{d}$ ou bien des paramètres environnementaux $\boldsymbol{e}$ fixés. Les disciplines communiquent entre elles par des variables de couplage $\boldsymbol{l}$, et on retrouve en sortie les objectifs $\boldsymbol{f}$ et les contraintes $\boldsymbol{g}$ du problème. Dans le cadre de l'optimisation multidisciplinaire, les variables d'entrées sont aussi parfois différenciées en variables locales (pour une variable qui n'affecte qu'une seule discipline) et variables partagées (pour les autres variables). Dans notre exemple, $\boldsymbol{e}$ serait ainsi un ensemble de variables locales à la Discipline 1 et $\boldsymbol{d}$ un ensemble de variables partagées par les deux disciplines.

Certaines entrées de disciplines proviennent de sorties d'autres disciplines. Ces liens interdisciplinaires sont représentés par les variables de couplage $\boldsymbol{l}$ (parfois appelées variables d'interaction). La modification de la sortie de l'une des disciplines peut donc impacter les autres par l'intermédiaire de ces variables, ce qui va altérer les sorties des autres disciplines qui pourront à leur tour avoir un effet sur la discipline initiale. Selon l'influence de ces interactions entre les disciplines et le nombre de variables de couplage dans le système, on parle de couplage faible ou fort entre les disciplines. Un couplage faible traduit une certaine indépendance entre les disciplines, alors qu'un couplage fort indique que la moindre modification au sein d'une discipline aura un impact important sur l'ensemble des autres sous-systèmes. Le problème de l'analyse multidisciplinaire est de parvenir à un équilibre entre toutes les disciplines en résolvant les différents bouclages présents (c'est-à-dire en trouvant des valeurs stables pour les variables de couplage $\boldsymbol{l}$ qui conviennent à toutes les disciplines). Plusieurs méthodes existent pour faire converger ces variables de couplage. Elles peuvent être gérées par le système lui-même, ou bien être intégrées à la formulation du problème d'optimisation. Les différentes méthodes d'optimisation multidisciplinaire sont présentées dans la section suivante.

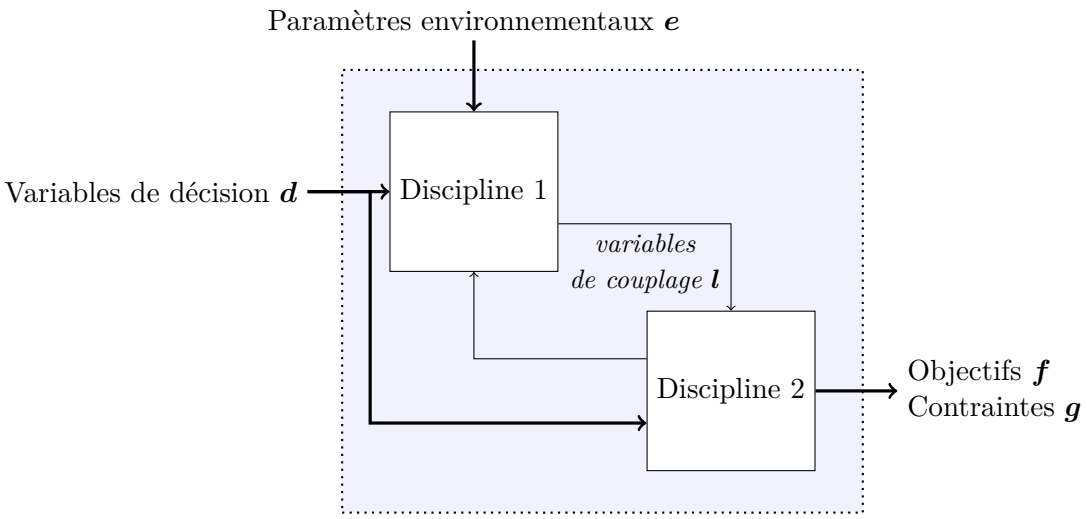


FIGURE 6.1 – Un exemple simple de système multidisciplinaire.

6.2 Méthodes d'optimisation multidisciplinaire

Les systèmes multidisciplinaires ne peuvent pas être traités avec les approches classiques du fait de leurs particularités. Des méthodes spécifiques ont donc été développées. On distingue deux approches générales pour la résolution de problèmes d'optimisation multidisciplinaire [TM06] : d'un côté la stratégie mononiveau dans laquelle le système est optimisé d'un seul tenant par une optimisation globale, et de l'autre la stratégie multiniveaux où l'optimisation se déroule aussi plus localement au niveau des disciplines. Nous présentons ci-dessous les principales formulations rencontrées dans chacune de ces stratégies (voir aussi [Cle09] pour une présentation générale des différentes formulations d'optimisation multidisciplinaire).

6.2.1 Méthodes mononiveau

Dans les méthodes mononiveau, les choix de conception sont réalisés au niveau global. Les disciplines permettent quant à elles de modéliser les différents phénomènes physiques qui entrent en jeu dans le système. Les trois principales formulations mononiveau sont présentées dans [CDF+94] : il s'agit des formulations MDF (« Multi Discipline Feasible »), AAO (pour « All At Once ») et IDF (« Individual Discipline Feasible »). Ces méthodes sont directes et faciles à implémenter, mais elles sont toutefois restreintes aux problèmes de petite taille car elles impliquent une optimisation globale du système entier.

MDF

La formulation « Multi Discipline Feasible » est la méthode la plus classique et la plus répandue dans l'industrie. L'optimisation des variables de décision est réalisée à un niveau global, le système tout entier étant considéré comme une boîte noire. Le couplage entre les disciplines est résolu par une analyse multidisciplinaire interne au système à chaque évaluation demandée par l'optimiseur global. C'est donc le système lui-même qui assure en interne la convergence des variables de couplage. L'avantage de ce procédé est

que la faisabilité du système est garantie durant toute l'optimisation (d'où le nom de cette méthode). L'analyse couplée effectuée à chaque itération peut toutefois rendre cette méthode assez coûteuse car elle ne tire pas parti de la décomposition en sous-systèmes.

IDF

Afin de ne pas multiplier le nombre d'analyses couplées, la formulation « Individual Discipline Feasible » traite les variables de couplage de manière différente. Les liens directs entre les disciplines sont en effet remplacés par des contraintes d'égalités dans le problème d'optimisation global. Ces contraintes spécifient que les différentes variables de couplage doivent avoir la même valeur en entrée et en sortie des disciplines. Il y a ainsi autant de contraintes d'égalités que de variables de couplage. C'est donc l'optimiseur global qui contrôle les communications interdisciplinaires et qui garantit la cohésion entre les disciplines. Le fait de ne pas réaliser d'analyse couplée en interne permet de gérer le cas de configurations impossibles pour lesquelles on ne peut pas assurer de cohérence physique du système et donc trouver de valeur commune pour les variables de couplage. L'optimiseur n'aura aucun problème avec ce genre de configurations, alors que l'approche MDF pourrait rester bloquée dans de vaines tentatives de convergence interne. Par contre, si le nombre de variables de couplage est grand il faut traiter un grand nombre de contraintes et l'optimisation devient plus complexe. La faisabilité globale du système n'est ici garantie qu'à la convergence, mais la faisabilité locale de chaque discipline est maintenue en permanence (car les disciplines gèrent elles-mêmes leurs contraintes locales). On ne perd ainsi pas de temps à assurer la faisabilité globale lorsqu'on est loin de l'optimum, mais il est nécessaire d'attendre la fin de l'optimisation pour obtenir une solution réalisable.

AAO

Comme dans la formulation IDF, la cohésion des variables de conception dans la formulation « All At Once » est assurée par des contraintes d'égalité globales. La différence est qu'ici les sorties disciplinaires sont aussi déterminées au niveau global. Les équations régissant le comportement des disciplines sont en effet intégrées au problème général sous la forme de contraintes d'égalité. Cela permet de gérer directement l'ensemble du problème afin de converger plus rapidement vers une solution optimale. La totalité du système se retrouve explicitée au niveau global, qui assure lui-même le calcul de chaque discipline et la résolution des couplages, d'où le nom de cette méthode. Cette formulation peut être très efficace pour la résolution de certains problèmes, mais sa convergence n'est pas toujours assurée. Dans le cas où le nombre de disciplines et de variables de couplage est important et lorsque les équations disciplinaires sont fortement non linéaires, le problème devient rapidement complexe. Ici aussi, la faisabilité du système n'est garantie qu'à la convergence.

6.2.2 Méthodes multiniveaux

Au delà des méthodes mononiveau qui sont restreintes à des problèmes de petite taille, des méthodes plus élaborées ont été développées afin de mieux tirer parti du découpage en disciplines et de pouvoir ainsi traiter des systèmes de taille plus importante. Si les méthodes mononiveau utilisent un seul optimiseur global, les méthodes multiniveaux mettent quant à elles en place des boucles d'optimisation à plusieurs étages différents. On distingue

le plus souvent deux niveaux : l'un global comme précédemment, et l'autre plus local relatif aux disciplines qui prennent à leur charge une partie de l'optimisation. La gestion de certaines variables est ainsi décentralisée vers les disciplines, qui optimisent elles-mêmes leurs variables locales par exemple. Ces méthodes permettent à chaque discipline de travailler indépendamment et de participer aux choix de conception. La cohérence globale du système est gérée par l'optimiseur global qui fédère le tout et s'assure qu'on se dirige vers un optimum global. On parle alors de conception distribuée.

La difficulté des approches multiniveaux est d'assurer la convergence globale de l'optimisation. Le fait de déléguer certaines optimisations aux disciplines rend en effet le contrôle global du système beaucoup plus complexe. Dans le cas de systèmes de taille modeste, on préfère parfois utiliser une approche mononiveau plus simple et qui permet de garantir la convergence de l'optimiseur. Mais pour des systèmes plus importants la décentralisation est une nécessité, et différentes stratégies ont donc été proposées pour tenter de converger vers une « bonne solution ». Le principe des trois principales formulations multiniveaux utilisées aujourd'hui est présenté ci-dessous.

CO

Au sein de la formulation « Collaborative Optimisation » [AL02], l'optimisation se fait sur deux niveaux et la gestion des couplages entre disciplines est réalisée par des contraintes d'égalité au niveau global. Cette formulation reproduit schématiquement le fonctionnement d'une entreprise, où une entité générale fournit des consignes aux disciplines qui essayent de les respecter au mieux. Des consignes sur les valeurs des variables de couplage sont ainsi envoyées par l'optimiseur global aux disciplines. Celles-ci cherchent alors les valeurs des paramètres de décision pour répondre au mieux à ces consignes (elles réalisent une minimisation de l'erreur par rapport aux consignes) tout en respectant leurs contraintes locales. La faisabilité de chaque discipline est donc assurée en permanence. Le système global tente ensuite de minimiser les objectifs généraux en respectant les contraintes globales et les contraintes d'égalité pour les variables de couplage et les variables partagées en réitérant cet envoi de consignes jusqu'à la convergence.

BLISS

La formulation « Bi-Level Integrated System Synthesis » [SSAS98] est elle aussi une méthode à deux niveaux, mais dans laquelle les variables de couplages sont gérées en interne par une analyse multidisciplinaire. L'analyse couplée du système pour une solution initiale donnée permet d'évaluer les fonctions objectifs et les contraintes du problème global en ce point. Des approximations linéaires de ces fonctions sont alors construites à partir de leur gradient, et les variables locales aux disciplines sont optimisées par rapport à ces approximations. On fait de même au niveau global pour ajuster les variables globales. Le calcul est ensuite poursuivi de manière itérative à partir de la nouvelle solution obtenue, jusqu'à converger vers une solution intéressante.

CSSO

La formulation « Concurrent Sub-Space Optimisation » [Sob89] est une approche décentralisée dans laquelle chaque discipline tente d'optimiser le problème global en utilisant des modèles approchés des autres disciplines. Dans cette méthode, les paramètres

de conception sont distribués entre les disciplines : chaque discipline va ainsi optimiser certains paramètres du problème. La responsabilité de la satisfaction des contraintes est aussi répartie entre les différentes disciplines. Pour mener à bien leur optimisation, les disciplines se basent sur des approximations du second ordre construites à partir des dérivées (ou bien plus généralement des modèles polynomiaux selon les implémentations) qui leur permettent de modéliser de manière locale le comportement des autres disciplines du système. Ces modèles approchés sont améliorés progressivement au cours des itérations de l'optimisation. Au niveau global, l'optimiseur utilise aussi des approximations des disciplines pour guider le système vers un optimum global.

6.3 Modèles de substitution en optimisation multidisciplinaire

L'optimisation d'un système multidisciplinaire est un processus complexe qui peut très rapidement devenir coûteux dès lors que certaines disciplines font appel à des simulations assez lourdes et peu réactives. Pour accélérer la résolution d'un problème, on peut alors s'aider de modèles de substitution plus rapides qui vont guider l'optimisation de manière efficace. Toutes les techniques de modélisation avancée peuvent être employées dans ce cadre. L'article [STBV08] rappelle l'historique de l'utilisation de modèles de substitution en optimisation multidisciplinaire et fait le point sur les nouvelles techniques mises en œuvre aujourd'hui (comme la construction de plans d'expériences, les modèles multi-fidélité ou encore l'utilisation de mélanges d'experts).

Une première utilisation possible des modèles de substitution est de remplacer une discipline trop coûteuse. [PLRRB09] présente ainsi une résolution de problème couplé grâce à des modèles de type POD (« Proper Orthogonal Decomposition ») pour représenter chacune des disciplines d'un système. Les POD sont des modèles spécifiques partiellement basés sur la connaissance de la physique particulière de la discipline, mais n'importe quel autre type de modèle pourrait être utilisé pareillement. Les modèles de substitution sont construits et validés a priori, puis on les intègre au système à la place de la discipline qu'ils représentent. Ces modèles doivent être précis sur l'ensemble du domaine de variation des entrées de la discipline, car on ne sait pas au préalable la zone qui sera parcourue au cours de l'optimisation. On peut par exemple utiliser les techniques de plans d'expériences adaptatifs pour l'amélioration globale de modèles (voir la section 3.8.2) afin d'obtenir un modèle d'erreur restreinte. Une fois les disciplines coûteuses remplacées, on peut alors résoudre le problème d'optimisation multidisciplinaire de façon classique avec n'importe quelle formulation. L'avantage de ces modèles de substitution disciplinaires est qu'ils peuvent être réutilisés pour différentes optimisations incluant la même discipline, voire au sein de différents systèmes.

On peut encore construire directement des modèles de substitution des objectifs et des contraintes du problème afin de faciliter la recherche du point optimal, comme cela se fait classiquement en optimisation monodisciplinaire. Les modèles construits dans ce but peuvent bénéficier des techniques de plans d'expériences adaptatifs pour l'optimisation. Dans le cadre de formulations MDF, [SKMM98] et [Jur09] approchent par exemple les objectifs par des modèles polynomiaux, des modèles de krigeage ou bien des SVM (« Support Vector Machines »).

Les formulations multiniveaux ont aussi été adaptées à l'utilisation de modèles de sub-

stitution pour les rendre plus efficaces. L'article [ZTW09] présente ainsi une version de CO basée sur des modèles MLS (« Moving Least Squares »). La modélisation y intervient à la fois au niveau disciplinaire avec des modèles multifidélité, et au niveau global où l'optimisation est réalisée par une méthode de région de confiance. Une extension de la formulation BLISS (nommée BLISS 2000 [SSAPS03]) fait aussi utilisation d'approximations polynomiales pour guider les itérations de l'algorithme. [SBR96] présente enfin une adaptation de CSSO qui utilise des réseaux de neurones.

6.4 Incertitudes en optimisation multidisciplinaire

Tout comme les autres systèmes plus classiques, les systèmes multidisciplinaires peuvent être soumis à des incertitudes affectant leurs paramètres d'entrée. Une solution optimale choisie à partir des résultats d'une optimisation purement déterministe pourra alors voir ses performances se dégrader du fait de ces incertitudes. Dans un tel contexte, il est important de quantifier l'impact des paramètres incertains sur les solutions obtenues. Si des approches classiques de gestion d'incertitudes peuvent être appliquées à certains systèmes multidisciplinaires (en considérant le système complet sous la forme d'une boîte noire), elles sont généralement bien trop onéreuses pour être utilisées en pratique. Des méthodes spécifiques ont donc été développées afin de prendre en compte la décomposition en différents sous-systèmes. Nous présentons ainsi ci-dessous le principe de la propagation d'incertitudes au sein d'un système multidisciplinaire. La quantification des incertitudes et de leur impact sur le système permet par la suite de rechercher des optima robustes, c'est-à-dire des solutions relativement peu sensibles aux variations engendrées par les paramètres incertains.

6.4.1 Propagation d'incertitudes

Pour étudier l'effet des incertitudes sur un système multidisciplinaire, il faut réaliser une propagation d'incertitudes au sein du système. La propagation consiste à suivre l'évolution des incertitudes au travers de l'ensemble du système pour connaître leur effet final sur les objectifs et les contraintes du problème. Un exemple de système multidisciplinaire soumis à des incertitudes sur les paramètres environnementaux $\boldsymbol{e}$ est présenté sur la figure 6.2. Les incertitudes en entrée sont définies sous la forme de distributions de probabilité qui engendrent des distributions résultantes sur les sorties du système. Ces distributions en sortie ne sont initialement pas connues, et le but de la propagation d'incertitudes est donc de les estimer.

Le principe de la propagation d'incertitudes est simple (voir [DWC00] par exemple). Elle consiste à calculer l'influence des incertitudes sur les sorties d'une discipline donnée, puis à transmettre cette information aux autres disciplines qui feront de même. Les incertitudes vont ainsi se propager à travers tout le système. Pour mettre en place une propagation d'incertitudes, il faut donc que chaque discipline calcule la distribution de probabilité résultante sur ses sorties à partir de l'incertitude présente sur ses entrées. Cette incertitude de sortie peut ensuite être transmise aux autres disciplines qui prendront à leur tour ces données comme entrées. La propagation continue ainsi entre les différentes disciplines jusqu'à ce qu'un équilibre soit atteint dans le système. Au final, on obtient les distributions de l'incertitude sur les objectifs et les contraintes du problème. La connaissance des incertitudes en sortie pour une solution donnée permet d'évaluer sa robustesse

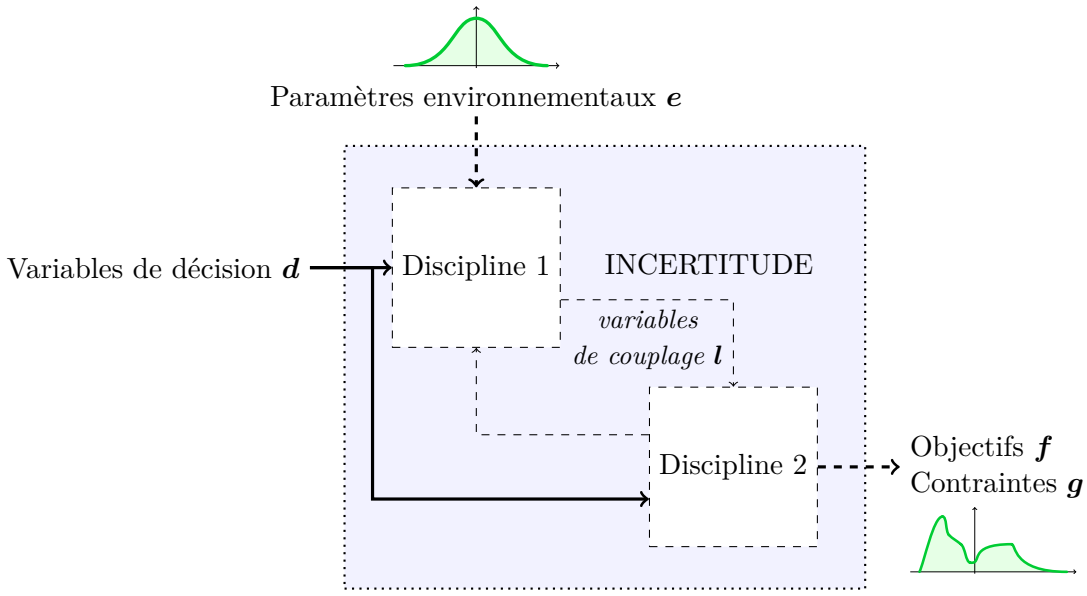


FIGURE 6.2 – Exemple de propagation d'incertitudes au sein d'un système multidisciplinaire simple. Les incertitudes proviennent des paramètres environnementaux et sont propagées jusqu'aux objectifs et aux contraintes du problème.

et les possibles variations que pourront subir sa performance et sa fiabilité.

La propagation d'incertitudes nécessite que les disciplines puissent échanger entre elles des informations sur l'incertitude affectant les variables de couplage $\boldsymbol{l}$. Dans l'idéal, il faudrait que chacune de ces variables soit représentée par une distribution de probabilité décrivant ses possibles fluctuations. Cela semble compliqué en pratique, et on transmet donc seulement certaines caractéristiques de cette distribution. Dans leurs études, les auteurs des articles [DC05] et [LCK+06] propagent ainsi uniquement la moyenne et la variance des distributions en supposant que les incertitudes sont gaussiennes, et [GRB+00] et [LA08] transmettent l'incertitude entre les disciplines sous la forme d'intervalles de variation. Dans ce cas, une variable de couplage initiale sera donc remplacée par deux nouvelles variables différentes. Il faut de plus que chaque discipline soit capable de traiter ces informations incertaines pour calculer les caractéristiques de ses distributions d'incertitude en sortie. Cela nécessite à chaque appel de réaliser plusieurs simulations de la discipline afin d'évaluer les variations de ses sorties (on peut bien évidemment se baser sur des modèles de substitution pour faciliter ces calculs). La propagation d'incertitudes au sein d'un système multidisciplinaire peut donc devenir très coûteuse selon la complexité de ces évaluations. L'équilibre général du système sera aussi plus long à atteindre du fait du plus grand nombre de variables de couplage présentes (qui implique un nombre plus important de bouclages à résoudre).

Comme expliqué dans [CAR+05] par exemple, la mise en œuvre d'une propagation d'incertitudes au sein d'un système multidisciplinaire déjà existant est relativement difficile. Elle nécessite en effet d'apporter de nombreuses modifications au système afin que les disciplines puissent traiter les incertitudes et échanger des informations incertaines les unes avec les autres. À titre d'illustration, les modifications à apporter au système de la figure 6.1 pour mettre en place une propagation d'incertitudes sont représentées sous

la forme de pointillés sur la figure 6.2 : l'ensemble des disciplines ainsi que les variables de couplage devront être adaptés. Il parait donc plus sage de prévoir la propagation des incertitudes lors de la phase de conception d'un nouveau système plutôt qu'a posteriori, afin d'implémenter directement les liens interdisciplinaires essentiels à la prise en compte des incertitudes ainsi que les traitements afférents à l'intérieur de chaque discipline.

La gestion des incertitudes dans un système multidisciplinaire est un procédé relativement complexe mais c'est le seul moyen de connaître avec précision certaines caractéristiques des distributions des incertitudes sur les objectifs et les contraintes du problème. On peut alors déterminer la robustesse des solutions étudiées, et incorporer ce calcul dans une boucle d'optimisation afin de rechercher des solutions robustes vis-à-vis des incertitudes présentes.

6.4.2 Optimisation multidisciplinaire robuste

Une présentation générale de l'optimisation robuste dans le contexte des systèmes multidisciplinaires est donnée dans [ASCM06]. La définition d'un problème d'optimisation multidisciplinaire robuste est similaire au cas monodisciplinaire décrit dans le chapitre 5 et peut se baser sur les mêmes mesures de robustesse. La seule différence réside dans les méthodes de calcul de ces mesures, qui peuvent ici s'appuyer sur la propagation d'incertitudes présentée ci-dessus.

Plusieurs articles détaillent l'application de la propagation d'incertitudes à la résolution de problèmes d'optimisation multidisciplinaire robuste. [GRB$^+$00] présentent par exemple une méthode d'optimisation robuste basée sur la mesure du pire cas, et [LA08] réalisent une optimisation robuste en imposant une contrainte de variabilité maximale autorisée pour les objectifs. Dans les deux cas la robustesse est évaluée en propageant des intervalles d'incertitude dans le problème. Une méthode de propagation approchée est aussi développée dans [GRP06] pour réaliser une optimisation fiabiliste. La méthode est appliquée à l'optimisation d'un design d'avion en considérant des mesures de type pire cas sur les contraintes.

Pour des problèmes de taille plus réduite et qui peuvent être résolus par une approche mononiveau, la propagation d'incertitudes n'est pas nécessairement obligatoire. Il est en effet possible de construire directement des modèles de substitution des objectifs et des contraintes et de calculer la robustesse des solutions sur ces modèles. [AR04] présentent ainsi une optimisation fiabiliste d'un problème simple de mécanique des structures en calculant des probabilités de défaillance grâce à des approximations polynomiales des contraintes. Les travaux de [KWGS02] concernent quant à eux l'optimisation robuste d'un bateau soumis à des incertitudes qui affectent plusieurs de ses disciplines. L'objectif de robustesse prend la forme d'une agrégation des mesures d'espérance et de variance, estimées à l'aide d'un modèle de krigeage de l'objectif initial (des mesures de robustesse similaires sont utilisées pour les contraintes du problème). Les techniques d'optimisation robuste par modèles globaux restent toutefois limitées à des systèmes simples et relativement peu coûteux pour lesquels les objectifs ont des comportements suffisamment lisses pour pouvoir être modélisés correctement.

Certaines approches d'optimisation multidisciplinaire déterministe classique se sont aussi inspirées des techniques utilisées en optimisation robuste. C'est le cas de la méthode MORDACE (« Multidisciplinary Optimisation and Robust Design Approaches applied to Concurrent Engineering ») présentée dans [GBM04]. Dans cette approche, chaque disci-

pline réalise sa propre optimisation locale en utilisant des modèles simplifiés pour représenter les autres disciplines. Un certain degré d'incertitude sur les variables partagées est pris en compte durant cette optimisation, afin de prévenir de possibles modifications dues aux choix réalisés simultanément par les autres disciplines. Les interactions avec les autres sous-systèmes sont en effet considérées comme une possible source de perte de performance. Chaque discipline cherche donc à minimiser la variance des fonctions objectifs par rapport aux variables partagées. On espère ainsi parvenir plus rapidement à une solution de compromis entre les différentes disciplines. Dans la même idée, [KHL01] utilise des formulations issues de l'optimisation robuste pour diminuer les effets des décisions globales sur les différentes disciplines. Cela permet d'augmenter la robustesse des sous-systèmes en les rendant moins dépendants les uns des autres. Ces approches ont pour but de faciliter la convergence des méthodes d'optimisation multidisciplinaire, mais il ne s'agit toutefois pas d'optimisation multidisciplinaire robuste à proprement parler car on ne considère pas d'incertitudes extérieures au système. Les formulations employées au sein des disciplines devraient toutefois permettre de se prémunir aussi contre de telles incertitudes, mais nous ne nous intéresserons pas à ces approches ici.

Troisième partie

Démarche

Chapitre 7

Méthodologie d'optimisation robuste

Sommaire

7.1 Identification des incertitudes 140
7.1.1 Identification des paramètres incertains 141
7.1.2 Modélisation de l'incertitude 141
7.1.3 Quantification de l'incertitude 142
7.1.4 Réduction des incertitudes 146
7.1.5 Définition du problème stochastique 146
7.2 Choix des mesures de robustesse et formulation du problème robuste . . 147
7.2.1 Choix des mesures de robustesse 147
7.2.2 Choix d'une formulation pour le problème d'optimisation robuste 149
7.3 Choix des stratégies de calcul de la robustesse 150
7.4 Exemple sur le cas test des deux barres 151
7.4.1 Évaluation directe de la robustesse sur les fonctions de référence 153
7.4.2 Modèles de substitution des fonctions de référence 154
7.4.3 Modèles de substitution des mesures de robustesse 157
7.5 Résolution du problème d'optimisation robuste 159
7.6 Conclusion 161

La synthèse de l'état de l'art réalisée au cours de cette étude nous a permis de définir une méthodologie pratique pour traiter des problèmes d'optimisation sous incertitude. Ce chapitre présente les différentes étapes à suivre pour poser un problème d'optimisation robuste et le résoudre.

La figure 7.1 montre le déroulement de la définition d'un problème d'optimisation robuste. Chacune des phases de ce processus est décrite dans la première partie du chapitre, avec l'aide d'expérimentations sur des exemples analytiques qui permettent de mieux appréhender la problématique de l'optimisation robuste. Le déroulement n'est en réalité pas nécessairement aussi linéaire que ce qui est représenté car tous les choix à effectuer sont liés les uns aux autres pour former un tout cohérent.

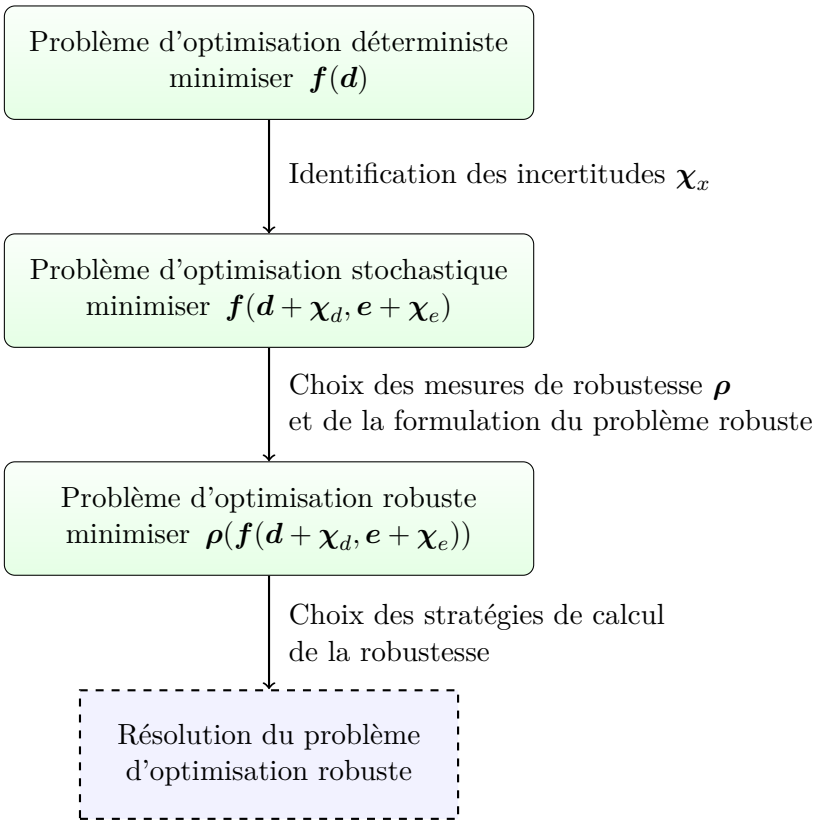


FIGURE 7.1 – Processus de définition d'un problème d'optimisation robuste.

La deuxième partie de ce chapitre se penche sur les stratégies de calcul des mesures de robustesse et de résolution des problèmes d'optimisation robuste grâce à des modèles de substitution. Nous les avons mises en application sur un cas test simple qui nous a permis de valider la plateforme d'expérimentation pour l'optimisation robuste par modèles de substitution que nous avons développée.

7.1 Identification des incertitudes

La définition d'un problème d'optimisation sous incertitude s'appuie souvent au départ sur un problème d'optimisation déterministe classique de la forme :

$$\underset{\boldsymbol{d}\in\boldsymbol{D}}{\text{minimiser}}\ \boldsymbol{f}(\boldsymbol{d}), \tag{7.1}$$

avec $\boldsymbol{d}$ les variables de décision et $\boldsymbol{f}$ les objectifs initiaux. La résolution de ce problème mène parfois à une solution optimale relativement sensible aux fluctuations qui peuvent apparaître du fait de paramètres incertains. Si on souhaite trouver une solution plus robuste, il faut dans un premier temps expliciter les incertitudes présentes dans le problème. Cela implique d'identifier les différentes sources d'incertitude puis de quantifier et de modéliser de manière appropriée les variations des paramètres incertains. La définition des incertitudes permettra alors de transformer le problème initial en un problème stochastique.

7.1.1 Identification des paramètres incertains

La première chose à faire lorsqu'on est face à un problème d'optimisation pour lequel on souhaite trouver des solutions robustes est de bien identifier les paramètres qui pourraient être soumis à des incertitudes. Le plus souvent, on sait déjà quelles sont les variables incertaines à prendre en compte car on a précisément remarqué une grande variabilité des solutions par rapport à celles-ci.

Comme présenté dans l'état de l'art, les incertitudes peuvent provenir de différentes sources (voir la section 5.1). La source la plus courante concerne les variables de décision $\boldsymbol{d}$: on souhaite par exemple tenir compte des tolérances de fabrication des machines utilisées pour réaliser le système à optimiser. Les incertitudes peuvent aussi affecter des paramètres environnementaux $\boldsymbol{e}$ sous-jacents qui ne sont pas forcément présents dans le problème d'optimisation déterministe initial. Il faut donc identifier ces paramètres supplémentaires qui devront être intégrés dans la formulation du problème.

Si la gestion d'incertitudes sur les variables de décision ne pose pas de problème particulier, cela peut nécessiter un peu plus de travail dans le cas des paramètres environnementaux. Pour pouvoir étudier l'impact de leurs variations sur le système, il faut en effet pouvoir manipuler ces paramètres comme on le souhaite sur les fonctions de référence. Il est donc nécessaire de les rendre contrôlables par l'optimiseur alors que ceux-ci n'étaient que de simples constantes dans le cadre de l'optimisation déterministe classique. La prise en compte d'incertitudes environnementales peut ainsi nécessiter quelques modifications dans la simulation numérique du système pour faire ressortir ces paramètres s'ils ne sont pas directement accessibles.

Lorsqu'on ne sait pas exactement quelles variables incertaines considérer, on peut partir de la liste complète des paramètres dont dépend la simulation du système et se demander pour chacun d'entre eux si une variabilité peut exister. Il semble évident que si on oublie d'identifier certains paramètres incertains les solutions ne seront pas robustes par rapport à ceux-ci et leur performance pourra donc toujours être affectée par leurs fluctuations. Si au contraire on ajoute trop de variables incertaines dotées d'une incertitude faible voire inexistante, cela n'aura pas d'impact sur la robustesse des solutions mais la résolution du problème deviendra plus ardue car on le complexifie inutilement. En règle générale, on préfère se concentrer sur quelques paramètres incertains uniquement, choisis parmi les plus influents sur les performances du système. Si nécessaire, on peut réaliser au préalable une étude de sensibilité des objectifs du problème par rapport aux différents paramètres pour faciliter leur sélection.

7.1.2 Modélisation de l'incertitude

Une fois que les variables incertaines du problème ont été identifiées, il faut dans un second temps modéliser leur incertitude. Nous considérons ici des représentations sous la forme d'intervalles de variation ou bien de distributions de probabilité, l'utilisation de la logique floue restant relativement lourde à mettre en œuvre même si elle offre une plus grande généralité.

La modélisation de l'incertitude doit être réalisée en fonction des données dont on dispose. Si seules des limites de variation sont connues pour un paramètre sans aucune information supplémentaire concernant la distribution de son incertitude, il vaut mieux représenter cette dernière par un simple intervalle plutôt que de supposer qu'elle décrit une loi uniforme par exemple. Définir une distribution sur un intervalle autorise en effet

l'utilisation de mesures spécifiques aux distributions comme l'espérance ou la variance, notions qui n'ont aucun sens dans le cadre des intervalles de variation.

La façon dont sont modélisées les incertitudes va influer sur les mesures de robustesse qui pourront être considérées par la suite, ainsi que sur les méthodes de calcul qui en découlent. On aura recours à des mesures et des méthodes éventuellement différentes dans le cas d'intervalles de variation et dans le cas de distributions de probabilité. Inversement, on peut aussi fixer la représentation des incertitudes en fonction des mesures de robustesse qu'on souhaite utiliser ou bien des méthodes de calcul qui peuvent être mises en œuvre. Il est par exemple inutile de s'efforcer de définir des distributions de probabilité sur les paramètres incertains pour ne considérer ensuite que le pire cas des objectifs, tout comme des intervalles de variation seront inappropriés en entrée d'un système qui ne peut calculer que des valeurs moyennes sur ses sorties. Le choix des mesures de robustesse et leur calcul (deux autres étapes de la définition d'un problème d'optimisation robuste que nous détaillerons par la suite) peuvent donc entrer en jeu dès la modélisation des incertitudes.

Choisir la modélisation appropriée pour représenter l'incertitude de chaque paramètre du problème a son importance. En effet, cela a un impact sur l'estimation de la robustesse des solutions : la valeur des mesures peut être modifiée selon le type de distribution de probabilité choisie. La figure 7.2 compare sur une fonction exemple f des distributions obtenues en sortie à partir d'une incertitude normale ou uniforme en entrée. On observe que les distributions en sortie sont légèrement différentes selon le choix de modélisation effectué, ce qui va se retrouver dans la mesure de la robustesse de chaque solution.

La figure 7.3 présente le tracé de l'espérance de cette même fonction dans les deux cas de modélisation de l'incertitude en entrée (loi normale ou bien uniforme). Si les deux courbes ont globalement le même comportement car le support des incertitudes reste relativement restreint par rapport aux variations de la fonction de référence f, on peut remarquer que la position de l'optimum n'est pas exactement la même dans les deux cas. Le choix d'une modélisation ou d'une autre va donc modifier les résultats de l'optimisation robuste qui sera réalisée par la suite. Il est primordial de représenter dès le départ les paramètres incertains avec une distribution d'incertitude proche de la réalité afin que les solutions robustes obtenues aient un sens.

7.1.3 Quantification de l'incertitude

De même que le type de modélisation des incertitudes a son importance, leur quantification joue un rôle évident : les solutions robustes seront différentes si l'ampleur des incertitudes est modifiée. Cette ampleur correspond par exemple à la taille des intervalles de variation ou bien à la variance des distributions gaussiennes définies sur les paramètres incertains. Une incertitude surestimée mènera à des solutions trop robustes et peu performantes, alors qu'une incertitude sous-évaluée pourra donner des solutions trop sensibles aux fluctuations. L'incertitude présente sur un paramètre donné doit donc être quantifiée au plus juste. Cela peut être effectué à partir de connaissances d'experts du domaine d'application, de spécifications techniques existantes ou encore de données expérimentales (par mesure des variations observées sur des systèmes réels).

Nous présentons ci-après deux exemples illustrant l'influence de la quantification des incertitudes sur les solutions d'un problème d'optimisation robuste : un cas simple en deux dimensions sur la fonction de Rosenbrock, et un problème plus complexe issu de la littérature.

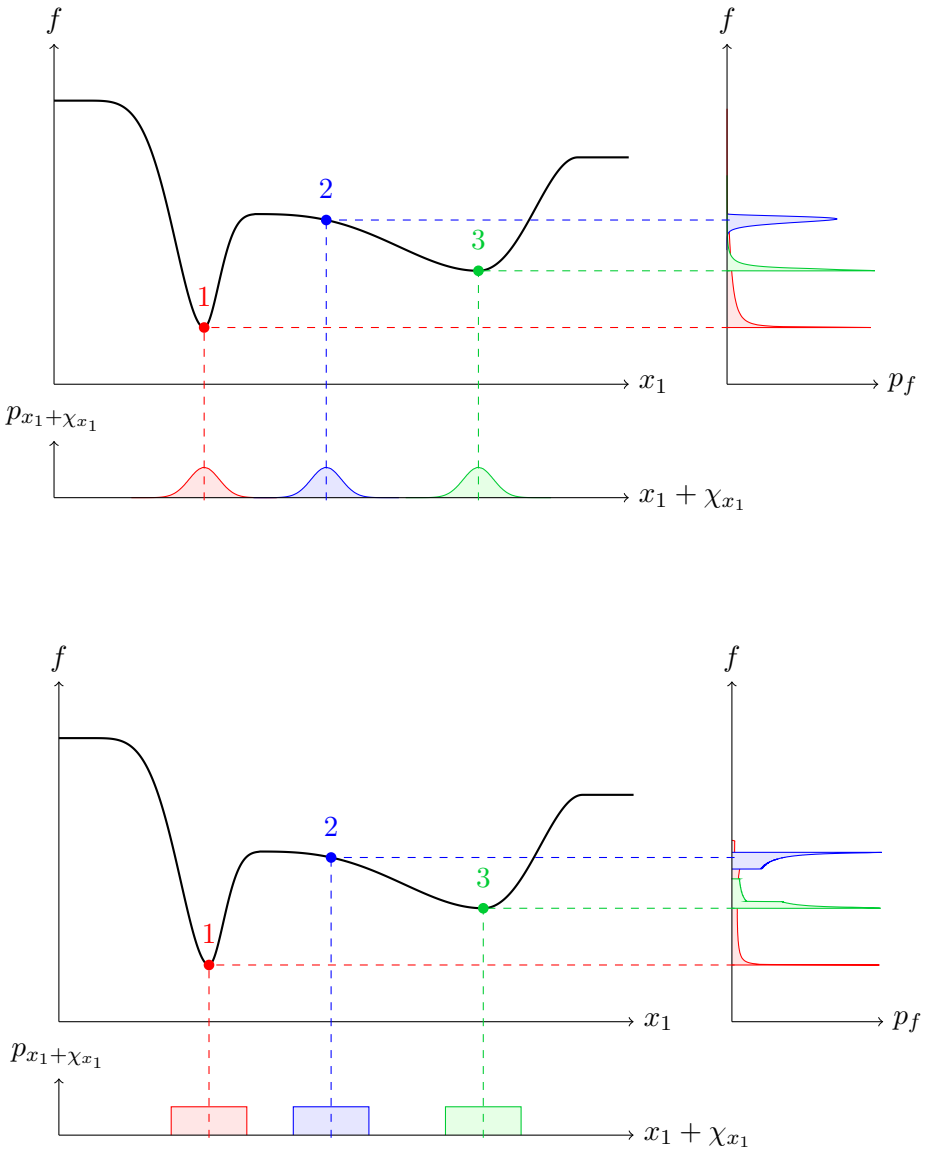


FIGURE 7.2 – Exemples de distributions obtenues en sortie d'une fonction f à partir d'une incertitude gaussienne (en haut) et uniforme (en bas) sur x_1. Les graphes des distributions en sortie ont été renversés afin de conserver le même axe des ordonnées que les représentations de f.

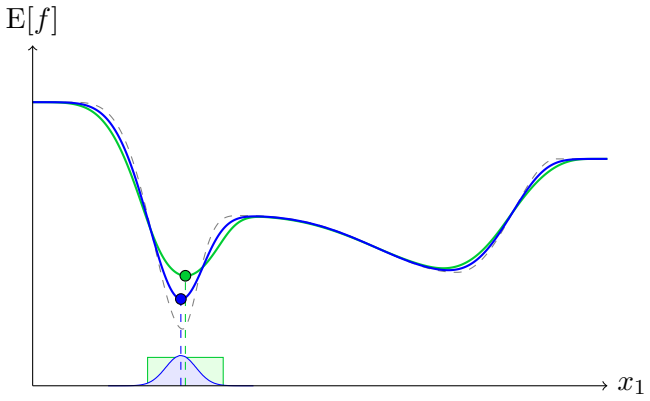


FIGURE 7.3 – Représentation de l'espérance (traits pleins) de la fonction f (en pointillés) dans le cas d'incertitudes d'entrée gaussienne (en bleu) et uniforme (en vert). La position de l'optimum est légèrement différente d'un cas à l'autre (points bleu et vert).

Exemple sur la fonction de Rosenbrock

La figure 7.4 présente les résultats d'optimisations robustes réalisées sur la fonction de Rosenbrock (définie dans l'équation (4.5)) avec une incertitude donnée par des intervalles de variation sur chacun des deux paramètres d'entrée x_1 et x_2. La robustesse est évaluée par la valeur du pire cas au sein de la région d'incertitude (représentée sous la forme de carrés). On observe que la position de l'optimum robuste change lorsque la taille des intervalles de variation est modifiée. Si la solution déterministe classique (qui correspond à des intervalles de taille nulle) est située au point (1,1), l'optimum robuste se déplace peu à peu le long de la vallée de la fonction et converge autour du point (0,0) pour de grandes tailles d'intervalles.

Cet exemple met en avant l'importance de la quantification des incertitudes sur les paramètres d'entrée afin d'obtenir des résultats en accord avec la réalité.

Exemple du problème DG2

Les effets d'une mauvaise quantification de l'incertitude peuvent aussi être visualisés sur l'exemple un peu plus complexe défini par le problème DG2 (issu de [DG06], voir l'équation (5.18)). Ce problème vise à minimiser les espérances ρ_1 et ρ_2 de deux objectifs f_1 et f_2 dans un espace d'entrée de dimension 5. Chacune des variables de décision x_i est dotée d'une incertitude modélisée par une loi uniforme sur un intervalle de 2δ de large pour la première variable x_1 et de 4δ de large pour les autres. On peut alors observer l'évolution du front de Pareto robuste pour différentes valeurs de $\delta \in \mathbb{R}^+$. Les résultats théoriques ont été présentés dans l'état de l'art (sur la figure 5.15).

Pour résoudre ce problème, nous avons utilisé l'algorithme NSGA-II [DPAM02] et calculé les mesures de robustesse grâce à 500 tirages de Monte-Carlo. Les résultats obtenus sont présentés sur la figure 7.5. Comme on peut le voir, on retrouve expérimentalement les fronts théoriques à peu de choses près. Quand il n'y a pas d'incertitude sur les paramètres d'entrée ($\delta = 0$), on obtient les solutions du front de Pareto déterministe classique. Mais lorsque la taille des intervalles de variation augmente le front de Pareto robuste évolue. La valeur de l'espérance des solutions optimales croît et la forme en elle-même du front

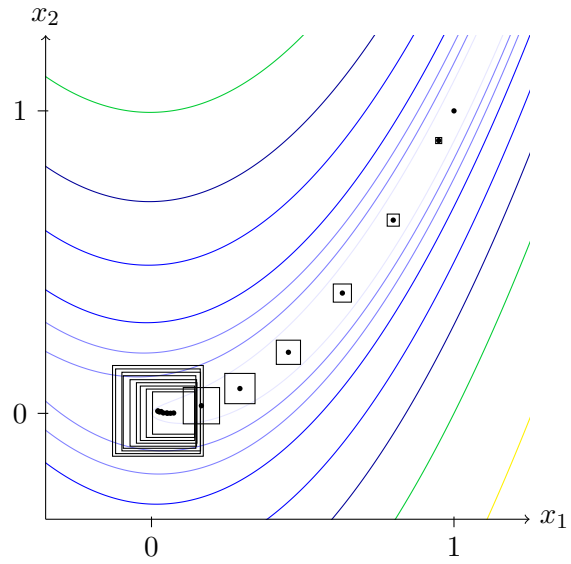


FIGURE 7.4 – Position des optima robustes (points noir) obtenus avec la mesure du pire cas sur la fonction de Rosenbrock (courbes de niveau) pour une incertitude sur les paramètres d'entrée (carrés) définie par des intervalles de variation de différentes tailles (allant de 0 pour le point en haut à droite jusqu'à 0.3 pour le point en bas à gauche, par pas de 0.02).

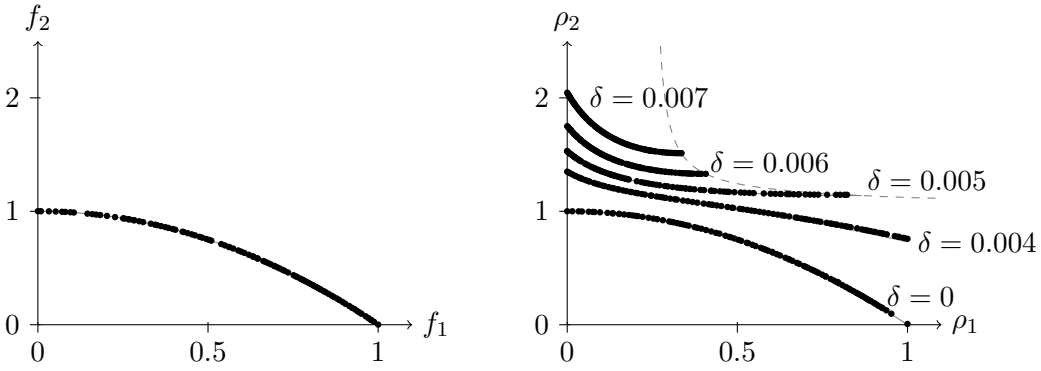


FIGURE 7.5 – Fronts de Pareto du problème DG2 obtenus expérimentalement : front déterministe classique à gauche, et fronts robustes pour différentes valeurs de δ à droite.

change : à l'origine concave, il devient convexe. Cela a un impact sur l'ensemble des solutions robustes du problème. En effet, si pour de faibles valeurs de δ les solutions restent les mêmes que dans le cas déterministe, lorsque l'incertitude augmente la convexité du front fait disparaître certaines de ces solutions (car elles deviennent dominées par d'autres). L'évolution de la limite du front robuste est ainsi représentée en pointillées sur la figure 7.5. Si jamais on définit une incertitude trop grande dans un problème de ce type, on va donc se priver d'une certaine partie des solutions optimales.

Ces deux exemples montrent l'importance de fixer avec précaution la taille des incertitudes sur les paramètres d'entrée d'un problème afin de pouvoir retrouver les solutions robustes attendues lors de sa résolution.

7.1.4 Réduction des incertitudes

Après avoir quantifié les incertitudes qui entrent en jeu dans le problème, on peut si on le souhaite tenter de les faire diminuer avant de formuler véritablement le problème d'optimisation stochastique auquel on a affaire. De manière générale, moins on a d'incertitudes dans un problème et plus le système optimal aura de chances de rester performant par la suite. Une solution au problème de robustesse serait donc d'arriver à supprimer toutes les incertitudes présentes en entrée. Cela n'est bien entendu pas toujours possible car certaines incertitudes proviennent de variabilités intrinsèques de paramètres comme la température ambiante par exemple (on parle d'incertitudes stochastiques). On ne peut donc pas réduire ce type d'incertitudes mais simplement les modéliser au mieux pour pouvoir en tenir compte correctement. Il existe par contre des incertitudes qui traduisent un manque de connaissance sur certains aspects du problème et qui peuvent donc être améliorées (il s'agit d'incertitudes épistémiques). Il peut par exemple y avoir une incertitude sur la surface d'une pièce car cette surface est évaluée de manière grossière alors qu'un calcul plus précis permettrait de connaître sa valeur exacte.

La réduction des incertitudes épistémiques demande un certain investissement. Il peut être intéressant de déterminer quels efforts seront rentables ou non en termes d'impact sur les performances et la robustesse du système. Si on veut par exemple réduire certaines tolérances de fabrication, on peut acheter de nouvelles machines outils plus précises qui permettront peut-être de construire des systèmes plus robustes et plus performants. Mais jusqu'à quel point doit-on améliorer la fabrication du système pour obtenir un gain significatif ? On peut aussi étudier l'influence des différents paramètres incertains sur les objectifs du problème (par le biais d'une analyse de sensibilité par exemple) afin de déterminer quelles sont les incertitudes qui pénalisent le plus le système. Cela permet d'identifier les paramètres pour lesquels une diminution d'incertitude serait intéressante.

La réduction d'incertitudes n'est pas l'objet de cette étude. Nous considérons ici que les efforts possibles pour minimiser les sources d'incertitudes ont déjà été faits. On ne cherche donc pas à identifier les variables pour lesquelles l'incertitude gagnerait à être réduite, mais plutôt à prendre en compte les incertitudes existantes afin de trouver des solutions robustes vis-à-vis de celles-ci.

7.1.5 Définition du problème stochastique

L'identification et la modélisation des incertitudes sur les paramètres d'entrée permet d'exprimer le problème d'optimisation à résoudre sous sa forme stochastique, c'est-à-dire sous la forme d'un problème dans lequel interviennent des variables aléatoires $\boldsymbol{\chi}$. On remplace pour cela les paramètres $\boldsymbol{x}$ (qu'ils soient de décision ou bien environnementaux) par leur équivalent aléatoire $(\boldsymbol{x} + \boldsymbol{\chi}_x)$. Le problème (7.1) devient alors :

$$\underset{\boldsymbol{d} \in \boldsymbol{D}}{\text{minimiser}} \ \boldsymbol{f}(\boldsymbol{d} + \boldsymbol{\chi}_d, \boldsymbol{e} + \boldsymbol{\chi}_e). \tag{7.2}$$

On différencie désormais l'espace de recherche $\boldsymbol{D} \subseteq \mathbb{R}^{n_d}$ (qui est l'espace de variation des paramètres de décision $\boldsymbol{d}$ du problème d'optimisation) de l'espace incertain $\boldsymbol{\Omega} \subseteq \mathbb{R}^{n_x}$ dans lequel évoluent les variables $(\boldsymbol{x} + \boldsymbol{\chi}_x)$. La dimension n_x de l'espace incertain est égale au nombre n_d de variables de décision auquel s'ajoute le nombre n_e de paramètres environnementaux incertains. Nous avons aussi introduit dans l'état de l'art l'espace de

recherche $\boldsymbol{X} \subseteq \mathbb{R}^{n_x}$ qui rassemble les variables de décision $\boldsymbol{d}$ et les paramètres environnementaux $\boldsymbol{e}$ dans leur version déterministe (ce sont des constantes), et qui est donc en correspondance directe avec $\boldsymbol{D}$.

Le domaine de définition d'une variable $\boldsymbol{x}$ peut différer de celui de $(\boldsymbol{x} + \boldsymbol{\chi}_x)$. L'espace incertain est en effet généralement plus étendu que l'espace de recherche afin de prendre en compte les fluctuations possibles des paramètres environnementaux ainsi que celles des variables de décision au delà des bornes de leur domaine de définition initial. On a donc $\boldsymbol{X} \subseteq \boldsymbol{\Omega}$. L'espace incertain $\boldsymbol{\Omega}$ représente l'espace dans lequel on peut faire varier l'ensemble des entrées du système en vue d'étudier sa réactivité face aux incertitudes, alors que l'espace de recherche $\boldsymbol{D}$ (ou $\boldsymbol{X}$) correspond à l'espace dans lequel seront exprimées les solutions du problème.

7.2 Choix des mesures de robustesse et formulation du problème robuste

Afin de trouver des solutions peu sensibles aux incertitudes présentes dans le problème d'optimisation stochastique (7.2), on va le transformer en un nouveau problème déterministe grâce à des mesures de robustesse $\boldsymbol{\rho}$ qui évaluent différentes caractéristiques de la distribution de probabilité des incertitudes sur les objectifs $\boldsymbol{f}$. Il faut pour cela définir les mesures de robustesse qu'on va faire intervenir et choisir une formulation pour le problème robuste.

7.2.1 Choix des mesures de robustesse

Une phase importante dans la définition d'un problème d'optimisation robuste est le choix des mesures de robustesse. C'est au travers de ces mesures qu'on va spécifier les qualités attendues pour une solution robuste. Le choix de mesures de robustesse adaptées se fait en fonction de trois points particuliers : le type de modélisation des incertitudes présentes en entrée, l'adéquation de la signification des mesures au problème à traiter, et enfin les possibilités de calcul de ces mesures.

Le premier aspect qui peut déterminer le choix des mesures de robustesse est la façon dont ont été modélisées les incertitudes du problème. On ne peut en effet pas utiliser les mêmes mesures lorsqu'on a affaire à des intervalles de variation ou lorsque les incertitudes sont représentées par des distributions de probabilité. Les mesures que nous avons décrites dans l'état de l'art (mesures basées sur les espérances ou bien sur les quantiles, voir la section 5.4) font référence à une description probabiliste de l'incertitude. Dans le cas où les incertitudes sont modélisées avec des intervalles de variation, la plupart de ces mesures ne sont pas utilisables. Les seules notions qu'on peut définir sur une représentation par intervalles sont les quantiles d'ordre 0 et 1. On utilisera donc la mesure du pire cas, ou bien une différence de quantiles pour mesurer la dispersion de l'incertitude sur les objectifs.

La mesure de la robustesse d'un point doit être choisie en fonction des qualités qu'on souhaite atteindre au niveau des solutions optimales. Les mesures n'ont pas toutes la même signification, et selon le problème à traiter certaines peuvent avoir plus d'intérêt que d'autres. Nous avons distingué les mesures basées sur les espérances, qui travaillent sur un comportement moyen des objectifs, et les mesures basées sur les quantiles qui offrent certaines garanties de performance. En fonction des résultats qu'on souhaite obtenir (bonne performance en moyenne ou bien performance garantie), on choisira donc une mesure

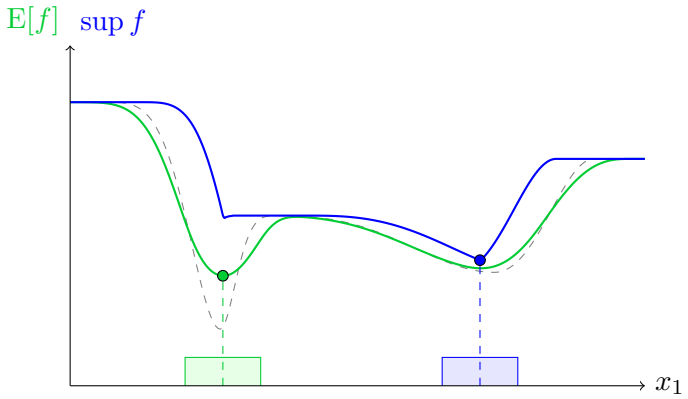


FIGURE 7.6 – Espérance (en vert) et pire cas (en bleu) de la fonction f (en pointillés) pour une incertitude uniforme en entrée. Les points représentent les solutions optimales robustes dans les deux cas.

de robustesse ou une autre. Le choix des mesures a un impact fort sur les résultats de l'optimisation robuste. La figure 7.6 présente par exemple les mesures d'espérance et de pire cas sur notre fonction de test f munie d'une incertitude uniforme sur sa variable d'entrée x_1. Les deux optima robustes obtenus sont totalement différents. Si l'espérance estime que la performance reste correcte en moyenne dans la zone de l'optimum global, le pire cas privilégie la région du minimum local situé sur la partie droite de la fonction afin de garantir une bonne performance même dans le cas de variations extrêmes du paramètre d'entrée. Le but d'une optimisation robuste peut aussi être de trouver une solution dont les performances varient le moins possible (pour avoir des produits uniformisés dans le cas d'une production en série par exemple). On s'appuiera alors sur des mesures de robustesse pure (comme la mesure de variance ou la différence de quantiles) qui évaluent la dispersion de l'incertitude en sortie. Ces mesures vont focaliser la recherche sur les régions « plates » des objectifs.

Le dernier paramètre qui entre en compte dans le choix d'une mesure de robustesse est sa facilité de calcul. Nous avons vu dans l'état de l'art que les mesures pouvaient être estimées de plusieurs manières différentes (section 5.8). Certaines approches ne sont applicables qu'à un nombre restreint de mesures de robustesse, comme le calcul analytique sur un modèle de substitution par exemple. De manière générale, les mesures basées sur les espérances sont les plus faciles à évaluer car elles bénéficient d'un grand nombre d'approches différentes. L'estimation des mesures de quantile repose quant à elle presque uniquement sur des méthodes à base de tirages de Monte-Carlo. Cela explique peut-être le fait que les mesures de l'espérance et de la variance soient les plus répandues en optimisation robuste. Si on souhaite employer une mesure de quantile d'ordre k dans un problème, on peut aussi tirer parti des facilités de calcul des autres mesures en faisant une approximation gaussienne de la distribution de l'incertitude sur l'objectif concerné. Le quantile peut alors être estimé à partir de l'espérance et de la variance de cette distribution. Nous appelons cette mesure un « quantile approché » noté $\hat{\mathrm{Q}}_k$:

$$\rho(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \hat{\mathrm{Q}}_k(f(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \mathrm{E}[f(\boldsymbol{x}+\boldsymbol{\chi}_x)] + \Phi^{-1}(k)\sqrt{\mathrm{Var}(f(\boldsymbol{x}+\boldsymbol{\chi}_x))}, \tag{7.3}$$

où Φ^{-1} représente l'inverse de la fonction de répartition de la loi normale réduite centrée

($\Phi^{-1}(k) = \sqrt{2}\,\mathrm{erf}^{-1}(2k-1)$). On retrouve alors une formulation agrégée de l'espérance et de la variance très courante dans la littérature (voir l'équation (5.12)). Sa considération sous la forme d'un quantile approché peut aider à fixer le paramètre α de cette agrégation. Au delà des facilités de calcul des mesures de robustesse elles-mêmes, on peut aussi considérer les possibilités d'estimation de leur erreur avant de faire un choix. Dans le cadre de l'utilisation de modèles de substitution, il est en effet intéressant d'avoir accès à des intervalles de confiance pour les valeurs de robustesse obtenues. Le calcul d'erreur est là aussi plus ou moins simple en fonction des mesures (voir la section 5.8.5).

7.2.2 Choix d'une formulation pour le problème d'optimisation robuste

Une fois les mesures de robustesse adaptées sélectionnées, la définition du problème robuste passe ensuite par le choix d'une formulation permettant de les intégrer au sein du problème stochastique (7.2). Il sera ainsi transformé en un problème déterministe prenant en compte les incertitudes présentes sur les paramètres d'entrée et qui pourra être résolu avec des méthodes d'optimisation classiques. Le choix d'une formulation donnée dépend du résultat recherché pour l'application à traiter, mais aussi de la complexité du problème initial.

Comme détaillé dans l'état de l'art, on peut utiliser trois approches différentes pour formuler un problème d'optimisation robuste (voir la section 5.5) : soit en modifiant les objectifs du problème initial, soit en lui ajoutant de nouveaux objectifs, soit en lui ajoutant de nouvelles contraintes (ces trois possibilités pouvant aussi être combinées). Le remplacement de chaque objectif par la mesure de robustesse qui lui correspond permet d'obtenir des solutions robustes qui sont des compromis entre les différents objectifs du problème comme dans le cas déterministe initial. L'impact des modifications liées à l'intégration de la robustesse est donc minime pour le décideur. L'ajout de nouveaux objectifs au problème présente par contre l'intérêt de pouvoir faire a posteriori un arbitrage entre robustesse et performance. Mais le décideur sera alors face à un choix plus complexe. Quant à l'ajout de contraintes, il permet de limiter la recherche de solutions optimales parmi les solutions dont la robustesse est satisfaisante. Le décideur doit pour cela fixer au départ des limites de robustesse qu'il considère acceptables, mais ensuite les solutions obtenues peuvent être traitées de manière classique.

Le deuxième critère de choix d'une formulation concerne la complexité du problème. Il faut en effet tenir compte de la complexité du problème initial : si celui-ci présente déjà un grand nombre d'objectifs différents, on évitera par exemple d'ajouter des objectifs supplémentaires afin que l'extraction d'une solution finale par le décideur ne se complique pas davantage. D'autre part, on peut aussi considérer les capacités des algorithmes de résolution utilisés. Un simple remplacement des objectifs du problème initial ne change pas trop sa complexité (mis à part qu'il est alors nécessaire d'évaluer des mesures de robustesse au lieu des objectifs classiques). Les deux autres formulations rendent par contre le problème plus ardu en y ajoutant des objectifs ou bien des contraintes. Selon les possibilités offertes par les algorithmes d'optimisation mis en œuvre, on pourra s'autoriser à ajouter un nombre plus ou moins grand d'objectifs ou de contraintes. Si par exemple le problème initial est déjà fortement contraint, il est fort probable que les algorithmes spécifiques utilisés pour le résoudre n'auront aucun mal à prendre en compte des contraintes supplémentaires, alors que d'autres algorithmes pourront rencontrer des difficultés si le nombre de contraintes à gérer devient trop important.

Si on prend l'exemple le plus classique d'un remplacement d'objectifs par leur équivalent robuste, le problème d'optimisation robuste une fois formulé devient ainsi :

$$\underset{\boldsymbol{d}\in\boldsymbol{D}}{\text{minimiser}}\ \boldsymbol{\rho}(\boldsymbol{f}(\boldsymbol{d}+\boldsymbol{\chi}_d,\boldsymbol{e}+\boldsymbol{\chi}_e)). \tag{7.4}$$

7.3 Choix des stratégies de calcul de la robustesse

Lorsque le problème d'optimisation robuste a été formulé, il peut être résolu grâce à des algorithmes d'optimisation déterministe classique. La seule difficulté supplémentaire par rapport au problème initial est qu'il est maintenant nécessaire d'évaluer des mesures de robustesse. Différentes méthodes existent pour cela : nous avons notamment distingué dans l'état de l'art l'estimation par tirages de Monte-Carlo, l'utilisation de modèles locaux comme les polynômes de chaos ainsi que le calcul analytique sur certains modèles de substitution globaux (voir la section 5.8). Il faut donc sélectionner dans un premier les approches qui vont être mises en place avant de pouvoir passer à l'étape de résolution.

Les mesures de robustesse du problème ont possiblement été choisies en tenant compte de leur facilité d'évaluation. Mais une fois ces mesures fixées il reste tout de même plusieurs options pour les calculer (notamment dans le cas des mesures d'espérance et de variance). Une méthode de calcul particulière doit être retenue en fonction de sa rapidité d'exécution et de la précision de ses estimations. On peut aussi privilégier des méthodes faciles à mettre en œuvre dans le cadre de l'application concernée. Une méthode efficace permettra à l'algorithme d'optimisation de converger plus rapidement vers les solutions optimales robustes, et la précision des évaluations garantira quand à elle de se diriger vers des solutions véritablement robustes. En ce qui concerne les trois approches citées plus haut, la complexité ainsi que la précision de l'évaluation de la robustesse par la méthode de Monte-Carlo dépend du nombre de tirages effectués. Ce nombre peut donc être réglé pour s'adapter au problème à résoudre. De son côté, le chaos polynomial est plus efficace lorsque le nombre de paramètres incertains est faible, car on a alors besoin de moins d'échantillons pour construire le modèle local. Sa précision dépend aussi de la non linéarité de la fonction de référence. Le degré du polynôme peut être adapté pour obtenir de meilleurs résultats ou bien réduire le temps de calcul de cette approche. Le calcul analytique présente quant à lui une bonne précision car son évaluation est exacte sur un modèle de substitution global. Sa complexité dépend du nombre de coefficients du modèle (qui correspond par exemple au nombre d'échantillons d'apprentissage dans le cas d'un modèle de krigeage classique). On peut parfois jouer là aussi sur ce nombre de coefficients pour s'adapter à l'application à traiter.

Au delà du choix d'une approche efficace pour évaluer la robustesse d'une solution, il faut de plus mettre en place une stratégie générale de calcul qui va prendre en charge les nombreuses estimations de robustesse que nécessite une optimisation. Différentes stratégies sont présentées sur la figure 7.7. Si une estimation directe sur la fonction de référence f peut sembler pratique pour évaluer la robustesse d'une unique solution (en utilisant la méthode de Monte-Carlo ou bien un polynôme de chaos par exemple), cette approche devient généralement trop coûteuse dans le cadre d'une optimisation du fait du grand nombre d'appels de la fonction à réaliser. C'est pourquoi on utilise plutôt un modèle de substitution $\hat{f}$ de cette fonction de référence sur lequel on peut effectuer des estimations de robustesse de manière beaucoup plus rapide (en s'appuyant sur les différentes techniques évoquées plus haut). Mais il existe aussi une troisième possibilité : construire un modèle

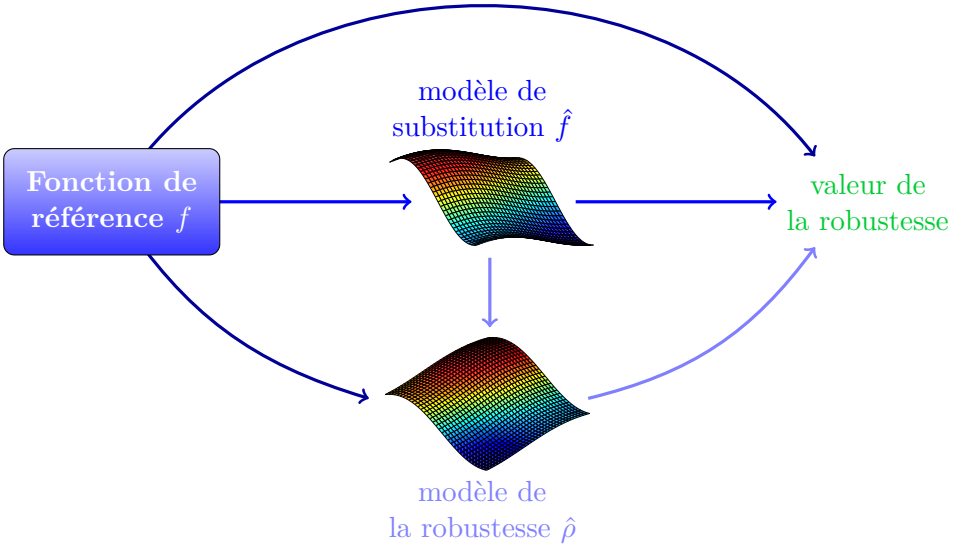


FIGURE 7.7 – Stratégies d'évaluation d'une mesure de robustesse ρ sur une fonction de référence f : estimation directe sur la fonction, utilisation d'un modèle de substitution $\hat{f}$ de la fonction, ou construction d'un modèle de substitution $\hat{\rho}$ de la mesure de robustesse.

de substitution $\hat{\rho}$ de la mesure de robustesse elle-même. On peut en effet faire un parallèle avec l'optimisation déterministe classique où les objectifs du problème sont approchés par des modèles de substitution. En optimisation robuste, on a aussi affaire à un problème déterministe dans lequel les objectifs sont des mesures de robustesse. On peut donc établir des modèles de ces objectifs et les optimiser ensuite de manière tout à fait ordinaire. Deux démarches sont offertes pour construire un modèle de substitution $\hat{\rho}$ de la mesure de robustesse ρ : on peut évaluer la robustesse au niveau de quelques échantillons d'apprentissage sur la fonction de référence pour construire un modèle $\hat{\rho}(f(\boldsymbol{x}+\boldsymbol{\chi}_x))$, ou bien utiliser le modèle $\hat{f}$ pour évaluer la robustesse des échantillons d'apprentissage et construire un modèle $\hat{\rho}(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x))$. L'inconvénient de la première approche réside dans le coût que représente l'évaluation de la robustesse des échantillons sur la fonction de référence. Dans le second cas on construit un modèle de substitution à partir de résultats eux-mêmes issus d'un modèle, ce qui rend la gestion de l'erreur de modélisation relativement complexe. Mais une fois ces modèles de la robustesse construits l'optimisation peut se dérouler de façon classique sans nécessiter de calculs supplémentaires à chaque évaluation.

Ces différentes approches pour estimer la robustesse de solutions nécessitent plus ou moins d'évaluations sur la fonction référence et sur les modèles de substitution. Selon le coût de ces évaluations, une stratégie pourra être plus intéressante que les autres.

7.4 Exemple sur le cas test des deux barres

Pour illustrer le déroulement de la définition d'un problème d'optimisation robuste que nous venons de présenter, nous avons choisi l'exemple du cas test des deux barres [JDC03] détaillé dans la section 5.6 et que nous rappelons ici. Il s'agit dans ce problème d'optimiser

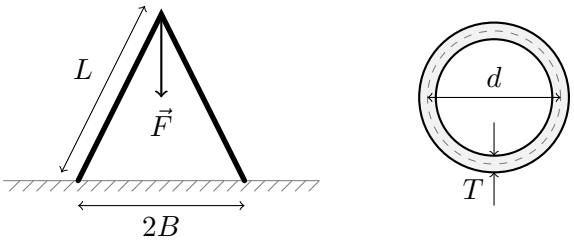


FIGURE 7.8 – Représentation schématique des deux barres à gauche, avec une vue de leur section à droite.

un système assez simple constitué de deux tubes apposés l'un contre l'autre au niveau de leur sommet et soumis à une certaine force $\vec{F}$ (voir le schéma de la figure 7.8). Le but est de minimiser le volume V de matière employé tout en garantissant la résistance du système à la force appliquée. Le problème d'optimisation déterministe initial est le suivant :

$$\begin{aligned} &\min_{d,L} \; V(T,d,L) \\ &s.c. \begin{cases} s(F,L,T,d,B) \leq s_{\max}, \\ s(F,L,T,d,B) \leq s_{\text{crit}}(E,T,d,L), \end{cases} \end{aligned} \tag{7.5}$$

où d et L sont les variables de décision qui représentent respectivement le diamètre et la longueur des tubes, et T, F, B, E et $s_{\max}$ des paramètres environnementaux fixés. Si on note :

$$\begin{aligned} f_1(T,d,L) &= V(T,d,L), \\ g_1(F,L,T,d,B,s_{\max}) &= s(F,L,T,d,B) - s_{\max}, \\ g_2(F,L,T,d,B,E) &= s(F,L,T,d,B) - s_{\text{crit}}(E,T,d,L), \end{aligned}$$

le problème peut se réécrire sous la forme standard :

$$\begin{aligned} &\min_{d,L} \; f_1(T,d,L) \\ &s.c. \begin{cases} g_1(F,L,T,d,B,s_{\max}) \leq 0, \\ g_2(F,L,T,d,B,E) \leq 0. \end{cases} \end{aligned} \tag{7.6}$$

Partant de ce problème initial, on peut identifier les incertitudes présentes. Les auteurs ont choisi de considérer six paramètres incertains : les deux variables de décision et quatre paramètres environnementaux (seul le paramètre $s_{\max}$ qui est une borne pour une contrainte est connu avec certitude). Le problème fait déjà intervenir toutes ces variables, qu'elles soient de décision ou bien environnementales : on n'aura donc pas à rajouter de paramètre supplémentaire. Les incertitudes ont été modélisées à l'aide de lois de probabilité normales car on considère que ces variables ont une valeur nominale qui reste la plus probable mais autour de laquelle elles peuvent varier avec une probabilité plus faible lorsqu'on s'en éloigne. La variance des distributions a été fixée à partir de connaissances empiriques sur les différents paramètres (le détail de ces distributions est donné dans le tableau 5.2). Le problème stochastique qui en résulte est similaire au problème déterministe (7.6) mis à part que toutes les variables incertaines sont maintenant exprimées sous

la forme de variables aléatoires $(x + \chi_x)$:

$$\begin{array}{l} \min\limits_{d,L} \; f_1(T+\chi_T, d+\chi_d, L+\chi_L) \\ s.c. \left\{ \begin{array}{l} g_1(F+\chi_F, L+\chi_L, T+\chi_T, d+\chi_d, B+\chi_B, s_{\max}) \leq 0, \\ g_2(F+\chi_F, L+\chi_L, T+\chi_T, d+\chi_d, B+\chi_B, E+\chi_E) \leq 0. \end{array} \right. \end{array} \tag{7.7}$$

Pour résoudre ce problème en prenant en compte ses incertitudes, il faut le transformer en problème d'optimisation robuste grâce à des mesures de robustesse. Les auteurs ont choisi de retenir le quantile d'ordre 0.9 du volume : on souhaite que le volume de matière utilisé soit garanti dans 90% des cas lorsque les paramètres incertains varient. On remarque aussi que la solution déterministe classique de ce problème est très peu fiable dès qu'on considère les incertitudes. Les contraintes sont en effet violées dans plus de la moitié des cas. Afin que le système puisse résister à la force qui lui est soumise, il faut donc introduire aussi une robustesse vis-à-vis des contraintes. La mesure choisie est la probabilité de satisfaction des contraintes (on souhaite maximiser le taux de fiabilité du système), qui correspond à un seuil probabiliste. Pour la formulation du problème robuste, les auteurs proposent de remplacer l'objectif initial par son équivalent robuste ρ_1 et de rassembler les deux contraintes en un second objectif robuste ρ_2. Le problème bi-objectifs final s'écrit donc (en omettant les variables dont dépendent les fonctions f_1, g_1, g_2, V, s et s_{crit}) :

$$\underset{d,L}{\text{minimiser}} \; \{\rho_1(f_1(...)), -\rho_2(g_1(...), g_2(...))\}, \tag{7.8}$$

avec :

$$\begin{aligned} \rho_1(f_1(...)) &= \mathrm{Q}_{0.9}(V(...)), \\ \rho_2(g_1(...), g_2(...)) &= P((s(...) \leq s_{\max}) \cap (s(...) \leq s_{\text{crit}}(...))). \end{aligned}$$

Ce problème déterministe peut être résolu par un algorithme d'optimisation classique, sous réserve de mettre en place une méthode de calcul des mesures de robustesse. Les quantiles seront ici évalués par la méthode de Monte-Carlo. Nous avons par la suite mis en œuvre les trois stratégies générales de calcul de la robustesse présentées sur la figure 7.7 : évaluation directe sur la fonction de référence, utilisation d'un modèle de substitution de la fonction de référence et construction d'un modèle de la robustesse. Ces trois approches de résolution sont détaillées ci-dessous.

7.4.1 Évaluation directe de la robustesse sur les fonctions de référence

Bien que cette stratégie soit rarement utilisable en pratique du fait de son coût particulièrement élevé, nous avons tout d'abord résolu le problème (7.8) en calculant la robustesse des solutions sur les fonctions de référence f_1, g_1 et g_2. Ce calcul n'est pas trop lourd dans le cadre de cet exemple car le système est décrit par des équations relativement simples. Cette résolution directe va nous servir de référence pour la suite. Pour peu que le nombre de tirages de Monte-Carlo soit suffisant, c'est en effet la méthode la plus précise pour résoudre un problème car on n'utilise aucun modèle de substitution intermédiaire.

Afin de trouver le front de Pareto « idéal », le problème a été résolu grâce à l'algorithme génétique NSGA-II avec une population de 50 individus sur 1000 générations. La robustesse a été évaluée avec 1000 tirages de Monte-Carlo autour de chaque point considéré, évalués sur l'objectif et les contraintes du problème initial. Le front obtenu est représenté

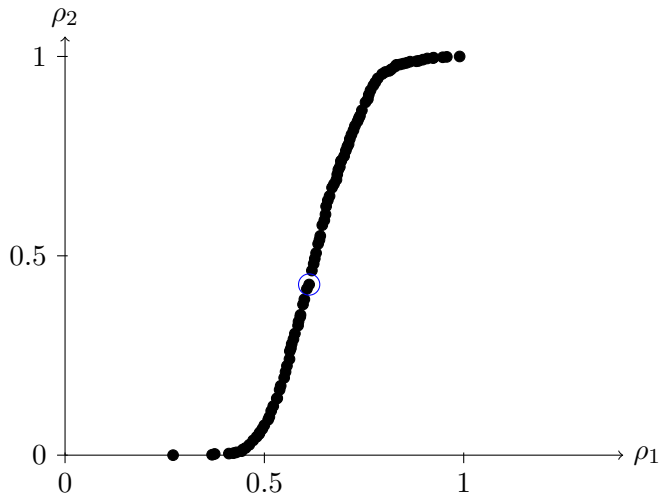


FIGURE 7.9 – Front de Pareto « idéal » pour le problème des deux barres (points noirs). ρ_1 correspond au quantile du volume (en L) et ρ_2 est le taux de fiabilité du système. La solution déterministe classique est indiquée par un cercle bleu.

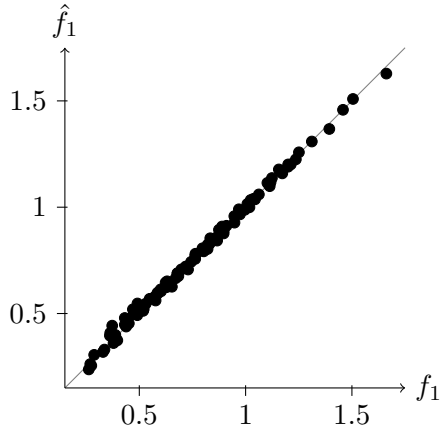
sur la figure 7.9. Il offre un choix de solutions plus ou moins fiables. On peut voir que la solution du problème déterministe classique reste optimale dans le cadre robuste, mais qu'elle présente un défaut de fiabilité conséquent car la probabilité que les contraintes soient satisfaites dans cette configuration n'est que de 0.43 environ. On pourra donc lui préférer une autre solution possédant un volume de matière légèrement plus élevé mais qui résistera mieux aux contraintes malgré les incertitudes du problème.

7.4.2 Modèles de substitution des fonctions de référence

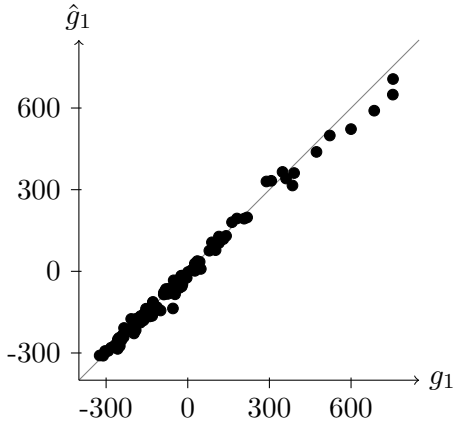
L'évaluation directe de la robustesse sur les fonctions de référence est généralement impossible. La seconde stratégie consiste alors à établir des modèles de substitution de ces fonctions sur lesquels la robustesse pourra être calculée plus rapidement. C'est par exemple l'approche retenue dans l'article [PLRRB09] qui utilise des modèles de krigeage pour résoudre le cas test des deux barres.

Les modèles utilisés ici ne sont pas forcément identiques à ceux d'une optimisation déterministe. En effet, il faut considérer à la fois l'espace de recherche $\boldsymbol{D}$ et l'espace incertain $\boldsymbol{\Omega}$. La dimension d'entrée des modèles est donc souvent plus importante que dans le cas déterministe car on doit tenir compte des paramètres environnementaux incertains. Pour le cas test des deux barres, une optimisation déterministe classique nécessiterait des modèles à deux entrées (d et L). Pour l'optimisation robuste, on doit aussi pouvoir faire varier T, F, B et E. Les modèles que nous allons construire auront donc 6 entrées.

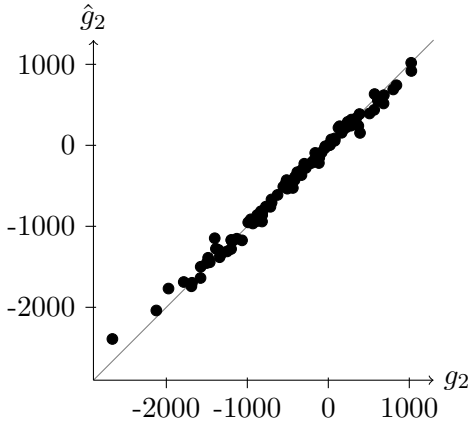
Afin de générer un plan d'expériences pour l'apprentissage des modèles, il faut définir la plage de variation de chacune de ces entrées. Les intervalles de variation doivent être fixés par rapport à ceux du problème initial en prenant en compte les possibles fluctuations dues aux incertitudes. Les paramètres environnementaux ont par exemple initialement une valeur fixée, mais l'incertitude qui les touche les fait varier autour de cette valeur. Dans le cas de distributions à support infini comme les lois normales du problème qui nous intéresse, les intervalles de variation des paramètres incertains sont eux aussi infinis en



(a) ErrMax = 0.070, ErrMoy = 0.013



(b) ErrMax = 104.23, ErrMoy = 18.45



(c) ErrMax = 275.17, ErrMoy = 51.27

FIGURE 7.10 – Qualité des réseaux de neurones représentant les objectifs (a) et contraintes (b) (c) initiaux pour le cas test des deux barres : on a représenté sur chaque graphe la valeur prédite par le modèle par rapport à la valeur réelle en 100 points de test (points noirs).

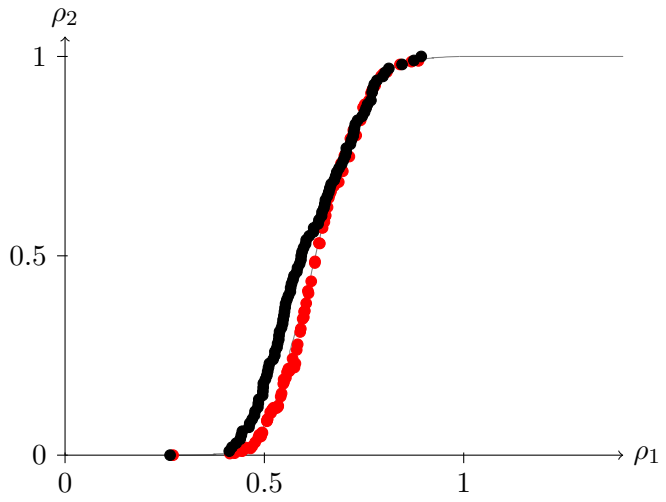
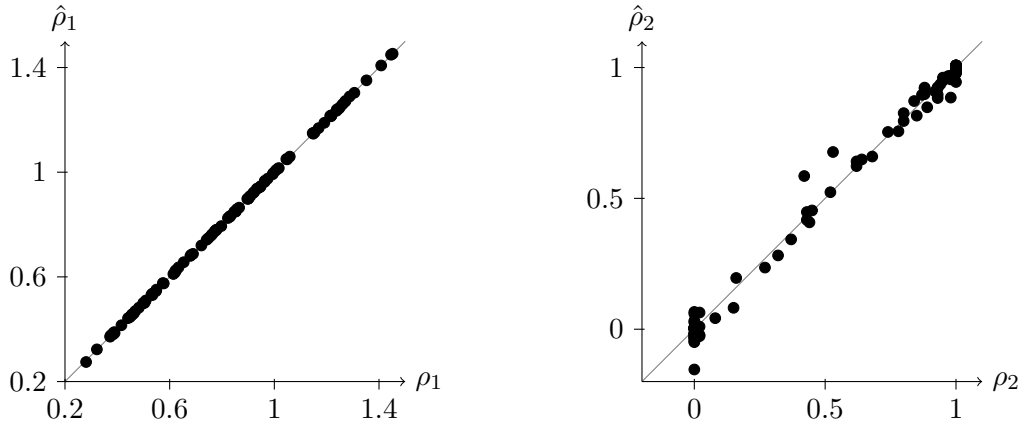


FIGURE 7.11 – Front de Pareto obtenu grâce aux modèles de substitution des fonctions de référence. La ligne grise indique le front de Pareto idéal de la figure 7.9, les points noirs le front de Pareto obtenu grâce aux modèles et les points rouges le même front recalculé sur les fonctions de référence.

théorie. On préfère néanmoins se restreindre aux valeurs les plus probables uniquement afin de représenter au mieux les fonctions de référence dans les régions les plus influentes, plutôt que d'essayer de modéliser des zones quasiment inaccessibles du fait des incertitudes. Nous avons choisi de prendre ici des échantillons d'apprentissage dans un intervalle de confiance de $\pm$ 3 écarts-type autour de la valeur moyenne ou des bornes de chaque variable incertaine (ce qui correspond à une probabilité de plus de 99%). La variable de décision d varie ainsi par exemple dans l'intervalle $[20 - 3 \times 1, 80 + 3 \times 1] = [17, 83]$.

Nous avons construit trois réseaux de neurones $\hat{f}_1$, $\hat{g}_1$ et $\hat{g}_2$ pour représenter l'objectif et les deux contraintes du problème initial. Comme dans l'article [PLRRB09], un LHS de 60 échantillons d'apprentissage a été utilisé. Une étude préalable a permis de fixer le nombre optimal de neurones cachés à 5. Ces modèles ont ensuite été testés à l'aide de 100 échantillons supplémentaires des fonctions de référence. Le résultat de ces tests est présenté sur la figure 7.10. Si les modèles ne sont pas parfaits du fait du faible nombre d'échantillons d'apprentissage disponibles, ils permettent tout de même d'avoir une bonne représentation des fonctions de référence du problème.

La résolution du problème d'optimisation robuste est réalisée grâce à l'algorithme NSGA-II avec une population de 20 individus sur 500 générations. Les mesures de robustesse sont évaluées à l'aide de 100 points de Monte-Carlo sur les modèles de substitution. La figure 7.11 présente le front de Pareto obtenu, ainsi que la valeur « réelle » de ces solutions recalculée grâce à 1000 tirages de Monte-Carlo sur les fonctions de référence. Les résultats sur les réseaux de neurones sont légèrement différents du front idéal du fait des erreurs de modélisation (la fiabilité est surestimée par endroits), mais lorsqu'on recalcule la valeur de la robustesse des points on voit que les solutions trouvées correspondent bien aux solutions optimales du problème. L'approche par modèle de substitution des fonctions de référence permet donc de résoudre convenablement le problème d'optimisation robuste.



(a) ErrMax = 0.0059, ErrMoy = 0.0017

(b) ErrMax = 0.17, ErrMoy = 0.023

FIGURE 7.12 – Qualité des réseaux de neurones représentant les mesures de robustesse pour le cas test des deux barres : on a représenté sur chaque graphe la valeur prédite par le modèle par rapport à la valeur réelle (estimée par la méthode de Monte-Carlo) en 100 points de test (points noirs).

7.4.3 Modèles de substitution des mesures de robustesse

La troisième et dernière stratégie envisagée pour résoudre un problème d'optimisation robuste est la construction de modèles de substitution des mesures de robustesse. Ces modèles permettent de réaliser ensuite l'optimisation sans se soucier du calcul des mesures. Cette stratégie est par exemple mise en œuvre dans l'article [LRPG+09] où les auteurs utilisent un modèle de krigeage de la mesure du quantile pour optimiser un engrenage soumis à des incertitudes.

Les modèles de la robustesse possèdent le même espace d'entrée que les modèles utilisés en optimisation déterministe classique : il s'agit de l'espace de recherche $\boldsymbol{D}$ défini par les variables de décision. Les incertitudes sont en effet prises en compte lors du calcul de la robustesse et n'apparaissent plus ensuite. La robustesse doit simplement être évaluée au préalable en un certain nombre d'échantillons d'apprentissage afin d'établir les modèles. Nous avons choisi ici de calculer la robustesse des échantillons d'apprentissage grâce aux fonctions de référence directement (mais on peut aussi utiliser des modèles de substitution de ces fonctions).

Deux réseaux de neurones $\hat{\rho}_1$ et $\hat{\rho}_2$ à deux entrées (et toujours avec 5 neurones cachés) ont été construits pour représenter le quantile ρ_1 du volume et le taux de fiabilité ρ_2 du système. Ces modèles sont basés sur un LHS de 60 échantillons d'apprentissage dont la robustesse a été estimée grâce à 100 tirages de Monte-Carlo autour de chaque échantillon. La figure 7.12 présente les résultats du test de ces modèles, réalisé avec 100 autres points pour lesquels la robustesse a été calculée de la même manière. On remarque que le premier modèle est quasiment parfait, alors le second est moins précis (particulièrement pour des taux de fiabilité proches de 0). La probabilité de respect des contraintes est donc plus difficile à modéliser que le quantile de l'objectif initial, ce qui peut se comprendre car ρ_2 se rapporte à deux contraintes distinctes qui rendent certainement son comportement plus complexe.

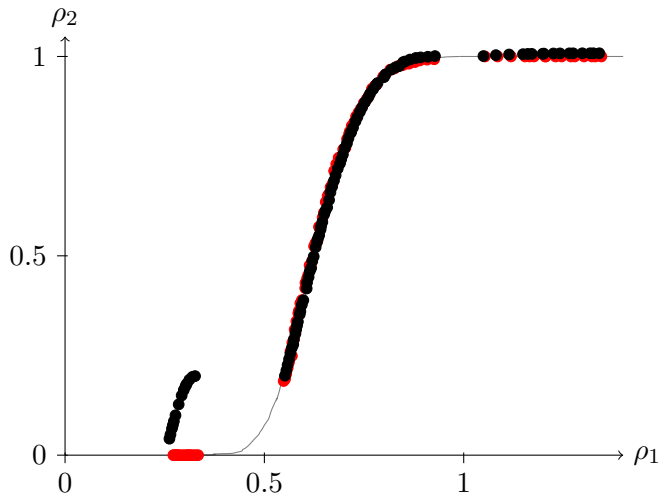


FIGURE 7.13 – Front de Pareto obtenu grâce aux modèles de substitution des mesures de robustesse. La ligne grise indique le front de Pareto idéal de la figure 7.9, les points noirs le front de Pareto obtenu grâce aux modèles et les points rouges le même front recalculé sur les fonctions de référence.

Une fois les modèles des mesures de robustesse construits, le problème est résolu par l'algorithme NSGA-II avec une population de 20 individus sur 500 générations. Le front de Pareto obtenu est représenté sur la figure 7.13. Les solutions de ce front ont été recalculées en évaluant leur robustesse à l'aide de 1000 tirages de Monte-Carlo sur les fonctions de référence. On observe que les deux fronts sont très proches, mis à part dans la zone où le taux de fiabilité est faible. La prédiction de $\hat{\rho}_2$ est complètement fausse à ce niveau (ce qu'on avait déjà remarqué lors du test du modèle), et cela fait perdre une partie du front idéal une fois les solutions recalculées. Cela n'est toutefois pas très grave car on s'intéresse surtout aux solutions dotées d'une fiabilité supérieure à 0.5 (c'est-à-dire des solutions plus fiables que l'optimum déterministe). Dans la région de forte fiabilité, les modèles des mesures de robustesse permettent de trouver des solutions optimales au problème robuste.

Le principal inconvénient de cette stratégie est qu'il est souvent trop coûteux de calculer la robustesse des échantillons d'apprentissage sur les fonctions de référence. Il faut alors utiliser des modèles de substitution de ces fonctions, mais la robustesse évaluée sur ces modèles est soumise à leur erreur. Les modèles des mesures de robustesse sont donc construits à partir de données d'apprentissage approximatives. Sachant que ces modèles présentent eux-mêmes une certaine erreur, on se rend compte que l'erreur finale sur les estimations de robustesse peut être importante et difficile à évaluer. Pour cela, il est possible de mettre en place des techniques de bootstrap sur les deux niveaux de modèles mais cela reste relativement coûteux. Les seuls travaux qui se sont intéressés à cette problématique (et dont nous n'avons eu connaissance qu'a posteriori) sont ceux de [DKM09]. Cet article présente la résolution d'un problème d'optimisation robuste à l'aide de modèles de krigeage des mesures de robustesse construits avec les deux approches que nous venons d'évoquer : la robustesse des échantillons d'apprentissage est évaluée par la méthode de Monte-Carlo directement sur la fonction de référence ou bien sur un modèle de cette fonction. Sur l'exemple traité, la superposition des deux modèles de substitution donne de plus mauvaises estimations de robustesse et le front de Pareto est plus éloigné du front idéal.

Cela explique peut-être pourquoi on préfère généralement évaluer la robustesse directement sur le modèle de substitution de la fonction de référence plutôt que d'introduire des imprécisions supplémentaires en construisant un autre modèle à partir de ses prédictions. Les modèles de substitution étant relativement rapides, des estimations de robustesse répétées ne représentent pas un coût trop important. L'établissement d'un modèle de la robustesse peut par contre se justifier lorsqu'on souhaite réaliser un nombre d'estimations vraiment important. On veillera alors à utiliser suffisamment d'échantillons d'apprentissage pour que les modèles soient de bonne qualité.

7.5 Résolution du problème d'optimisation robuste

Le schéma 7.1 détaillé au cours des sections précédentes a permis de définir un problème d'optimisation robuste prenant en compte les incertitudes sur les variables d'entrée. Nous allons maintenant nous attarder plus précisément sur la dernière étape, qui est la résolution de ce problème. Le problème robuste ayant la forme d'un problème déterministe, sa résolution est similaire à celle d'un problème classique utilisant des modèles de substitution. Son déroulement est présenté sur la figure 7.14.

À partir d'un plan d'expériences évalué sur les fonctions de référence, on construit tout d'abord l'ensemble des modèles de substitution nécessaires au calcul des mesures de robustesse du problème, selon la stratégie qui a été choisie. L'inconvénient de l'utilisation de ces modèles est qu'ils ne sont parfois pas suffisamment précis (comme on a pu le constater sur l'exemple du cas test des deux barres). Les résultats obtenus lors de l'optimisation sont ainsi plus ou moins approximatifs. Pour remédier à cela, on peut augmenter le nombre d'échantillons d'apprentissage du plan d'expériences initial, mais bien souvent le nombre de points qu'il est permis d'évaluer sur les fonctions de référence est limité car ces évaluations sont relativement coûteuses. Il faut donc les utiliser avec parcimonie. On a alors recours à des techniques de complétion adaptative du plan d'expériences initial qui visent à sélectionner de manière astucieuse les échantillons à évaluer pour n'améliorer les modèles qu'aux endroits où c'est nécessaire. On réalise ainsi plusieurs itérations d'amélioration en ajoutant peu à peu des échantillons à la base d'apprentissage des modèles jusqu'à ce qu'ils atteignent une qualité suffisante (ou bien jusqu'à ce que le quota d'évaluations des fonctions de référence soit dépassé). Cette amélioration adaptative peut chercher à faire diminuer l'erreur des modèles dans leur ensemble ou bien à améliorer les zones possiblement optimales uniquement. Dans ce second cas, la détermination des échantillons à évaluer fait intervenir le problème d'optimisation robuste à résoudre afin de localiser les régions d'intérêt. La construction adaptative de plans d'expériences sera développée en détail dans le prochain chapitre.

Lorsque la qualité des modèles est satisfaisante, on peut ensuite déterminer les solutions optimales du problème. Plusieurs cas de figure se présentent. Une première possibilité est d'extraire ces solutions parmi les échantillons du plan d'expériences final obtenu, si on considère que seuls ces points sont recevables car leur robustesse est connue avec plus de précision que celle des autres points de l'espace. Lorsque la construction adaptative du plan d'expériences a pour but d'enrichir les zones optimales, les points améliorants sont choisis en résolvant un problème proche du problème initial. Les solutions optimales font alors certainement partie du plan d'expériences et elles sont ainsi disponibles sans calcul supplémentaire. On peut aussi envisager de mener une optimisation plus précise sur les modèles finaux afin de bien converger vers le front de Pareto idéal du problème. Cette

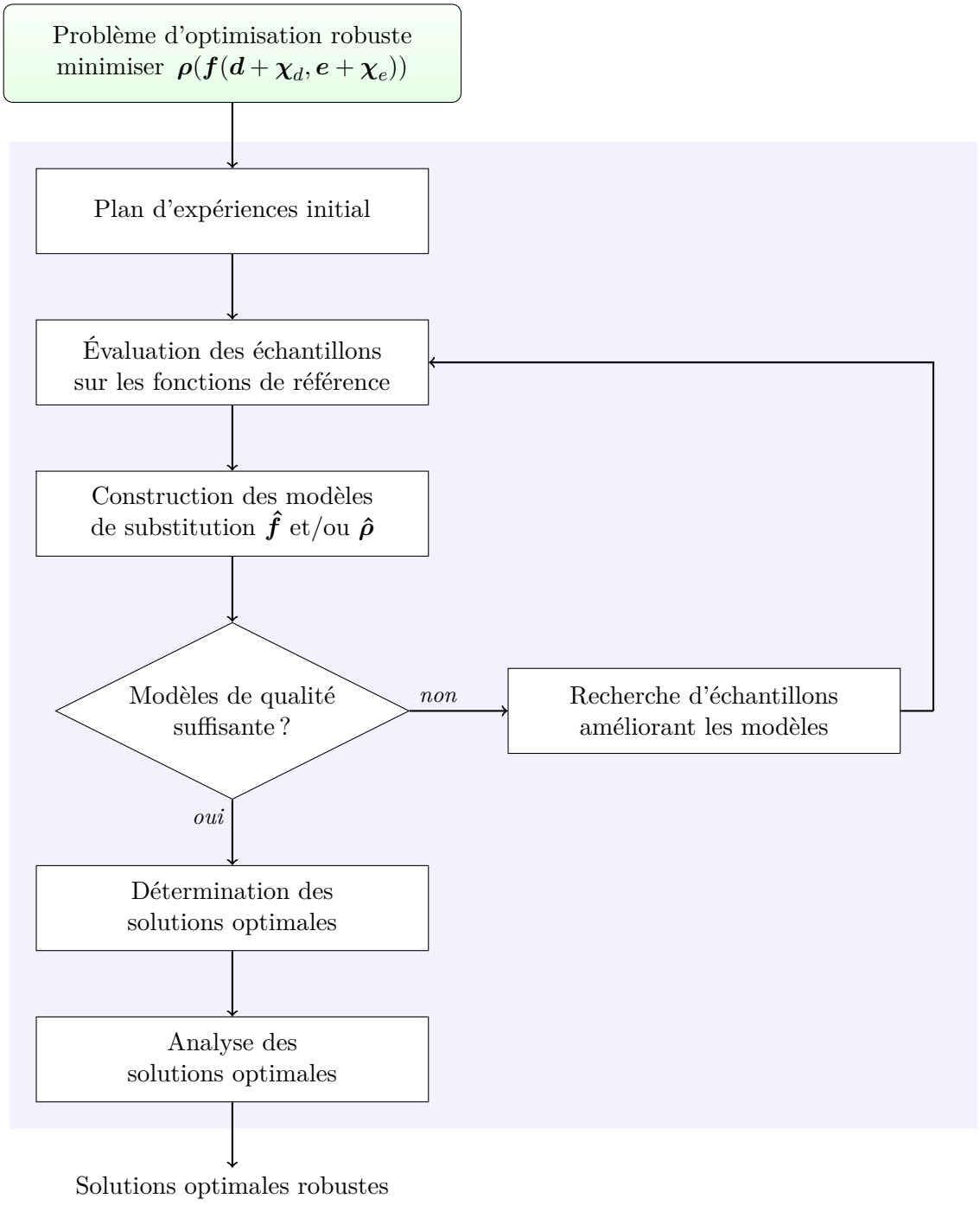


FIGURE 7.14 – Processus de résolution d'un problème d'optimisation robuste à l'aide de modèles de substitution.

optimisation a posteriori est d'autant plus nécessaire quand les modèles sont améliorés de manière globale sans recherche des zones optimales. Les algorithmes d'optimisation multiobjectifs retournent par contre en règle générale un nombre assez important de solutions. Afin de ne pas submerger le décideur avec de multiples propositions différentes, on peut alors sélectionner quelques solutions optimales parmi les plus représentatives en réalisant un élagage du front de Pareto obtenu (voir la section 4.4). On obtient au final un certain nombre de solutions robustes pour le problème posé.

Une dernière étape facultative consiste à analyser plus en détail les solutions optimales choisies. La robustesse de ces points a en effet été estimée à l'aide de modèles de substitution. Même si la valeur exacte de leur robustesse n'est pas accessible (car cela nécessiterait de connaître la valeur des fonctions de référence sur l'ensemble de leur voisinage), on peut néanmoins la calculer de manière plus précise dans le but de valider les résultats de l'optimisation. Si cela est possible, leur robustesse peut par exemple être évaluée sur les fonctions de référence elles-mêmes à l'aide de la méthode de Monte-Carlo ou bien d'un modèle local de type polynôme de chaos. Dans le cas test des deux barres, nous avons ainsi recalculé la robustesse des solutions du front de Pareto par tirages de Monte-Carlo sur les fonctions de référence. Pour un problème réel, cela ne serait bien sûr effectué que pour les quelques solutions sélectionnées au préalable. Une fois que ces solutions optimales robustes ont été validées, elles peuvent enfin être transmises au décideur qui fera son choix final parmi elles.

7.6 Conclusion

Nous avons décrit dans ce chapitre le déroulement de la définition et de la résolution d'un problème d'optimisation robuste. À partir d'un problème déterministe initial, l'identification des incertitudes présentes sur les variables d'entrée permet d'exprimer ce problème sous sa forme stochastique. Les incertitudes doivent être modélisées de manière réaliste à partir des informations disponibles afin de ne pas fausser les résultats futurs. Le problème stochastique est ensuite converti en un nouveau problème déterministe (mais robuste cette fois) à l'aide de mesures de robustesse. Ces mesures visent à quantifier la sensibilité des solutions par rapport aux fluctuations induites par les paramètres incertains, et doivent être choisies avec soin car elles déterminent les propriétés des solutions optimales recherchées. Afin de résoudre le problème d'optimisation robuste ainsi formulé, on met en place une stratégie d'estimation de la robustesse des solutions. Ces stratégies se basent en général sur des modèles de substitution des fonctions de référence, car un calcul direct serait bien trop coûteux. Il devient alors nécessaire de pouvoir maîtriser l'erreur de ces modèles pour garantir la qualité de leurs prédictions. La construction adaptative d'un plan d'expériences peut ainsi être intégrée à la boucle d'optimisation. C'est ce à quoi nous allons nous intéresser dans le chapitre suivant.

Chapitre 8

Plans d'expériences adaptatifs pour l'optimisation robuste multiobjectifs

Sommaire

8.1 Plans d'expériences adaptatifs pour l'amélioration globale de modèles . 164
8.1.1 Amélioration d'un unique modèle 164
8.1.2 Ajout de plusieurs échantillons à la fois 169
8.1.3 Amélioration simultanée de plusieurs modèles 172
8.2 Plans d'expériences pour l'optimisation déterministe (méthode PareBO) 180
8.2.1 Prise en compte de l'erreur des modèles en optimisation multiobjectifs . . . 180
8.2.2 Adaptation de l'algorithme NSGA-II 182
8.2.3 Enrichissement de modèles pour l'optimisation multiobjectifs 183
8.3 Plans d'expériences pour l'optimisation robuste (méthode PareBRO) . . 189
8.4 Conclusion 205

L'optimisation de fonctions coûteuses est limitée par le nombre restreint d'évaluations des objectifs qu'il est possible d'effectuer en un temps « raisonnable ». Pour obtenir des résultats satisfaisants, on se tourne alors vers des modèles de substitution capables de prédire le comportement de ces fonctions à partir de quelques échantillons seulement. L'utilisation de modèles pose toutefois le problème de la gestion de leur erreur d'approximation. Les données exploitées durant l'optimisation sont en effet des valeurs approchées obtenues grâce aux modèles et non sur les fonctions de référence elles-mêmes. Cela peut amener l'algorithme d'optimisation à ignorer certaines solutions optimales qu'il considère comme inintéressantes à cause de l'erreur des modèles, ou bien à sélectionner des solutions qui sont en réalité peu performantes. Il est ainsi utile de prendre en compte l'erreur de modélisation durant la résolution d'un problème. Le moyen le plus efficace pour minimiser son impact sur les résultats de l'optimisation serait d'établir des modèles de substitution les plus précis possibles à partir d'un grand nombre d'échantillons d'apprentissage. Mais le quota de points qu'il est permis d'évaluer sur les fonctions de référence est limité, et

on cherche plutôt à planifier ces évaluations le plus intelligemment possible. Le but est de calculer uniquement les échantillons les plus pertinents, c'est-à-dire ceux qui amélioreront notablement la qualité des modèles et qui permettront de résoudre le problème d'optimisation avec le moins d'erreur possible.

Peu de solutions ont été proposées à ce sujet dans le cadre de l'optimisation robuste. La motivation de ce chapitre est donc de définir une méthode de construction adaptative de plans d'expériences dédiés à l'optimisation robuste. Le contexte multiobjectifs de nos applications nous oblige de plus à gérer plusieurs modèles de substitution simultanément. Ces modèles sont construits à partir des sorties d'un même code de calcul, et ils partagent donc le même plan d'expériences. Les évaluations des fonctions de référence doivent alors être choisies en vue d'améliorer tous les modèles à la fois. Nous avons enfin la possibilité d'évaluer plusieurs échantillons en parallèle afin d'accélérer le processus d'optimisation. La méthode recherchée devra donc tenir compte de toutes ces particularités.

Nous avons développé notre méthode adaptative de façon séquentielle, en complexifiant le problème considéré peu à peu. Ce parcours est retracé dans ce chapitre. Nous allons ainsi nous intéresser dans un premier temps à l'amélioration globale de modèles de substitution dans le but de faire diminuer leur erreur sur l'ensemble de leur domaine de définition. Nous étudierons ensuite la prise en compte de l'erreur des modèles dans le contexte de l'optimisation déterministe puis robuste, afin d'améliorer les modèles au niveau des zones optimales uniquement et de prédire au mieux les solutions d'un problème.

8.1 Plans d'expériences adaptatifs pour l'amélioration globale de modèles

Avant d'aborder la construction adaptative de plans d'expériences pour l'optimisation (qui se concentre sur les régions où sont possiblement situés les optima), nous allons nous intéresser à l'amélioration globale de modèles. Nous cherchons ici à accroître leur qualité sur l'ensemble de leur espace de définition. Une stratégie d'amélioration globale de modèle peut par exemple être utilisée dans l'optique de remplacer totalement une fonction par un modèle de substitution au sein d'un système multidisciplinaire : il faut que le modèle possède une qualité suffisante sur la totalité de l'espace car on ne sait pas a priori quelles zones vont être explorées par la suite. On peut aussi employer cette stratégie avant de réaliser une optimisation, afin de démarrer la recherche de solutions optimales sur un modèle relativement représentatif de la fonction de référence.

De nombreux travaux se sont penchés sur l'amélioration globale d'un unique modèle de substitution, en se basant sur différentes mesures de son erreur. Nous revenons dans un premier temps sur ces méthodes pour ensuite étudier les possibilités offertes par l'évaluation en parallèle de plusieurs échantillons sur les fonctions de référence. Nous étudierons enfin l'amélioration simultanée de plusieurs modèles.

8.1.1 Amélioration d'un unique modèle

Nous avons décrit dans l'état de l'art les méthodes existantes pour l'amélioration globale d'un modèle (voir la section 3.8.2). Ces méthodes visent à réduire l'erreur du modèle et dépendent donc de la mesure d'erreur considérée. Les différentes mesures de l'erreur globale d'un modèle permettent ainsi de définir différentes stratégies d'amélioration.

Algorithme 8.1 : Construction d'un plan d'expériences adaptatif pour l'amélioration globale d'un modèle

```
1 PDE ← PDEinitial
2 Évaluer_avec_fonction_de_référence(PDE)
3 tant que le critère d'arrêt n'est pas vérifié faire
4     f̂ ← Construire_modèle(PDE)
5     p ← Chercher_point_à_ajouter(f̂)
6     Évaluer_avec_fonction_de_référence(p)
7     PDE ← PDE ∪ {p}
8 fin
```

Mesures de l'erreur d'un modèle

La qualité d'un modèle est généralement évaluée en termes d'erreur moyenne (ErrMoy) ou bien d'erreur maximale (ErrMax). On retrouve ici les deux types de mesures utilisées en robustesse : soit une espérance pour l'erreur moyenne, soit un quantile pour l'erreur maximale (on pourrait aussi choisir de considérer l'erreur à 95% par exemple pour avoir une mesure moins conservatrice). Ces mesures d'erreur peuvent être estimées à partir de données analytiques fournies par le modèle de substitution (la variance du modèle de krigeage par exemple), ou bien grâce à des méthodes statistiques (en utilisant une base de points de test ou bien la méthode du bootstrap).

Amélioration de l'erreur d'un modèle

Le principe de l'amélioration adaptative d'un modèle $\hat{f}$ est résumé dans l'algorithme 8.1. En partant d'un plan d'expériences initial $\boldsymbol{PDEinitial}$ (qui peut être un LHS par exemple), on construit un premier modèle de substitution $\hat{f}$ de la fonction de référence puis on recherche le point p qui va permettre d'améliorer le plus possible l'erreur de ce modèle. Une fois trouvé, ce point est évalué avec la fonction de référence et ajouté au plan d'expériences. On recommence ensuite la construction de modèle et la recherche de point améliorant jusqu'à ce qu'un certain critère d'arrêt soit satisfait. Ce critère peut par exemple être un seuil de qualité du modèle à atteindre, ou bien un quota d'évaluations de la fonction de référence à ne pas dépasser.

La recherche du meilleur point à ajouter au plan d'expériences est au cœur des méthodes adaptatives. On veut trouver le point qui une fois évalué assurera une erreur de modèle minimale. Il faut donc résoudre le problème suivant :

$$\underset{\boldsymbol{p}\in\boldsymbol{X}}{\operatorname{argmin}} \ \operatorname{Err}(\hat{f}^{+\boldsymbol{p}}), \tag{8.1}$$

avec $\hat{f}^{+\boldsymbol{p}}$ le modèle $\hat{f}$ auquel on a ajouté l'échantillon d'apprentissage $\boldsymbol{p}$, et Err une mesure de l'erreur globale de ce modèle. La difficulté vient du fait que l'image du point $\boldsymbol{p}$ par la fonction de référence n'est pas connue lors de la résolution de ce problème. On ne peut donc qu'estimer son impact sur l'erreur globale du modèle (cette erreur devant elle-même être approchée à partir des données qu'on possède). Si on considère la mesure de l'erreur maximale ErrMax, le critère d'amélioration correspondant est le critère MMSE pour lequel

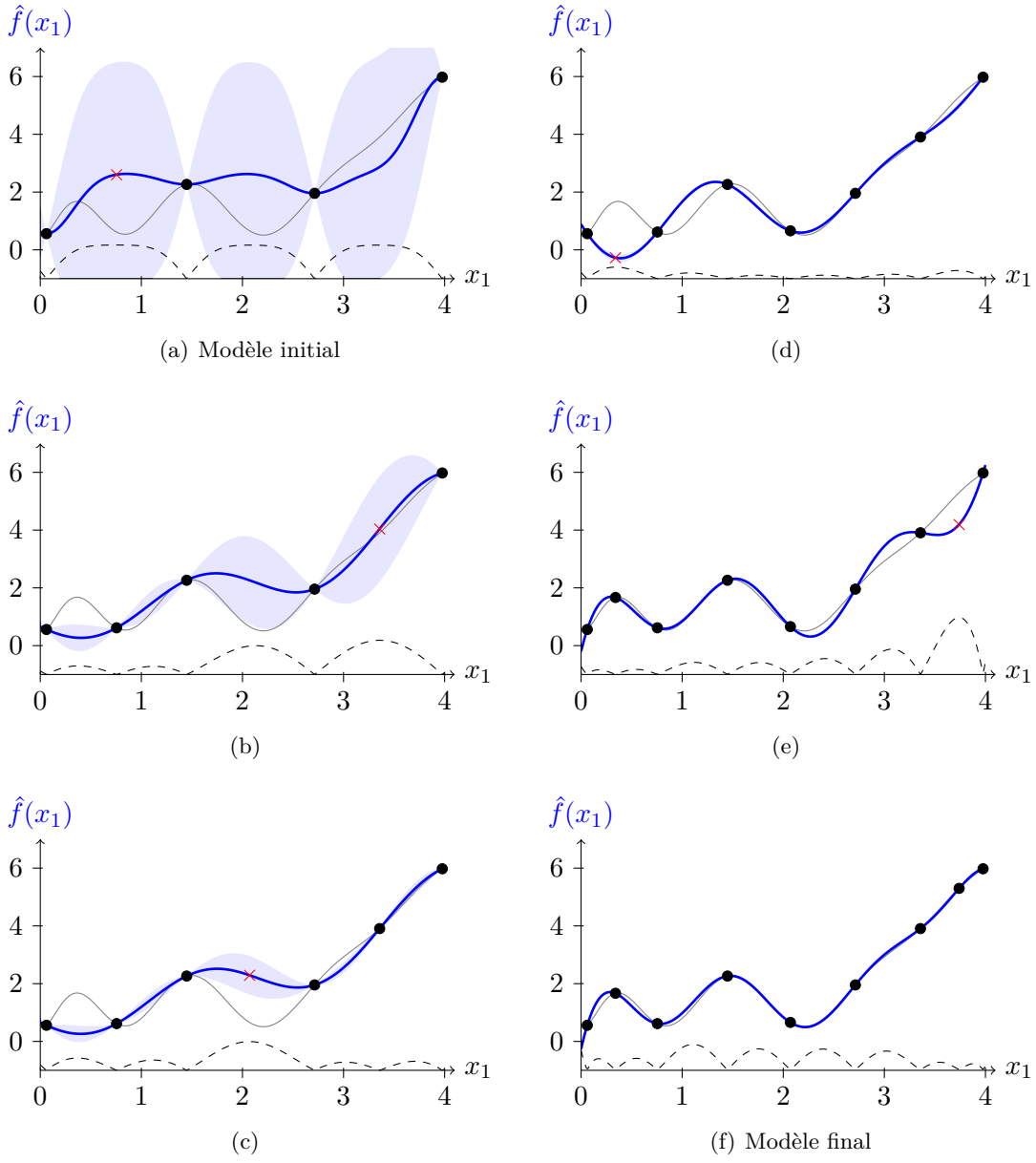
le problème (8.1) revient tout simplement à trouver le point qui possède la plus grande variance de modélisation sur le processus stochastique $\hat{F}$ lié au modèle $\hat{f}$:

$$\mathrm{MMSE}(\hat{F}) = \underset{\boldsymbol{p}\in\boldsymbol{X}}{\operatorname{argmax}}\ \mathrm{Var}(\hat{F}(\boldsymbol{p})). \tag{8.2}$$

La figure 8.1 présente un exemple d'amélioration d'un modèle de krigeage par ajout du point de plus grande variance à chaque itération. Partant d'un modèle grossier avec de larges intervalles de confiance, on obtient au final un modèle qui approche relativement bien la fonction à représenter.

Nous avons expérimenté cette méthode d'amélioration globale avec le critère MMSE sur un exemple en deux dimensions visant à approcher la fonction de Rosenbrock (définie par l'équation (4.5)) par un modèle de krigeage. Il s'agit de construire un plan d'expériences de manière adaptative à partir d'un LHS initial de 3 points. Un exemple de plan final obtenu (constitué de 18 échantillons) est présenté sur la figure 8.2. On voit que ce critère d'amélioration a tendance à ajouter des points sur les frontières de l'espace, car c'est souvent dans ces zones que le modèle est le plus mal défini. Les maxima de la fonction de Rosenbrock sont de plus situés sur des frontières et l'erreur absolue du modèle y est donc possiblement élevée. Pour modéliser des fonctions possédant une large plage de variation comme c'est le cas ici, il serait plus judicieux d'utiliser une erreur relative afin que les régions aux valeurs importantes ne soient pas privilégiées par rapport aux autres lors de l'enrichissement. Nous avons toutefois employé ici l'erreur absolue afin de rendre les particularités du critère MMSE plus visibles. La fonction de Rosenbrock est globalement bien représentée par le modèle final, même si les zones de faible valeur sont moins bien décrites que les autres (mais leur erreur absolue reste faible). La figure 8.3 présente l'évolution des erreurs maximale et moyenne du modèle de krigeage lors de l'enrichissement adaptatif du plan d'expériences. Ces erreurs ont été calculées grâce à une base de test de 1000 points dont la valeur réelle est connue, et moyennées sur 100 exécutions différentes de l'enrichissement. Nous avons comparé l'ajout de points selon le critère MMSE à l'utilisation d'un LHS de taille équivalente sans construction adaptative, ainsi qu'à une amélioration « idéale » de l'erreur maximale du modèle. Le critère MMSE est en effet basé sur une estimation de la position du point le plus mal décrit par le modèle (obtenue à partir de la variance du modèle de krigeage). Sur notre exemple analytique, il est possible d'ajouter le véritable point de plus grande erreur trouvé en comparant le modèle à la fonction de référence. On obtient ainsi l'erreur de modèle idéale qui aurait pu être atteinte si on avait eu connaissance de l'erreur exacte en chaque point. On voit sur le graphique d'évolution de l'erreur maximale que le critère MMSE n'obtient pas d'aussi bons résultats que l'amélioration idéale du fait de l'approximation réalisée lors de l'estimation de l'erreur du modèle, mais il permet tout de même de construire des plans d'expériences donnant de meilleurs modèles qu'un LHS de même taille. L'erreur moyenne est par contre moins bonne que celle obtenue avec un LHS, ce qui est normal car on n'a pas cherché à l'améliorer ici en rajoutant le point du modèle le plus mal décrit. L'amélioration d'un modèle dépend donc du critère utilisé et de la mesure d'erreur considérée. Ces résultats montrent que l'utilisation du critère MMSE permet effectivement d'aboutir à des modèles possédant une bonne erreur maximale.

Si on souhaite au contraire réduire l'erreur moyenne du modèle de substitution, on peut se baser sur le critère IMSE pour rechercher le point qui apportera la plus grande amélioration moyenne une fois ajouté à la base d'apprentissage. La construction de plans d'expériences adaptatifs avec ce critère est par contre plus complexe car son optimisation



(a) Modèle initial (b) (c) (d) (e) (f) Modèle final

FIGURE 8.1 – Exemple d'amélioration adaptative d'un modèle de krigeage par ajout du point de variance maximale. La fonction f a modéliser est représentée en gris. Les points noirs sont les échantillons d'apprentissage, à partir desquels est construit un modèle de krigeage $\hat{f}$ représenté en bleu (avec son intervalle de confiance à 95%). La variance du modèle est tracée en pointillés au bas de chaque graphe (sans respect d'échelle). Le point maximisant cette variance est indiqué d'une croix rouge.

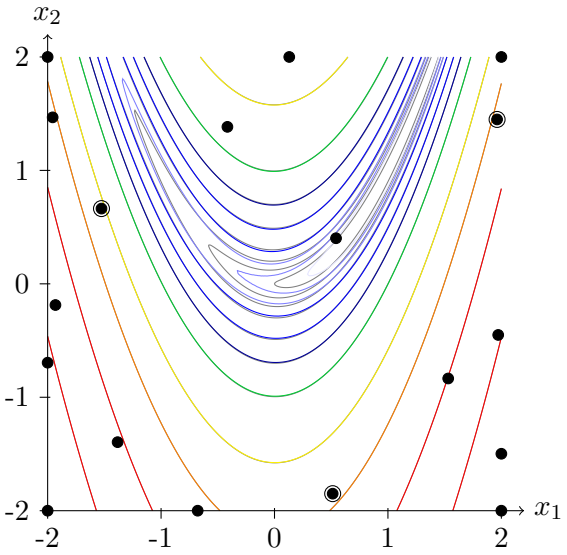


FIGURE 8.2 – Exemple de plan d'expériences (points noirs) construit par amélioration adaptative à partir d'un LHS initial de trois points (indiqués par des cercles) pour représenter la fonction de Rosenbrock par un modèle de krigeage. Le point possédant la plus grande variance de modèle a été rajouté à chaque itération. Les courbes de niveau de la fonction de référence sont représentées en gris et celles du modèle final en couleur.

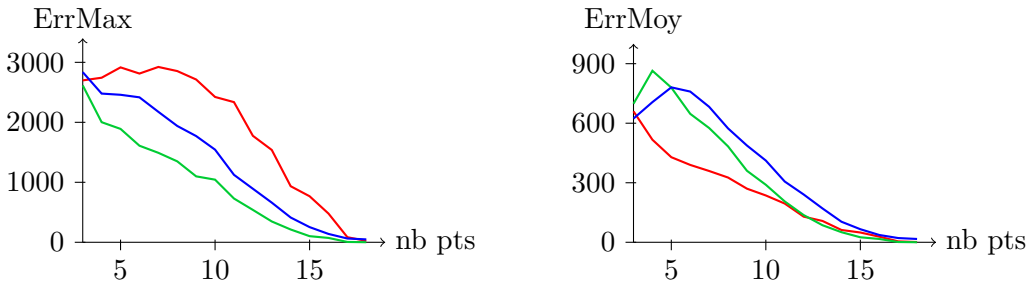


FIGURE 8.3 – Évolution moyenne en fonction du nombre de points d'apprentissage des erreurs maximale (à gauche) et moyenne (à droite) du modèle de krigeage de la fonction de Rosenbrock. La méthode MMSE est représentée en bleu, l'approche « idéale » par ajout du point de plus grande erreur réelle du modèle en vert, et l'utilisation d'un LHS sans construction adaptative en rouge.

nécessite de simuler l'ajout de points au modèle (voir la section 3.8.3) afin de voir leur effet sur son erreur moyenne, alors que le critère MMSE dépend directement de la variance du modèle en cours.

Le choix d'un critère d'amélioration ou d'un autre se fait en fonction de la manière dont l'erreur globale du modèle de substitution est mesurée. Dans les problèmes industriels qui nous intéressent, l'erreur d'un modèle est souvent évaluée en termes de « pire cas » : on souhaite avoir une erreur de modèle inférieure à un seuil donné (absolu ou bien relatif). L'amélioration va donc devoir se focaliser sur le point qui présente la plus grande erreur d'approximation. Nous utiliserons ainsi par la suite le critère MMSE. Ce critère a l'avantage d'être simple à calculer, mais il a tendance à favoriser les frontières de l'espace de recherche. On pourrait aussi imaginer des approches mixtes visant à améliorer l'erreur maximale du modèle dans un premier temps, puis son erreur moyenne. On obtiendrait ainsi des modèles corrects à la fois en erreur maximale et en erreur moyenne.

8.1.2 Ajout de plusieurs échantillons à la fois

L'approche adaptative que nous venons d'étudier détermine à chaque itération la position d'un seul et unique point à ajouter au plan d'expériences. Mais on a parfois à disposition des moyens de calcul qui permettent d'effectuer plusieurs évaluations de la fonction de référence en parallèle pour obtenir un certain nombre d'échantillons d'apprentissage (noté $NombrePointsParallèles$) dans la durée d'un seul calcul. L'amélioration de la qualité du modèle est ainsi accélérée grâce à la parallélisation des évaluations. Dans ce cadre, on cherche à déterminer à chaque itération de l'algorithme adaptatif la position de plusieurs points prometteurs à calculer simultanément.

La recherche de plusieurs points améliorants nécessite de mettre en place des techniques plus évoluées que dans le cas d'un point unique. Il faut en effet estimer l'impact sur le modèle de l'ensemble des points choisis en tentant compte de leurs interactions. Si on ne prend pas ces dernières en considération, il est possible que deux points sélectionnés soient très proches l'un de l'autre (voire identiques), et leur évaluation sur la fonction de référence n'apportera alors pas plus d'information que l'évaluation d'un seul point. Les échantillons retenus doivent donc être suffisamment distincts pour que chacun puisse participer à l'amélioration globale du modèle.

Pour sélectionner différents points à ajouter au plan d'expériences, on peut par exemple découper l'espace en plusieurs parties et rechercher un point améliorant dans chacune d'elles. Cette approche relativement simple permet d'éviter de rajouter des points trop proches (sauf si ceux-ci sont situés près de la frontière de deux zones adjacentes), mais son inconvénient est qu'elle considère au même plan chacune des régions alors que certaines sont peut-être déjà suffisamment bien décrites par le modèle. Des stratégies plus avancées de raffinement de la partition de l'espace pourraient être envisagées pour favoriser les zones difficiles à modéliser par rapport aux régions où le modèle est assez précis.

Une autre approche pour choisir de multiples échantillons à évaluer en parallèle est de trouver un premier point par les techniques traditionnelles (en utilisant par exemple le critère MMSE), puis de simuler l'ajout de ce point à la base d'apprentissage du modèle avant de chercher le point améliorant suivant de la même manière. On prend ainsi en considération les points sélectionnés précédemment lors de la recherche d'un nouveau point améliorant, même si ceux-ci n'ont pas encore été réellement évalués avec la fonction de référence. Le principe de cette méthode est décrit dans l'algorithme 8.2. Elle repose

Algorithme 8.2 : Construction d'un plan d'expériences adaptatif pour l'amélioration globale d'un modèle avec ajout de plusieurs échantillons à la fois

```
1  PDE ← PDEinitial
2  Évaluer_avec_fonction_de_référence(PDE)
3  tant que le critère d'arrêt n'est pas vérifié faire
4      f̂ ← Construire_modèle(PDE)
5      NouveauxPoints ← ∅
6      pour i ∈ {1, 2, ..., NombrePointsParallèles} faire
7          p ← Chercher_point_à_ajouter(f̂)
8          NouveauxPoints ← NouveauxPoints ∪ {p}
9          Simuler_ajout(p, f̂)
10     fin
11     Évaluer_avec_fonction_de_référence(NouveauxPoints)
12     PDE ← PDE ∪ NouveauxPoints
13 fin
```

donc sur la simulation de l'ajout d'un nouveau point p au modèle de substitution $\hat{f}$. Ce point sera ainsi pris en compte dans l'estimation de l'erreur du modèle lors de la recherche du point améliorant suivant. Comme pour tous les échantillons d'apprentissage, l'erreur sera relativement faible autour de p, ce qui évitera de sélectionner un autre point améliorant dans son voisinage. Les diverses possibilités pour simuler l'ajout d'un échantillon d'apprentissage à un modèle de substitution sont décrites dans la section 3.8.3. Selon le type de modèle considéré, cet ajout « virtuel » peut être effectué à partir de la position du point uniquement ou bien en supposant une valeur fictive pour son image par la fonction de référence. Le modèle peut aussi être reconstruit ou non après ajout du nouveau point. Il ne s'agit bien entendu que d'une méthode approchée car elle repose sur une estimation de l'impact de chaque point sur l'erreur du modèle sans connaître sa vraie valeur sur la fonction de référence. Les estimations s'éloignent donc de plus en plus de la réalité au fur et à mesure des ajouts « virtuels » de points. Cette méthode est ainsi à réserver aux cas où le nombre de calculs en parallèle reste relativement limité.

À titre d'illustration, la figure 8.4 présente une itération de cet algorithme sur un exemple. Il s'agit de rajouter simultanément deux points d'apprentissage au plan d'expériences initial afin d'améliorer la qualité d'un modèle de krigeage. En suivant le critère MMSE, on commence donc par sélectionner le point de plus forte variance dont on simule l'ajout au modèle. La prise en compte de ce point supplémentaire dans l'estimation de l'erreur du modèle de krigeage est relativement simple, et permet d'obtenir un second point améliorant correspondant au nouveau maximum de la variance du modèle (celle-ci étant devenue nulle au niveau du premier point sélectionné). Une fois ces deux points choisis, ils sont réellement évalués sur la fonction de référence et le modèle est reconstruit avec ces nouvelles informations. On peut ensuite continuer à effectuer des itérations d'amélioration de la même manière jusqu'à ce que le modèle atteigne une qualité suffisante ou bien jusqu'à ce que le quota d'évaluations qu'il est permis de réaliser sur la fonction de référence soit atteint.

Nous avons mis en œuvre cette stratégie d'ajout simultané de plusieurs points sur l'exemple de la fonction de Rosenbrock de la section précédente (figure 8.2). Nous suppo-

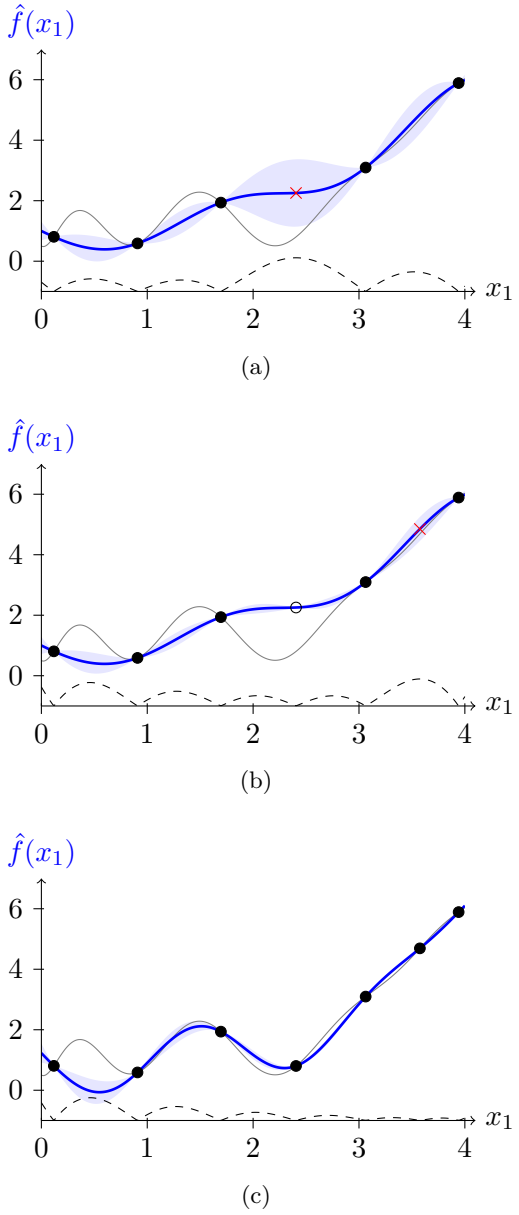


FIGURE 8.4 – Exemple d'amélioration d'un modèle $\hat{f}$ par ajout simultané de deux points. La fonction de référence est représentée en gris, les échantillons d'apprentissage par des points noirs, le modèle en bleu (avec son intervalle de confiance à 95%) et la variance du modèle en pointillés (sans respect d'échelle). Un premier point de variance maximale est sélectionné (croix rouge en haut), puis ajouté « virtuellement » au modèle (cercle noir au centre) pour rechercher un second point de variance maximale (croix rouge au centre). Les deux points sont ensuite évalués avec la fonction de référence et le modèle est reconstruit (en bas).

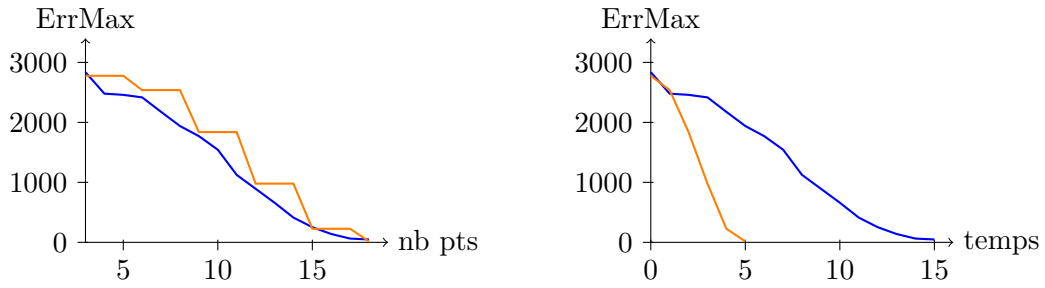


FIGURE 8.5 – Évolution moyenne de l'erreur maximale du modèle de krigeage de la fonction de Rosenbrock en fonction du nombre de points d'apprentissage (à gauche) et du temps de calcul (à droite, une unité correspondant au temps d'évaluation d'un échantillon). L'amélioration point par point est représentée en bleu, et l'amélioration en rajoutant trois points à la fois en orange.

sons ici que les échantillons peuvent être évalués trois par trois. L'évolution de l'erreur du modèle obtenu est comparée à l'approche point par point initiale sur la figure 8.5. On peut voir que les erreurs des deux méthodes sont relativement similaires lorsqu'on les trace en fonction du nombre de points présents dans le plan d'expérience. L'erreur de la nouvelle approche évolue par paliers car plusieurs points sont ajoutés simultanément. Elle reste tout de même légèrement plus mauvaise que l'approche point par point, car à chaque itération la détermination des trois points améliorants est réalisée à partir des mêmes informations de départ, alors que dans la stratégie initiale on a connaissance de la valeur réelle de tous les points un par un. Les estimations par ajout simulé engendrent donc une légère perte de précision, mais si on compare l'évolution de l'erreur en fonction du temps de calcul on voit que la parallélisation des évaluations de la fonction de référence permet de construire un modèle de qualité beaucoup plus rapidement. Ces résultats valident ainsi l'approche par simulation d'ajout de point qui offre la possibilité d'évaluer plusieurs échantillons en parallèle pour accélérer l'enrichissement d'un modèle.

8.1.3 Amélioration simultanée de plusieurs modèles

Dans le cadre de simulations numériques complexes comme celles utilisées dans cette étude, on peut aussi établir plusieurs modèles de substitution en même temps. Chaque simulation permet en effet d'évaluer simultanément différents observables qui vont être représentés par autant de modèles. On souhaite alors améliorer l'erreur de tous les modèles à la fois en ajoutant au plan d'expériences commun des échantillons qui profitent à tous. À notre connaissance, il y a peu d'études dans la littérature qui visent à améliorer simultanément plusieurs modèles de substitution de manière globale.

Pour poursuivre dans la même optique que précédemment en considérant le critère MMSE, nous allons ici chercher à améliorer le plus mauvais point de chaque modèle (des méthodes similaires pourraient être employées pour améliorer l'erreur moyenne en employant d'autres critères). On peut imaginer deux stratégies d'amélioration différentes : soit on améliore chacun des modèles alternativement avec des méthodes classiques, soit on tente d'améliorer simultanément l'ensemble des modèles. Ces deux possibilités sont détaillées ci-après.

Algorithme 8.3 : Construction d'un plan d'expériences adaptatif pour plusieurs modèles par amélioration alternative

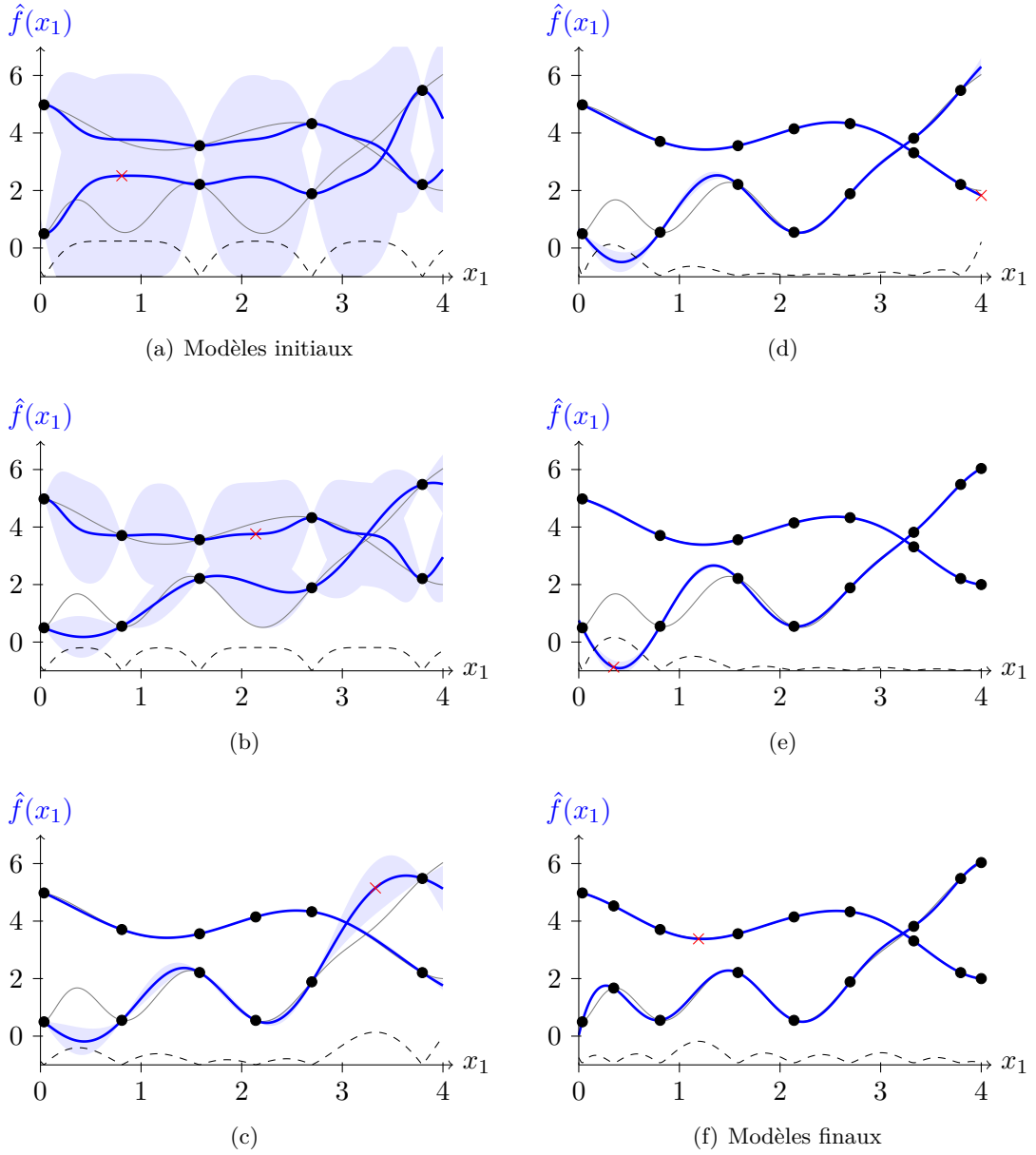
```
1  PDE ← PDEinitial
2  Évaluer_avec_fonctions_de_référence(PDE)
3  i ← 1
4  tant que le critère d'arrêt n'est pas vérifié faire
5      f̂_i ← Construire_modèle(PDE, i)
6      p ← Chercher_point_à_ajouter(f̂_i)
7      Évaluer_avec_fonctions_de_référence(p)
8      PDE ← PDE ∪ {p}
9      i ← (i mod n) + 1
10 fin
```

Amélioration alternative de chaque modèle

Une première idée pour enrichir plusieurs modèles à la fois est d'améliorer chacun des modèles tour à tour. On peut alors utiliser les méthodes classiques d'amélioration d'un unique modèle en changeant de modèle considéré à chaque itération. On gère donc maintenant un vecteur $\hat{\boldsymbol{f}} = (\hat{f}_1, \hat{f}_2, ..., \hat{f}_n)$ de n modèles de substitution, et la recherche de point améliorant se fait sur l'un des modèles seulement à chaque étape. Le principe de cette approche est présenté dans l'algorithme 8.3.

La figure 8.6 illustre cet algorithme sur un exemple simple d'amélioration de deux modèles de krigeage. Partant d'un plan d'expériences initial composé de quatre échantillons, on commence par rechercher le point le plus mal décrit par le premier modèle (c'est-à-dire le point de plus forte variance), puis ce point est calculé avec les fonctions de référence et les modèles reconstruits. On ajoute alors au plan d'expériences le plus mauvais point du second modèle, et ainsi de suite alternativement. On obtient au final deux modèles relativement représentatifs des fonctions initiales.

L'avantage de cette méthode est que la recherche du point à ajouter au plan d'expériences est relativement simple et qu'on n'a besoin de construire qu'un seul modèle de substitution à chaque itération (celui sur lequel va porter l'amélioration). Mais le point sélectionné n'est pas toujours le plus adapté à l'ensemble des modèles. Il se peut en effet qu'un des modèles soit plus précis que les autres à un moment donné, comme c'est le cas par exemple au niveau du graphe (d) de la figure 8.6. Le modèle amélioré lors de cette étape approche déjà relativement bien sa fonction de référence, et il aurait peut-être été profitable de chercher à améliorer le second modèle à la place. On pourrait ainsi imaginer de modifier l'algorithme pour enrichir de préférence le modèle qui présente la plus grande erreur. La stratégie alternative fonctionne bien lorsqu'on a peu de modèles à améliorer simultanément, mais si leur nombre augmente chacun des modèles n'est considéré que plus rarement. S'il peut arriver que les points de plus grande variance des modèles soient situés au même endroit, ce qui permet alors à tous les modèles de bénéficier de l'amélioration apportée par l'évaluation de cet échantillon, ces points ne sont pas nécessairement les mêmes en règle générale. Chacun doit alors attendre son tour avant de voir son erreur diminuer significativement. C'est pourquoi nous avons imaginé une autre stratégie d'amélioration qui prend en compte l'ensemble des modèles à chaque itération.



(a) Modèles initiaux (d)

(b) (e)

(c) (f) Modèles finaux

FIGURE 8.6 – Exemple d'amélioration simultanée de deux modèles de krigeage par ajout du point de plus grande variance de chaque modèle alternativement (visualisé par une croix rouge). Les fonctions de référence sont représentées en gris, les échantillons d'apprentissage par des points noirs et les modèles en bleu (avec leur intervalle de confiance à 95%). La variance du modèle concerné à chaque étape est tracée en pointillés (sans respect d'échelle).

Amélioration simultanée de tous les modèles

La stratégie d'amélioration alternative présentée ci-dessus favorise un des modèles à chaque itération. Mais on peut aussi chercher à enrichir tous les modèles de substitution à la fois en faisant des compromis. Le problème d'amélioration simultanée de plusieurs modèles peut en effet être vu comme une recherche multiobjectifs, chaque objectif étant d'améliorer un modèle donné.

L'amélioration d'un unique modèle $\hat{f}_i$ revient à déterminer le point qui, une fois évalué, permettra de faire diminuer au maximum son erreur globale (il s'agit par exemple du point de plus grande variance si on considère le critère MMSE). Nous noterons ce point $PointsÀAméliorer_i$. Lorsque plusieurs modèles entrent en jeu, on obtient ainsi autant de points qu'on aimerait voir évalués que de modèles. S'il n'est pas possible d'effectuer des évaluations en parallèle sur la fonction de référence, on ne peut pas connaître la valeur exacte de tous ces points lors d'une même itération. Mais on peut par contre essayer de l'estimer aussi précisément que possible : nous allons ainsi chercher à réduire simultanément la taille des intervalles de confiance des prédictions de ces points, ce qui améliorera par là même l'erreur de tous les modèles à la fois. Le problème à résoudre s'écrit :

$$\underset{\boldsymbol{p}\in\boldsymbol{X}}{\text{minimiser}} \begin{cases} |\operatorname{IC}(\hat{F}_1^{+\boldsymbol{p}}(PointsÀAméliorer_1))|, \\ |\operatorname{IC}(\hat{F}_2^{+\boldsymbol{p}}(PointsÀAméliorer_2))|, \\ \vdots \\ |\operatorname{IC}(\hat{F}_n^{+\boldsymbol{p}}(PointsÀAméliorer_n))|, \end{cases} \tag{8.3}$$

avec $\hat{F}_i^{+\boldsymbol{p}}$ le processus stochastique associé au modèle $\hat{f}_i$ auquel on a ajouté l'échantillon d'apprentissage $\boldsymbol{p}$, et IC l'intervalle de confiance à 95% de ses prédictions. Dans le cas du critère MMSE que nous avons choisi ici, cela revient à chercher un point de l'espace qui améliore la prédiction des points de plus grande variance de chaque modèle. Le fait de réaliser la recherche de point améliorant en deux temps (on localise d'abord les points de plus grande variance des modèles puis un point qui améliore les points précédents) sera de plus adapté au contexte de l'optimisation robuste dans lequel les deux espaces de recherche ne sont pas forcément les mêmes comme nous le verrons plus loin.

Le principe de la méthode d'amélioration simultanée est résumé dans l'algorithme 8.4. La seule différence par rapport aux approches précédentes est la recherche du point améliorant général visant à mieux prédire les points les plus mal décrits par chaque modèle (ligne 8 de l'algorithme). Cette recherche est réalisée en résolvant le problème (8.3). On ne souhaite toutefois obtenir qu'une seule solution optimale de ce problème multiobjectifs, puisqu'on ne va évaluer qu'un unique point sur les fonctions de référence à chaque itération. On peut donc agréger les différents objectifs afin de le transformer en problème mono-objectif. Nous avons choisi ici de normaliser ces objectifs avant de les sommer (on pourrait aussi utiliser d'autres agrégations comme le produit par exemple). Nous considérons ainsi l'amélioration relative AR_i apportée par un échantillon $\boldsymbol{p}$ à la prédiction d'un point $PointsÀAméliorer_i$ sur un modèle $\hat{f}_i$:

$$\mathrm{AR}_i(\boldsymbol{p}) = \frac{|\operatorname{IC}(\hat{F}_i(PointsÀAméliorer_i))| - |\operatorname{IC}(\hat{F}_i^{+\boldsymbol{p}}(PointsÀAméliorer_i))|}{|\operatorname{IC}(\hat{F}_i(PointsÀAméliorer_i))|}. \tag{8.4}$$

On peut noter que $\mathrm{AR}_i(\boldsymbol{p})$ est nul si $\boldsymbol{p}$ fait déjà partie des échantillons d'apprentissage de $\hat{f}_i$ (ajouter un échantillon déjà présent n'apporte aucune amélioration). Nous recherchons

Algorithme 8.4 : Construction d'un plan d'expériences adaptatif pour plusieurs modèles par amélioration simultanée

$\boldsymbol{PDE} \leftarrow \boldsymbol{PDEinitial}$
Évaluer_avec_fonctions_de_référence($\boldsymbol{PDE}$)
tant que *le critère d'arrêt n'est pas vérifié* **faire**
    pour $i \in \{1, 2, ..., n\}$ **faire**
        $\hat{f}_i \leftarrow$ Construire_modèle($\boldsymbol{PDE}, i$)
        $PointsÀAméliorer_i \leftarrow$ Chercher_point_à_ajouter($\hat{f}_i$)
    fin
    $p \leftarrow$ Chercher_point_améliorant_général($\boldsymbol{PointsÀAméliorer}, \hat{\boldsymbol{f}}$)
    Évaluer_avec_fonctions_de_référence(p)
    $\boldsymbol{PDE} \leftarrow \boldsymbol{PDE} \cup \{p\}$
fin

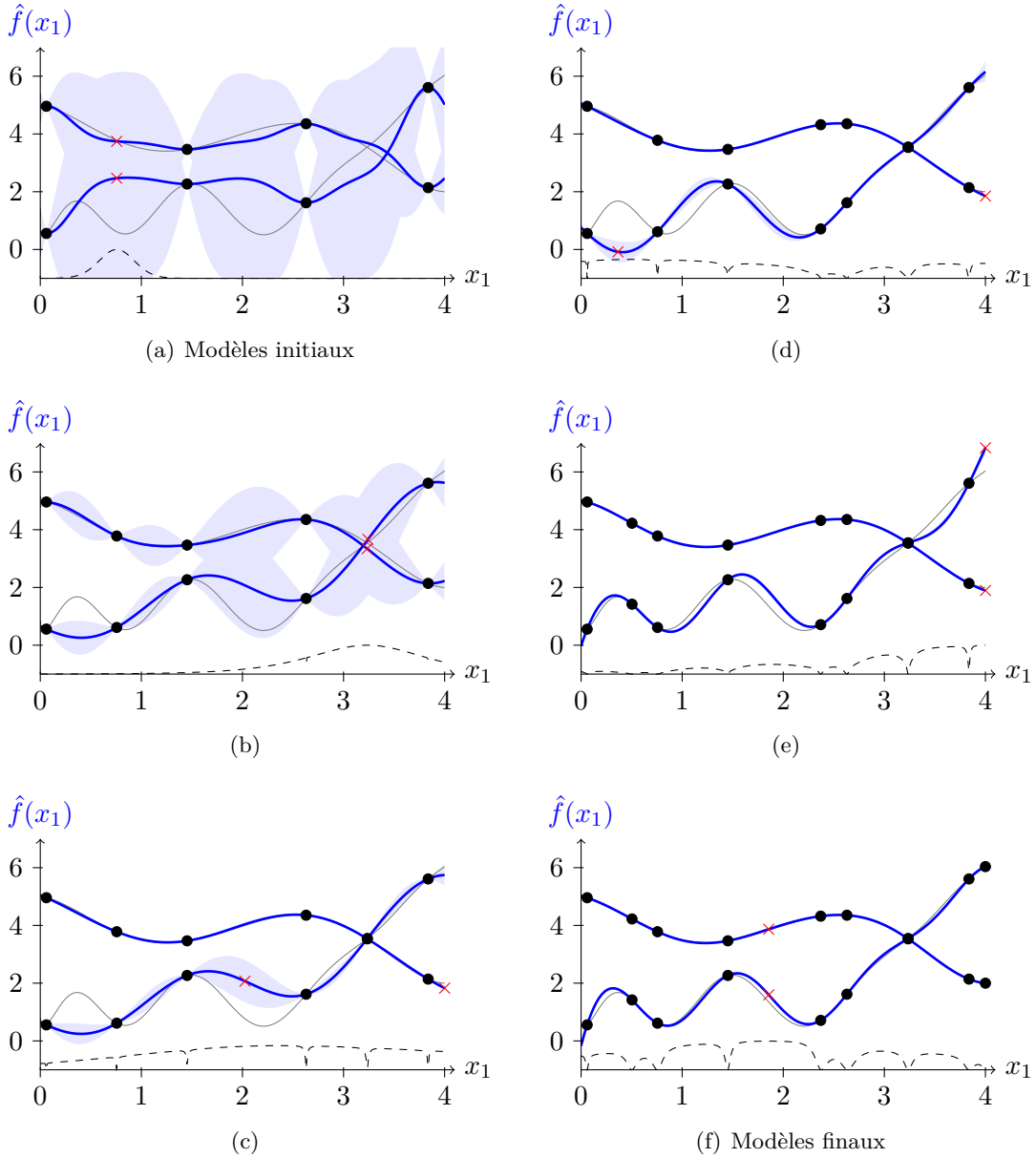
ensuite le point de l'espace qui maximise la somme des améliorations relatives :

$$\underset{\boldsymbol{p} \in \boldsymbol{X}}{\operatorname{argmax}} \sum_{i=1}^{n} \mathrm{AR}_i(\boldsymbol{p}). \tag{8.5}$$

On a ainsi un simple problème mono-objectif à traiter, dont la résolution fournit un point améliorant simultanément l'ensemble des modèles de substitution considérés.

La figure 8.7 présente un exemple d'amélioration de deux modèles de krigeage selon la méthode que nous venons de décrire. On peut constater que lorsque les points de plus grande variance des deux modèles sont situés au même endroit, c'est ce point qui est choisi pour être ajouté au plan d'expérience comme dans le cas d'une amélioration alternative (voir le graphe (a) par exemple). Quand cette situation se présente, on pourrait ainsi sélectionner directement le point sans effectuer de recherche avancée via le problème (8.5) afin d'accélérer un peu la procédure. Par contre, lorsque les points de plus grande variance sont différents, l'approche essaie de trouver un compromis entre les deux modèles. Au niveau du graphe (c) par exemple, on voit que l'amélioration atteint son maximum entre les deux points les plus mal décrits. Ce maximum est plus proche du point de gauche qui possède une variance relativement importante. L'autre modèle étant déjà bien défini, il ne peut pas bénéficier d'une amélioration aussi significative. Le nouveau point évalué se trouve donc à proximité du point de plus grande variance, tout en étant décalé en direction du second point à améliorer. Cette méthode permet ainsi de résoudre les conflits d'intérêt entre modèles lorsque les points qu'ils souhaitent voir améliorés ne sont pas les mêmes.

Nous avons aussi appliqué cette approche sur un cas test analytique en deux dimensions. Il s'agit de représenter conjointement la fonction de Rosenbrock (équation (4.5)) et la fonction « chameau » (équation (4.7)) à l'aide de modèles de krigeage. Ces deux fonctions sont représentées sur la figure 8.8. Partant d'un LHS initial constitué de trois échantillons d'apprentissage, le plan d'expériences est complété de manière adaptative en vue d'améliorer simultanément les deux modèles. Un exemple de plan d'expériences final obtenu est donné sur la figure 8.9. On constate que l'amélioration simultanée de plusieurs modèles a moins tendance à favoriser les frontières de l'espace de définition que l'enrichissement d'un seul modèle avec le critère MMSE. On est en effet amené à faire des compromis entre les différents modèles, et ces compromis se situent plus souvent à l'intérieur de l'espace de



(a) Modèles initiaux (d)

(b) (e)

(c) (f) Modèles finaux

FIGURE 8.7 – Exemple d'amélioration simultanée de deux modèles de krigeage avec notre méthode (algorithme 8.4). Les fonctions de référence sont représentées en gris, les échantillons d'apprentissage par des points noirs et les modèles en bleu (avec leur intervalle de confiance à 95%). On recherche tout d'abord les points de plus grande variance de chaque modèle (croix rouges) puis on estime l'amélioration apportée en ces deux points par tout point de l'espace (courbes en pointillés, sans respect d'échelle). Le point maximisant l'amélioration est évalué sur les fonctions de référence et ajouté au plan d'expériences avant de reconstruire les modèles.

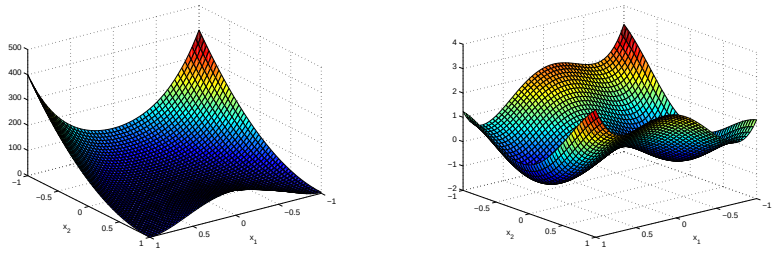


FIGURE 8.8 – Les fonctions de Rosenbrock et chameau qu'on souhaite modéliser simultanément.

définition plutôt qu'en bordure du domaine. Les modèles finaux représentent assez bien les deux fonctions de référence, même si on peut noter que la fonction chameau est plus difficile à approcher que la fonction de Rosenbrock. La figure 8.10 présente l'évolution de l'erreur maximale de chacun des modèles lors de la construction adaptative du plan d'expériences. Ces erreurs ont été calculées grâce à une base de test de 1000 points dont la valeur réelle est connue, et moyennées sur 100 exécutions différentes de l'enrichissement. Nous avons comparé l'ajout de points selon notre méthode à l'utilisation d'un LHS de taille équivalente sans construction adaptative, ainsi qu'à une amélioration alternative des deux modèles (approche présentée dans l'algorithme 8.3). On note que notre méthode et l'ajout alternatif ont des comportements similaires en termes d'erreur, et font globalement mieux qu'un LHS de même taille. L'approche d'amélioration simultanée permet donc de réduire l'erreur de modèles au même titre que l'amélioration alternative, mais sans pour autant la surpasser sur cet exemple. Les ressemblances dans l'évolution de l'erreur obtenue avec chacune des méthodes peuvent être liées au fait qu'il n'y a pas beaucoup de conflits entre les modèles durant la recherche de points améliorants, car les modèles sont ici en nombre restreint. Les points les plus mal décrits par chaque modèle peuvent donc se retrouver fréquemment au même endroit, et quand c'est le cas les deux méthodes deviennent similaires. L'avantage de notre méthode est qu'elle se comporte mieux dans le cas de conflits entre les modèles en compétition, conflits dont la fréquence augmente avec le nombre de modèles à prendre en compte.

Nous avons jusqu'ici considéré l'ajout d'un unique point au plan d'expériences à chaque itération de l'algorithme. S'il est possible d'effectuer plusieurs évaluations des fonctions de référence en parallèle, on peut imaginer de rechercher différentes solutions du front de Pareto du problème (8.3), que nous avions transformé en problème mono-objectif (8.5) pour n'obtenir qu'une seule solution. On aura ainsi plusieurs échantillons à évaluer qui améliorent plus ou moins chacun des modèles. Mais cette approche n'est pas très intéressante car les points ainsi trouvés ne tiennent pas compte les uns des autres : on peut donc se retrouver avec des points proches ou bien dont l'évaluation n'apporte pas plus d'information que celle d'un unique point. Il est alors préférable d'utiliser le même principe que lors de la recherche de plusieurs échantillons améliorants pour un seul modèle (algorithme 8.2) : on choisit d'abord un seul point selon la méthode proposée plus haut, puis on simule son ajout à la base d'apprentissage des modèles avant de rechercher le point améliorant suivant. Chaque point obtenu tient ainsi compte de tous ceux qui ont été sélectionnés auparavant. La figure 8.11 présente l'évolution de l'erreur des modèles des fonctions de Rosenbrock et chameau de l'exemple précédent lors de l'ajout de trois points

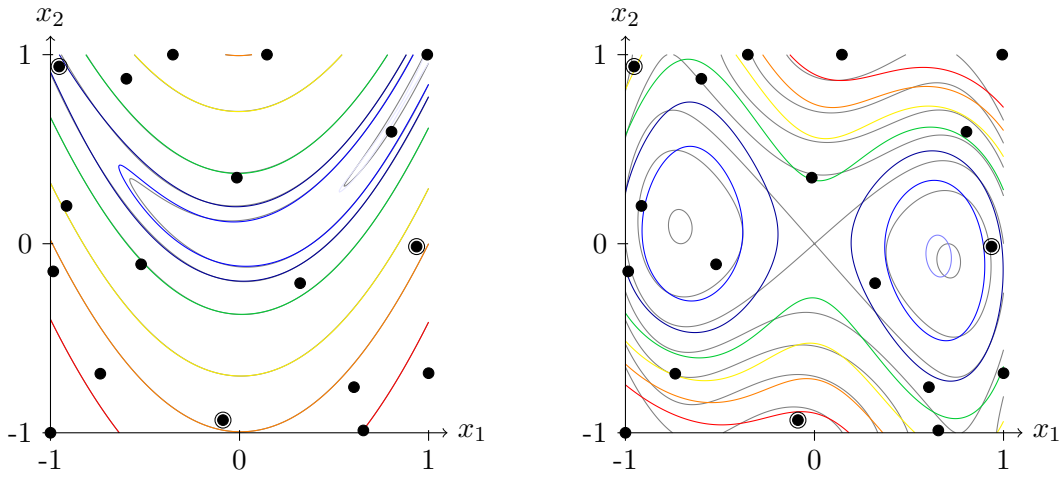


FIGURE 8.9 – Exemple de plan d'expériences (points noirs) obtenu par amélioration adaptative à partir d'un LHS initial de trois points (indiqués par des cercles) pour représenter simultanément les fonctions de Rosenbrock (à gauche) et chameau (à droite) par des modèles de krigeage. Les courbes de niveau des fonctions de référence sont représentées en gris et celles des modèles finaux en couleur.

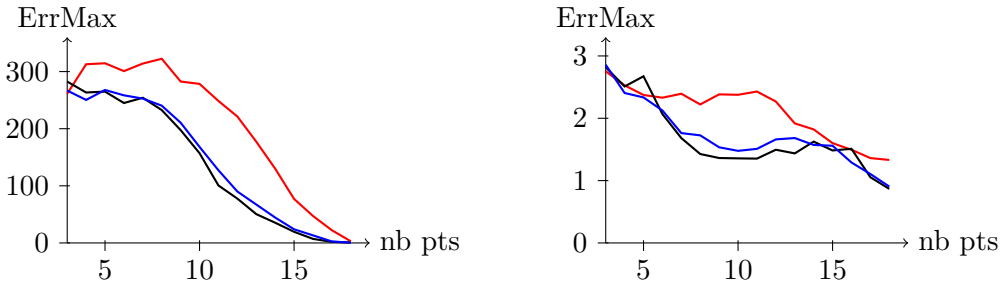


FIGURE 8.10 – Évolution moyenne en fonction du nombre de points d'apprentissage de l'erreur maximale des modèles de la fonction de Rosenbrock (à gauche) et de la fonction chameau (à droite). Notre méthode est représentée en bleu, l'amélioration alternative des modèles en noir, et l'utilisation d'un LHS sans construction adaptative en rouge.

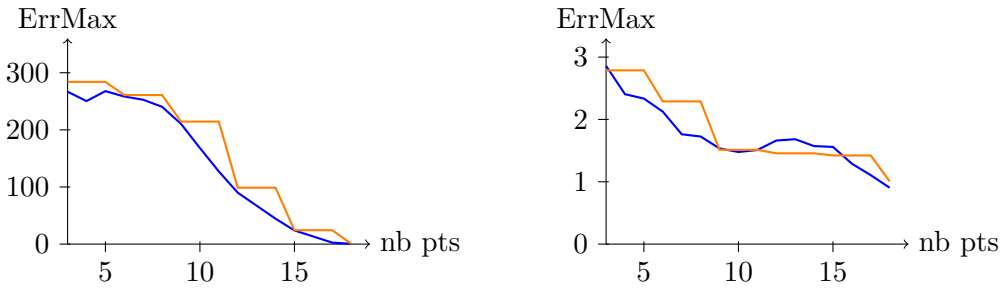


FIGURE 8.11 – Évolution moyenne en fonction du nombre de points d'apprentissage de l'erreur maximale des modèles de la fonction de Rosenbrock (à gauche) et de la fonction chameau (à droite). L'amélioration point par point est représentée en bleu, et l'amélioration en rajoutant trois points à la fois en orange.

à la fois à chaque itération. On retrouve une erreur similaire à celle obtenue en ajoutant les points un par un. L'évaluation d'échantillons en parallèle permet donc une fois de plus d'accélérer le processus d'amélioration de la qualité des modèles.

8.2 Plans d'expériences adaptatifs pour l'optimisation déterministe multiobjectifs (méthode PareBO)

Après avoir étudié l'amélioration globale de modèles de substitution, nous allons maintenant nous intéresser à leur enrichissement dans le cadre d'une optimisation multiobjectifs déterministe. Le but recherché n'est plus d'obtenir des modèles précis dans leur ensemble mais de rajouter des points au plan d'expériences au niveau des zones optimales du problème, afin que ses solutions soient connues avec la plus grande précision possible.

La méthode adaptative la plus répandue en optimisation mono-objectif est l'algorithme EGO [JSW98]. Quelques adaptations de cette approche à l'optimisation multiobjectifs ont été proposées dans la littérature (voir la section 4.6.3), mais d'autres auteurs préfèrent utiliser les intervalles de confiance des prédictions des modèles pour prendre en compte leur erreur au cours de l'optimisation (comme [EN04] par exemple). C'est sur ce principe que nous allons nous baser ici. Nous présentons ainsi un procédé pour intégrer l'erreur des modèles de substitution à une optimisation multiobjectifs, et adaptons l'algorithme NSGA-II en conséquence. À partir de cela, nous développons ensuite une méthode de construction adaptative de plans d'expériences pour l'optimisation multiobjectifs, nommée PareBO.

8.2.1 Prise en compte de l'erreur des modèles en optimisation multiobjectifs

Plutôt que de considérer lors d'une optimisation que les modèles de substitution utilisés sont exacts, il est possible de prendre en compte les intervalles de confiance de leurs prédictions afin d'éviter de passer à côté de certaines solutions optimales du fait d'erreurs de modélisation. Ces intervalles peuvent être obtenus directement comme dans le cas du krigeage ou bien évalués par bootstrap. On associe ainsi un intervalle de confiance $\mathrm{IC}(\hat{F}_i(\boldsymbol{p}))$ à la prédiction d'un modèle $\hat{f}_i$ en un point $\boldsymbol{p}$. Lorsque plusieurs modèles entrent en jeu, on obtient une « région de confiance » $\mathbf{RC}(\hat{\boldsymbol{F}}(\boldsymbol{p}))$ autour de la valeur prédite dans l'es-

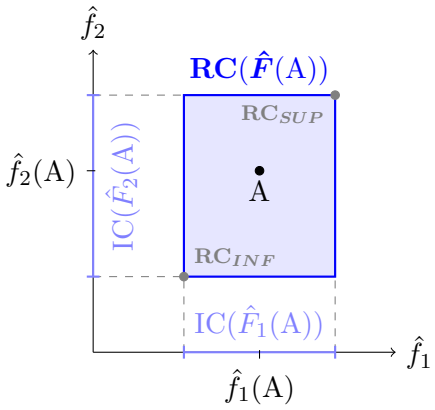


FIGURE 8.12 – Région de confiance **RC** (en bleu) autour d'un point A (point noir) définie à partir des intervalles de confiance IC sur les prédictions des modèles $\hat{f}_1$ et $\hat{f}_2$.

pace des objectifs ($\hat{\boldsymbol{F}}$ représentant l'ensemble des processus stochastiques correspondant aux modèles $\hat{\boldsymbol{f}}$ du problème). Cette région de confiance est un hypercube défini comme le produit des intervalles de confiance de chacun des modèles $\hat{f}_i$ (voir l'équation (4.17)). Un exemple d'illustration est donné sur la figure 8.12. On distingue notamment les points inférieur $\mathbf{RC}_{\boldsymbol{INF}}$ et supérieur $\mathbf{RC}_{\boldsymbol{SUP}}$ de la région de confiance **RC**.

Pour prendre en compte l'erreur des modèles au cours d'une optimisation multiobjectifs, nous allons redéfinir la dominance de Pareto en nous appuyant sur ce concept de région de confiance. Nous dirons ainsi qu'un point $\boldsymbol{a}$ domine un point $\boldsymbol{b}$ si l'ensemble des points de la région de confiance de $\boldsymbol{a}$ domine l'ensemble des points de la région de confiance de $\boldsymbol{b}$. Cette nouvelle relation de dominance, que nous noterons $\prec_{\mathrm{RC}}$, peut être déterminée à partir du plus mauvais point de la région de confiance de $\boldsymbol{a}$ (qui correspond à $\mathbf{RC}_{\boldsymbol{SUP}}(\hat{\boldsymbol{F}}(\boldsymbol{a}))$) et du meilleur point de la région de confiance de $\boldsymbol{b}$ (soit $\mathbf{RC}_{\boldsymbol{INF}}(\hat{\boldsymbol{F}}(\boldsymbol{b}))$) :

$$\boldsymbol{a} \prec_{\mathrm{RC}} \boldsymbol{b} \Leftrightarrow \mathbf{RC}_{\boldsymbol{SUP}}(\hat{\boldsymbol{F}}(\boldsymbol{a})) \prec \mathbf{RC}_{\boldsymbol{INF}}(\hat{\boldsymbol{F}}(\boldsymbol{b})), \tag{8.6}$$

avec $\prec$ l'opérateur de la dominance de Pareto classique défini par l'équation (4.3). Il s'agit là d'une caractérisation relativement conservative de la dominance car on demande que tout point de la région de confiance de $\boldsymbol{a}$ domine ceux de $\boldsymbol{b}$. On peut toutefois ajuster la permissivité de cette relation à l'aide du niveau de confiance considéré pour les modèles. Nous avons choisi ici d'utiliser des intervalles de confiance à 95% sur les prédictions, mais des intervalles à 90% auraient par exemple donné des régions de confiance moins étendues et donc une relation de dominance plus sélective. Il est ainsi possible de jouer sur ce paramètre si jamais les intervalles de confiance des modèles semblent surestimés au point qu'il n'est plus possible de distinguer les solutions dominantes des autres.

Avec cette nouvelle définition de la dominance, le front de Pareto devient une « bande de Pareto ». La figure 8.13 en présente un exemple pour un problème à deux objectifs. Les solutions qui seraient retenues lors d'une optimisation classique sans considération des erreurs de modèles sur cet exemple seraient les points A, C et D. Le fait de prendre en compte les régions de confiance ajoute les points B et E, car ils sont eux aussi possiblement optimaux. La zone de l'espace dominée par les solutions de la bande de Pareto est définie par les plus mauvais points des régions de confiance de ces solutions. Pour qu'un point

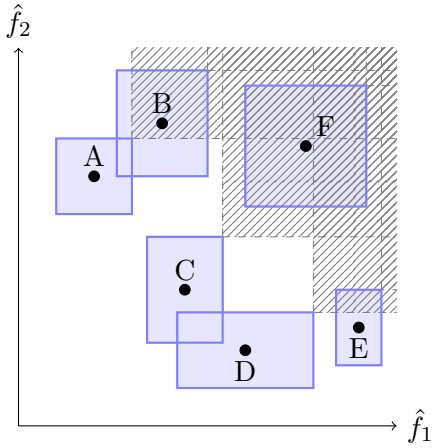


FIGURE 8.13 – Exemple de bande de Pareto dans l'espace des objectifs $\hat{f}_1$ et $\hat{f}_2$. Chaque point noir est une solution avec sa région de confiance en bleu. Les zones dominées sont hachées en gris. Tous les points représentés appartiennent à la bande de Pareto sauf le point F qui est dominé par C.

ne soit pas Pareto-optimal, il faut que sa région de confiance soit totalement incluse dans cette zone dominée. Le point F est ainsi complètement dominé par C, et ce n'est donc pas une solution optimale du problème. Le point E n'est par contre que partiellement dominé par D et il se peut donc qu'il soit Pareto-optimal. Ce sera par exemple le cas s'il s'avère que sa vraie valeur est le point E lui même et que celle de D correspond au plus mauvais point de sa région de confiance $\mathbf{RC}_{\boldsymbol{SUP}}(\hat{\boldsymbol{F}}(\mathrm{D}))$. La bande de Pareto rassemble ainsi toutes les solutions possiblement optimales du problème en prenant en compte l'erreur des modèles. Si on évalue véritablement ces solutions avec les fonctions de référence, certaines se révèleront non optimales mais on est par contre assuré de retrouver les solutions optimales réelles parmi celles de la bande de Pareto (sous réserve que les intervalles de confiance des prédictions des modèles soient correctement estimés).

8.2.2 Adaptation de l'algorithme NSGA-II

Afin de pouvoir rechercher la bande de Pareto d'un problème d'optimisation, nous avons intégré la nouvelle définition de la dominance $\prec_{\mathrm{RC}}$ au sein de l'algorithme génétique NSGA-II [DPAM02]. Cette modification est très facile à réaliser car elle nécessite simplement de comparer les points extrêmes des régions de confiance en lieu et place des valeurs ponctuelles lors du classement des individus de la population (au niveau de la ligne 5 de l'algorithme 4.2). On obtient ainsi un algorithme d'optimisation multiobjectifs qui prend en compte l'erreur des modèles de substitution.

Notre but final est d'arriver à améliorer les modèles de substitution dans les régions de l'espace possiblement optimales. Comme nous le verrons par la suite, ce sont les solutions de la bande de Pareto qui vont nous permettre de localiser ces régions. Mais il faut pour cela que les solutions soient bien réparties dans l'espace des variables d'entrée $\boldsymbol{x}$ du problème afin de n'oublier aucune de ces zones d'intérêt. Si on observe le comportement de l'algorithme NSGA-II traditionnel, on peut remarquer que les solutions qu'il fournit sont bien réparties le long du front de Pareto (c'est-à-dire dans l'espace des objectifs $\boldsymbol{f}$), mais

pas nécessairement dans l'espace des $\boldsymbol{x}$. Cela est dû au critère de nichage mis en place (voir l'algorithme 4.1). En effet, lorsque le nombre de solutions optimales devient trop important, celles-ci sont triées de manière à éliminer les solutions voisines sur le front de Pareto. Le nichage permet donc de sélectionner des solutions bien réparties sur le front, mais sans prendre en considération leur situation dans l'espace de recherche.

Nous avons illustré ce fait sur le problème standard ZDT6 [ZDT00] défini par l'équation (4.9) avec une dimension d'entrée $n_x = 2$. Il possède un front de Pareto concave (présenté sur la figure 4.12) dont les solutions constituent dans l'espace des variables de décision le segment $x_1 \in [0, 1]$ et $x_2 = 0$. Ce problème a été résolu à l'aide de l'algorithme NSGA-II classique avec une population de 50 individus sur 500 générations. La figure 8.14 présente les résultats obtenus. Les solutions sont regroupées en quatre classes pour avoir une meilleure visualisation des régions de l'espace de décision représentées. On voit que dans le cadre du nichage « classique » réalisé sur l'espace des objectifs, le front obtenu est effectivement très bien décrit mais ses solutions ne couvrent qu'une faible partie de la région optimale dans l'espace des $\boldsymbol{x}$. L'utilisation de ce type de nichage a donc pour conséquence que les zones définies par les solutions optimales ne représentent pas l'ensemble des zones potentiellement intéressantes du problème. On peut au contraire baser le nichage sur les valeurs des variables de décision : les solutions sont alors réparties sur l'ensemble du segment optimal de l'espace de recherche mais le front de Pareto est mal représenté dans l'espace des objectifs. La dernière possibilité est de prendre en compte à la fois les variables de décision et les objectifs dans le processus de nichage pour obtenir des solutions relativement bien réparties dans les deux espaces.

Cet exemple utilise la dominance de Pareto classique, mais le principe reste exactement le même dans le cas d'une bande de Pareto. Nous avons donc modifié la procédure de nichage de NSGA-II afin de prendre en compte à la fois les variables de décision et les objectifs. On souhaite en effet obtenir des solutions bien réparties dans l'espace des $\boldsymbol{x}$ tout en offrant une bonne représentation des possibilités offertes dans l'espace des objectifs. Ce nichage « mixte » reprend le principe du nichage classique mis à part qu'on considère les valeurs des variables de décision des individus au même titre que les valeurs de leurs objectifs. Le détail des opérations effectuées pour classer un ensemble de solutions $\boldsymbol{F}_i$ est donné dans l'algorithme 8.5.

Les adaptations de NSGA-II que nous avons réalisées permettent ainsi d'obtenir les solutions qui constituent la bande de Pareto d'un problème et qui sont représentatives de l'ensemble des régions possiblement optimales de l'espace des variables de décision.

8.2.3 Enrichissement de modèles pour l'optimisation multiobjectifs

Maintenant que nous sommes à même de déterminer les solutions possiblement optimales d'un problème en prenant en compte les erreurs de modélisation, nous allons développer une méthode de construction adaptative de plans d'expériences pour l'optimisation multiobjectifs. Nous avons nommé cette méthode PareBO (pour « Pareto Band Optimization ») car elle est basée sur l'exploitation des solutions de la bande de Pareto. Ces solutions permettent en effet de localiser les régions possiblement optimales de l'espace de recherche. Ces régions contiennent des véritables solutions optimales du problème, ou bien sont trop mal décrites par les modèles pour qu'on puisse affirmer qu'aucun optimum ne s'y trouve. Nous allons donc chercher à approcher plus précisément ces zones en ajoutant de nouveaux échantillons d'apprentissage au plan d'expériences.

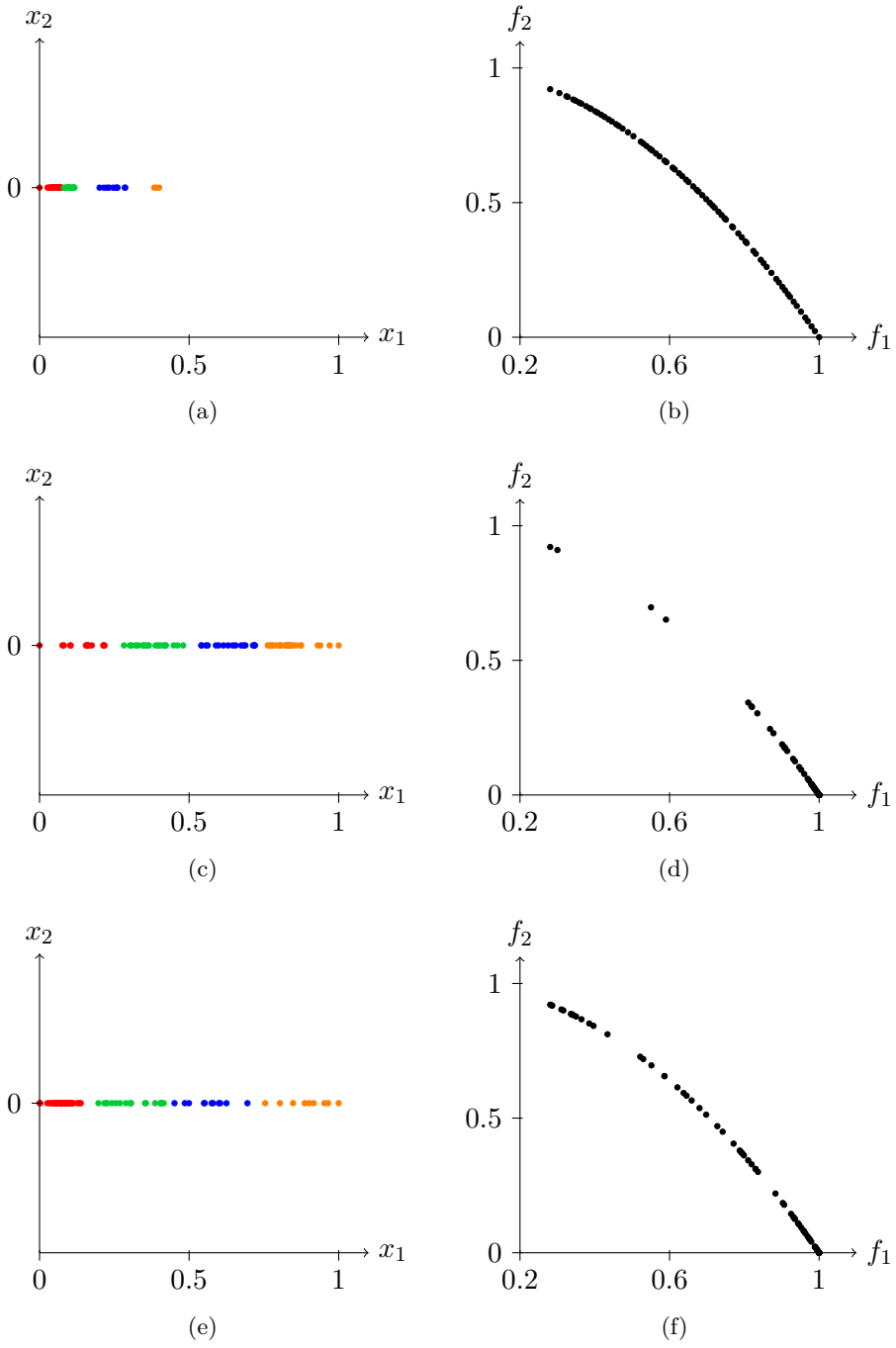


FIGURE 8.14 – Solutions (figurées par des points) obtenues pour le problème ZDT6 dans l'espace des variables de décision (à gauche) et dans l'espace des objectifs (à droite) pour différents types de nichage : en haut un nichage sur les objectifs uniquement, au centre sur les variables de décision uniquement et en bas sur les deux à la fois.

Algorithme 8.5 : Classement_nichage_mixte($\boldsymbol{F}_i$)

$NombreSolutions \leftarrow |\boldsymbol{F}_i|$
pour $j \in \{1, 2, ..., NombreSolutions\}$ **faire**
    $\boldsymbol{DistancesNichage}[j] \leftarrow 0$
fin
pour $j \in \{1, 2, ..., n_f\}$ **faire**
    $\boldsymbol{T} \leftarrow$ Trier_selon_valeur_objectif($\boldsymbol{F}_i, f_j$)
    $\boldsymbol{DistancesNichage}[\boldsymbol{T}_1] \leftarrow +\infty$
    $\boldsymbol{DistancesNichage}[\boldsymbol{T}_{NombreSolutions}] \leftarrow +\infty$
    $VariationGlobalef \leftarrow f_j(\boldsymbol{F}_i[\boldsymbol{T}_{NombreSolutions}]) - f_j(\boldsymbol{F}_i[\boldsymbol{T}_1])$
    pour $k \in \{2, 3, ..., NombreSolutions - 1\}$ **faire**
        $\Delta f \leftarrow f_j(\boldsymbol{F}_i[\boldsymbol{T}_{k+1}]) - f_j(\boldsymbol{F}_i[\boldsymbol{T}_{k-1}])$
        $\boldsymbol{DistancesNichage}[\boldsymbol{T}_k] \leftarrow \boldsymbol{DistancesNichage}[\boldsymbol{T}_k] + \frac{\Delta f}{VariationGlobalef}$
    fin
fin
pour $j \in \{1, 2, ..., n_x\}$ **faire**
    $\boldsymbol{T} \leftarrow$ Trier_selon_valeur_variable_de_décision($\boldsymbol{F}_i, j$)
    $\boldsymbol{DistancesNichage}[\boldsymbol{T}_1] \leftarrow +\infty$
    $\boldsymbol{DistancesNichage}[\boldsymbol{T}_{NombreSolutions}] \leftarrow +\infty$
    $VariationGlobalex \leftarrow \boldsymbol{F}_i[\boldsymbol{T}_{NombreSolutions}]_j - \boldsymbol{F}_i[\boldsymbol{T}_1]_j$
    pour $k \in \{2, 3, ..., NombreSolutions - 1\}$ **faire**
        $\Delta x \leftarrow \boldsymbol{F}_i[\boldsymbol{T}_{k+1}]_j - \boldsymbol{F}_i[\boldsymbol{T}_{k-1}]_j$
        $\boldsymbol{DistancesNichage}[\boldsymbol{T}_k] \leftarrow \boldsymbol{DistancesNichage}[\boldsymbol{T}_k] + \frac{\Delta x}{VariationGlobalex}$
    fin
fin
Trier_selon_distance($\boldsymbol{F}_i, \boldsymbol{DistancesNichage}$)

Afin de mieux cerner les différentes régions d'intérêt du problème, nous réalisons tout d'abord une classification des solutions de la bande de Pareto dans l'espace des $\boldsymbol{x}$. Nous allons ensuite rechercher des échantillons permettant d'améliorer les modèles au sein de chacune de ces classes (notées S_i) : le nombre de classes constituées doit donc être égal au nombre d'évaluations des fonctions de référence qu'il est possible de mener en parallèle (noté $NombrePointsParallèles$). Une fois la classification effectuée, on considère les groupes de solutions optimales obtenus les uns après les autres. Pour déterminer quel est l'échantillon à ajouter au plan d'expériences pour une classe S_i donnée, nous reprenons une idée similaire à celle utilisée pour l'amélioration simultanée de plusieurs modèles. La seule différence est qu'on s'intéresse désormais à l'amélioration d'une certaine région de l'espace définie par les solutions de la classe S_i plutôt que d'enrichir les modèles dans leur globalité. Dans l'algorithme 8.4, nous recherchions dans un premier temps les points permettant de réduire l'erreur de chaque modèle (qui correspondent par exemple aux points de plus grande variance si on utilise le critère MMSE), avant de choisir un point améliorant général dans le but d'affiner les prédictions des points précédents. Pour rendre l'amélioration locale à une zone d'intérêt particulière, il suffit ici de restreindre la première recherche aux solutions de la classe S_i considérée. Nous allons ainsi déterminer quels sont les points de S_i qui, une fois évalués, permettraient de diminuer l'erreur de chaque modèle sur l'en-

semble de cette classe. Avec le critère MMSE, cela revient à choisir le point de plus grande variance parmi ceux de la classe. Cette recherche de la solution de S_i la plus mal décrite est réalisée pour chaque modèle du problème, et on obtient finalement un ensemble de points qu'on aimerait connaître plus précisément afin de réduire l'erreur de modélisation au niveau de la région d'intérêt que représentent les solutions de S_i. On aurait aussi pu utiliser un critère du type IMSE en considérant l'erreur moyenne des points de la classe et en recherchant la solution de S_i qui une fois évaluée aurait apporté la plus grande amélioration de cette erreur. Cette approche est légèrement plus coûteuse et son objectif est différent. Nous conservons donc ici le critère MMSE dans la lignée de ce qui a été fait précédemment.

Les différentes solutions de S_i qu'on souhaiterait voir améliorées ayant été définies (une solution par modèle), on recherche ensuite un point améliorant général pour affiner les prédictions de cet ensemble de solutions. Le point améliorant général est choisi comme auparavant en résolvant le problème (8.5). Il est à noter que la recherche de ce point est réalisée dans l'espace tout entier et n'est donc pas restreinte aux seules solutions de S_i (même si le point sélectionné a de grandes chances d'être situé à proximité de la région d'intérêt définie par cette classe). Lorsque l'échantillon à évaluer pour améliorer la classe S_i a été choisi, on simule son ajout à la base d'apprentissage des modèles avant de passer à la classe suivante. Chaque échantillon tiendra ainsi compte de ceux qui ont été sélectionnés avant lui afin d'éviter de choisir des points similaires. On obtient au final autant d'échantillons que de classes de solutions de la bande de Pareto constituées initialement. Après avoir évalué ces points sur les fonctions de référence, on peut relancer la construction des modèles de substitution $\hat{\boldsymbol{f}}$ et la recherche de points améliorants. Les itérations d'enrichissement du plan d'expériences continuent ainsi jusqu'à ce que les modèles aient atteint une qualité suffisante ou bien jusqu'à ce qu'un quota d'évaluations des fonctions de référence soit atteint.

Le principe de la méthode PareBO que nous venons de décrire est résumé dans l'algorithme 8.6. La fonction « NSGA-II_avec_erreur_modèles($\hat{\boldsymbol{f}}$) » correspond à la résolution du problème de minimisation des modèles $\hat{\boldsymbol{f}}$ en prenant en compte leur erreur. Elle est effectuée grâce à l'algorithme NSGA-II modifié doté de la nouvelle définition de la dominance de Pareto $\prec_{\mathrm{RC}}$. La fonction « Classification($\boldsymbol{BandePareto}, NombrePointsParallèles$) » correspond quant à elle à la partition de l'ensemble de points $\boldsymbol{BandePareto}$ en différentes classes regroupant des points similaires dans l'espace des $\boldsymbol{x}$. Le nombre de classes à constituer est donné par $NombrePointsParallèles$, et la classification est réalisée ici à l'aide de l'algorithme des k-moyennes (algorithme 4.3). « Chercher_point_à_ajouter($\hat{f}_j, S_i$) » correspond à la recherche du point de plus grande variance sur le modèle $\hat{f}_j$ parmi les solutions constituant la classe S_i (ce point peut aussi être choisi différemment en considèrant un autre critère d'amélioration que le critère MMSE). Enfin, la fonction « Chercher_point_améliorant_général($\boldsymbol{PointsÀAméliorer}, \hat{\boldsymbol{f}}$) » correspond à la recherche du point de l'espace permettant de réduire les intervalles de confiance des prédictions des points $\boldsymbol{PointsÀAméliorer}$ par les modèles $\hat{\boldsymbol{f}}$ (obtenu en résolvant le problème (8.5)).

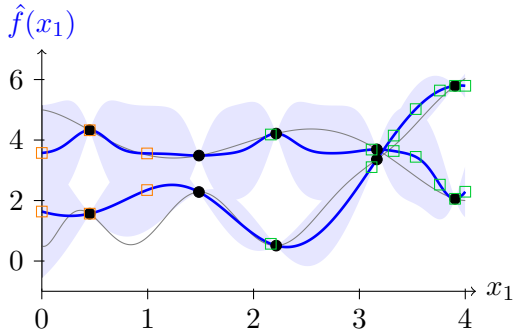
La figure 8.15 illustre sur un exemple en une dimension le déroulement d'une itération de l'algorithme PareBO. Dans cet exemple, on cherche à minimiser deux fonctions approchées à l'aide de modèles de krigeage $\hat{f}_1$ et $\hat{f}_2$. On a la possibilité d'évaluer deux échantillons en parallèle à chaque itération. Du fait d'une erreur de modélisation importante, la bande de Pareto est au départ relativement large et ne permet pas d'avoir une

Algorithme 8.6 : Méthode PareBO (« Pareto Band Optimization ») de construction adaptative d'un plan d'expériences pour l'optimisation déterministe multiobjectifs

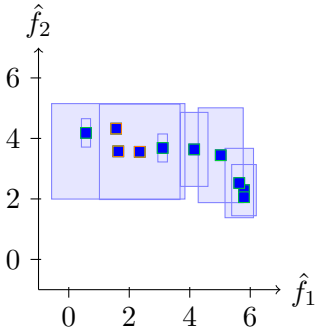
$\boldsymbol{PDE} \leftarrow \boldsymbol{PDEinitial}$
Évaluer_avec_fonctions_de_référence($\boldsymbol{PDE}$)
tant que *le critère d'arrêt n'est pas vérifié* **faire**
  $\hat{\boldsymbol{f}} \leftarrow$ Construire_modèles($\boldsymbol{PDE}$)
  $\boldsymbol{BandePareto} \leftarrow$ NSGA-II_avec_erreur_modèles($\hat{\boldsymbol{f}}$)
  $\boldsymbol{S} \leftarrow$ Classification($\boldsymbol{BandePareto}, NombrePointsParallèles$)
  $\boldsymbol{NouveauxPoints} \leftarrow \emptyset$
  pour $i \in \{1, 2, ..., NombrePointsParallèles\}$ **faire**
    pour $j \in \{1, 2, ..., n_f\}$ **faire**
      $PointsÀAméliorer_j \leftarrow$ Chercher_point_à_ajouter($\hat{f}_j, S_i$)
    fin
    $p \leftarrow$ Chercher_point_améliorant_général($\boldsymbol{PointsÀAméliorer}, \hat{\boldsymbol{f}}$)
    $\boldsymbol{NouveauxPoints} \leftarrow \boldsymbol{NouveauxPoints} \cup \{p\}$
    Simuler_ajout($p, \hat{\boldsymbol{f}}$)
  fin
  Évaluer_avec_fonctions_de_référence($\boldsymbol{NouveauxPoints}$)
  $\boldsymbol{PDE} \leftarrow \boldsymbol{PDE} \cup \boldsymbol{NouveauxPoints}$
fin

idée précise de la qualité de ses solutions. En vue de l'améliorer, les solutions optimales sont donc réparties en deux classes différentes qui vont être traitées successivement. On recherche tout d'abord au sein de la première classe S_1 les solutions qui possèdent la plus forte variance sur les deux modèles : il se trouve qu'il s'agit ici de la même solution. La recherche d'un point améliorant général indique bien évidemment que c'est l'évaluation de cette solution elle-même qui permettra de réduire au maximum l'erreur des modèles dans cette classe. Cet échantillon est donc ajouté « virtuellement » à la base d'apprentissage des modèles avant d'étudier la seconde classe S_2. Selon le même procédé, on détermine les deux solutions de plus grande variance sur chacun des modèles puis on recherche le point qui permettra de les améliorer. L'amélioration maximum est apportée par l'une de ces deux solutions. On peut enfin évaluer en parallèle les deux échantillons sélectionnés et reconstruire les modèles en y intégrant ces informations supplémentaires. La nouvelle bande de Pareto obtenue est beaucoup mieux définie qu'initialement, $\hat{f}_2$ approchant de manière quasi parfaite sa fonction de référence.

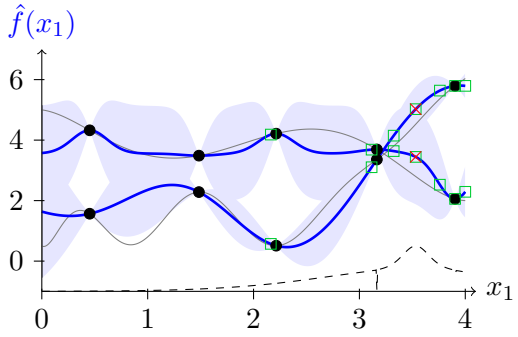
Nous avons aussi mis en œuvre cette méthode de construction adaptative de plans d'expériences pour l'optimisation multiobjectifs sur l'exemple de la section précédente qui met en jeu les fonctions de Rosenbrock et chameau (figure 8.8). Nous cherchons maintenant à minimiser simultanément ces deux fonctions. Les solutions « idéales » de ce problème ont été calculées à l'aide de l'algorithme NSGA-II classique sur les fonctions de référence avec une population de 100 individus durant 1000 générations. Elles sont présentées sur la figure 8.16. Ces solutions sont situées dans la partie basse de la vallée de la fonction de Rosenbrock ainsi qu'à proximité d'un des minima de la fonction chameau. Nous avons ensuite effectué une optimisation adaptative avec la méthode PareBO à partir d'un LHS initial constitué de 3 échantillons. Trois points supplémentaires ont été évalués en parallèle



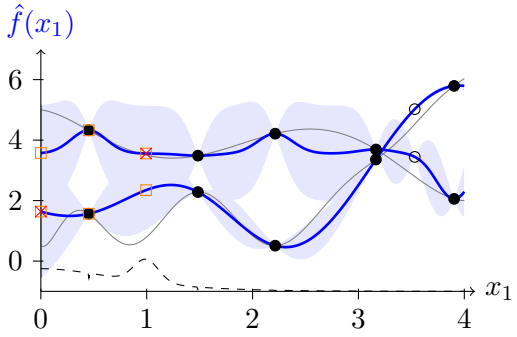
(a) Modèles initiaux et classification des solutions optimales en deux classes S_1 et S_2



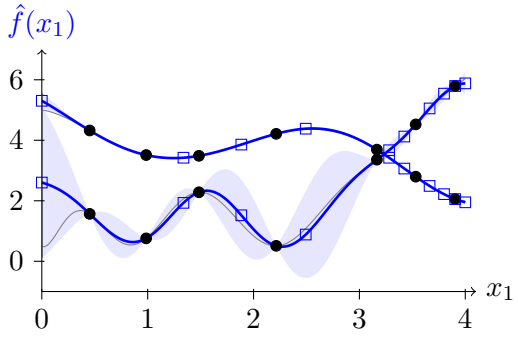
(b) Solutions optimales dans l'espace des objectifs



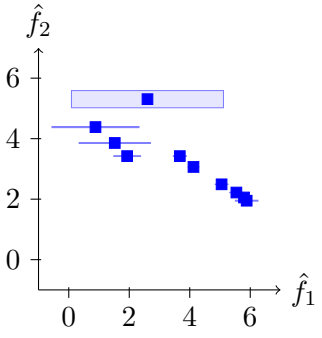
(c) Amélioration de la première classe S_1



(d) Amélioration de la seconde classe S_2



(e) Modèles reconstruits après évaluation des deux nouveaux échantillons sélectionnés



(f) Nouvelle bande de Pareto

FIGURE 8.15 – Exemple d'une itération de la méthode PareBO sur un problème de minimisation bi-objectifs. Les fonctions de référence sont représentées en gris, les modèles de krigeage en bleu (avec leur intervalle de confiance à 95%) et les échantillons d'apprentissage par des points noirs. Les carrés de couleur figurent les solutions optimales de la bande de Pareto réparties en deux classes S_1 et S_2 (a). Dans chaque classe, les solutions de plus forte variance sont marquées d'une croix rouge (c) (d). L'amélioration générale apportée en ces solutions par tout point de l'espace est représentée en pointillés (sans respect d'échelle). L'échantillon ajouté de manière simulée aux modèles lors de l'étude de la classe S_2 est indiqué par un rond noir (d).

sur les fonctions de référence à chaque itération, pour arriver à un plan d'expériences final de 15 échantillons. Les deux fonctions à minimiser ont été approchées à l'aide de modèles de krigeage comme auparavant. La recherche des solutions de la bande de Pareto (ligne 5 de l'algorithme 8.6) a été effectuée grâce à l'algorithme NSGA-II modifié pour prendre en compte l'erreur des modèles, avec une population de 20 individus sur 300 générations. La recherche du point améliorant général de chaque classe de solutions (ligne 12 de l'algorithme 8.6) a été réalisée quant à elle avec l'algorithme NSGA-II classique en utilisant une population de 20 individus sur 200 générations. La figure 8.17 présente le plan d'expériences ainsi que les solutions de la bande de Pareto obtenus au terme des itérations de l'algorithme. On voit qu'on s'approche assez bien du front de Pareto idéal, même si la bande de Pareto contient des solutions supplémentaires du fait de l'erreur de modélisation qui subsiste encore sur le second objectif (la fonction chameau est en effet plus difficile à approcher). L'algorithme a cerné les deux zones optimales de l'espace de recherche, et en considère aussi une troisième au niveau du second minimum de la fonction chameau à cause de l'erreur des modèles dans cette région. Pour vérification, nous avons recalculé les solutions de la bande de Pareto à l'aide des fonctions de référence (graphe (d) de la figure 8.17). On retrouve bien une partie des solutions optimales idéales, réparties le long du front de Pareto idéal. La méthode PareBO a donc correctement résolu ce problème. À titre de comparaison, nous avons aussi construit des modèles des deux fonctions à optimiser à l'aide d'un LHS de 15 échantillons sans enrichissement adaptatif. Les solutions possiblement optimales ont été recherchées sur ces modèles avec l'algorithme NSGA-II modifié. Les résultats sont présentés sur la figure 8.18. On constate que l'erreur des modèles est plus importante que précédemment au niveau des points qui composent la bande de Pareto. Le front de Pareto idéal est donc moins bien décrit, même si on retrouve tout de même une partie de ses solutions. Les points de la bande de Pareto sont aussi répartis différemment dans l'espace de recherche : plusieurs d'entre eux sont situés au niveau des frontières de l'espace, où l'erreur de modélisation est plus importante. Les modèles n'ayant pas été améliorés dans ces zones, les intervalles de confiance des prédictions y sont assez larges et viennent encombrer la bande de Pareto. Cet exemple montre donc l'importance de raffiner les modèles dans des régions ciblées afin de pouvoir mieux discerner les solutions optimales d'un problème.

La méthode PareBO permet ainsi de mener une optimisation multiobjectifs à l'aide de modèles de substitution en prenant en compte leur erreur pour n'oublier aucune solution optimale. L'enrichissement adaptatif de ces modèles sélectionne les échantillons les plus pertinents pour améliorer la prédiction des optima tout en limitant le nombre d'évaluations des fonctions de référence. Plusieurs de ces échantillons peuvent être sélectionnés à chaque itération en vue de les évaluer en parallèle pour accélérer le processus d'optimisation.

8.3 Plans d'expériences adaptatifs pour l'optimisation robuste multiobjectifs (méthode PareBRO)

Après avoir étudié la construction adaptative de plans d'expériences pour l'optimisation déterministe classique, nous pouvons maintenant revenir au sujet principal de cette étude, c'est-à-dire l'optimisation robuste. La difficulté d'un problème d'optimisation robuste réside principalement dans sa formulation et dans le calcul des mesures de robustesse qu'il fait intervenir. Cela mis à part, le problème peut être résolu à l'aide d'algorithmes

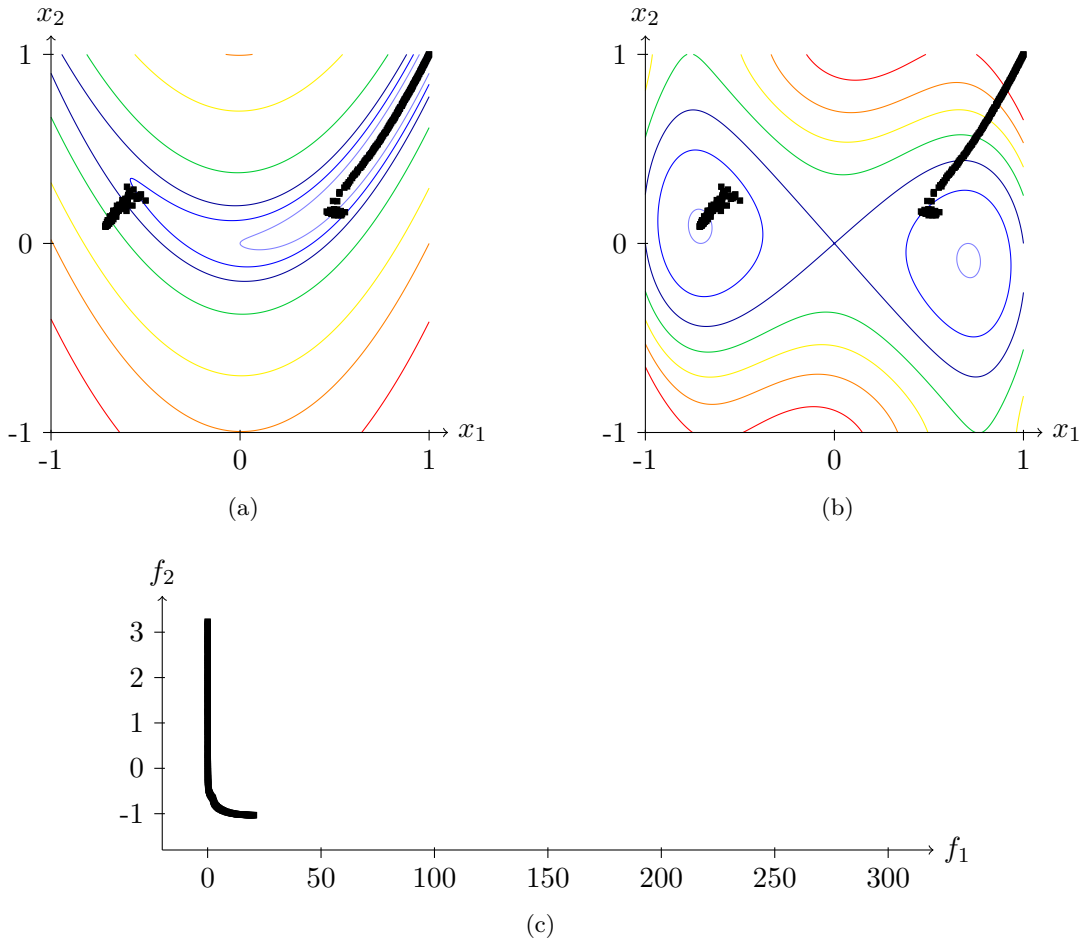


FIGURE 8.16 – Solutions « idéales » (carrés noirs) dans l'espace de recherche pour le problème d'optimisation déterministe bi-objectifs des fonctions de Rosenbrock (a) et chameau (b), dont les courbes de niveau sont tracées en couleur. Le graphe (c) présente ces mêmes solutions au sein de l'espace des objectifs (il s'agit donc du front de Pareto idéal qu'on cherche à atteindre).

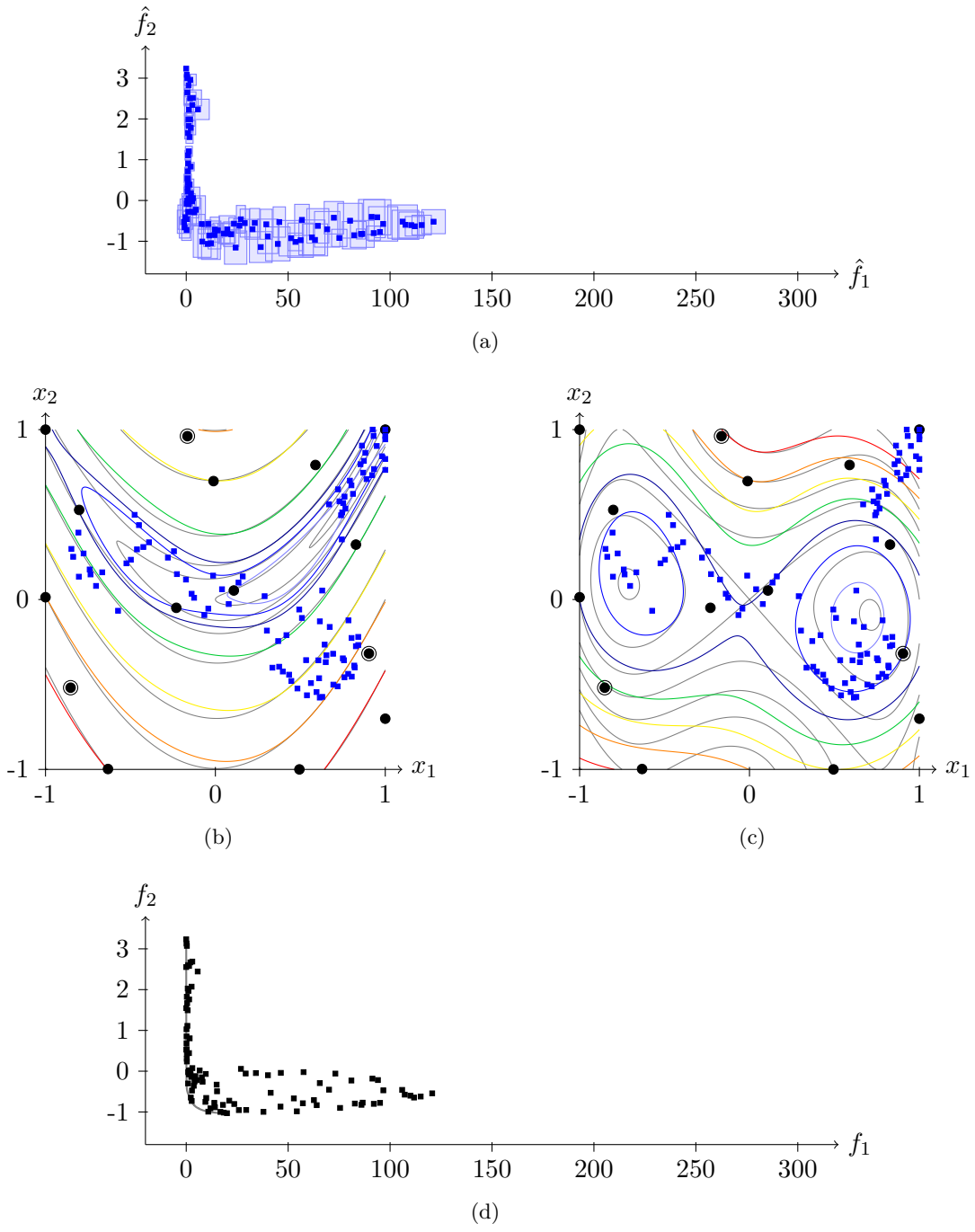


FIGURE 8.17 – Plan d'expériences (points noirs) obtenu par amélioration adaptative à partir d'un LHS initial de trois points (indiqués par des cercles) pour minimiser simultanément les fonctions de Rosenbrock (b) et chameau (c) à l'aide de modèles de krigeage. Les courbes de niveau des fonctions de référence sont représentées en gris et celles des modèles finaux en couleur. Les solutions de la bande de Pareto (a) sont figurées par des carrés bleus. Le graphe (d) présente les valeurs de ces mêmes solutions recalculées sur les fonctions de référence, avec en gris la position du front de Pareto idéal de la figure 8.16(c).

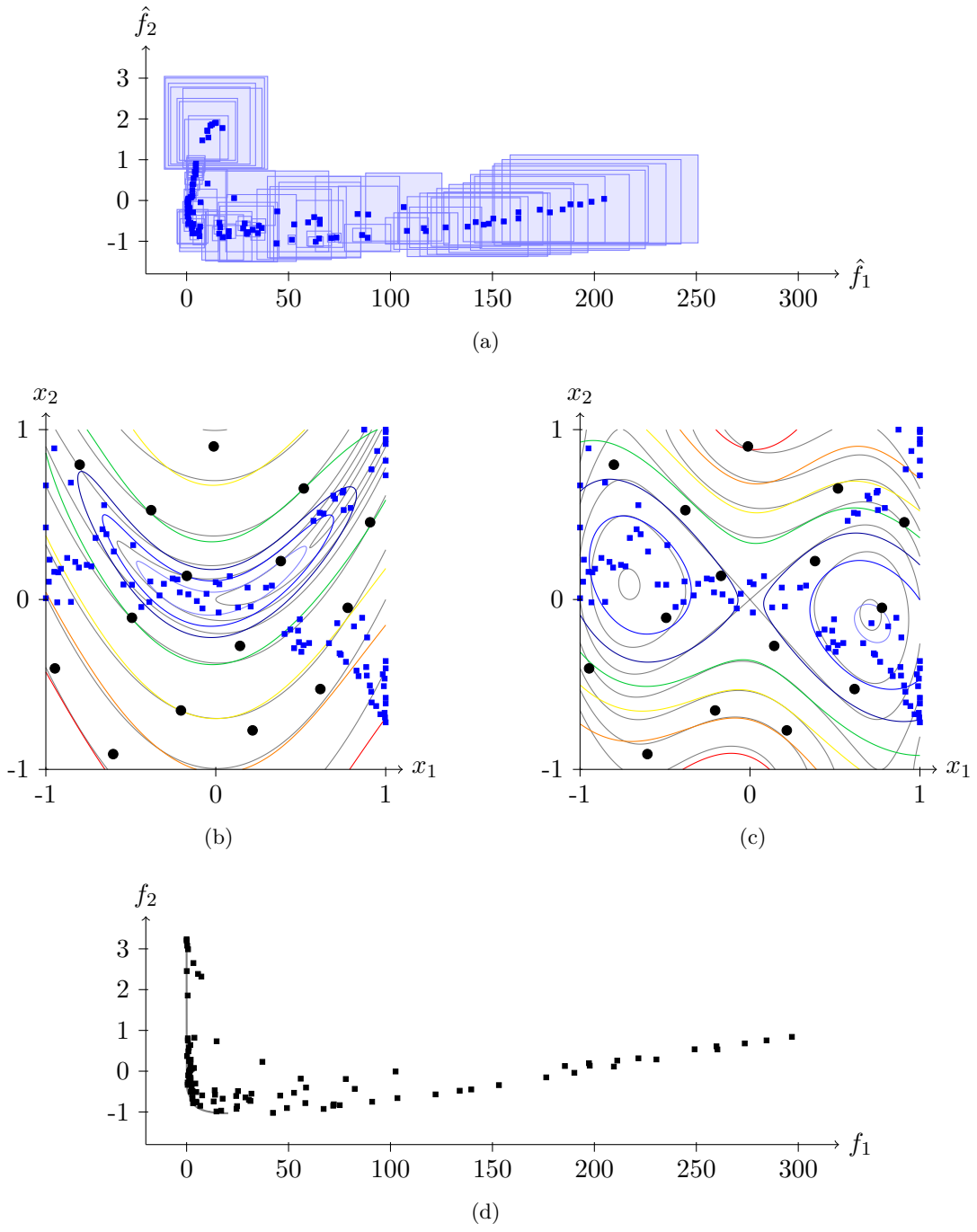


FIGURE 8.18 – Bande de Pareto (a) obtenue lors de la minimisation des fonctions de Rosenbrock (b) et chameau (c) à l'aide de modèles de krigeage construits sur un LHS de 15 points (points noirs). Les courbes de niveau des fonctions de référence sont représentées en gris et celles des modèles finaux en couleur. Les solutions optimales trouvées sont figurées par des carrés bleus. Le graphe (d) présente les valeurs de ces mêmes solutions recalculées sur les fonctions de référence, avec en gris la position du front de Pareto idéal de la figure 8.16(c).

d'optimisation tout à fait classiques. Les méthodes adaptatives peuvent ainsi se baser sur des principes similaires au cas déterministe, avec seulement quelques spécificités liées à la gestion des incertitudes. Peu de travaux se sont intéressés à l'enrichissement de modèles de substitution pour l'optimisation robuste (et encore moins dans le cadre de problèmes multiobjectifs). On peut néanmoins citer ceux de [JLR12, Jur07, LSN04, WSN00] qui décrivent des stratégies de construction adaptative de plans d'expériences pour l'optimisation robuste mono-objectif. Bien que ces approches soient basées sur le principe d'amélioration espérée (« Expected Improvement ») que nous n'avons pas utilisé ici, la recherche de point améliorant y est effectuée en deux étapes successives dans la même idée que ce que nous proposons. Nous présentons dans cette section l'algorithme PareBRO (« Pareto Band Robust Optimization »), version adaptée de la méthode PareBO pour l'optimisation robuste multiobjectifs.

La méthode PareBO est basée sur la recherche des solutions possiblement optimales qui forment la bande de Pareto d'un problème. Ces solutions sont obtenues en tenant compte des erreurs sur la prédiction des objectifs approchés par des modèles de substitution. Dans le cadre de l'optimisation déterministe, on considérait ainsi des intervalles de confiance fournis par les modèles. En optimisation robuste, les objectifs du problème sont maintenant des mesures de robustesse. On a donc besoin de déterminer l'erreur réalisée sur les estimations de robustesse afin de donner des intervalles de confiance autour des valeurs calculées. La robustesse étant généralement estimée à l'aide de méthodes approchées sur des modèles de substitution des fonctions de référence, il faut prendre en compte d'une part l'erreur due à la méthode d'estimation elle-même et d'autre part l'erreur inhérente au modèle sur lequel sont effectués les calculs. Évaluer l'erreur sur une mesure de robustesse peut donc être relativement complexe. Il n'y a que dans le cas du calcul analytique des mesures d'espérance et de variance que la seule source d'erreur à considérer est celle provenant du modèle, car l'évaluation analytique de la robustesse est exacte. Nous avons choisi ici de ne pas nous attarder sur l'erreur des méthodes d'estimation pour nous concentrer uniquement sur celle provenant des modèles de substitution. Ce sont en effet ces modèles que nous allons tenter d'améliorer lors de la complétion adaptative du plan d'expériences.

L'erreur due à l'utilisation d'un modèle de substitution pour estimer une mesure de robustesse peut être évaluée à l'aide de tirages de Monte-Carlo ou bien analytiquement à partir des coefficients du modèle, en fonction de la méthode de calcul de robustesse retenue et du modèle employé (voir la section 5.8.5). Ces techniques permettent de déterminer des intervalles de confiance autour des valeurs de robustesse prédites. Ces intervalles de confiance peuvent ensuite être pris en compte au cours de l'optimisation de la même manière que dans le cas déterministe afin de trouver la bande de Pareto robuste d'un problème. Le seul inconvénient est que le calcul de l'erreur sur les mesures de robustesse est généralement plus coûteux que celui de l'erreur d'un objectif classique, ces mesures étant déjà au départ plus complexes à évaluer que la simple valeur ponctuelle d'un objectif déterministe. De la même manière qu'un problème d'optimisation robuste est généralement plus complexe à résoudre qu'un problème déterministe classique, la gestion de l'erreur des modèles en optimisation robuste demande un temps de calcul plus important que dans le cadre d'une optimisation déterministe. Le principe de la prise en compte des erreurs de modélisation grâce à une bande de Pareto reste néanmoins le même, hormis que les intervalles de confiance concernent maintenant les estimations de robustesse.

Pour adapter la méthode PareBO au contexte de l'optimisation robuste, il faut aussi tenir compte d'une seconde différence par rapport à l'optimisation déterministe classique.

Elle se situe au niveau de la recherche des points améliorants. Dans la méthode PareBO, ces points sont obtenus en deux temps : une première analyse détermine la solution permettant d'améliorer chaque modèle $\hat{f}_j$ sur une classe S_i donnée, puis on choisit ensuite un point améliorant général pour réduire l'erreur d'approximation de ces solutions et améliorer ainsi l'ensemble des modèles sur la classe considérée. En optimisation robuste, les objectifs du problème ne sont plus représentés par les modèles $\hat{\boldsymbol{f}}$ des fonctions de référence eux-mêmes mais par des estimations de robustesse $\hat{\boldsymbol{\rho}}$ (dont le calcul est basé sur $\hat{\boldsymbol{f}}$). On cherche donc à améliorer les estimations $\hat{\boldsymbol{\rho}}$ via l'amélioration des modèles $\hat{\boldsymbol{f}}$. La séparation de la recherche des échantillons améliorants en deux étapes est ainsi totalement adaptée à l'optimisation robuste.

La première recherche est effectuée de manière similaire au cas déterministe : on détermine le point de la classe S_i qui permettrait s'il était connu avec exactitude de réduire l'erreur de prédiction des objectifs sur l'ensemble de la classe. Au lieu de prendre le point de S_i qui présente la plus grande variance du modèle $\hat{f}_j$ (dans le cadre du critère MMSE), on choisit simplement le point de plus grande variance de $\hat{\rho}_j$ (ou celui qui présente le plus grand intervalle de confiance), déterminé à partir des estimations d'erreur sur les mesures de robustesse abordées plus haut. Les points ainsi sélectionnés appartiennent à l'espace $\boldsymbol{D}$ des variables de décision sur lequel sont définies les mesures de robustesse $\hat{\boldsymbol{\rho}}$ (elles associent une valeur de robustesse à chaque point de l'espace de recherche $\boldsymbol{D}$). En optimisation déterministe, il était possible d'évaluer ces points à l'aide des fonctions de référence pour connaître leur valeur exacte (même si nous avons préféré dans la méthode PareBO n'évaluer qu'un unique échantillon les améliorant tous à la fois). La situation est différente dans le cadre de l'optimisation robuste. En effet, la robustesse des points sélectionnés ne peut pas être évaluée avec exactitude sur les fonctions de référence car cela nécessiterait par exemple de réaliser un nombre infini de tirages de Monte-Carlo. La seule possibilité offerte reste donc d'améliorer au maximum les estimations de la robustesse en ces points.

On retrouve là le but de la recherche du point améliorant général, qui vise à augmenter la qualité des modèles $\hat{\boldsymbol{f}}$ pour mieux prédire les mesures de robustesse $\hat{\boldsymbol{\rho}}$ au niveau des points considérés. Les modèles de substitution $\hat{\boldsymbol{f}}$ des fonctions de référence initiales sont définis sur un espace incertain $\boldsymbol{\Omega}$ qui comprend à la fois les variables de décision $\boldsymbol{d}$ (qui peuvent être incertaines ou non) et les paramètres environnementaux incertains $\boldsymbol{e}$. Le point améliorant général qui va être ajouté au plan d'expériences doit ainsi être recherché dans $\boldsymbol{\Omega}$ et non dans $\boldsymbol{D}$ uniquement. Sinon, on ne fera jamais varier les paramètres environnementaux incertains et il sera alors bien difficile d'évaluer leur effet sur les objectifs du problème déterministe initial. La méthode utilisée pour trouver ce point reste néanmoins similaire au cas déterministe : on cherche le point de $\boldsymbol{\Omega}$ qui une fois ajouté à la base d'apprentissage des modèles $\hat{\boldsymbol{f}}$ permettra de réduire au maximum la taille des intervalles de confiance des estimations de robustesse $\hat{\boldsymbol{\rho}}$ au niveau des points à améliorer sélectionnés dans la première étape. On résout pour cela le problème (8.5) en définissant l'amélioration relative sur les intervalles de confiance des mesures de robustesse plutôt que de $\hat{\boldsymbol{f}}$ directement. Le problème devient ainsi :

$$\underset{\boldsymbol{p}\in\boldsymbol{\Omega}}{\operatorname{argmax}} \sum_{i=1}^{n_\rho} \mathrm{ARR}_i(\boldsymbol{p}), \tag{8.7}$$

avec :

$$\mathrm{ARR}_i(\boldsymbol{p}) = \frac{|\,\mathrm{IC}(\hat{\rho}_i(\mathit{PointsÀAméliorer}_i))| - |\,\mathrm{IC}(\hat{\rho}_i^{+\boldsymbol{p}}(\mathit{PointsÀAméliorer}_i))|}{|\,\mathrm{IC}(\hat{\rho}_i(\mathit{PointsÀAméliorer}_i))|}. \tag{8.8}$$

Algorithme 8.7 : Méthode PareBRO (« Pareto Band Robust Optimization ») de construction adaptative d'un plan d'expériences pour l'optimisation robuste

1 $\boldsymbol{PDE} \leftarrow \boldsymbol{PDEinitial}$
2 Évaluer_avec_fonctions_de_référence($\boldsymbol{PDE}$)
3 **tant que** *le critère d'arrêt n'est pas vérifié* **faire**
4 $\hat{\boldsymbol{f}} \leftarrow$ Construire_modèles($\boldsymbol{PDE}$)
5 $\hat{\boldsymbol{\rho}} \leftarrow$ Mesures_de_robustesse($\hat{\boldsymbol{f}}, \boldsymbol{\chi}$)
6 $\boldsymbol{BandeParetoRobuste} \leftarrow$ NSGA-II_avec_erreur_modèles($\hat{\boldsymbol{\rho}}$)
7 $\boldsymbol{S} \leftarrow$ Classification($\boldsymbol{BandeParetoRobuste}, NombrePointsParallèles$)
8 $\boldsymbol{NouveauxPoints} \leftarrow \emptyset$
9 **pour** $i \in \{1, 2, ..., NombrePointsParallèles\}$ **faire**
10 **pour** $j \in \{1, 2, ..., n_\rho\}$ **faire**
11 $PointsÀAméliorer_j \leftarrow$ Chercher_point_à_ajouter($\hat{\rho}_j, S_i$)
12 **fin**
13 $p \leftarrow$ Chercher_point_améliorant_général($\boldsymbol{PointsÀAméliorer}, \hat{\boldsymbol{\rho}}$)
14 $\boldsymbol{NouveauxPoints} \leftarrow \boldsymbol{NouveauxPoints} \cup \{p\}$
15 Simuler_ajout($p, \hat{\boldsymbol{\rho}}$)
16 **fin**
17 Évaluer_avec_fonctions_de_référence($\boldsymbol{NouveauxPoints}$)
18 $\boldsymbol{PDE} \leftarrow \boldsymbol{PDE} \cup \boldsymbol{NouveauxPoints}$
19 **fin**

Le principe de la méthode PareBRO est décrit dans l'algorithme 8.7. Son déroulement est similaire à celui de PareBO. La seule instruction qui a été ajoutée est l'établissement des fonctions de calcul des mesures de robustesse $\hat{\boldsymbol{\rho}}$ à partir des modèles de substitution $\hat{\boldsymbol{f}}$ et des incertitudes $\boldsymbol{\chi}$ présentes sur les variables d'entrée du problème (ligne 5 de l'algorithme).

Pour mieux comprendre le fonctionnement de cette méthode, nous l'avons mise en œuvre sur un exemple en deux dimensions. Cet exemple est basé sur les fonctions de Michalewicz (équation (4.8)) et chameau (équation (4.7)). La fonction de Michalewicz a en fait été légèrement adaptée comme suit :

$$f_1(\boldsymbol{x}) = 100 - 100\sin(1 - 1.5x_1)\sin^2\left(\frac{2 - 3x_1}{\pi}\right) - 100\sin(1.5 + 1.5x_2)\sin^2\left(\frac{1.5 + 1.5x_2}{\pi}\right). \tag{8.9}$$

La fonction chameau est quant à elle notée f_2. Ces deux fonctions sont représentées sur la figure 8.19. Nous considérons dans ce problème que la première variable d'entrée est une variable de décision $d = x_1$, alors que la seconde est un paramètre environnemental fixé $e = x_2 = 0.3$. L'optimisation va donc se faire uniquement sur d, mais en considérant une incertitude sur e. Cette incertitude est modélisée à l'aide d'une variable aléatoire $\chi_e \sim \mathcal{N}(0, 0.0025)$ ajoutée au paramètre environnemental. Le problème d'optimisation robuste à résoudre est formulé comme un problème de minimisation de deux agrégations des mesures d'espérance et de variance de chacune des fonctions de référence f_1 et f_2. Ces agrégations sont définies sous la forme de quantiles approchés $\hat{\mathrm{Q}}_{0.95}$ d'ordre 0.95 (voir l'équation (7.3)). Le problème s'écrit ainsi :

$$\underset{d \in [-1,1]}{\text{minimiser}} \ \{\rho_1(f_1(d, e + \chi_e)), \rho_2(f_2(d, e + \chi_e))\}, \tag{8.10}$$

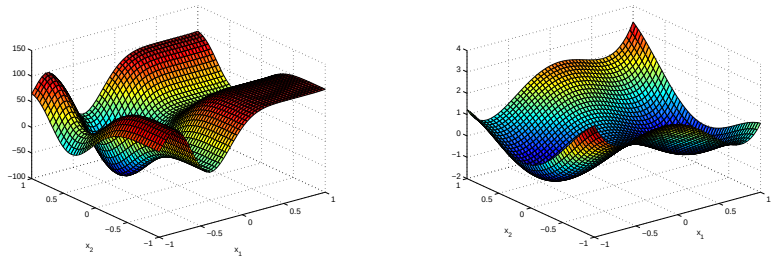


FIGURE 8.19 – Les fonctions de référence de type Michalewicz (à gauche) et chameau (à droite) utilisées dans notre problème d'optimisation robuste.

avec :

$$\rho_i(f_i(d, e+\chi_e)) = \hat{\mathrm{Q}}_{0.95}(f_i(d, e+\chi_e)) \simeq \mathrm{E}[f_i(d, e+\chi_e)] + 1.64\sqrt{\mathrm{Var}(f_i(d, e+\chi_e))}.$$

Nous avons tout d'abord recherché les solutions « idéales » de ce problème en réalisant une optimisation robuste sur les fonctions de référence f_1 et f_2 directement. La robustesse de chaque point étudié a été évaluée à l'aide de 1000 tirages de Monte-Carlo et l'optimisation réalisée grâce à l'algorithme NSGA-II classique avec une population de 100 individus sur 1000 générations. Les solutions idéales obtenues sont présentées sur la figure 8.20. Elles recouvrent une zone relativement restreinte de l'espace de recherche (qui correspond au segment défini par $d \in [-1, 1]$ et $e = 0.3$).

Après avoir déterminé les solutions de référence de notre problème, nous allons maintenant réaliser une résolution par enrichissement adaptatif de modèles de substitution à l'aide de la méthode PareBRO. Nous avons choisi d'utiliser ici des modèles de krigeage car ils permettent d'évaluer les mesures de robustesse impliquées ainsi que leur erreur de manière analytique (le détail de ces calculs est donné en annexe). Nous sommes partis d'un plan d'expériences initial constitué par un LHS de 5 échantillons d'apprentissage. Ces échantillons doivent être choisis au sein de l'espace incertain $\mathbf{\Omega} = [-1, 1] \times \mathbb{R}$ (car le support de la distribution de l'incertitude sur e est infini) afin de modéliser le comportement des fonctions de référence à la fois sur les dimensions de décision et sur les dimensions incertaines. Il aurait été astucieux de restreindre cet espace autour des valeurs les plus probables pour e (par exemple dans un voisinage de ± 3 écarts-types autour de sa moyenne comme représenté sur les figures) pour ne pas trop disperser les échantillons initiaux. Nous avons préféré ici laisser varier e entre -1 et 1 pour mieux observer le comportement de notre méthode (qui concentrera par elle-même ses évaluations autour de la valeur nominale de ce paramètre incertain). On a donc $\mathbf{\Omega} = [-1, 1]^2$. Nous avons établi deux modèles de krigeage $\hat{f}_1$ et $\hat{f}_2$ sur cet espace à partir du plan d'expériences initial, et recherché les solutions de la bande de Pareto correspondantes à l'aide de l'algorithme NSGA-II modifié pour prendre en compte l'erreur de ces modèles, avec une population de 20 individus sur 500 générations. La figure 8.21 présente les modèles et les solutions possiblement optimales obtenues. On voit que l'erreur importante des modèles fait que ces solutions recouvrent pour l'instant la totalité de l'espace de recherche. Il faut donc enrichir les modèles à l'aide d'échantillons d'apprentissage supplémentaires judicieusement choisis. Nous supposons ici que trois échantillons peuvent être évalués en parallèle sur les fonctions de référence à chaque itération. Les solutions de la bande de Pareto ont donc été séparées

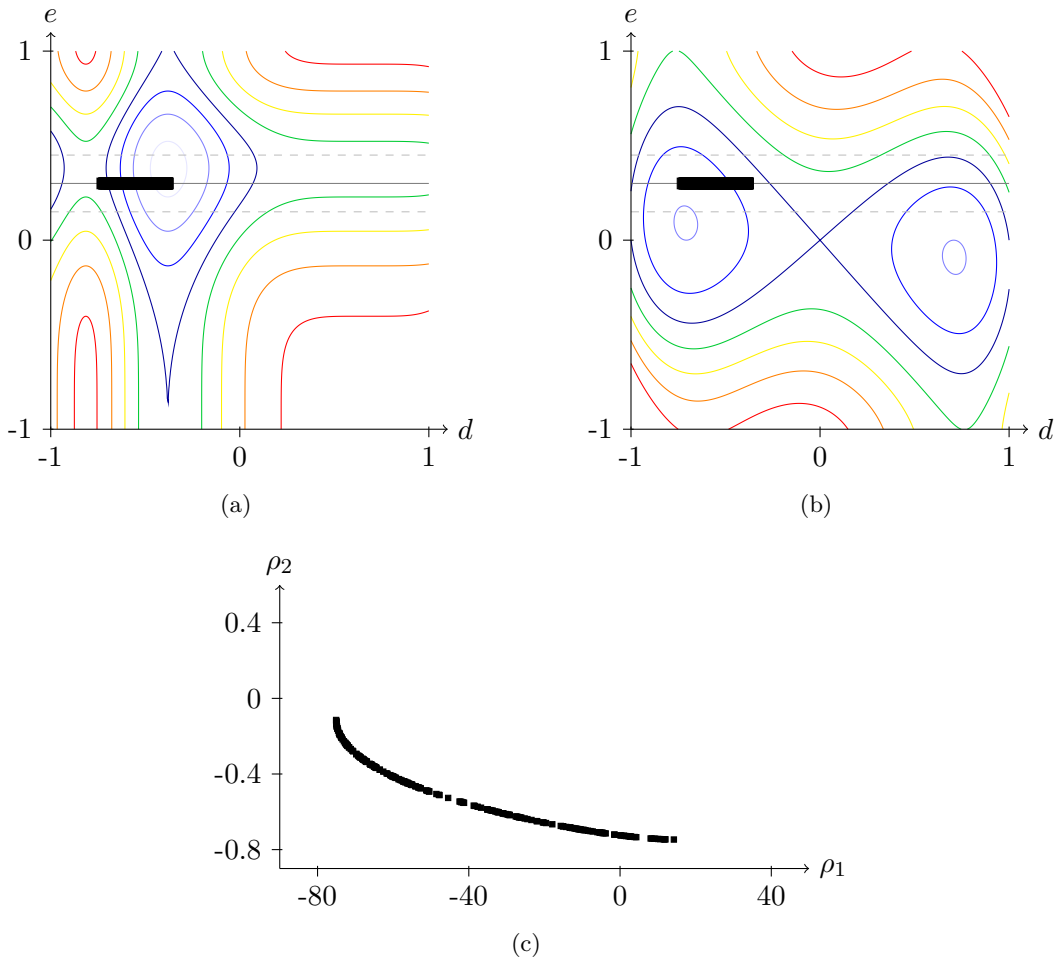


FIGURE 8.20 – Solutions « idéales » (carrés noirs) pour le problème d'optimisation robuste 8.10, représentées dans l'espace d'entrée (a) (b) ainsi que dans l'espace des objectifs (c). Les lignes de niveau des fonctions de référence f_1 (a) et f_2 (b) sont tracées en couleur. Sur ces graphes, la valeur nominale de la variable incertaine $(e + \chi_e)$ est indiquée par une ligne grise, et son intervalle de variation à ± 3 écarts-types par des pointillés.

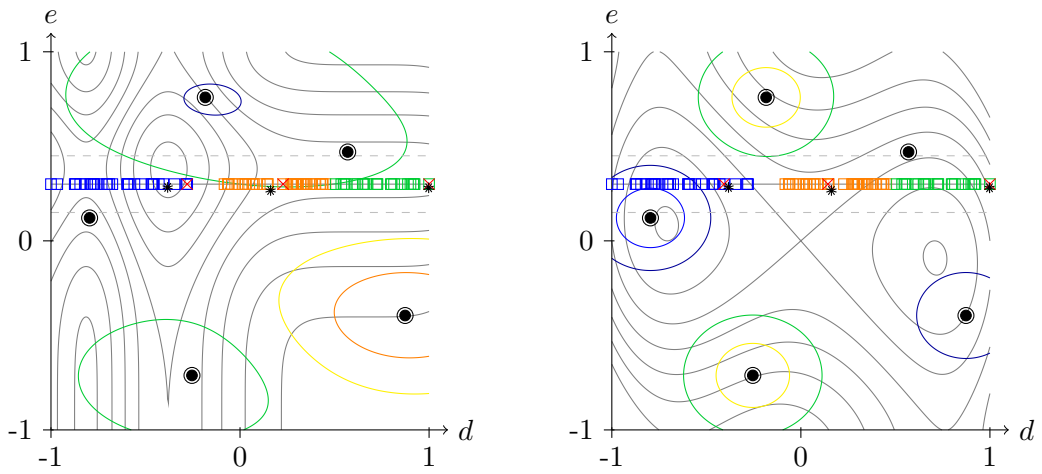


FIGURE 8.21 – Modèles de krigeage et solutions de la bande de Pareto obtenus à partir du LHS initial (points noirs entourés d'un cercle). Les courbes de niveau des modèles $\hat{f}_1$ (à gauche) et $\hat{f}_2$ (à droite) sont tracées en couleur, celles des fonctions de référence en gris. Les solutions possiblement optimales sont représentées par des carrés et réparties en trois classes distinctes. Les points les plus mal décrits de chaque classe sont indiqués par des croix rouges sur chaque modèle. Les étoiles représentent les échantillons améliorants sélectionnés pour être évalués avec les fonctions de référence.

en trois classes S_1, S_2 et S_3 distinctes sur d à l'aide de l'algorithme des k-moyennes. Nous allons chercher à présent un échantillon améliorant pour chacune de ces classes.

On commence par étudier le premier groupe S_1 de solutions possiblement optimales (les solutions bleues). La figure 8.22 présente la valeur prédite en ces points pour chacun des objectifs de robustesse ρ_i. Les mesures de robustesse ainsi que leurs intervalles de confiance ont été calculées analytiquement sur les modèles de krigeage $\hat{f}_1$ et $\hat{f}_2$. On peut remarquer que la taille de ces intervalles est légèrement inférieure dans les zones situées à proximité d'un échantillon d'apprentissage, mais contrairement aux modèles de substitution interpolants classiques il n'y a pas de point connu avec certitude. Les intervalles de confiance sur la robustesse permettent de déterminer les solutions de S_1 les plus mal décrites pour chacun des objectifs. Ce sont ces deux solutions qu'on va chercher à améliorer dans le but de réduire l'erreur sur les estimations de robustesse de l'ensemble de la classe. Pour cela, on recherche dans l'espace incertain $\mathbf{\Omega}$ le point qui maximise la somme des améliorations relatives sur la prédiction de la robustesse de ces solutions (problème (8.7)). Les variations de la valeur de cette somme sont représentées sur le graphe (c) de la figure 8.22. Son maximum a été déterminé à l'aide de l'algorithme NSGA-II classique avec une population de 10 individus sur 250 générations. On voit que le point qui promet une amélioration maximale est situé entre les deux solutions les plus mal décrites, à proximité de celle qui provient de ρ_2. Le point retourné par NSGA-II ne correspond pas exactement au maximum réel (ce qui est normal car il s'agit d'un algorithme de résolution approchée), mais il en est tout de même très proche. La valeur de e en ce point est pratiquement égale à sa valeur nominale, ce qui n'est pas étonnant car celle-ci a une grande influence sur la précision de l'estimation de la robustesse. S'il y avait eu un échantillon d'apprentissage présent à proximité, la recherche du point améliorant général aurait aussi bien pu favoriser

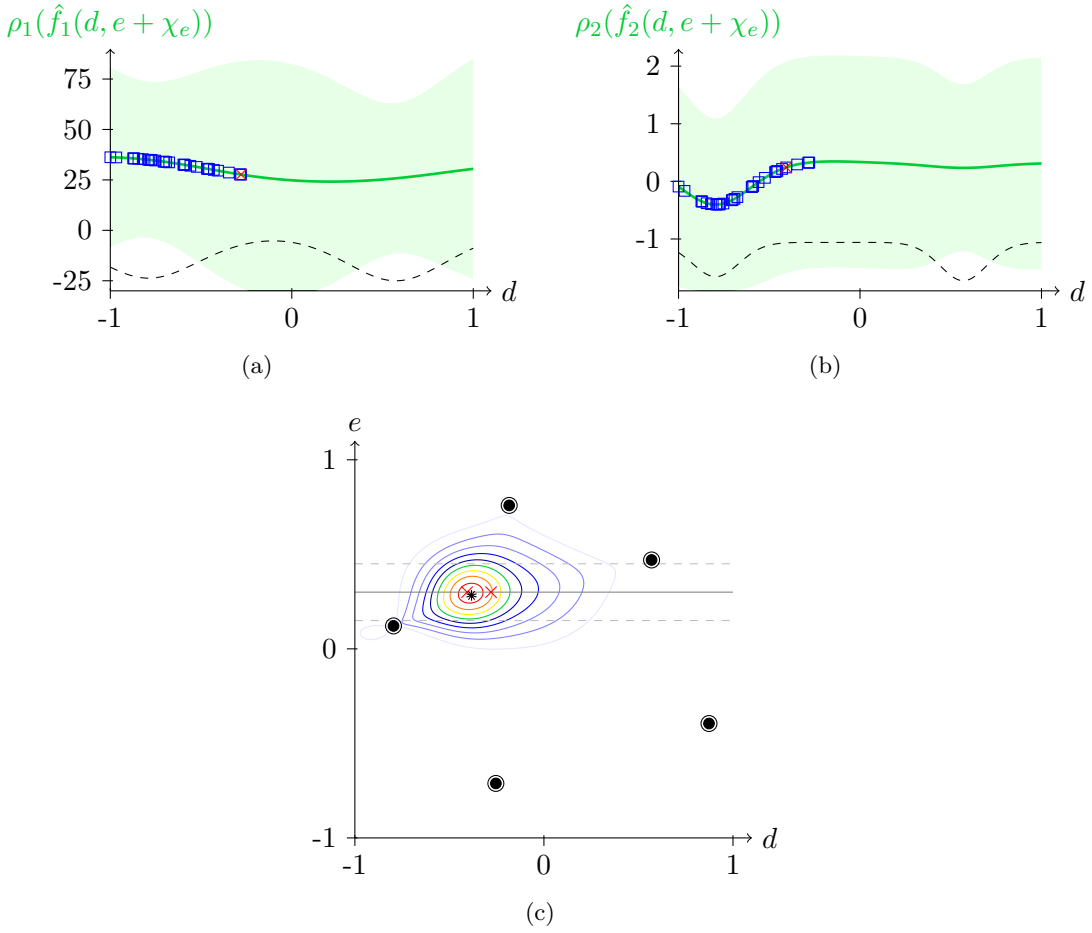


FIGURE 8.22 – Objectifs de robustesse ρ_1 (a) et ρ_2 (b) du problème avec leur intervalle de confiance à 95% (en vert). L'évolution de la taille de ces intervalles est tracée en pointillés sans respect d'échelle. Les solutions de la classe S_1 sont représentées par des carrés bleus, et la solution la plus mal décrite est indiquée par une croix rouge sur chaque modèle. Le graphe (c) présente les lignes de niveau de la somme des améliorations relatives pour la recherche du point améliorant les deux solutions les plus mal décrites (croix rouges). Les échantillons du LHS initial sont figurés par des points noirs entourés d'un cercle. Le point améliorant général obtenu est marqué d'une étoile.

une autre valeur pour ce paramètre. Nous avons donc finalement obtenu un premier point améliorant pour les solutions de la classe S_1. Ce point ne va pas être évalué de suite sur les fonctions de référence car il faut encore déterminer les échantillons équivalents pour les deux autres classes. On simule toutefois son ajout aux modèles $\hat{f}_1$ et $\hat{f}_2$ afin qu'il soit pris en compte dans les futures estimations d'erreur sur les mesures de robustesse.

Nous nous intéressons ensuite au deuxième groupe S_2 de solutions de la bande de Pareto (les solutions oranges). La valeur de ces solutions sur chacun des objectifs du problème est visualisée sur la figure 8.23. On peut voir sur cette figure que les intervalles de confiance des estimations de robustesse sont devenus beaucoup moins larges aux alentours du point que nous avons sélectionné pour S_1. L'ajout simulé de ce point à la base d'apprentissage des modèles va ainsi permettre d'éviter de choisir un autre échantillon trop proche. Comme auparavant, la recherche du point améliorant général de S_2 nécessite dans un premier temps de déterminer les deux solutions de la classe les plus mal décrites (c'est-à-dire les solutions pour lesquelles les intervalles de confiance des estimations de robustesse sont les plus grands), puis de trouver la position du maximum de la somme des améliorations relatives de ces solutions. Le point obtenu est une fois de plus situé entre les deux solutions à améliorer. Cet échantillon est alors ajouté aux modèles $\hat{f}_1$ et $\hat{f}_2$ de façon simulée, et on procède de la même manière pour déterminer l'échantillon améliorant le dernier groupe S_3 de solutions possiblement optimales. Les trois échantillons sélectionnés lors de cette première itération de l'algorithme PareBRO sont localisés sur la figure 8.21.

Nous avons évalué ces trois échantillons avec les fonctions de référence avant de reconstruire les modèles de krigeage $\hat{f}_1$ et $\hat{f}_2$. Les nouveaux modèles obtenus ainsi que les solutions possiblement optimales correspondantes sont représentés sur la figure 8.24. Si les modèles ne sont pas encore de très bonne qualité, on peut tout de même observer qu'un des échantillons ajoutés a déjà permis d'écarter la partie extrême droite de l'espace de recherche des zones possiblement optimales. L'erreur des modèles est en effet suffisamment faible dans cette région pour pouvoir affirmer qu'aucune solution optimale ne s'y trouve. À partir de ces nouveaux modèles, l'algorithme est déroulé selon le même principe pour déterminer les points les plus mal décrits de chaque classe de solutions ainsi que les points améliorants généraux correspondants. Les trois échantillons obtenus sont ensuite évalués et on commence une nouvelle itération. La figure 8.25 présente l'état des modèles et les solutions de la bande de Pareto trouvées au début de cette itération. On remarque que l'espace possiblement optimal a été considérablement réduit. Toute la partie droite de l'espace de recherche n'est ainsi plus considérée lors de l'amélioration des modèles. Trois échantillons améliorants supplémentaires sont alors sélectionnés et intégrés au plan d'expériences. Les résultats finaux après ajout de ces trois nouveaux points sont présentés sur la figure 8.26. On voit que les modèles sont maintenant relativement corrects dans la zone des solutions optimales, où la densité d'échantillons d'apprentissage est la plus importante. Ces échantillons sont de plus concentrés autour des valeurs les plus probables du paramètre e. Partout ailleurs, la qualité des modèles est nettement inférieure, mais il s'agit de régions de l'espace dont la connaissance ne présente pas beaucoup d'intérêt. La méthode PareBRO a ainsi permis de focaliser les efforts d'amélioration des modèles dans les zones les plus judicieuses. La partie de l'espace de recherche recouverte par les solutions de la bande de Pareto reste par contre légèrement plus grande que celle des solutions idéales de la figure 8.20 du fait des erreurs de modélisation qui persistent encore. Ces erreurs sont visibles sur le tracé de la bande de Pareto dans l'espace des objectifs : on remarque que certains points sont inclus dans la bande optimale du fait d'intervalles

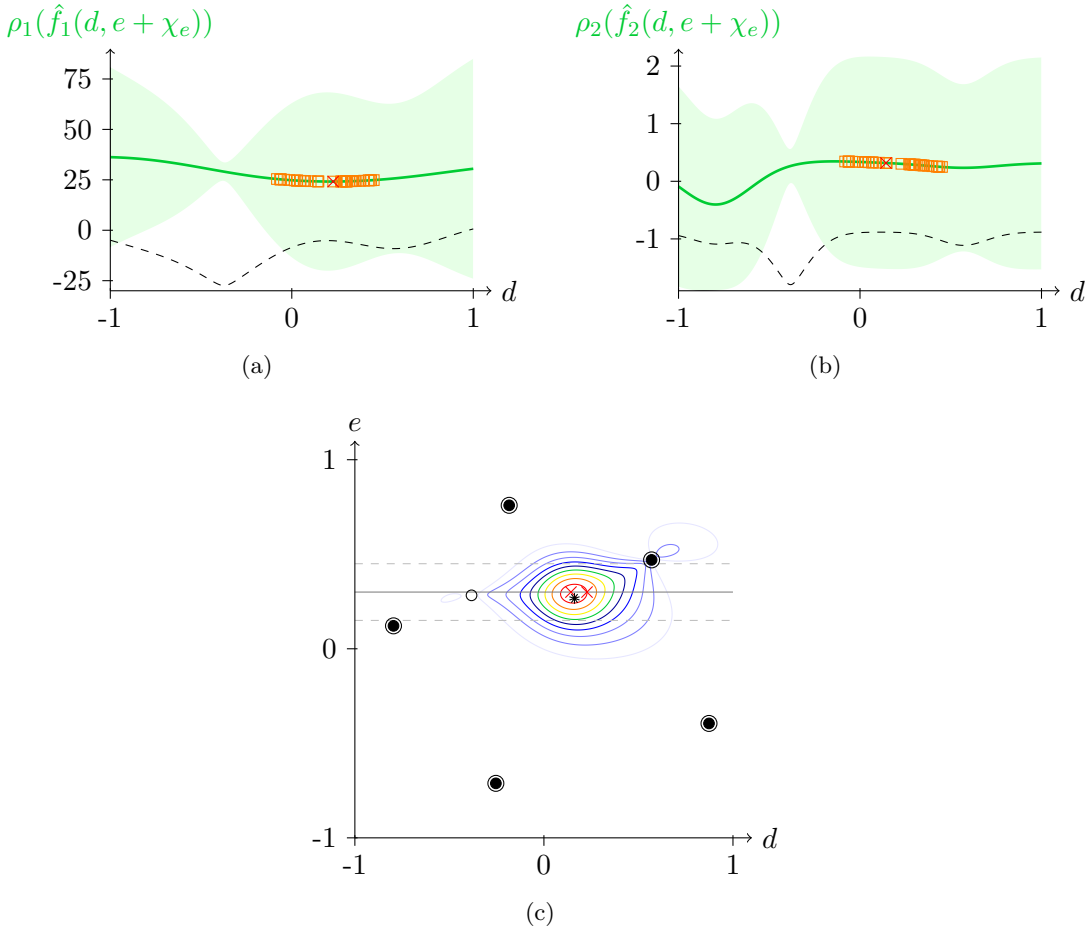


FIGURE 8.23 – Objectifs de robustesse ρ_1 (a) et ρ_2 (b) du problème avec leur intervalle de confiance à 95% (en vert). L'évolution de la taille de ces intervalles est tracée en pointillés sans respect d'échelle. Les solutions de la classe S_2 sont représentées par des carrés oranges, et la solution la plus mal décrite est indiquée par une croix rouge sur chaque modèle. Le graphe (c) présente les lignes de niveau de la somme des améliorations relatives pour la recherche du point améliorant les deux solutions les plus mal décrites (croix rouges). Les échantillons du LHS initial sont figurés par des points noirs entourés d'un cercle, et la position de l'échantillon améliorant précédent dont l'ajout a été simulé est indiquée par un cercle. Le nouveau point améliorant général obtenu est marqué d'une étoile.

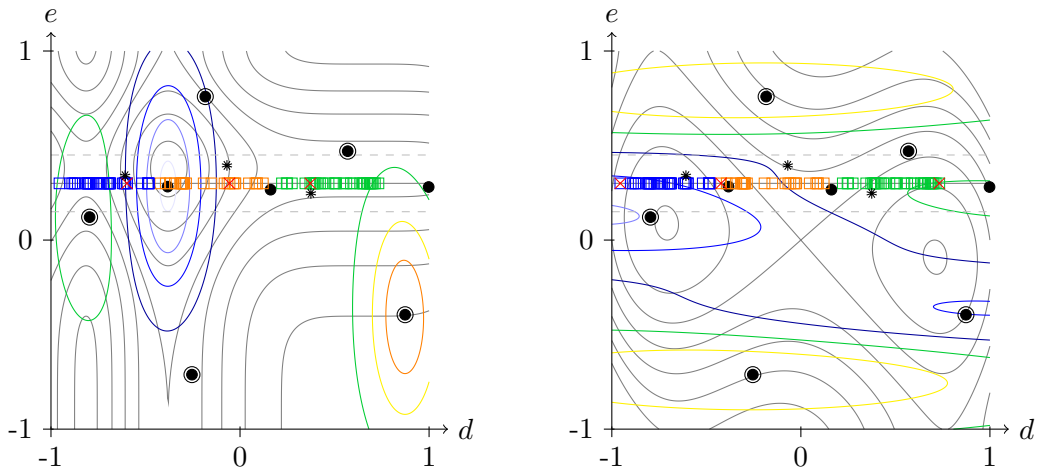


FIGURE 8.24 – Modèles de krigeage et solutions de la bande de Pareto obtenus à partir de 8 échantillons d'apprentissage (points noirs). Les échantillons provenant du LHS initial sont entourés d'un cercle. Les courbes de niveau des modèles $\hat{f}_1$ (à gauche) et $\hat{f}_2$ (à droite) sont tracées en couleur, celles des fonctions de référence en gris. Les solutions possiblement optimales sont représentées par des carrés et réparties en trois classes distinctes. Les points les plus mal décrits de chaque classe sont indiqués par des croix rouges sur chaque modèle. Les étoiles représentent les échantillons améliorants sélectionnés pour être évalués avec les fonctions de référence.

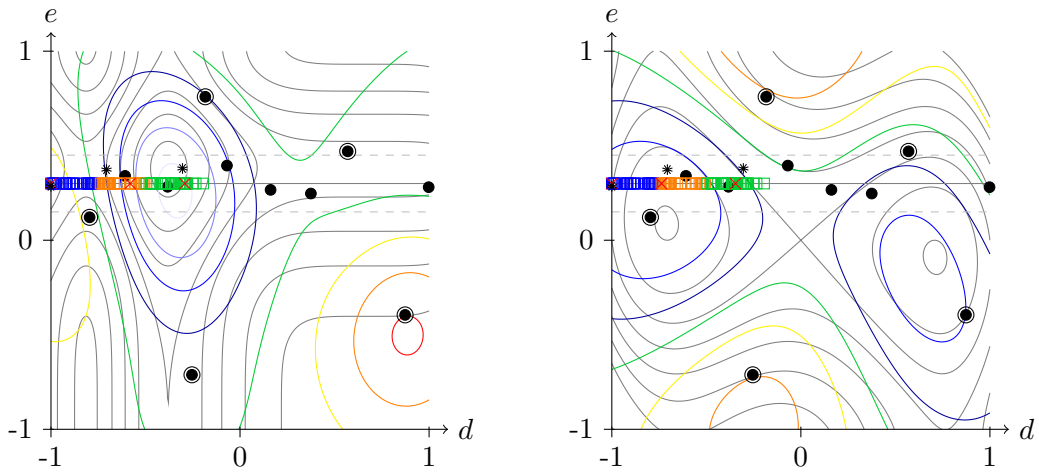


FIGURE 8.25 – Modèles de krigeage et solutions de la bande de Pareto obtenus à partir de 11 échantillons d'apprentissage (points noirs). Les échantillons provenant du LHS initial sont entourés d'un cercle. Les courbes de niveau des modèles $\hat{f}_1$ (à gauche) et $\hat{f}_2$ (à droite) sont tracées en couleur, celles des fonctions de référence en gris. Les solutions possiblement optimales sont représentées par des carrés et réparties en trois classes distinctes. Les points les plus mal décrits de chaque classe sont indiqués par des croix rouges sur chaque modèle. Les étoiles représentent les échantillons améliorants sélectionnés pour être évalués avec les fonctions de référence.

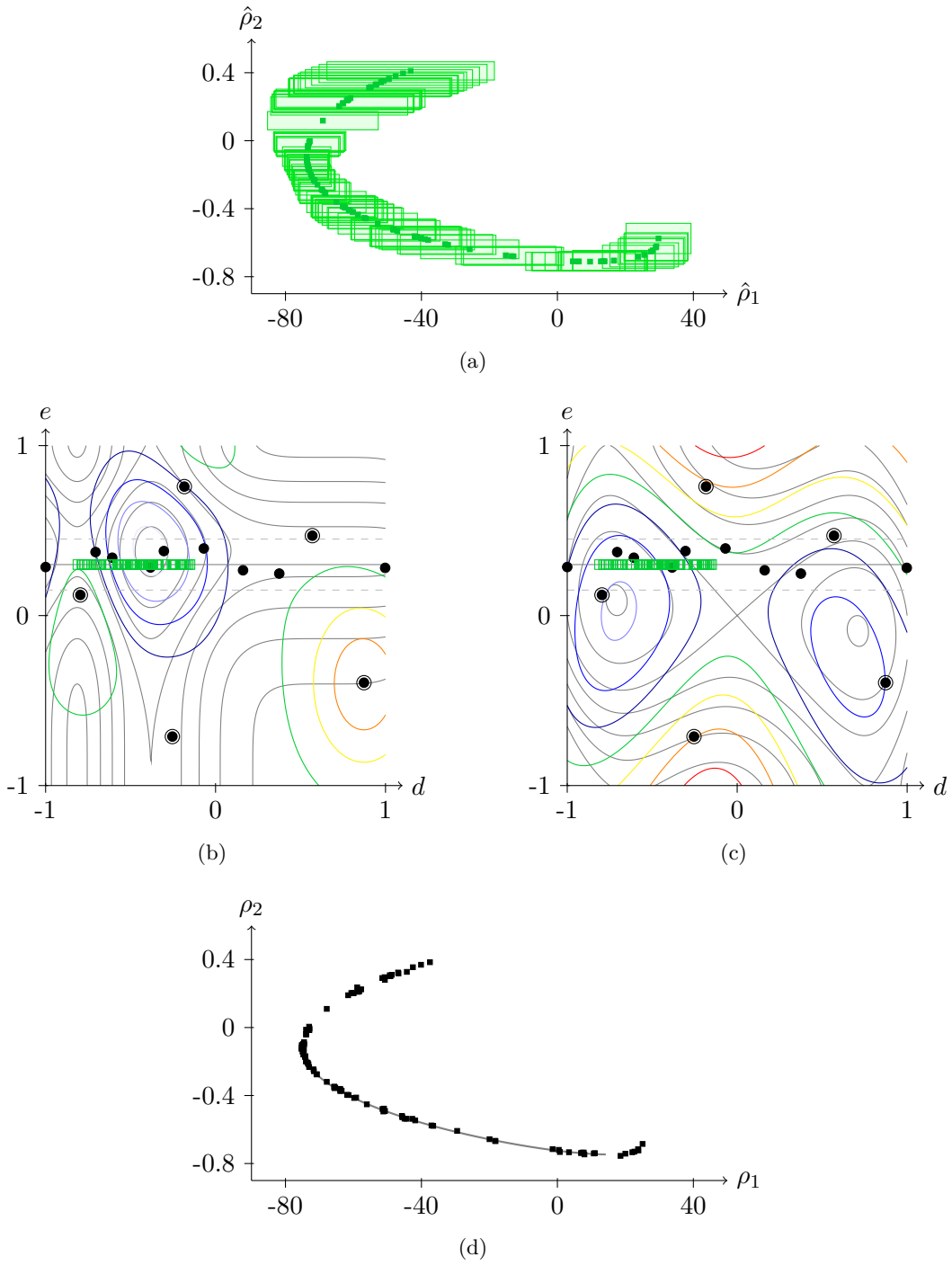


FIGURE 8.26 – Solutions de la bande de Pareto obtenus avec le plan d'expériences final constitué de 14 échantillons d'apprentissage (points noirs). Les échantillons provenant du LHS initial sont entourés d'un cercle. Les courbes de niveau des modèles $\hat{f}_1$ (b) et $\hat{f}_2$ (c) sont tracées en couleur, celles des fonctions de référence en gris. Les solutions possiblement optimales sont représentées par des carrés verts. La graphe (a) présente ces même solutions dans l'espace des objectifs, et le graphe (d) leur robustesse recalculée avec les fonctions de référence. La front de Pareto idéal de la figure 8.20 est indiqué par une ligne grise.

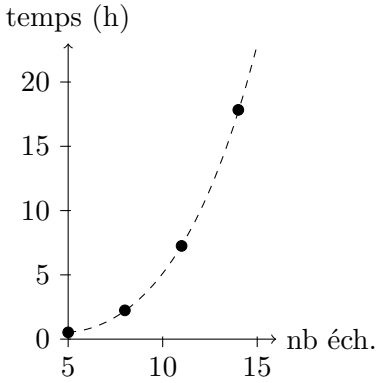


FIGURE 8.27 – Évolution du temps de calcul (en heures) pour la recherche des points améliorants en fonction du nombre d'échantillons d'apprentissage déjà présents dans le plan d'expériences.

de confiance importants (notamment au niveau du premier objectif). Les résultats présentés font néanmoins apparaître clairement cette information, et le choix final du décideur se portera donc certainement vers des solutions mieux décrites. On aurait aussi pu faire encore quelques itérations d'enrichissement des modèles pour réduire leur erreur et mieux se rapprocher du front de Pareto idéal. Afin de vérifier la qualité des solutions obtenues, nous avons recalculé la robustesse des points de la bande de Pareto grâce à 1000 tirages de Monte-Carlo réalisés sur les fonctions de référence f_1 et f_2 directement. Les valeurs obtenues sont présentées sur le graphe (d) de la figure 8.26. On voit qu'une partie de ces solutions ne sont effectivement pas Pareto-optimales, mais par contre on obtient bien des points répartis sur l'ensemble du front de Pareto idéal.

L'inconvénient majeur de la méthode PareBRO reste son coût, qui peut rapidement devenir prohibitif du fait des multiples boucles d'optimisation imbriquées (l'instruction la plus coûteuse étant la recherche du point améliorant général à la ligne 13 de l'algorithme 8.7). Dans l'exemple que nous venons de traiter, ce coût est lié à l'évaluation analytique de la robustesse et de ses intervalles de confiance sur les modèles de krigeage. En effet, la complexité de l'estimation des mesures de quantile approché évolue relativement au nombre d'échantillons d'apprentissage à la puissance quatre. La figure 8.27 présente l'évolution du temps de calcul mesuré lors de la recherche des points améliorants de notre problème en fonction du nombre d'échantillons d'apprentissage présents dans le plan d'expériences (ce nombre augmente de trois en trois au cours de l'enrichissement). On retrouve bien le comportement quartique du coût des calculs expérimentalement. La durée de planification entre deux évaluations d'échantillons sur les fonctions de référence monte ainsi à près de 18h alors que le plan d'expériences contient 14 échantillons seulement. On aurait pu se ramener à une complexité quadratique beaucoup plus acceptable si le problème d'optimisation robuste à résoudre n'avait fait intervenir que la mesure d'espérance (la mesure de variance présente au sein du quantile approché étant la plus onéreuse à évaluer).

Il est à noter que ce temps de calcul aurait aussi été important si on avait utilisé un autre type de modèles de substitution que le krigeage. La recherche des points améliorants généraux nécessite en effet de simuler l'ajout de points aux modèles, puis d'estimer les intervalles de confiance des mesures de robustesse associées pour un certain nombre de

solutions. Les modèles de krigeage ont l'avantage de fournir directement une estimation de l'erreur de leurs prédictions et de pouvoir prendre facilement en compte de nouveaux échantillons d'apprentissage. Pour les autres types de modèles, l'erreur doit être évaluée par des techniques statistiques comme le bootstrap par exemple, et l'intégration de nouveaux échantillons nécessite généralement une reconstruction complète du modèle. Lors de la recherche de points améliorants avec des modèles quelconques, il faut donc pour chaque point considéré reconstruire les modèles en intégrant ce point à leur base d'apprentissage, puis déterminer l'erreur sur les estimations de robustesse par bootstrap (en construisant plusieurs modèles sur des bases différentes et en évaluant les mesures de robustesse sur chacun d'eux). Le processus est donc là aussi relativement coûteux, bien qu'il puisse être préférable à l'utilisation du krigeage en fonction du nombre d'échantillons d'apprentissage à considérer ainsi que des techniques de calcul retenues.

La méthode PareBRO nous a ainsi permis d'établir un plan d'expériences ciblant les zones potentiellement optimales d'un problème d'optimisation robuste multiobjectifs grâce à la prise en compte de l'erreur des modèles de substitution utilisés pour l'estimation des mesures de robustesse. Nous avons finalement obtenu un ensemble de solutions contenant les solutions optimales réelles du problème.

8.4 Conclusion

Nous avons présenté dans ce chapitre le développement d'une approche de construction adaptative de plans d'expériences avec évaluation de plusieurs échantillons en parallèle dédiée à l'optimisation multiobjectifs, aussi bien déterministe avec la méthode PareBO que robuste avec la méthode PareBRO. Cette approche est basée sur la recherche de la bande de Pareto d'un problème, une extension de la notion de front de Pareto qui tient compte des intervalles de confiance des prédictions des modèles de substitution utilisés. Les solutions qui composent cette bande de Pareto sont représentatives de l'ensemble des régions possiblement optimales de l'espace de recherche. Ces différentes régions sont distinguées par classification afin de sélectionner des échantillons améliorant la qualité des modèles sur chacune d'entre elles.

Ces deux méthodes pourraient être améliorées afin de réduire leur coût, notamment dans le cadre de l'optimisation robuste où celui-ci peut devenir problématique. L'étape la plus coûteuse étant la recherche du point améliorant général, c'est sur celle-ci qu'il faudrait concentrer les efforts. Nous avons vu que cette recherche pourrait parfois être évitée dans le cadre d'une optimisation déterministe si les points à améliorer de chaque modèle sont situés au même endroit (on sélectionne alors directement un de ces points). En optimisation robuste, on pourrait aussi envisager une stratégie plus simple de recherche du point améliorant général en limitant par exemple la zone de recherche à la classe considérée (car le point obtenu est souvent situé dans cette région).

Lors de leur mise au point, nous avons testé les méthodes PareBO et PareBRO sur différents exemples analytiques relativement basiques. Nous allons maintenant les appliquer à un cas test réel de taille plus conséquente, dont la résolution est présentée dans le chapitre suivant.

Chapitre 9

Optimisation robuste d'un système d'injection

Sommaire

9.1 Bref rappel du problème 208
9.2 Amélioration globale des modèles de substitution 210
9.3 Formulation du problème d'optimisation robuste 214
9.3.1 Comparaison des méthodes de calcul de la robustesse 215
9.3.2 Comparaison des méthodes d'estimation d'erreur sur la robustesse 217
9.4 Optimisation robuste adaptative à l'aide de la méthode PareBRO 218
9.5 Détermination des solutions optimales robustes 223
9.5.1 Étude des solutions obtenues grâce à l'enrichissement adaptatif 223
9.5.2 Sélection de quelques solutions intéressantes 225
9.6 Analyse des solutions optimales obtenues 227
9.6.1 Validation par Monte-Carlo 227
9.6.2 Comparaison des solutions 229
9.7 Conclusion 231

Le premier cas d'application de cette étude concerne l'optimisation d'un système d'injection multipoints en vue de réduire les émissions polluantes d'une chambre de combustion de turbomachine aéronautique. Le système est soumis à une incertitude sur la répartition du débit de carburant entre ses différents injecteurs, car il peut arriver que ceux-ci se bouchent de manière impromptue. Nous avons ainsi affaire à un problème d'optimisation robuste. Ce chapitre présente la résolution de ce problème à l'aide de la méthode PareBRO développée précédemment, qui permet de guider la complétion adaptative d'un plan d'expériences pour l'optimisation robuste multiobjectifs par modèles de substitution.

Après un bref rappel des données de l'application à traiter, nous abordons chacune des étapes successives de cette résolution. Le déroulement général est présenté sur la figure 9.1. À partir d'un plan d'expériences initial de taille réduite (30 échantillons d'apprentissage), nous avons choisi de réaliser tout d'abord une amélioration globale des modèles des fonctions de référence du problème. Cela permet d'obtenir des modèles de départ relativement

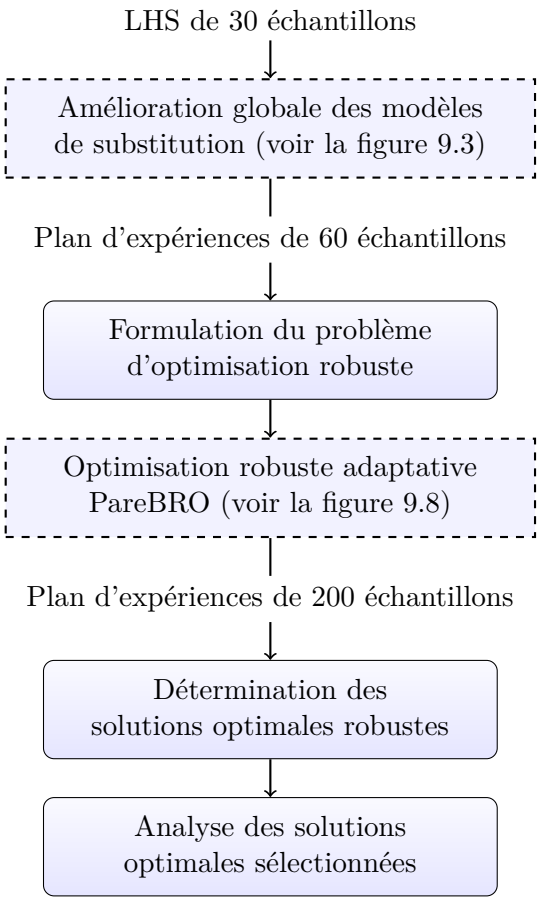


FIGURE 9.1 – Déroulement de la résolution du cas d'application en combustion.

corrects et qui ont du sens avant de démarrer toute optimisation. 30 échantillons supplémentaires sont ainsi rajoutés au plan d'expériences dans ce but. Les modèles établis sont ensuite utilisés pour comparer les différentes approches de calcul des mesures de robustesse afin de définir la formulation finale du problème d'optimisation robuste auquel nous allons nous intéresser. Ce problème est résolu de manière adaptative à l'aide de la méthode PareBRO. Le plan d'expériences qui en résulte contient 200 échantillons des fonctions de référence, à partir desquels nous déterminons un ensemble de solutions optimales robustes choisies. Ces solutions sont enfin analysées plus en détail pour valider leur robustesse et comparer leurs caractéristiques.

9.1 Bref rappel du problème

Nous nous intéressons à l'optimisation du système d'injection multipoints présenté dans le chapitre 1. Ce système permet d'alimenter en kérosène la chambre de combustion d'une turbomachine aéronautique. Le comportement du foyer est simulé à l'aide du code CEDRE [CCD+05] de l'Onera via une modélisation simplifiée 2D RANS axisymétrique qui permet de bénéficier d'un temps de calcul réduit. Chaque évaluation d'une configuration donnée prend tout de même deux jours environ sur une machine standard, mais on a la possibilité d'effectuer dix simulations en parallèle. La représentation numérique de la

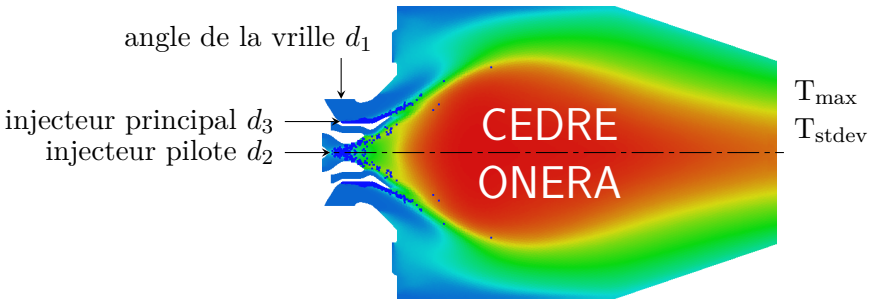


FIGURE 9.2 – Visualisation des variables de décision et des objectifs du système d'injection à optimiser.

Variables de décision			
d_1	Angle de la vrille	[30,60]	°
d_2	Position de l'injecteur pilote	[0.00341,0.0101]	m
d_3	Position de l'injecteur principal	[0.0018,0.0249]	m
Paramètre environnemental			
e_1	Répartition du carburant entre les deux injecteurs	0	
Incertitude			
χ_{e_1}	Répartition du carburant entre les deux injecteurs	$\mathcal{N}(0, 0.000576)$	
Objectifs			
f_1	Température maximale à l'intérieur de la chambre T_{max}	minimiser	K
f_2	Écart-type de la température en sortie T_{stdev}	minimiser	K

TABLE 9.1 – Variables et objectifs du problème d'optimisation du système d'injection.

physique du système fait intervenir des modélisations stochastiques, ce qui peut engendrer un léger bruit sur les résultats obtenus.

Le but de l'optimisation de ce système d'injection est de minimiser la température maximale T_{max} dans la chambre de combustion ainsi que son écart-type T_{stdev} en sortie, afin de réduire autant que possible les émissions de gaz polluants. Pour cela, il est possible de jouer sur l'angle de la vrille d'une des entrées d'air de la chambre, ainsi que sur la position des injecteurs pilote et principal. Il peut par contre arriver de manière aléatoire que ces injecteurs se bouchent partiellement au cours du fonctionnement de la turbomachine, modifiant alors le partage du kérosène entre les deux points d'injection. On souhaite donc prendre en compte une incertitude sur la répartition du débit de carburant entre les injecteurs. Cette incertitude est modélisée sous la forme d'une variable aléatoire χ_{e_1} dotée d'une loi de probabilité normale. Sa valeur moyenne correspond à la répartition nominale du kérosène lorsque les deux injecteurs fonctionnent correctement. Le problème à résoudre possède ainsi trois paramètres de conception $\boldsymbol{d}$, un paramètre environnemental

incertain χ_{e_1} et deux objectifs f_1 et f_2. Ils sont rappelés sur la figure 9.2 ainsi que dans le tableau 9.1. Les deux objectifs sont calculés simultanément par la simulation numérique du système à partir des valeurs spécifiées pour les paramètres d'entrée. Le problème stochastique multiobjectifs auquel on est confronté s'exprime comme suit :

$$\underset{\boldsymbol{d}}{\text{minimiser}} \ \{f_1(\boldsymbol{d}, \chi_{e_1}), f_2(\boldsymbol{d}, \chi_{e_1})\}. \tag{9.1}$$

9.2 Amélioration globale des modèles de substitution

La simulation numérique du fonctionnement de la chambre de combustion étant assez coûteuse, il semble évident que l'optimisation ne pourra pas être menée directement sur celle-ci. Nous avons donc commencé par établir deux modèles de krigeage $\hat{f}_1$ et $\hat{f}_2$ pour représenter les objectifs du problème (9.1). Le krigeage a été choisi ici car il permet d'obtenir de bonnes représentations de fonctions dont les caractéristiques sont inconnues dans un espace de dimension restreinte, et car il offre de plus des possibilités de formulation analytique de certaines mesures de robustesse. Afin de nous prémunir contre le léger bruit sur les résultats de la simulation du système, nous avons utilisé des modèles de krigeage régressants qui sont à même d'effectuer un lissage des données d'apprentissage (voir la section 3.4.4).

Les modèles initiaux sont construits grâce à un LHS de 30 points dans un espace de dimension 4 afin de considérer les variations des trois variables de décision $\boldsymbol{d}$ ainsi que du paramètre incertain χ_{e_1}. Malgré le support infini de la distribution de probabilité de ce dernier, sa plage de variation a été limitée pour cette étude à l'intervalle $[-0.05, 0.05]$ qui recouvre tout de même en probabilité plus de 95% des valeurs possibles. Le plan d'expériences évolue donc dans l'espace :

$$\boldsymbol{\Omega} = [30, 60] \times [0.00341, 0.0101] \times [0.0018, 0.0249] \times [-0.05, 0.05] \subset \mathbb{R}^4.$$

Au lieu de démarrer directement l'optimisation sur ces modèles initiaux, il nous a semblé préférable de les améliorer dans un premier temps sur l'ensemble de leur domaine de définition $\boldsymbol{\Omega}$. Cette étape n'est pas obligatoire, mais elle permet de s'assurer que les modèles ont une qualité suffisante et que leurs prédictions ont un sens avant de lancer le processus d'optimisation. Celui-ci est en effet basé sur les valeurs ponctuelles et les intervalles de confiance estimés par les modèles, et il peut donc être souhaitable d'affiner légèrement ces prédictions en amont plutôt que d'effectuer des calculs complexes d'optimisation adaptative avec des estimations trop fantaisistes. Grâce aux possibilités de parallélisation des évaluations des fonctions de référence, nous avons ainsi ajouté des séries de dix points au plan d'expériences initial en vue d'améliorer globalement $\hat{f}_1$ et $\hat{f}_2$. Nous avons utilisé pour cela la méthode d'amélioration simultanée de plusieurs modèles présentée dans l'algorithme 8.4 du chapitre précédent. Son principe dans le cadre de cette application est schématisé sur la figure 9.3.

À chaque itération de la méthode adaptative, dix échantillons améliorants sont sélectionnés pour être évalués avec la simulation numérique du système. La sélection de chacun de ces échantillons se déroule en deux temps : on détermine tout d'abord les points qui maximisent la variance de chaque modèle de krigeage (il s'agit des deux points les plus mal décrits dont on souhaiterait améliorer la prédiction afin de réduire l'erreur des modèles), puis on recherche l'échantillon qui maximise l'amélioration des intervalles de confiance des

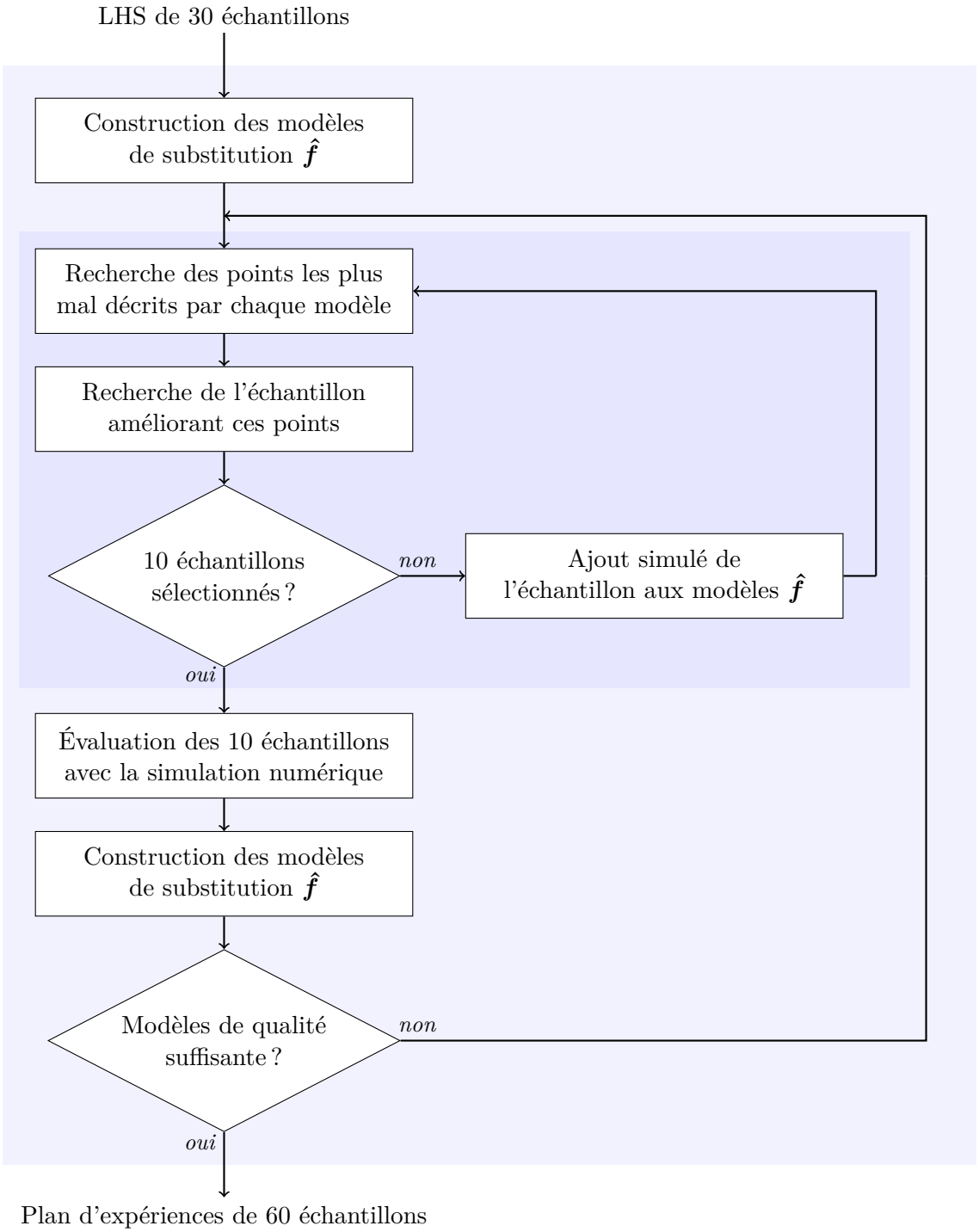


FIGURE 9.3 – Méthode d'amélioration globale simultanée de modèles de substitution utilisée pour notre cas d'application.

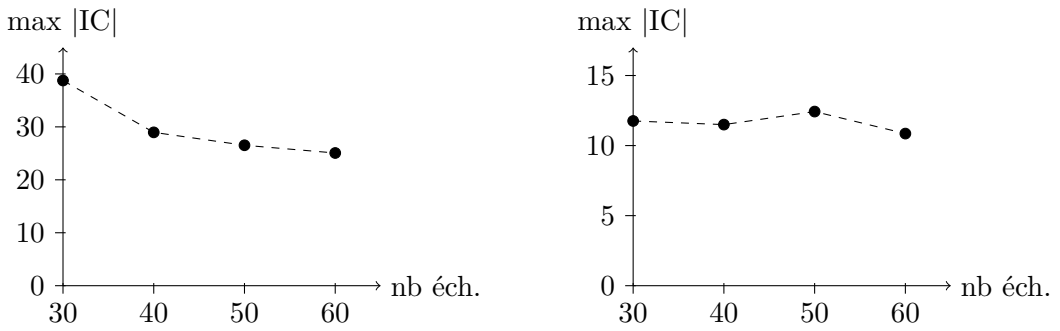


FIGURE 9.4 – Évolution de la taille maximale des intervalles de confiance des prédictions des modèles $\hat{f}_1$ (à gauche) et $\hat{f}_2$ (à droite) en 100 points d'observation en fonction du nombre d'échantillons d'apprentissage présents dans le plan d'expériences au cours de l'amélioration globale de ces modèles.

prédictions de ces points (en résolvant le problème (8.5)). Quand cet échantillon améliorant sera évalué, il viendra ainsi préciser la prédiction des deux points de plus forte variance et par conséquent améliorer $\hat{f}_1$ et $\hat{f}_2$ à la fois. Pour l'instant, son image par les fonctions de référence n'est pas connue et on ne peut donc que simuler son ajout à la base d'apprentissage des modèles afin qu'il soit pris en compte lors de la recherche des échantillons améliorants suivants. Ces échantillons sont déterminés exactement de la même manière, le calcul de la variance des modèles tenant compte à chaque fois des échantillons sélectionnés précédemment afin d'éviter de choisir deux points trop proches. Lorsque dix échantillons améliorants ont été désignés, ils sont évalués avec la simulation numérique et les modèles sont reconstruits. On démarre alors de nouvelles itérations d'enrichissement jusqu'à ce que les modèles aient atteint une qualité suffisante. Pour rechercher les points de variance maximale des modèles ainsi que les échantillons améliorants, nous avons utilisé ici l'algorithme NSGA-II classique avec une population de 30 individus sur 500 générations (n'importe quel autre algorithme d'optimisation mono-objectif aurait pu être employé sous réserve de considérer la présence de nombreux minima locaux dans les problèmes à résoudre).

La figure 9.4 présente l'évolution de la taille maximale des intervalles de confiance des prédictions des modèles $\hat{f}_1$ et $\hat{f}_2$ en 100 points d'observation (choisis au hasard dans l'espace de définition des modèles) au cours des itérations d'amélioration. Nous rappelons qu'il s'agit ici d'estimations de l'erreur des modèles de krigeage à partir des intervalles de confiance qu'ils fournissent autour de leurs prédictions, ce qui explique une certaine variabilité des résultats (la courbe remonte par exemple légèrement au niveau de 50 échantillons pour $\hat{f}_2$). On peut voir que l'erreur du premier modèle diminue visiblement avec l'ajout de nouveaux points au plan d'expériences, alors que l'amélioration est moins significative sur le second modèle. Nous avons arrêté l'algorithme adaptatif dès que l'erreur de $\hat{f}_1$ a commencé à stagner, car nous ne souhaitions pas évaluer trop de points durant cette étape préliminaire. Le plan d'expériences final est ainsi constitué de 60 échantillons d'apprentissage. Cette amélioration globale des modèles a pris moins d'une semaine (soit l'équivalent de l'évaluation successive de trois échantillons avec la simulation numérique) grâce aux calculs réalisés en parallèle.

La figure 9.5 illustre les changements engendrés sur la modélisation des objectifs par l'ajout des nouveaux échantillons d'apprentissage. Les valeurs prédites par les modèles

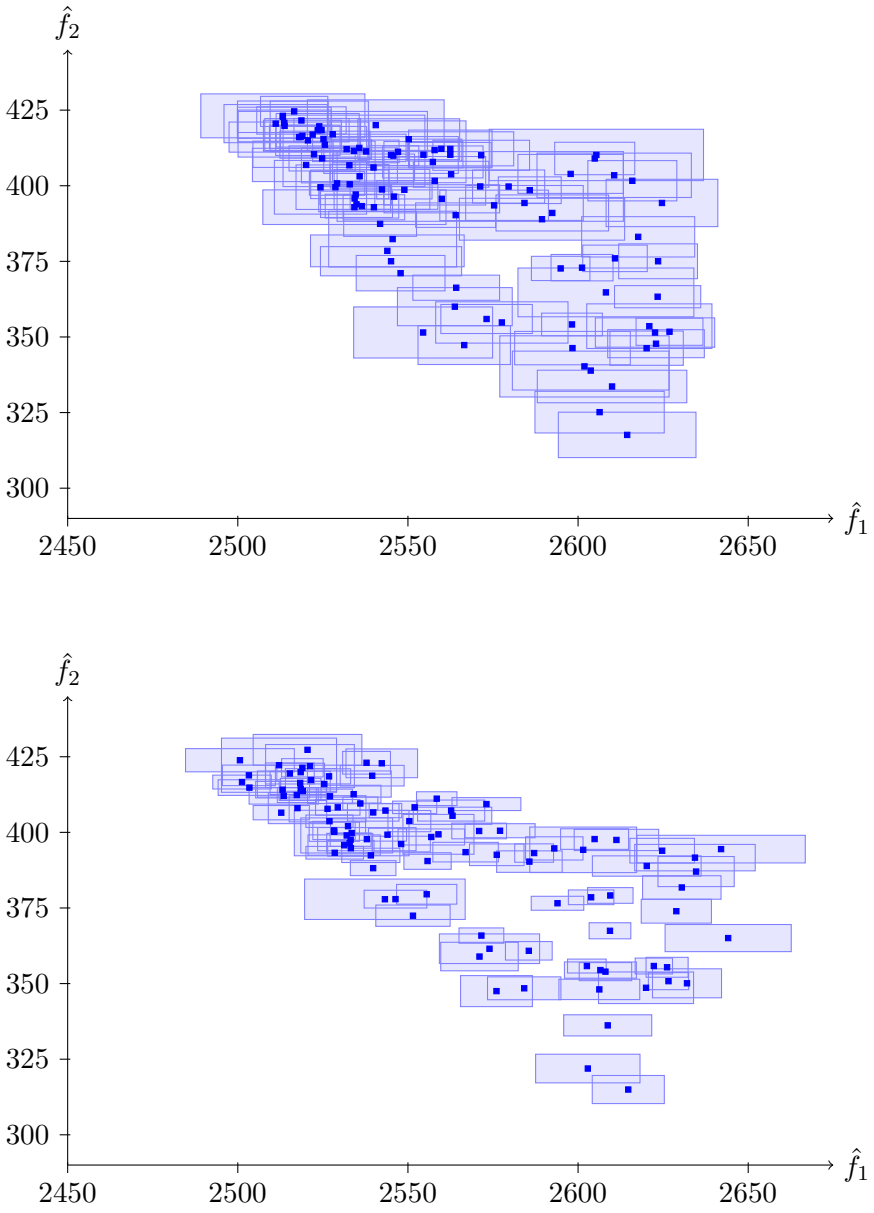


FIGURE 9.5 – Valeurs prédites (avec leur région de confiance à 95% en bleu) en 100 points d'observation par les modèles initiaux basés sur 30 échantillons d'apprentissage (en haut) et par les modèles obtenus après amélioration (60 échantillons d'apprentissage, en bas).

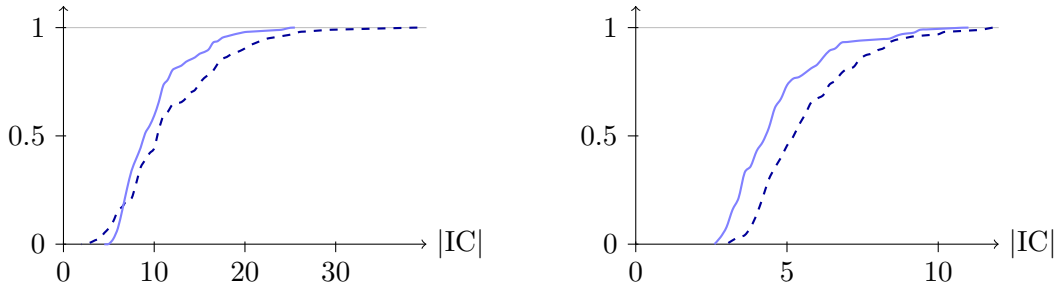


FIGURE 9.6 – Fonctions de répartition de la taille des intervalles de confiance en 100 points d'observation prédits par les modèles $\hat{f}_1$ (à gauche) et $\hat{f}_2$ (à droite). Les courbes en pointillés représentent les résultats obtenus avec les modèles initiaux (30 échantillons d'apprentissage) et les traits pleins ceux des modèles après amélioration (60 échantillons d'apprentissage).

initiaux au niveau des 100 points d'observation y sont comparées à celles des modèles obtenus après amélioration. On remarque que les régions de confiance ont globalement rétréci comme on s'y attendait. La figure 9.6 indique la répartition de la taille des intervalles de confiance dans les deux cas de figure (avant et après amélioration des modèles). Même si effectivement l'erreur maximale de $\hat{f}_2$ n'a pas beaucoup évolué au cours de l'enrichissement, on voit que les intervalles de confiance de ses prédictions ont tout de même diminué. Le second objectif semble d'ailleurs mieux prédit que le premier car ses intervalles de confiance sont relativement plus fins par rapport à l'étendue de sa plage de variation.

Cette première étape d'amélioration adaptative des modèles de substitution dans leur ensemble a permis d'établir une bonne base de représentation des fonctions de référence du problème stochastique (9.1). Nous allons maintenant formuler le problème d'optimisation robuste correspondant.

9.3 Formulation du problème d'optimisation robuste

Afin de pouvoir résoudre le problème stochastique (9.1), nous allons le transformer en problème d'optimisation déterministe robuste au moyen de mesures de robustesse. Ces mesures permettent de quantifier les incertitudes présentes sur les objectifs du fait des fluctuations du paramètre χ_{e_1}. Le choix des mesures à utiliser se fait en premier lieu en lien avec l'application concernée. Dans notre cas, il est souhaitable que les émissions de polluants restent en dessous d'un seuil donné quel que soit l'état d'obstruction des injecteurs. Le fait que la température des gaz en sortie de la chambre demeure aussi basse et homogène que possible malgré les incertitudes évitera de plus d'endommager la turbine située juste derrière. La mesure de robustesse qui semble adaptée à notre problème est donc une mesure de quantile. On pourra ainsi considérer les quantiles à 95% de chacun des deux objectifs (le pire cas n'a pas d'intérêt ici car la distribution de l'incertitude a un support infini). Mais le choix d'une mesure de robustesse doit aussi être effectué en fonction des ses possibilités de calcul, afin que l'optimisation se déroule en un temps raisonnable tout en bénéficiant d'estimations suffisamment précises. Nous avons ainsi réalisé des tests d'évaluation de robustesse selon différentes méthodes sur les modèles de substitution construits à l'aide du plan d'expériences de 60 échantillons obtenu précédemment.

9.3.1 Comparaison des méthodes de calcul de la robustesse

Nous avons tout d'abord comparé les différentes approches d'estimation de la robustesse présentées dans la section 5.8 : calcul analytique, polynômes de chaos et méthode de Monte-Carlo. Les mesures de quantile ne pouvant être estimées qu'à partir de tirages de Monte-Carlo, nous avons utilisé pour ces essais un quantile approché $\hat{\mathrm{Q}}_{0.95}$ qui correspond à une agrégation des mesures d'espérance et de variance (voir l'équation (7.3)) :

$$\hat{\mathrm{Q}}_{0.95}(f_i(\boldsymbol{d}, \chi_e)) \simeq \mathrm{E}[f_i(\boldsymbol{d}, \chi_e)] + 1.64\sqrt{\mathrm{Var}(f_i(\boldsymbol{d}, \chi_e))}. \tag{9.2}$$

Cette mesure peut être considérée comme une approximation du quantile classique sous l'hypothèse que la distribution de l'incertitude sur les objectifs est gaussienne. Elle a surtout l'avantage de pouvoir être estimée avec les trois méthodes qui nous intéressent. Nous les avons comparées sur l'évaluation de la robustesse de 100 points de test à l'aide du modèle $\hat{f}_1$ du premier objectif. La valeur analytique du quantile approché est calculée directement à partir des coefficients de ce modèle. Un polynôme de chaos d'ordre 4 est ensuite construit le long de la dimension incertaine au niveau de chaque point de test grâce à 10 échantillons d'apprentissage évalués sur $\hat{f}_1$ (l'ordre optimal du polynôme a été déterminé par une étude préalable). Enfin, la robustesse est estimée par la méthode de Monte-Carlo à l'aide de 1000 puis 10000 tirages sur $\hat{f}_1$. L'erreur commise ainsi que le temps de calcul de chaque approche sont indiqués dans le tableau 9.2. Les erreurs sont évaluées en prenant comme référence le calcul analytique (qui est exact par définition).

On constate que chacune de ces trois méthodes d'estimation de la robustesse est relativement précise, mais que par contre leurs temps de calcul diffèrent. L'approche la plus efficace pour l'application qui nous concerne est le chaos polynomial, qui bénéficie du fait qu'il n'y a qu'une seule dimension incertaine dans le problème. On peut donc approcher convenablement $\hat{f}_1$ sur cette dimension par un modèle polynomial d'ordre restreint, qui nécessite peu d'échantillons pour sa construction et dont les calculs de l'espérance et de la variance sont très rapides. L'estimation par la méthode de Monte-Carlo est la plus coûteuse. Son coût peut être réduit en jouant sur le nombre de tirages réalisés, mais on perd alors en précision (car la qualité des résultats obtenus dépend directement de ce nombre de tirages). D'autres techniques d'échantillonnage avancé ont aussi été développées afin de limiter la quantité de points utilisés, mais nous ne nous y sommes pas intéressés ici (voir la section 5.8.1). L'approche analytique est quant à elle assez efficace, son atout majeur étant son erreur nulle. Son temps de calcul est par contre sensible au nombre d'échantillons d'apprentissage utilisés pour construire le modèle de krigeage. Dans le cas de la mesure du quantile approché, la complexité du calcul analytique est par exemple relative au nombre d'échantillons à la puissance quatre. Il faut en tenir compte car le nombre d'échantillons du plan d'expériences évolue au cours de l'enrichissement des modèles.

Pour visualiser l'influence de ce paramètre, nous avons réalisé a posteriori les mêmes calculs de robustesse sur un modèle construit avec les 200 échantillons d'apprentissage obtenus à la fin de cette étude. Les résultats en termes de temps de calcul sont présentés dans la partie basse du tableau 9.2. On voit clairement que la durée des calculs analytiques augmente de manière importante avec la complexité du modèle, cette approche devenant même plus coûteuse que la méthode de Monte-Carlo basée sur un faible nombre de tirages. Le temps de calcul des autres approches est lui aussi modifié (mais de manière moins conséquente) car l'évaluation d'un point sur le modèle de krigeage est plus longue lorsqu'il y a plus d'échantillons d'apprentissage.

	Calcul analytique	**Chaos poly.** ordre 4, 10 pts	**Monte-Carlo** 1000 pts	10000 pts
KRIGEAGE 60 ÉCH.				
Erreur sur ρ (moyenne sur 100 éval.)	0.0%	0.00012%	0.012%	0.0044%
Temps de calcul (pour 1 évaluation)	0.30 s	0.0049 s	0.47 s	4.05 s
Durée d'une optim. (estim. pour 10000 éval.)	50 min	49 s	1h19	11h15
KRIGEAGE 200 ÉCH.				
Durée d'une optim. (estim. pour 10000 éval.)	9h12	2 min	3h17	29h13

TABLE 9.2 – Test du calcul de la mesure du quantile approché par différentes approches sur un modèle de krigeage doté de 60 puis 200 échantillons d'apprentissage. La robustesse a été évaluée de manière analytique, puis en utilisant un polynôme de chaos d'ordre 4 construit à partir de 10 points d'apprentissage, et enfin par la méthode de Monte-Carlo en effectuant 1000 et 10000 tirages.

	Espérance		**Variance**		**Quantile appr.**	
	Anal.	M.-C.	Anal.	M.-C.	Anal.	M.-C.
KRIGEAGE 60 ÉCH.						
Erreur sur ρ (moyenne sur 100 éval.)	0.0%	0.012%	0.0%	64.11%	0.0%	0.12%
Erreur sur $\lvert IC(\rho)\rvert$ (moyenne sur 100 éval.)	0.0%	2.70%	0.0%	7.04%	0.0%	7.99%
Temps de calcul (pour 1 évaluation)	0.012 s	144.70 s	816.58 s	131.11 s	940.73 s	124.89 s
Durée d'une optim. (estim. pour 10000 éval.)	2 min	17 j	95 j	15 j	109 j	14 j

TABLE 9.3 – Test du calcul des mesures d'espérance, de variance et de quantile approché avec leur intervalle de confiance sur un modèle de krigeage doté de 60 échantillons d'apprentissage. Les mesures ont été évaluées de manière analytique puis par la méthode de Monte-Carlo en effectuant 500 tirages sur 500 réalisations du processus gaussien défini par le modèle.

Ces tests permettent de conclure que l'approche par polynômes de chaos serait la plus adaptée pour réaliser une optimisation robuste avec la mesure du quantile approché sur notre cas d'application. Si on souhaite utiliser la mesure du quantile classique calculée par Monte-Carlo, l'optimisation prendra quelques heures (avec un nombre de tirages restreint). Mais il s'agit là d'optimisations qui ne prennent pas en considération l'erreur des modèles de substitution sur lesquels elles s'appuient. Pour s'assurer de la qualité des résultats obtenus, il est nécessaire de pouvoir estimer des intervalles de confiance autour des valeurs de robustesse prédites.

9.3.2 Comparaison des méthodes d'estimation d'erreur sur la robustesse

Les possibilités d'estimation de l'erreur commise sur la robustesse sont une autre composante du choix d'une mesure. L'intégration de l'erreur de modélisation aux calculs de robustesse permettra en effet de réaliser par la suite une optimisation robuste adaptative en améliorant les modèles dans les zones d'intérêt. Le calcul d'intervalles de confiance pour les prédictions de robustesse reste néanmoins relativement complexe. Nous avons comparé ici les deux approches décrites dans la section 5.8.5 (calcul analytique et méthode de Monte-Carlo) sur des estimations de robustesse et d'intervalles de confiance en 100 points de test. Ces estimations sont basées sur le modèle de krigeage $\hat{f}_1$ construit avec 60 échantillons d'apprentissage. Elles ont d'abord été effectuées de manière analytique à partir des coefficients du modèle, puis par la méthode de Monte-Carlo en simulant 500 réalisations du processus gaussien $\hat{F}_1$ associé à $\hat{f}_1$ (voir l'exemple de la figure 3.7) et en estimant la robustesse à l'aide de 500 tirages de Monte-Carlo sur chacune de ces réalisations. On produit ainsi 500 évaluations différentes de la robustesse d'un point, dont la distribution est représentative de la variabilité induite par le modèle de krigeage. Les résultats obtenus pour les mesures d'espérance, de variance et de quantile approché sont présentés dans le tableau 9.3. Nous précisons que l'erreur importante affichée sur la prédiction de la variance par Monte-Carlo est due au fait que dans notre exemple les valeurs prises par cette mesure sont moins élevées que celles des autres mesures (la variance varie ainsi entre 0 et 300 alors que l'espérance et le quantile approché évoluent entre 2500 et 2700 environ). Les écarts sur la variance sont donc plus marqués dans le calcul de l'erreur relative, même s'il est vrai que cette mesure reste plus difficile à estimer que les autres. L'erreur sur l'évaluation du quantile approché est quant à elle plus importante qu'auparavant car le nombre de tirages utilisés est différent.

Si la robustesse est globalement bien évaluée (hormis pour la mesure de variance), on peut noter que l'erreur sur l'estimation des intervalles de confiance par la méthode de Monte-Carlo n'est pas négligeable. Il aurait fallu augmenter le nombre de tirages effectués pour obtenir des résultats plus précis. Mais les temps de calcul sont déjà très élevés : cette approche demande ainsi une quinzaine de jours environ pour réaliser les évaluations de robustesse nécessaires à une optimisation (quelle que soit la mesure considérée). Les durées sont encore plus importantes dans le cas du calcul analytique, mis à part pour l'espérance dont les calculs sont beaucoup plus légers. Nous avons en effet pu constater dans un exemple du chapitre précédent qu'une optimisation adaptative utilisant la mesure du quantile approché pouvait être relativement coûteuse (voir la figure 8.27). Une troisième solution qui n'a pas été envisagée lors de cette étude aurait été de modifier l'approche par Monte-Carlo en calculant la robustesse à l'aide de polynômes de chaos sur les réalisations du processus gaussien $\hat{F}_1$. Les polynômes de chaos permettent en effet d'évaluer la ro-

bustesse de manière plus rapide que des tirages de Monte-Carlo (voir le tableau 9.2). Les intervalles de confiance obtenus ne prendraient toutefois pas en compte l'erreur commise lors de l'approximation locale du modèle de krigeage par un polynôme. Cette stratégie nécessite en effet de construire un modèle de substitution à partir de données fournies par un autre modèle, ce qui complexifie la gestion des erreurs. Le temps de calcul d'une telle approche resterait tout de même relativement important. Si on l'estime à partir de la durée d'une évaluation du quantile approché par chaos polynomial (sachant qu'elle serait similaire pour les autres mesures), en la multipliant par les 500 réalisations du processus gaussien $\hat{F}_1$ sur lesquelles elle doit être effectuée (sans tenir compte du temps nécessaire à la création de ces réalisations) et par les 10000 évaluations requises lors d'une optimisation standard, on arrive à 6h48 de calcul environ. Ce résultat est relativement intéressant pour la mesure du quantile approché par rapport aux autres méthodes, mais il représente tout de même une durée conséquente sachant qu'une optimisation de ce type va être réalisée à chaque itération de l'algorithme adaptatif (le nombre d'évaluations nécessaires étant aussi multiplié par le nombre de modèles qui entrent en jeu).

Au vu des résultats obtenus lors de ces tests, il apparaît que l'utilisation des mesures de quantile approché ou de quantile classique (évalué par Monte-Carlo) engendre des temps de calcul élevés dès lors qu'on souhaite prendre en compte l'erreur des modèles de substitution. Nous avons donc décidé de réaliser ici l'optimisation adaptative sur l'espérance des objectifs initiaux seulement. Le calcul analytique de cette mesure offre en effet des durées bien plus raisonnables. Même si son temps de calcul va évoluer avec l'augmentation du nombre d'échantillons (la complexité de l'estimation de l'espérance avec son intervalle de confiance étant liée au carré du nombre d'échantillons d'apprentissage du modèle de krigeage), celui-ci restera négligeable à côté de la durée d'une simulation numérique du système et en tout cas inférieur à la durée d'un calcul par une approche de type Monte-Carlo. Le problème que nous allons résoudre de manière adaptative comprend donc deux objectifs d'espérance à minimiser :

$$\underset{\boldsymbol{d}}{\text{minimiser}} \ \{\rho_1(f_1(\boldsymbol{d}, \chi_{e_1})), \rho_2(f_2(\boldsymbol{d}, \chi_{e_1}))\}, \tag{9.3}$$

avec :

$$\rho_i(f_i(\boldsymbol{d}, \chi_{e_1})) = \mathrm{E}[f_i(\boldsymbol{d}, \chi_{e_1})].$$

Nous allons ainsi optimiser dans un premier temps les performances moyennes du système. La variance et le quantile des solutions obtenues ne seront considérés qu'a posteriori en vue de sélectionner les points les plus intéressants.

9.4 Optimisation robuste adaptative à l'aide de la méthode PareBRO

Pour résoudre le problème (9.3) que nous venons de formuler, nous disposons d'un plan d'expériences de départ composé de 60 échantillons. Les modèles de krigeage $\hat{f}_1$ et $\hat{f}_2$ construits grâce à ces échantillons permettent d'estimer de manière analytique la valeur des objectifs ρ_1 et ρ_2 ainsi que les intervalles de confiance correspondants. Nous avons réalisé une première résolution du problème à l'aide de ces modèles afin de déterminer la bande de Pareto robuste initiale. Celle-ci est représentée sur la figure 9.7. Elle a été obtenue à l'aide de l'algorithme NSGA-II modifié pour prendre en compte les intervalles de confiance des objectifs (voir la section 8.2.2), avec une population de 20 individus sur 500 générations. On remarque que la bande de Pareto est pour l'instant très grossière du

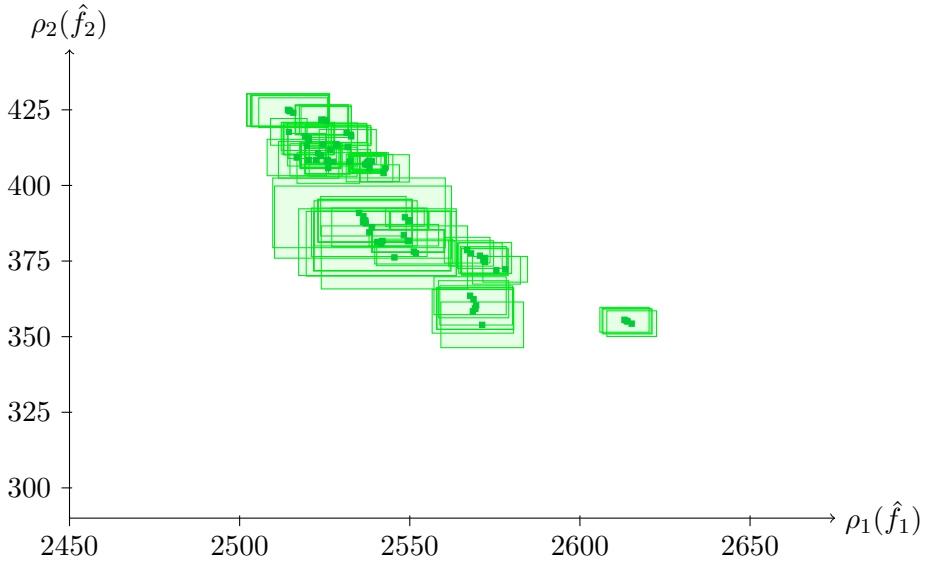


FIGURE 9.7 – Bande de Pareto robuste initiale obtenue grâce aux modèles $\hat{f}_1$ et $\hat{f}_2$ construits avec le plan d'expériences de 60 échantillons.

fait de l'erreur des modèles. Il est en effet difficile de distinguer des solutions réellement optimales. Nous allons donc améliorer la prédiction des objectifs en mettant en place une stratégie d'enrichissement adaptatif.

Nous avons utilisé pour cela la méthode PareBRO présentée dans la section 8.3 (voir l'algorithme 8.7). Cette approche vise à déterminer les zones possiblement optimales de l'espace afin d'affiner leur description par les modèles de substitution en ajoutant au plan d'expériences des échantillons judicieusement choisis. Le déroulement de cette méthode dans le cadre de notre application est présenté sur la figure 9.8. Les solutions de la bande de Pareto que nous venons de déterminer grâce aux modèles de départ représentent l'ensemble des régions possiblement optimales qu'on va chercher à améliorer. Comme nous avons la capacité d'évaluer dix échantillons en parallèle avec la simulation numérique du système, nous séparons ces solutions en dix classes S_i à l'aide de l'algorithme des k-moyennes. Un échantillon améliorant les modèles est ensuite recherché sur chacune de ces classes. La recherche est effectuée en deux temps comme lors de l'amélioration globale des modèles (la seule différence est qu'on chercher ici à améliorer les solutions de la classe considérée uniquement). On commence tout d'abord par déterminer les points de la classe qui possèdent le plus grand intervalle de confiance sur chaque objectif de robustesse ρ_i, puis on recherche ensuite dans l'espace $\mathbf{\Omega}$ tout entier l'échantillon qui permettra d'améliorer les prédictions de ces points (en résolvant le problème 8.7). Nous avons utilisé pour cela l'algorithme NSGA-II classique avec une population de 10 individus sur 250 générations. Lorsque l'échantillon améliorant est trouvé, on simule son ajout aux modèles afin qu'il soit pris en compte lors de l'étude de la classe suivante. Une fois que les dix échantillons améliorants correspondant aux dix classes de solutions de la bande de Pareto ont été ainsi sélectionnés, ils sont évalués avec la simulation numérique du système et les modèles sont reconstruits. On peut alors déterminer la nouvelle bande de Pareto de la même manière que pour la bande initiale. Les itérations d'enrichissement continuent ainsi jusqu'à ce que la bande de Pareto obtenue soit suffisamment fine.

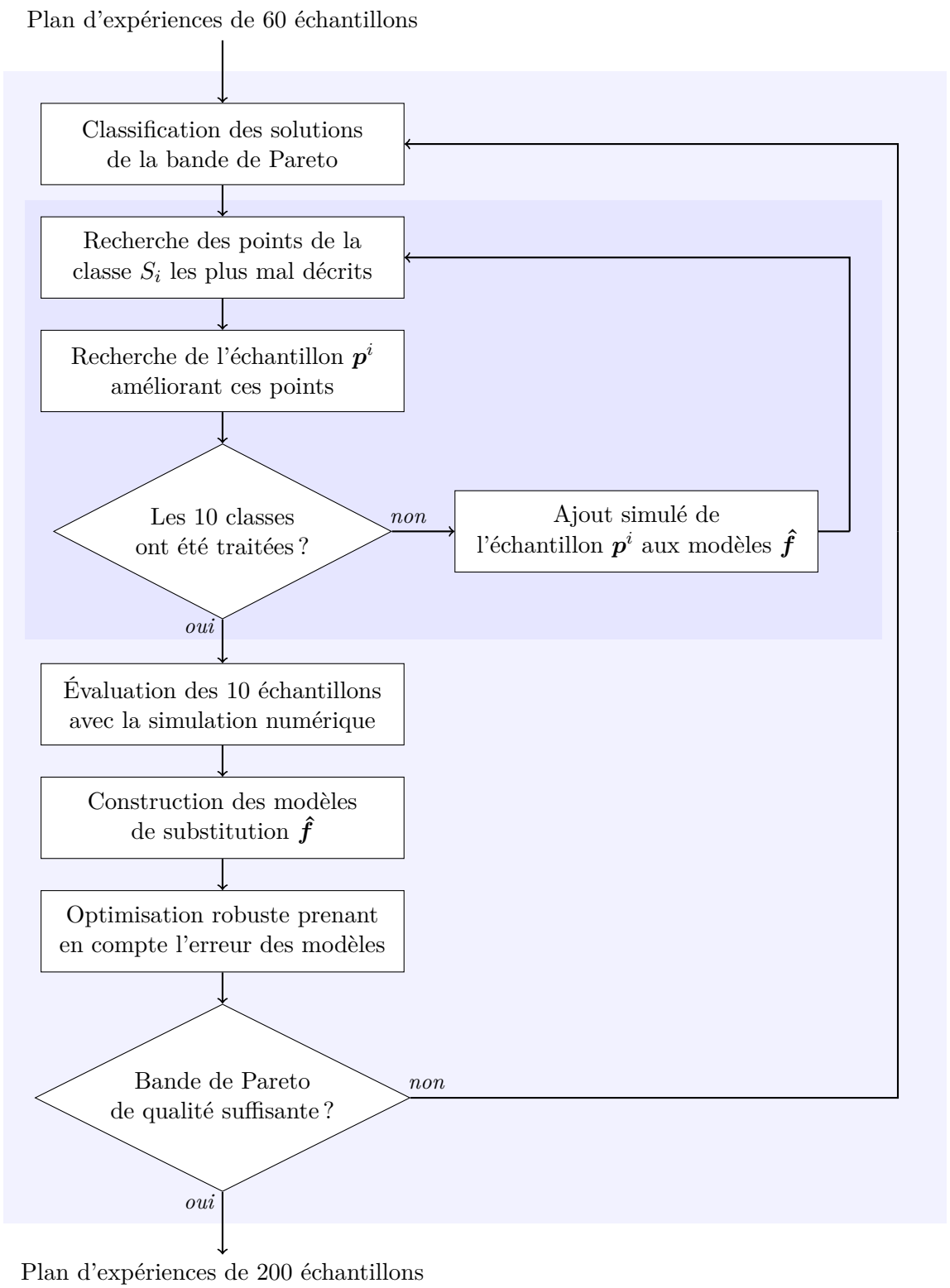


FIGURE 9.8 – Méthode PareBRO d'optimisation robuste adaptative utilisée pour notre cas d'application, où il est possible d'évaluer 10 échantillons en parallèle.

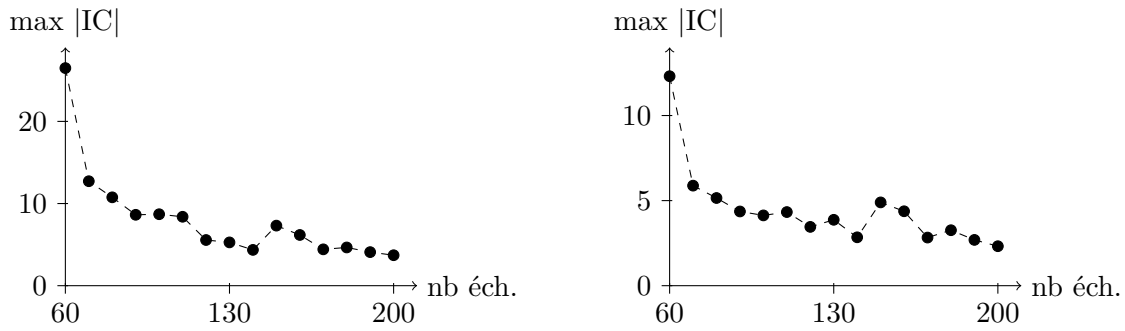


FIGURE 9.9 – Évolution de la taille maximale des intervalles de confiance sur $\rho_1(\hat{f}_1)$ (à gauche) et $\rho_2(\hat{f}_2)$ (à droite) des solutions de la bande de Pareto en fonction du nombre d'échantillons présents dans le plan d'expériences au cours du déroulement de la méthode PareBRO.

La figure 9.9 présente l'évolution de la taille maximale des intervalles de confiance de la robustesse des solutions de la bande de Pareto au cours de l'amélioration adaptative. On remarque une forte baisse de l'étendue de ces intervalles dès la première itération, puis elle continue par la suite à diminuer à un rythme de moins en moins soutenu. Les itérations ont été arrêtées lorsque cette erreur commençait à stagner sur les deux objectifs. Nous avons ainsi ajouté 140 nouveaux points au plan d'expériences, pour obtenir au final un ensemble de 200 échantillons d'apprentissage. On voit que l'enrichissement aurait aussi pu être stoppé lorsque le plan d'expériences était composé de 140 échantillons, car la taille des intervalles de confiance n'évolue plus beaucoup au delà de cette limite.

Pour se rendre compte de l'amélioration apportée, la figure 9.10 compare les valeurs de robustesse prédites en 100 points d'observation pris au hasard calculées avec les modèles de départ dotés de 60 échantillons d'apprentissage et les modèles finaux avec 200 échantillons d'apprentissage. Nous précisons que ces points d'observation sont différents des points précédents visualisés sur la figure 9.5 car on ne se trouve plus dans le même espace. L'espace de recherche pour les objectifs de robustesse est ici de dimension 3 (pour les trois variables de décision uniquement), alors qu'auparavant on se situait dans l'espace $\mathbf{\Omega}$ de dimension 4. Cette figure montre que l'incertitude sur les mesures de robustesse a été notablement réduite dans les zones optimales grâce à l'enrichissement des modèles de substitution. Les régions moins intéressantes ont quant à elles conservé des intervalles de confiance plus larges. La méthode PareBRO a donc effectivement permis d'améliorer les zones d'intérêt du problème.

Le fait de réaliser cette optimisation adaptative avec des mesures d'espérance a permis de limiter la durée des calculs visant à trouver les échantillons améliorants. La figure 9.11 présente l'évolution de cette durée en fonction du nombre d'échantillons présents dans le plan d'expériences. On retrouve expérimentalement le comportement quadratique dû au calcul des intervalles de confiance de la mesure d'espérance. L'augmentation du temps de calcul au cours des itérations d'amélioration est beaucoup moins importante que dans le cas du quantile approché dont la complexité est quartique (voir la figure 8.27). Le nombre d'échantillons qu'il est possible d'évaluer avec des simulations coûteuses comme celle utilisée ici reste généralement limité, et les durées obtenues sont donc tout à fait raisonnables en regard de celle d'une simulation numérique.

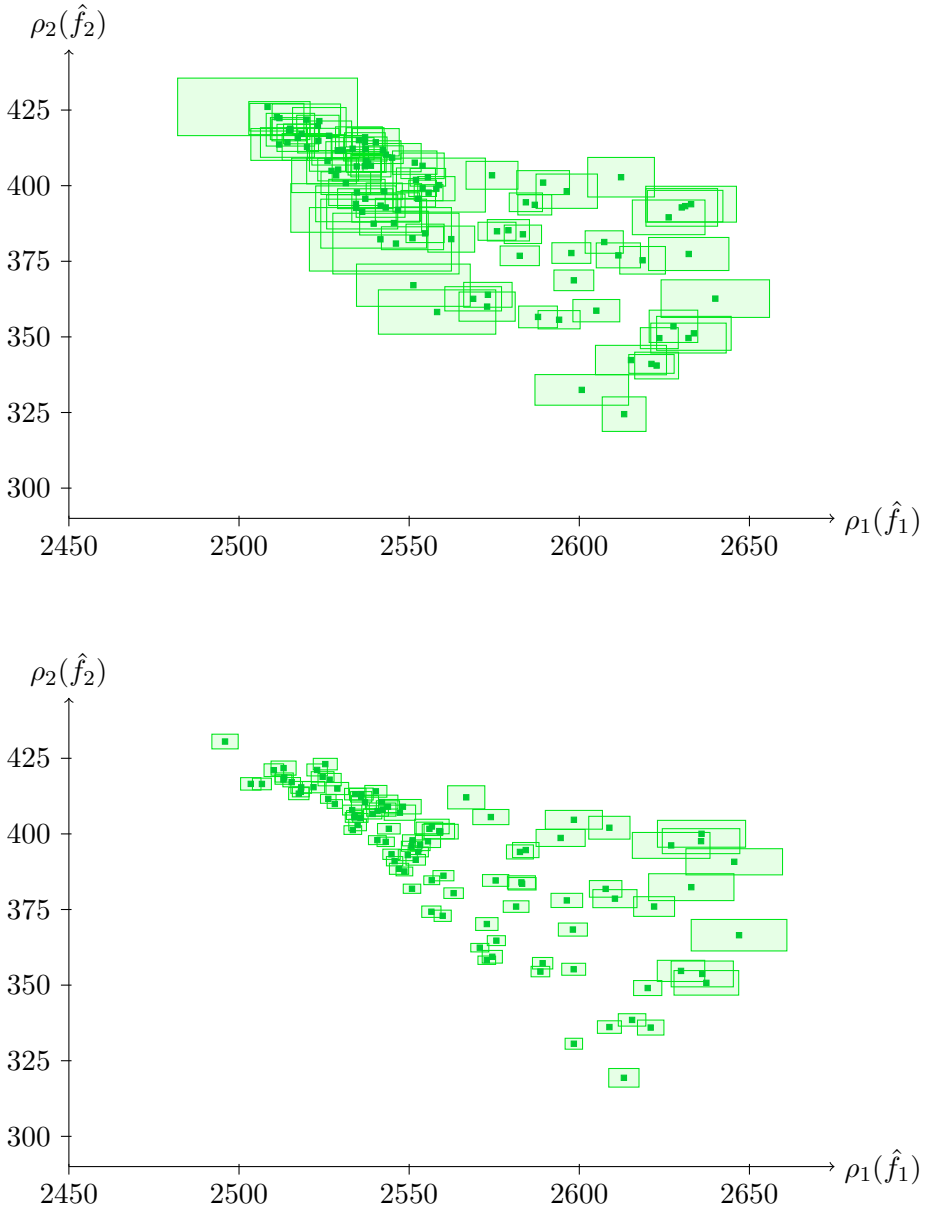


FIGURE 9.10 – Valeurs de robustesse prédites (avec leur région de confiance à 95% en vert) en 100 points d'observation grâce aux modèles de départ basés sur 60 échantillons d'apprentissage (en haut) et par les modèles obtenus après optimisation adaptative avec la méthode PareBRO (200 échantillons d'apprentissage, en bas).

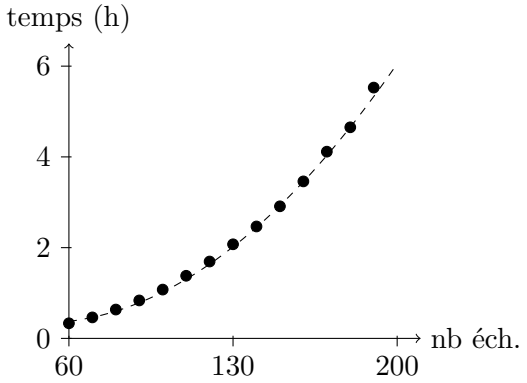


FIGURE 9.11 – Évolution du temps de calcul (en heures) pour la recherche des points améliorants de la méthode PareBRO en fonction du nombre d'échantillons d'apprentissage déjà présents dans le plan d'expériences.

9.5 Détermination des solutions optimales robustes

La construction adaptative du plan d'expériences que nous venons de mener a permis de mettre en avant les zones optimales du problème (9.3). Nous allons maintenant étudier les solutions robustes obtenues avant d'en sélectionner quelques-unes pour une analyse plus approfondie.

9.5.1 Étude des solutions obtenues grâce à l'enrichissement adaptatif

Afin de trouver l'ensemble des solutions optimales du problème (9.3), nous avons lancé une optimisation finale à l'issue des itérations d'enrichissement de la méthode PareBRO. Cette optimisation a été menée à l'aide des modèles construits avec le plan d'expériences de 200 échantillons d'apprentissage et grâce à l'algorithme NSGA-II modifié doté d'une population de 100 individus sur 1000 générations (pour bien converger vers les solutions optimales). La bande de Pareto robuste obtenue est présentée sur la figure 9.12. On constate que les solutions sont maintenant très bien décrites. Seuls quelques points non Pareto-optimaux au sens classique du terme sont présents du fait des légères erreurs de modélisation qui subsistent encore. Notre approche d'amélioration adaptative a donc permis de déterminer les solutions optimales du problème en prenant en compte l'erreur des modèles de substitution utilisés.

En recalculant la valeur de ces solutions dans l'espace des objectifs initiaux f_1 et f_2 à l'aide de leur modèle de substitution respectif (voir le graphe du bas de la figure 9.12), nous avons par contre remarqué que la valeur de la performance déterministe est tout à fait comparable à celle de son espérance. Cela est certainement dû à un comportement symétrique de f_1 et f_2 de part et d'autre de la valeur nominale de la variable incertaine χ_{e_1}. Si ce fait avait été détecté auparavant, nous aurions donc pu réaliser une optimisation classique sur les objectifs initiaux et obtenir les mêmes résultats. Pour nous en convaincre, nous avons effectué cette optimisation déterministe sur les modèles de krigeage finaux, toujours à l'aide de l'algorithme NSGA-II modifié avec 100 individus sur 1000 générations. La bande de Pareto déterministe obtenue est représentée sur la figure 9.13. On retrouve comme prévu des points similaires aux solutions robustes précédentes.

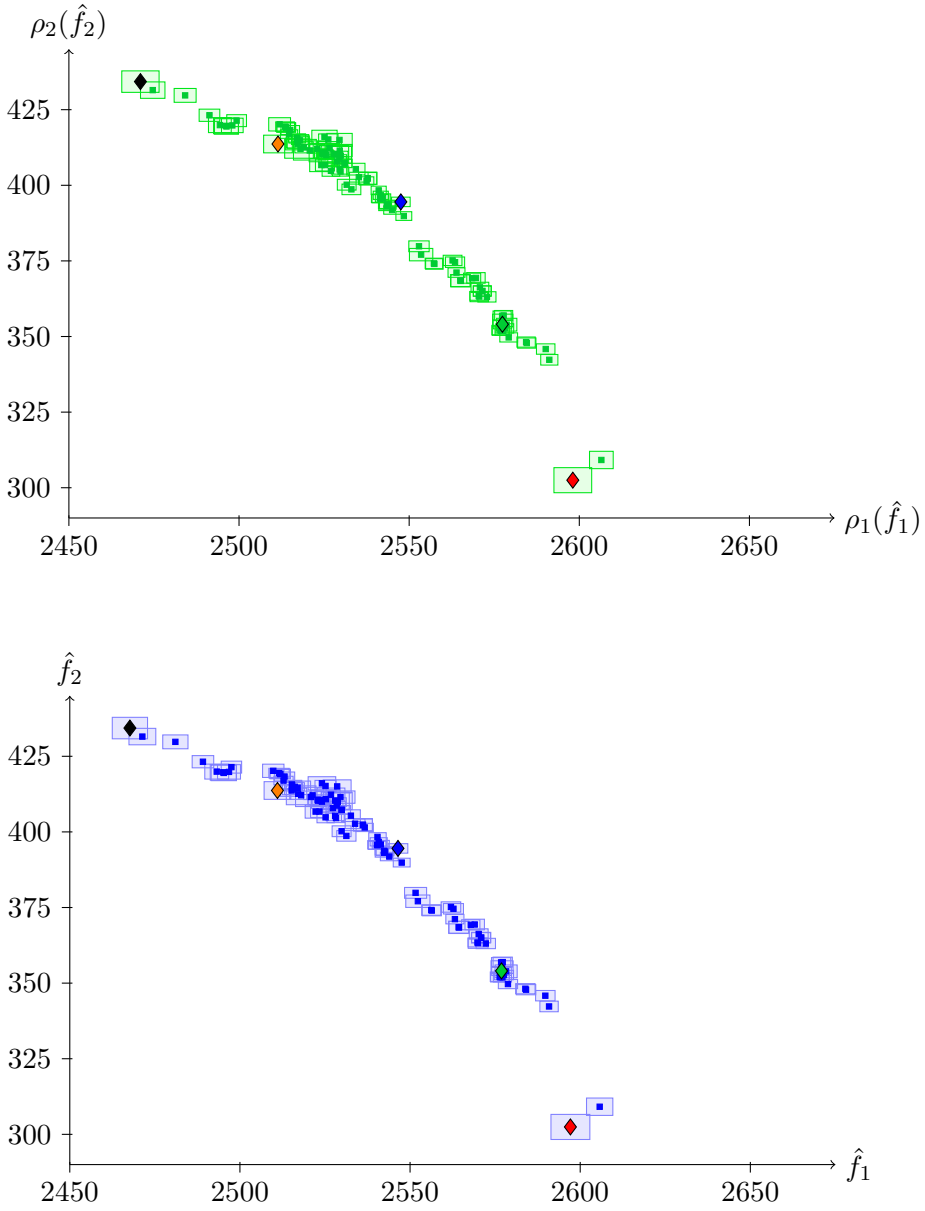


FIGURE 9.12 – Bande de Pareto robuste finale obtenu après amélioration des modèles (plan d'expériences de 200 échantillons, en haut), et valeur de ces points dans l'espace des objectifs initiaux (en bas). Les losanges représentent les solutions optimales qui seront sélectionnées par la suite.

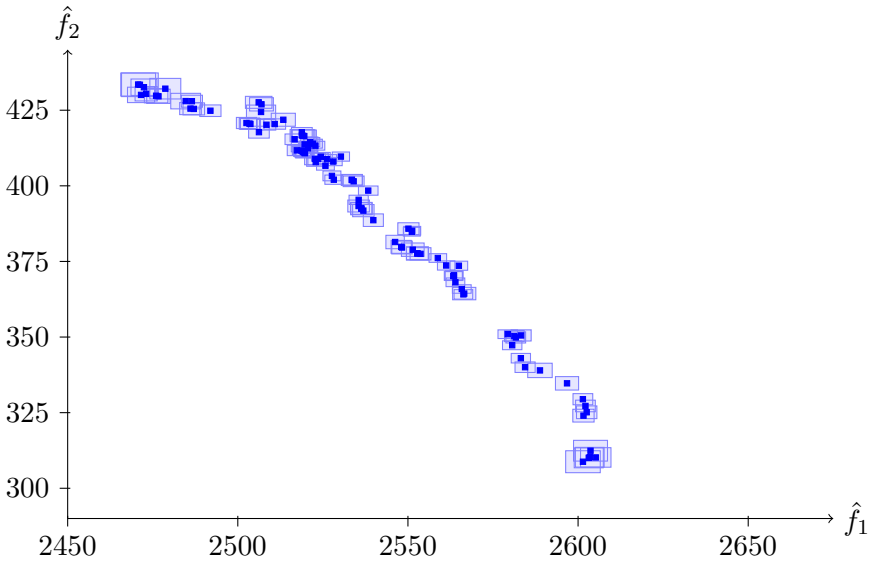


FIGURE 9.13 – Bande de Pareto déterministe classique obtenue à partir des modèles de substitution finaux (200 échantillons d'apprentissage).

L'optimisation robuste du système d'injection n'a donc pas mené à des solutions différentes des optima déterministes classiques, mais elle a tout de même eu l'intérêt de montrer que ces solutions sont effectivement robustes en termes de performance moyenne. Nous allons maintenant étudier leur variance et leur quantile pour en sélectionner quelques-unes parmi les plus intéressantes.

9.5.2 Sélection de quelques solutions intéressantes

Au delà d'une bonne performance moyenne, nous souhaitions en effet que les solutions du problème stochastique (9.1) soient aussi robustes du point de vue du quantile à 95% des objectifs initiaux. Avant d'estimer ce quantile, nous nous sommes tout d'abord penchés sur la variance des solutions de la bande de Pareto. Celle-ci a été évaluée de manière analytique à partir des modèles de krigeage finaux. Les valeurs obtenues sont présentées sur la figure 9.14. On observe que si les solutions optimales sont Pareto-équivalentes au regard de leur espérance, elles se distinguent par contre en ce qui concerne leur variance. La différence se fait surtout sur la variance de f_1 car celle du second objectif est plutôt faible (pour ne pas dire quasi nulle). Nous avons ainsi accès à un choix de solutions dont la température maximale T_{max} à l'intérieur de la chambre est plus ou moins sensible au niveau d'obstruction des injecteurs.

La connaissance de ces variances nous a ensuite permis d'évaluer les quantiles approchés des solutions de la bande de Pareto. Il sont présentés sur la figure 9.15. Le front obtenu est relativement similaire à celui de l'espérance (figure 9.12), mais les solutions sont néanmoins décalées en fonction de leur variance le long de la dimension correspondant au premier objectif. Par exemple, les solutions notées D et E se retrouvent ainsi très proches sur l'axe du quantile de f_1 alors que leurs espérances étaient beaucoup plus espacées. On peut vérifier que le point E possède une variance très faible alors que celle de D est plus élevée, ce qui explique les valeurs de quantiles analogues. On constate sur cette figure que toutes les

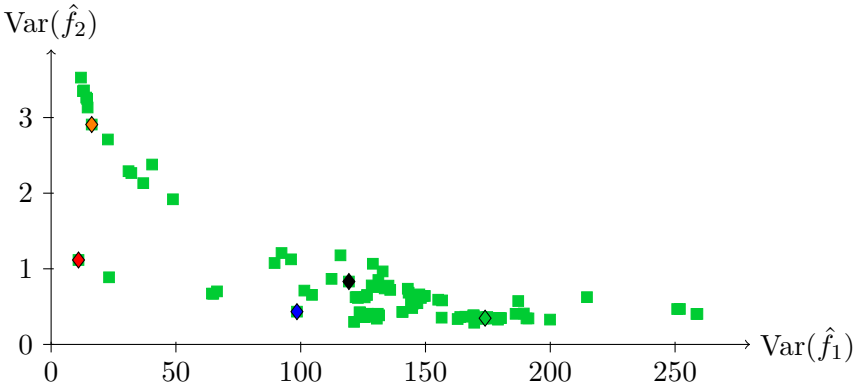


FIGURE 9.14 – Variance des solutions de la bande de Pareto robuste de la figure 9.12. Les losanges représentent les solutions optimales qui seront sélectionnées par la suite.

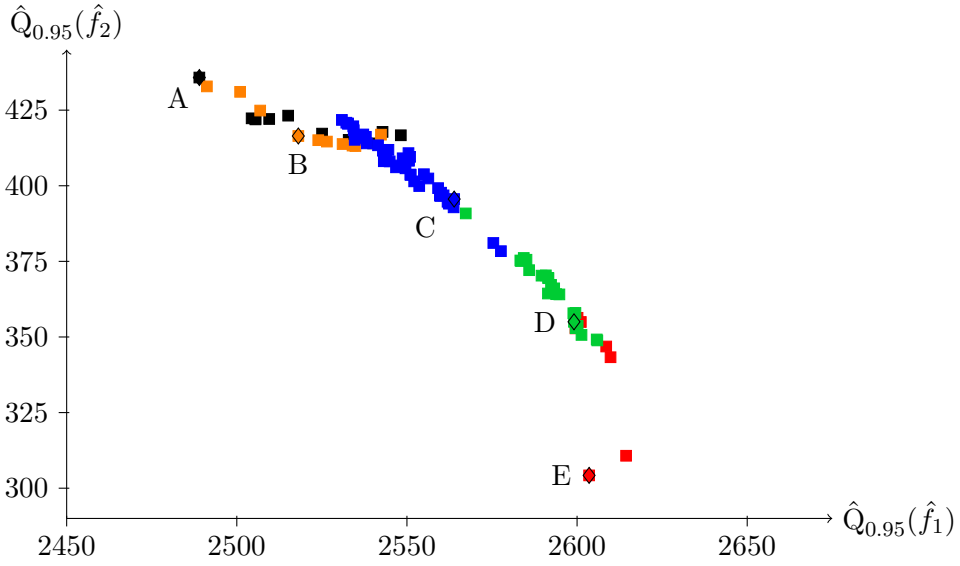


FIGURE 9.15 – Quantile approché des solutions de la bande de Pareto robuste de la figure 9.12. Les couleurs différencient les cinq classes constituées. Les solutions sélectionnées sont représentées par des losanges et identifiées par une lettre.

		d_1	d_2	d_3	f_1	f_2	$Q_{0.95}(f_1)$	$Q_{0.95}(f_2)$
A	noir	30.05	0.01010	0.0248	2471.8	432.5	2483.2	437.1
B	orange	46.96	0.00357	0.0249	2515.8	415.3	2519.1	416.8
C	bleu	38.32	0.00472	0.0078	2539.8	399.6	2560.0	401.3
D	vert	46.44	0.00792	0.0038	2578.3	354.0	2599.1	356.1
E	rouge	60.00	0.00460	0.0018	2594.3	304.2	2599.7	304.6

TABLE 9.4 – Valeurs des variables de décision, des objectifs déterministes ainsi que des quantiles à 95% pour les cinq solutions sélectionnées.

solutions de la bande de Pareto restent relativement intéressantes en termes de quantiles (d'autant plus qu'on n'a pas ici accès aux régions de confiance correspondantes du fait de leur coût de calcul élevé). La prise en compte de la variance n'a donc pas modifié en profondeur l'ensemble des solutions à considérer, mais elle nous a servi toutefois à mettre en évidence des différences entre ces optima.

Pour faire ressortir quelques solutions représentatives de la bande de Pareto, nous avons réalisé une classification sur les valeurs des variables de décision $\boldsymbol{d}$ et celles des deux quantiles approchés $\hat{Q}_{0.95}$ à l'aide de l'algorithme des k-moyennes. Le nombre de classes a été fixé à 5 pour obtenir un nombre raisonnable de solutions à présenter au décideur final. Les classes ainsi trouvées sont représentées en couleur sur la figure 9.15. Une solution caractéristique a ensuite été sélectionnée dans chacune d'elle. Le détail des cinq solutions choisies est donné dans le tableau 9.4, et elles sont indiquées par des losanges sur les figures précédentes. Ces solutions optimales décrivent l'ensemble des valeurs prises par les quantiles des deux objectifs initiaux. On peut aussi remarquer qu'aucune règle simple ne se dessine au niveau des variables de décision : leurs valeurs sont dispersées sur leur domaine de variation. Seule la position de l'injecteur principal d_3 semble être liée aux objectifs car ils varient de concert. Ces cinq solutions robustes seront donc exposées au choix du décideur final, mais nous allons auparavant les analyser de manière plus approfondie.

9.6 Analyse des solutions optimales obtenues

L'optimisation réalisée nous a permis de distinguer cinq solutions robustes pour notre application. Avant de comparer les qualités de ces optima, nous procédons à des évaluations complémentaires avec la simulation numérique du système afin de valider leur robustesse.

9.6.1 Validation par Monte-Carlo

La robustesse des solutions sélectionnées ayant été déterminée à partir de modèles de substitution (aussi précis soient-ils), il peut être intéressant de la recalculer a posteriori sur les fonctions de référence. Cette étape relativement onéreuse n'est bien sûr pas obligatoire, mais elle permet de valider les résultats obtenus et de donner au décideur des informations plus précises sur les solutions qu'on lui fournit. Dans le cadre d'une optimisation déterministe classique, la valeur exacte de la performance d'un point optimal peut être obtenue par une simple évaluation des fonctions de référence. Mais pour une optimisation robuste le problème est différent. L'estimation de la robustesse d'une solution n'est en effet jamais parfaite et nécessite une bonne connaissance de l'ensemble de son voisinage

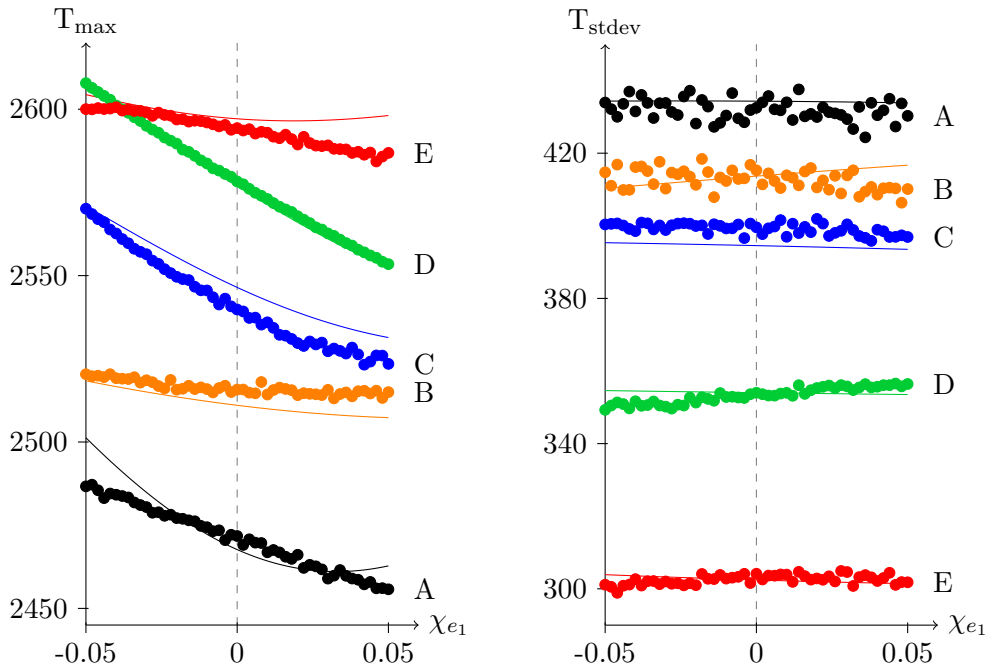


FIGURE 9.16 – Valeur des échantillons (points) évalués le long de la dimension incertaine au niveau des cinq solutions optimales sélectionnées. La ligne en pointillés indique la valeur nominale du paramètre incertain χ_{e_1}, et les traits pleins sont les prédictions des modèles de krigeage finaux $\hat{f}_1$ et $\hat{f}_2$ pour chacune de ces solutions.

le long des dimensions incertaines. On peut donc évaluer de nouveaux points au sein de ce voisinage afin d'en avoir une description plus précise. Il est par ailleurs probable que certains échantillons du plan d'expériences construit de manière adaptative en fassent déjà partie, et ils peuvent alors être pris en compte lors de l'ajout des points supplémentaires.

Afin de valider les valeurs de robustesse prédites par les modèles de substitution, nous avons ici estimé la robustesse « réelle » des cinq solutions choisies précédemment à l'aide de tirages de Monte-Carlo sur la simulation numérique du système. 51 échantillons ont ainsi été évalués pour chacune de ces solutions en faisant varier le paramètre incertain χ_{e_1} entre -0.05 et 0.05. Les points obtenus sont présentés sur la figure 9.16. On remarque tout de suite que le comportement des objectifs initiaux est effectivement globalement symétrique autour de la valeur nominale de χ_{e_1} (à part peut-être pour la solution C), ce qui confirme les résultats donnés par la mesure d'espérance. Malgré le bruit engendré par les simulations autour de certaines solutions, la variance du second objectif est aussi très faible comme nous avions pu le constater. Les solutions ont par contre des variances différentes sur f_1. Les modèles de krigeage avaient donc bien cerné les caractéristiques des fonctions de référence du problème.

La figure 9.16 indique par ailleurs les prédictions des modèles $\hat{f}_1$ et $\hat{f}_2$ au niveau des solutions sélectionnées. On observe que ces modèles sont assez fidèles à la réalité même si certaines imperfections sont visibles, notamment pour le premier objectif qui est plus difficile à approcher. Les modèles régressants n'ont eu aucun mal à s'affranchir du bruit existant sur f_2. La figure 9.17 compare enfin les valeurs estimées par les modèles pour les quantiles approchés à 95% des solutions choisies par rapport aux valeurs de quantiles

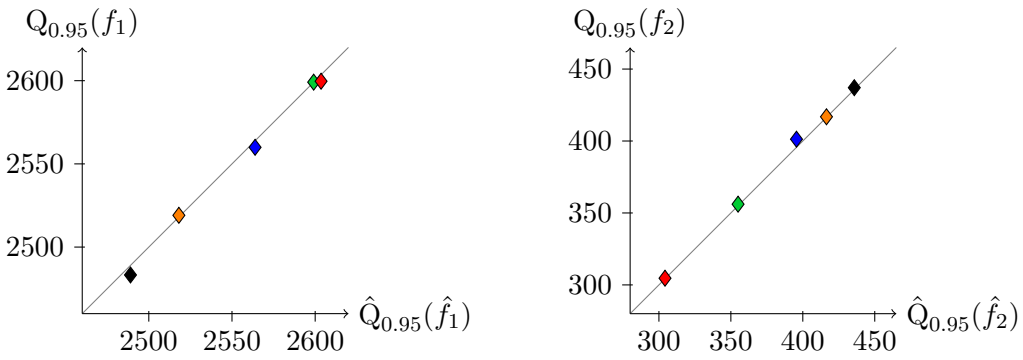


FIGURE 9.17 – Comparaison des valeurs de quantiles $\hat{Q}_{0.95}(\hat{f}_i)$ prédites par les modèles et $Q_{0.95}(f_i)$ calculées par Monte-Carlo pour le premier objectif initial (à gauche) et le second (à droite) au niveau des cinq solutions sélectionnées (losanges).

classiques calculées grâce aux échantillons de Monte-Carlo que nous venons d'évaluer. On peut voir que les quantiles sont relativement bien prédits malgré les erreurs de modèles qui persistent. Les simulations numériques complémentaires effectuées permettent donc de valider les résultats obtenus lors de l'optimisation robuste.

9.6.2 Comparaison des solutions

Après avoir vérifié la robustesse des solutions que nous avons sélectionnées, nous allons finalement comparer les configurations qu'elles décrivent en visualisant les résultats de leurs simulations numériques.

Nous avons tout d'abord observé l'impact des variations du paramètre incertain χ_{e_1} dans la configuration centrale du front de Pareto (point C). La figure 9.18 présente la répartition de la température des gaz dans la chambre de combustion pour les valeurs extrêmes de χ_{e_1}. On voit que l'obstruction des injecteurs a un effet conséquent sur le déroulement de la combustion. Dans le cas où $\chi_{e_1} = 0.05$, une plus grande quantité de kérosène est pulvérisée par l'injecteur pilote situé sur l'axe de la turbomachine. L'évaporation de ce carburant refroidit la zone à proximité de l'injecteur, et la combustion a lieu plus en aval avec une température moindre. Lorsque $\chi_{e_1} = -0.05$, la zone d'allumage se rapproche de l'injecteur pilote et les températures atteintes sont plus élevées. Le carburant supplémentaire issu de l'injecteur principal vient alors alimenter l'extérieur de la flamme. Cela peut engendrer des émissions de polluants ou bien des imbrûlés en sortie car on voit que certaines gouttes parviennent jusqu'au milieu de la chambre avant de s'évaporer. La combustion de ce kérosène va éventuellement se faire de manière incomplète, et il est aussi possible que quelques gouttes atteignent directement la sortie de la chambre (ce qui est bien entendu à éviter). Les imbrûlés ne sont pas simulés ici, et ils ne pouvaient donc pas être pris en compte durant l'optimisation. Malgré l'impact de χ_{e_1} sur la combustion à l'intérieur de la chambre, la distribution de la température en sortie reste quant à elle relativement similaire dans les deux cas. Il était donc intéressant de prendre en compte l'incertitude sur la répartition du carburant entre les deux injecteurs afin d'éviter d'atteindre des températures inattendues dans la chambre. Les températures élevées entraînent en effet une production accrue de gaz polluants et peuvent aussi détériorer des éléments de la turbomachine si elles ne sont pas suffisamment anticipées.

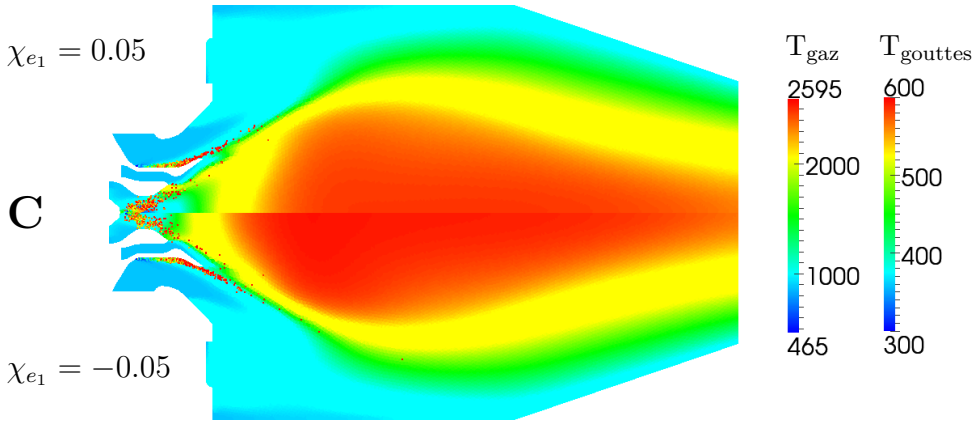


FIGURE 9.18 – Comparaison des iso-valeurs de la température des gaz (T_{gaz}) et des gouttes de carburant ($T_{gouttes}$) à l'intérieur de la chambre de combustion pour des variations extrêmes du paramètre incertain χ_{e_1} autour de la solution C.

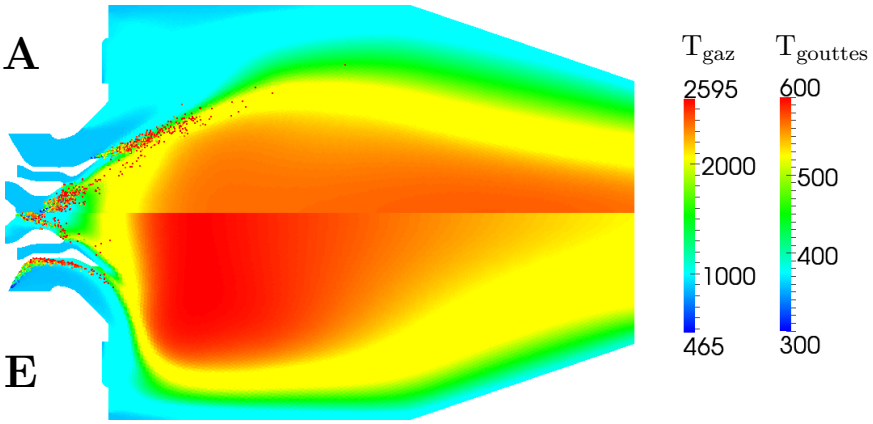


FIGURE 9.19 – Comparaison des iso-valeurs de la température des gaz (T_{gaz}) et des gouttes de carburant ($T_{gouttes}$) à l'intérieur de la chambre de combustion pour les deux solutions extrêmes du front de Pareto robuste (solution A sur la moitié haute, et solution E sur la moitié basse).

Nous avons ensuite visualisé les résultats de simulation pour les solutions extrêmes du front de Pareto (les points A et E). Ils sont présentés sur la figure 9.19. On voit bien sur cette figure les différences de température maximale ainsi que de distribution de la température en sortie qui existent entre ces deux solutions. Pour la configuration A, l'injection est réalisée très en aval dans la chambre. Cela permet d'obtenir des températures de combustion moins élevées, mais implique aussi une distribution de température peu homogène en sortie. Cette solution possède comme le point C un angle de vrille d_1 relativement faible, et peut donc de la même manière que précédemment produire des imbrûlés si l'air entrant ne permet pas d'évaporer correctement l'ensemble du carburant injecté. La solution E définit au contraire une très large zone transversale de gaz à haute température dans laquelle se déroule la combustion. Les gaz brûlés à cet endroit ont ensuite le temps de se refroidir avant d'atteindre l'extrémité de la chambre. On obtient ainsi un faible écart-type en sortie, mais les températures à l'intérieur de la chambre sont plus élevées. Dans cette configuration, on peut aussi noter que l'injecteur principal est placé assez haut dans sa veine d'air. Il est alors possible que le carburant ruisselle le long de la paroi, ce qui viendrait affecter les performances du système. Le ruissellement n'est pas visible ici car il n'y a pas de modélisation des films liquides dans la simulation numérique utilisée. Il faudra donc y prêter attention si cette solution est retenue.

Le choix final d'une solution optimale robuste unique est relativement complexe car il faut prendre en compte de nombreux paramètres complémentaires qui ne sont pas nécessairement exprimés (ou exprimables) dans le problème d'optimisation initial. C'est pourquoi nous avons conservé ici les cinq solutions possibles que nous venons d'analyser. Un expert du domaine pourra déterminer par la suite la configuration du système d'injection la plus convenable en toute connaissance de cause.

9.7 Conclusion

La résolution de ce problème d'optimisation sous incertitude d'un système d'injection nous a permis de mettre en œuvre sur une application industrielle l'approche adaptative que nous avons développée. La méthode PareBRO a ainsi conduit à un ensemble de solutions robustes dont les qualités ont ensuite été vérifiées à l'aide de la simulation numérique du système. Ces configurations pourraient dans un second temps être simulées de manière plus précise avec une modélisation 3D qui gère la production des polluants afin de valider le fait que les solutions trouvées permettent effectivement d'obtenir des chambres de combustion plus propres.

Nous avons aussi pu expérimenter lors du traitement de cette application les limites de notre approche. La prise en compte des erreurs sur les estimations de robustesse entraîne en effet des temps de calcul relativement importants. Cela nous a donc obligés à réaliser l'enrichissement adaptatif avec des mesures d'espérances alors que des quantiles auraient été plus adaptés au problème. Afin de réduire le coût de notre méthode, il pourrait être envisagé de simplifier l'estimation des intervalles de confiance sur la robustesse. Ceux-ci pourraient par exemple être fixés a priori au début de chaque itération à partir de quelques évaluations réelles, ou bien définis de manière empirique en fonction de l'erreur « classique » des modèles aux points considérés. Ce genre d'approximations permettrait de réaliser des optimisations adaptatives tout en limitant le nombre d'estimations coûteuses d'erreur sur la robustesse. Il faudrait tout de même pouvoir valider ces approximations afin que les résultats obtenus avec l'approche PareBRO gardent leur sens.

Chapitre 10

Optimisation robuste mutidisciplinaire d'un design d'avion

Sommaire

10.1 Bref rappel du problème 234
10.2 Prise en compte simplifiée des incertitudes (méthode LOUP) 236
10.2.1 Problèmes multidisciplinaires concernés 236
10.2.2 Traitement local des incertitudes 237
10.2.3 Formulation du problème d'optimisation robuste 239
10.3 Recherche de designs d'avion robustes avec la méthode LOUP 240
10.4 Expression d'un critère d'applicabilité de la méthode 242
10.4.1 Évaluation de corrélations a posteriori 243
10.4.2 Vérification de l'applicabilité pour le cas-test de conception avion 246
10.5 Analyse des solutions optimales obtenues 247
10.6 Conclusion 249

Nous nous sommes intéressés dans les chapitres précédents à des problèmes d'optimisation robuste multiobjectifs faisant intervenir des fonctions de type « boîte noire » dont aucune caractéristique n'est connue. Dans le cas où le système à optimiser est de taille importante, il peut s'avérer judicieux de le considérer sous la forme d'une « boîte grise », c'est-à-dire d'une entité contenant plusieurs sous-systèmes interconnectés (voir l'exemple de la figure 10.1). On se situe alors dans le contexte plus général de l'optimisation multidisciplinaire (voir le chapitre 6) auquel est liée la seconde application de cette étude. Elle concerne en effet l'optimisation d'un design conceptuel d'avion modélisé par un ensemble de disciplines en interaction. Le but recherché est de maximiser les performances de l'appareil sur une mission déterminée, tout en prenant en compte une certaine incertitude sur l'estimation de la surface de son empennage vertical. Nous allons ainsi étudier maintenant la résolution d'un problème d'optimisation robuste multidisciplinaire.

Après un bref rappel de l'application à traiter, nous développons dans ce chapitre une méthode heuristique pour tenir compte des incertitudes dans une optimisation multidis-

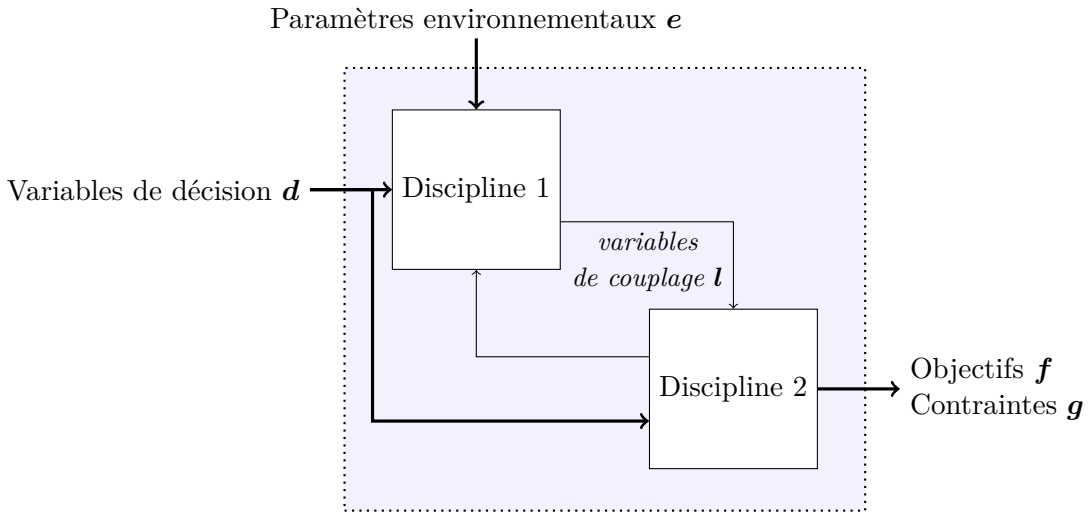


FIGURE 10.1 – Exemple simple d'un système multidisciplinaire incluant deux disciplines.

ciplinaire. Pour résoudre un problème d'optimisation robuste multidisciplinaire, on met habituellement en place une propagation d'incertitudes au sein du système. La propagation d'incertitudes est un procédé coûteux qui demande des modifications importantes au niveau de chaque discipline ainsi que des variables de couplage. Mais des solutions robustes peuvent aussi être obtenues plus facilement lorsque les disciplines affectées par les incertitudes jouent un rôle important sur les objectifs du problème. Nous introduisons ainsi le principe de la méthode LOUP (pour « LOcal Uncertainty Processing ») basée sur une gestion locale de l'incertitude au sein des disciplines qu'elle affecte directement, sans effectuer de propagation dans tout le système. Cette méthode permet de résoudre de manière approchée certains problèmes d'optimisation multidisciplinaire robuste avec des changements mineurs sur le système initial. Nous la mettons ici en œuvre sur le cas de conception avion qui nous intéresse. Dans un deuxième temps, nous proposons un critère pour vérifier a posteriori l'applicabilité de la méthode LOUP à un problème donné. Ce critère permet de valider les résultats obtenus avec cette approche sur notre cas-test. Nous analysons enfin à l'aide de nouvelles évaluations du système les optima identifiés, de manière à confirmer leur robustesse.

10.1 Bref rappel du problème

Le problème de l'optimisation d'un design d'avion lors de la phase d'étude conceptuelle a été abordé dans le chapitre 2. Les variables, objectifs et contraintes qui interviennent dans ce problème sont rappelés dans le tableau 10.1. Il s'agit de dimensionner un appareil à l'aide de 10 paramètres de conception $\boldsymbol{d}$ en vue de minimiser sa masse à vide équipée f_1 ainsi que la masse de carburant f_2 à emporter pour réaliser une mission moyen-courrier donnée. Les performances de l'avion sont évaluées à l'aide d'une simulation qui regroupe un ensemble de codes de calcul modélisant les différentes disciplines impliquées (structure, aérodynamique, etc.) au sein d'un logiciel d'optimisation multidisciplinaire. Le problème initial est ainsi résolu par un algorithme génétique avec une approche globale MDF qui permet d'assurer la convergence vers les solutions optimales.

Variables de décision			
d_1	Allongement λ	[5,15]	
d_2	Effilement ε	[0.1,0.5]	
d_3	Flèche φ	[0,30]	m
d_4	Surface de la voilure S	[90,200]	m^2
d_5	Position de la cassure Y_k	[5,10]	m
d_6	Épaisseur relative de la voilure à l'emplanture E_e	[0.12,0.16]	
d_7	Épaisseur relative de la voilure au niveau de la cassure E_k	[0.1,0.14]	
d_8	Épaisseur relative de la voilure à 70% de l'envergure E_r	[0.09,0.12]	
d_9	Altitude de croisière de la mission de référence $H_{\text{réf}}$	[9448,12450]	m
d_{10}	Altitude de croisière de la mission optimale H_{opt}	[9448,12450]	m
Paramètre environnemental			
e_1	Erreur sur la surface de l'empennage vertical	0	m^2
Sortie disciplinaire affectée par e_1			
$D_{1,1}$	Masse de l'empennage vertical W_{empV}		kg
Incertitude			
χ_{e_1}	Erreur sur la surface de l'empennage vertical	$U(-5,5)$	m^2
Objectifs			
f_1	Masse à vide équipée MTOW	minimiser	kg
f_2	Masse de fuel pour la mission optimale W_{fuel}	minimiser	kg
Contraintes			
g_1	Vitesse d'approche V_{app}	< 72	$m.s^{-1}$
g_2	Longueur de décollage $L_{\text{déco}}$	< 2200	m
g_3	Envergure B	< 35.95	m
g_4	Différence entre les cordes à la cassure et à l'extrémité $(C_k - C_t)$	> 0	m
g_5	Différence entre les épaisseurs relatives à l'emplanture et à la cassure $(d_6 - d_7)$	> 0	
g_6	Différence entre les épaisseurs relatives à la cassure et à 70% de l'envergure $(d_7 - d_8)$	> 0	

TABLE 10.1 – Variables, objectifs et contraintes du problème d'optimisation multidisciplinaire du design d'un avion.

On souhaite désormais considérer dans ce problème une incertitude sur l'estimation de la surface de l'empennage vertical de l'appareil. L'erreur commise lors de cette estimation est modélisée par une variable aléatoire χ_{e_1} dotée d'une loi de probabilité uniforme. Cette incertitude affecte directement le calcul de la masse de l'empennage, et par là même l'estimation de la masse totale et de la consommation de l'avion. Le problème multidisciplinaire stochastique auquel nous sommes confrontés est donc le suivant :

$$\begin{aligned} &\underset{\boldsymbol{d}}{\text{minimiser}}\ \{f_1(\boldsymbol{d}, \chi_{e_1}), f_2(\boldsymbol{d}, \chi_{e_1})\}, \\ &\text{s.c.}\ \ \boldsymbol{g}(\boldsymbol{d}, \chi_{e_1}) \leq \mathbf{0}. \end{aligned} \tag{10.1}$$

10.2 Prise en compte simplifiée des incertitudes (méthode LOUP)

La résolution d'un problème d'optimisation robuste issu du problème (10.1) nécessite de réaliser une propagation d'incertitudes dans le système, afin de déterminer la distribution des incertitudes engendrées par les fluctuations de la surface de l'empennage vertical sur les objectifs et les contraintes. Comme expliqué dans la section 6.4.1, cette propagation ne peut se faire qu'au prix de nombreuses modifications. La figure 10.2 présente ainsi un exemple des éléments du système de la figure 10.1 qu'il faudrait adapter afin de propager des incertitudes provenant des paramètres environnementaux. Si la gestion des incertitudes peut être envisagée lors de la conception d'un nouveau système multidisciplinaire, la question de son utilité se pose quand on a affaire à un système déjà existant. En effet, les disciplines sont parfois des boîtes noires impossibles à modifier (car on ne possède pas les sources de leur code par exemple), ou bien le coût de ces modifications est prohibitif. L'ajout de nouvelles variables de couplage au sein d'une implémentation existante peut être coûteux lui aussi.

C'est pourquoi nous nous sommes intéressés au développement d'une méthode approchée qui permette de faciliter la prise en compte d'incertitudes dans un système multidisciplinaire. Bien sûr, seule une propagation complète des incertitudes à travers le système peut assurer une bonne connaissance des distributions d'incertitude en sortie. Mais nous avons remarqué que pour certains problèmes cette propagation n'était pas forcément nécessaire pour trouver des solutions robustes. Nous proposons donc une méthode heuristique simple à implémenter pour tenir compte des incertitudes dans un système multidisciplinaire existant. Cette méthode nommée LOUP peut être vue comme un premier pas vers l'optimisation robuste multidisciplinaire, avant d'intégrer directement la propagation des incertitudes dans la construction de nouveaux systèmes.

10.2.1 Problèmes multidisciplinaires concernés

Notre méthode de prise en compte simplifiée de l'incertitude ne peut être appliquée qu'à certains problèmes spécifiques d'optimisation robuste multidisciplinaire. Pour pouvoir faciliter la gestion des incertitudes, il faut en effet que les sorties des disciplines affectées par des paramètres incertains aient une influence significative sur les objectifs du problème à traiter. Autrement dit, il faut que les variations des objectifs soient corrélées à celles des disciplines qui possèdent des variables de décision $\boldsymbol{d}$ ou des paramètres environnementaux $\boldsymbol{e}$ incertains comme entrées (nous appellerons celles-ci les disciplines « incertaines »). Sur l'exemple de la figure 10.2, la discipline incertaine est ainsi la Discipline 1 car elle

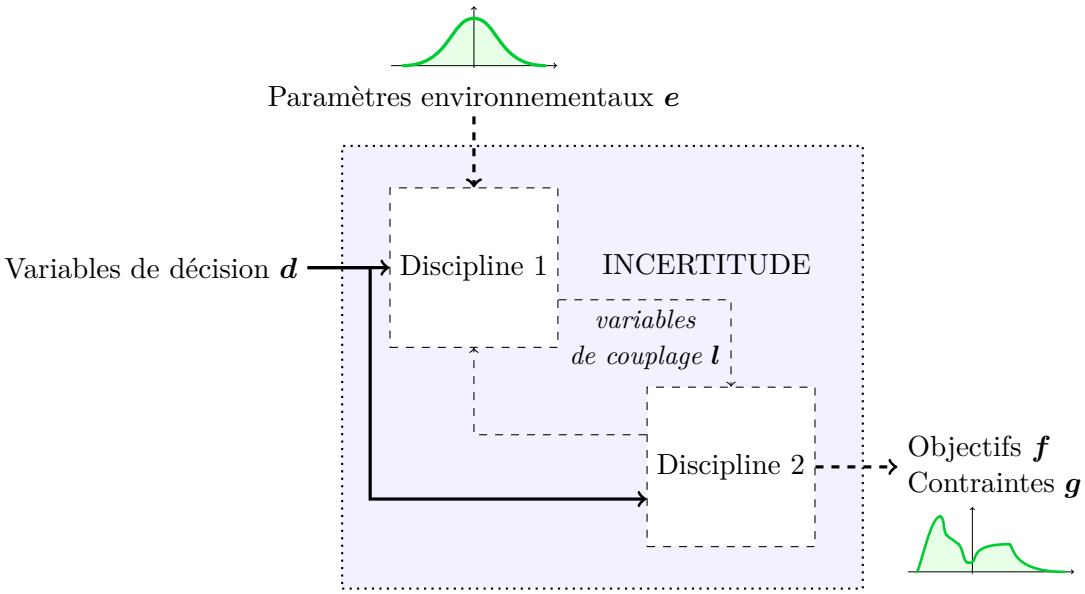


FIGURE 10.2 – Exemple de propagation d'incertitudes dans le système de la figure 10.1. Les incertitudes proviennent des paramètres environnementaux et sont propagées jusqu'aux objectifs et aux contraintes du problème. Les éléments affectés sont représentés en pointillés.

est directement affectée par l'incertitude des paramètres environnementaux, alors que la Discipline 2 est influencée par le biais des variables de couplage uniquement. Les variations de la sortie de la Discipline 1 doivent donc être corrélées aux variations des objectifs $\boldsymbol{f}$. Nous n'aborderons pas ici le cas des contraintes $\boldsymbol{g}$, mais elles pourraient néanmoins être traitées de manière similaire.

Grâce à l'hypothèse que nous venons de poser sur le problème à résoudre, on peut raisonnablement considérer de traiter l'incertitude localement au sein des disciplines incertaines au lieu de la propager dans tout le système. Cela a l'avantage de rendre la gestion des incertitudes beaucoup plus facile comme nous allons le voir. Cette hypothèse sera validée (ou réfutée) par la suite grâce à un critère d'applicabilité de la méthode que nous allons développer.

10.2.2 Traitement local des incertitudes

Le but final d'une optimisation robuste est de trouver des solutions dont la performance ne sera pas affectée par les fluctuations des paramètres incertains. Ces fluctuations pénètrent tout d'abord dans le système au niveau des disciplines incertaines qui sont directement en lien avec les paramètres d'entrée concernés. L'incertitude se propage ensuite vers les autres disciplines via les variables de couplage avant d'atteindre finalement les fonctions objectifs. Les fluctuations dues aux incertitudes interviennent donc dans le système par le biais des sorties des disciplines incertaines. Hors, nous savons que les variations de ces sorties sont ici directement liées aux variations des objectifs du fait de l'hypothèse précédente. On peut en déduire qu'il faut que ces sorties restent aussi stables que possible pour modérer les variations des objectifs. La variance des objectifs peut ainsi être réduite en minimisant la variance des sorties des disciplines incertaines.

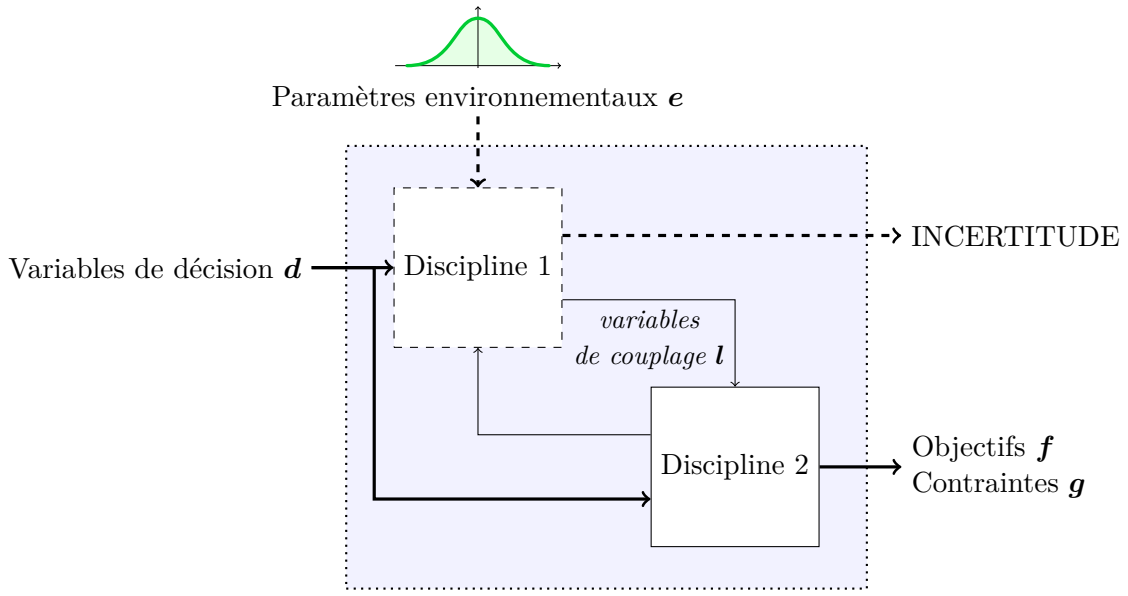


FIGURE 10.3 – Principe de la gestion locale des incertitudes : l'incertitude provenant des paramètres environnementaux est ici traitée par la Discipline 1 uniquement alors que le reste du système demeure inchangé. Les éléments affectés sont représentés en pointillés.

C'est sur ce principe qu'est basée la méthode LOUP (pour « LOcal Uncertainty Processing »). L'idée est d'estimer la robustesse d'une solution de manière locale au sein de chaque discipline affectée par les incertitudes. Ces robustesses locales seront ensuite prises en compte au niveau du problème d'optimisation global. Les solutions dotées des plus petites valeurs de variance des sorties disciplinaires vont être considérées comme plus robustes que les autres, car on sait qu'en ces points les incertitudes n'auront pas un grand impact sur le système. La figure 10.3 illustre cette prise en compte locale de l'incertitude sur l'exemple de la figure 10.1. On voit que les modifications à apporter au système multidisciplinaire initial sont ici marginales. Seules les disciplines incertaines doivent être adaptées afin de pouvoir gérer l'incertitude, et le reste du système demeure inchangé.

Si on note $D_{i,j}$ la j^{e} sortie de la i^{e} discipline du système, la robustesse locale $\rho_{\text{loc}_{D_{i,j}}}$ correspondant à cette sortie est définie comme suit :

$$\rho_{\text{loc}_{D_{i,j}}}(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e) = \text{Var}(D_{i,j}(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e, \dot{\boldsymbol{l}}(\boldsymbol{d},\boldsymbol{e}))), \tag{10.2}$$

où $\boldsymbol{\chi}_d$ et $\boldsymbol{\chi}_e$ représentent respectivement l'incertitude sur les variables de décision et les paramètres environnementaux du problème, et $\dot{\boldsymbol{l}}(\boldsymbol{d},\boldsymbol{e})$ la valeur des variables de couplage $\boldsymbol{l}$ une fois que le système a convergé en prenant les valeurs de $\boldsymbol{d}$ et $\boldsymbol{e}$ comme entrées globales. La robustesse locale de chaque sortie des disciplines incertaines du problème peut être exprimée de la sorte. Dans le cadre de l'exemple de la figure 10.3, on peut ainsi considérer la robustesse locale $\rho_{\text{loc}_{D_{1,1}}}$ de la première et unique sortie de la Discipline 1. $\boldsymbol{l}$ correspond alors à $D_{2,1}$, la première sortie de la Discipline 2 (celle qui est une entrée de la Discipline 1). Il est à noter que le calcul de variance ne prend pas en compte les variations de $D_{2,1}$ induites par les fluctuations des paramètres environnementaux, car cela reviendrait alors à considérer une propagation d'incertitude complète à travers le système.

10.2.3 Formulation du problème d'optimisation robuste

La méthode LOUP vise à réduire la variance des objectifs du problème multidisciplinaire initial. Pour un problème stochastique de la forme :

$$\underset{\boldsymbol{d}}{\text{minimiser}}\ \boldsymbol{f}(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e), \tag{10.3}$$

nous considérons ainsi le problème d'optimisation robuste suivant :

$$\underset{\boldsymbol{d}}{\text{minimiser}}\ \{\boldsymbol{f}(\boldsymbol{d}, \boldsymbol{e}), \boldsymbol{\rho}(\boldsymbol{f}(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e))\}, \tag{10.4}$$

avec :

$$\rho_i(f_i(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e)) = \mathrm{Var}(f_i(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e)).$$

Ce problème définit de nouveaux objectifs de robustesse aux côtés des objectifs déterministes initiaux afin d'obtenir un ensemble de solutions de compromis entre performance et robustesse.

Pour trouver ces solutions, nous allons nous appuyer sur les robustesses locales $\rho_{\mathrm{loc}_{D_{i,j}}}$ des sorties des disciplines incertaines. Leur intégration au processus d'optimisation dépend de la stratégie utilisée. Il existe en effet deux approches générales pour résoudre un problème multidisciplinaire (voir la section 6.2) : la stratégie mononiveau où le système global est optimisé d'un seul tenant, et la stratégie multiniveaux où l'optimisation se déroule aussi localement au niveau disciplinaire.

Dans le cas de l'optimisation mononiveau, les robustesses locales peuvent être prises en compte au niveau global directement. Nous définissons alors un nouvel objectif global ρ_{loc} qui est une agrégation de l'ensemble des robustesses locales du système. Toutes ces robustesses locales doivent être minimisées en même temps afin de réduire la variance des objectifs initiaux. On pose par exemple :

$$\rho_{\mathrm{loc}}(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e) = \sum_{i,j} \gamma_{i,j}\ \rho_{\mathrm{loc}_{Di,j}}(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e), \tag{10.5}$$

où les $\gamma_{i,j} \in \mathbb{R}$ sont des paramètres utilisés pour normaliser les mesures locales avant de les sommer. D'autres agrégations similaires pourraient aussi être envisagées. Le problème (10.4) devient ainsi :

$$\underset{\boldsymbol{d}}{\text{minimiser}}\ \{\boldsymbol{f}(\boldsymbol{d}, \boldsymbol{e}), \rho_{\mathrm{loc}}(\boldsymbol{d}+\boldsymbol{\chi}_d, \boldsymbol{e}+\boldsymbol{\chi}_e)\}. \tag{10.6}$$

On ne rajoute donc qu'un unique objectif de robustesse au problème de départ. Le front de Pareto de ce nouveau problème sera composé des solutions optimales déterministes standards ainsi que d'autres solutions moins performantes mais plus robustes (des solutions avec des variances locales inférieures).

Dans le cas d'une stratégie d'optimisation multiniveaux, plusieurs boucles d'optimisation sont présentes à différents niveaux hiérarchiques du système. Il existe de nombreuses formulations multiniveaux différentes. Dans l'approche bi-niveaux par exemple, chaque discipline possède sa propre boucle d'optimisation locale et un optimiseur global vient ensuite fédérer le tout. Dans ce cadre, les robustesses locales peuvent être intégrées au problème local de la discipline à laquelle elles sont rattachées en vue de favoriser des solutions robustes pour cette discipline, ou bien renvoyées au niveau supérieur pour être agrégées en un objectif global comme dans le cas de la stratégie mononiveau.

Nous nous sommes concentrés ici sur l'approche mononiveau car c'est la stratégie utilisée dans l'application qui nous concerne. Une fois le problème d'optimisation robuste (10.6) ainsi formulé, il peut être résolu avec le même optimiseur que celui employé pour le problème déterministe initial : on a simplement ajouté un nouvel objectif de robustesse au problème. Celui-ci est donc un peu plus complexe à résoudre, mais il reste traitable par les solveurs standards d'optimisation multidisciplinaire. Il suffit simplement de mettre en place le calcul des variances locales $\rho_{\mathrm{loc}_{D_{i,j}}}$ en se basant par exemple sur des modèles de substitution des sorties des disciplines incertaines.

10.3 Recherche de designs d'avion robustes avec la méthode LOUP

Nous allons utiliser la méthode LOUP pour trouver des solutions robustes à notre problème de conception avion (son applicabilité à ce problème sera vérifiée a posteriori). L'incertitude présente sur l'estimation de la surface de l'empennage vertical n'affecte directement que le calcul de la masse de cet empennage. Cette masse, que nous noterons $D_{1,1}$, correspond donc à l'unique sortie disciplinaire incertaine du système. Le problème d'optimisation robuste défini par la méthode LOUP est donc le suivant :

$$\begin{aligned} &\underset{\boldsymbol{d}}{\text{minimiser}}\ \{f_1(\boldsymbol{d}, e_1), f_2(\boldsymbol{d}, e_1), \rho_{\mathrm{loc}_{D_{1,1}}}(\boldsymbol{d}, e_1 + \chi_e)\}, \\ &\text{s.c.}\ \ \boldsymbol{g}(\boldsymbol{d}, e_1) \leq \mathbf{0}. \end{aligned} \tag{10.7}$$

Nous rappelons que nous ne prenons pas ici en compte l'effet des incertitudes sur la fiabilité du système (c'est-à-dire la robustesse vis-à-vis des contraintes).

Pour estimer la variance $\rho_{\mathrm{loc}_{D_{1,1}}}$ de la masse de l'empennage vertical, nous utilisons un modèle de krigeage $\hat{D}_{1,1}$ de cette sortie disciplinaire. Ce modèle possède trois entrées : le paramètre incertain χ_{e_1} et deux variables de couplage $\boldsymbol{l}$ (qui correspondent à la masse maximale au décollage et à la corde de l'empennage). Le modèle de substitution a été construit à partir d'un LHS de 11 échantillons d'apprentissage évalués sur la discipline réelle $D_{1,1}$. Le comportement de cette discipline étant suffisamment simple, il n'a pas été nécessaire de mettre en place une stratégie adaptative de construction de plan d'expériences pour l'amélioration globale de modèle comme présenté dans la section 8.1. La qualité du modèle de substitution a été évaluée grâce à 100 échantillons de test supplémentaires choisis au hasard et calculés sur la discipline initiale. Les valeurs réelles de ces échantillons sont comparées aux valeurs prédites par le modèle sur la figure 10.4. Ce graphe montre que le modèle établi est de bonne qualité, les points de test étant situés près de la diagonale $y = x$. L'obtention de tels résultats à partir d'un nombre restreint d'échantillons d'apprentissage s'explique par le fait que le calcul de la masse de l'empennage vertical est réalisé par un modèle simplifié réservé à la phase d'étude conceptuelle d'un avion. Les variations de cette sortie disciplinaire sont donc relativement faciles à approcher.

À partir de ce modèle précis de la masse de l'empennage vertical, nous pouvons maintenant estimer sa variance vis-à-vis du paramètre incertain χ_{e_1}. Comme détaillé dans la section 5.8, la mesure de variance peut être calculée de diverses manières. Nous avons comparé ici le calcul par la méthode de Monte-Carlo au calcul analytique sur une base de 100 points de test choisis au hasard. L'approche par polynôme de chaos n'a pas été considérée. Le tableau 10.2 indique les résultats de ces tests en termes d'erreur et de temps

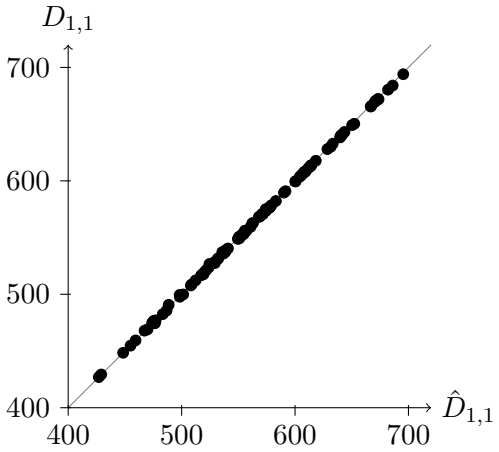


FIGURE 10.4 – Vérification de la qualité du modèle de krigeage $\hat{D}_{1,1}$: la valeur de 100 échantillons de test (points noirs) prédite par le modèle est comparée à la valeur réelle donnée par la fonction de référence.

	Calcul analytique	**Monte-Carlo** 1000 pts
Erreur sur ρ (moyenne sur 100 éval.)	0.0%	2.23%
Temps de calcul (pour 1 évaluation)	0.0054 s	0.13 s

TABLE 10.2 – Comparaison des méthodes de calcul de la variance en 100 points de test sur le modèle de krigeage $\hat{D}_{1,1}$ doté de 11 échantillons d'apprentissage.

de calcul (l'erreur étant définie par rapport à l'approche analytique qui est par définition exacte). On constate que le calcul analytique permet d'obtenir plus rapidement des estimations plus précises. C'est donc cette méthode qui a été retenue ici. Elle bénéficie du faible nombre d'échantillons d'apprentissage du modèle, nombre auquel sa complexité est liée.

Une fois le calcul de variance mis en place, le problème (10.7) a été résolu à l'aide d'un algorithme génétique au sein du logiciel d'optimisation multidisciplinaire Model Center. Nous avons utilisé une population de 50 individus sur 200 générations. Cette optimisation a duré environ deux jours sur un ordinateur personnel standard. Le problème possédant trois objectifs, elle a permis d'obtenir un front de Pareto en trois dimensions, composé de 301 solutions optimales. Ces solutions sont représentées sur la figure 10.5 dans l'espace des deux objectifs initiaux f_1 et f_2 (la valeur de l'objectif de robustesse étant visualisée par un dégradé de couleur). À des fins de comparaison nous avons par la suite réalisé de la même manière une optimisation déterministe classique du problème initial à deux objectifs. Les solutions déterministes obtenues sont affichées sur la même figure.

On peut voir que certaines solutions du front de Pareto robuste sont proches des solutions déterministes. Cela confirme le fait que la méthode proposée retourne bien des solutions déterministes optimales (car on a conservé les objectifs initiaux du problème),

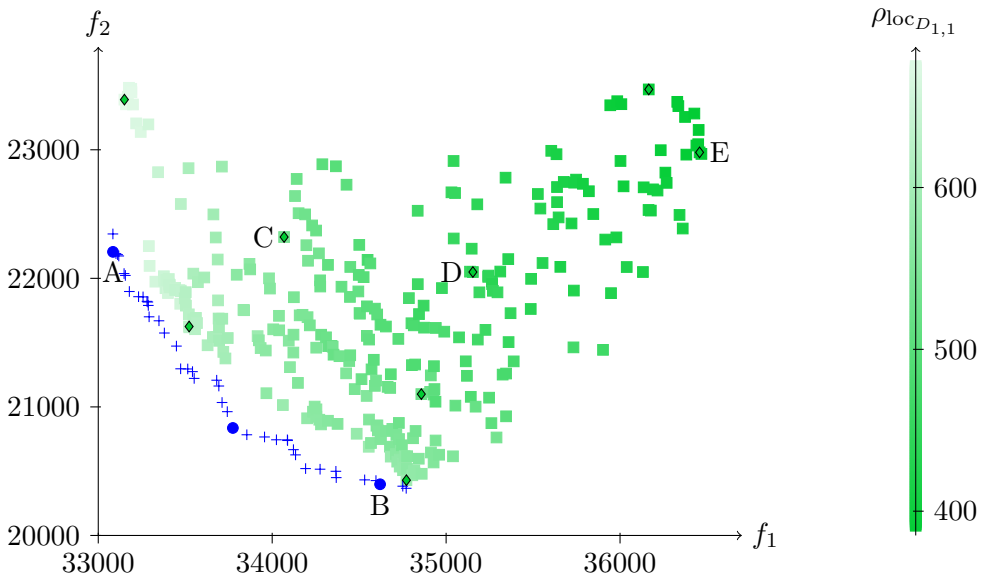


FIGURE 10.5 – Fronts de Pareto obtenus pour les problèmes déterministe (croix) et robuste (carrés) dans l'espace des objectifs initiaux. La couleur des carrés permet de visualiser l'évolution de la valeur de l'objectif de robustesse locale $\rho_{\mathrm{loc}_{D_{1,1}}}$ (plus les carrés sont foncés, plus les solutions sont robustes). Les losanges et les points représentent quant à eux les solutions qui seront analysées par la suite.

mais aussi d'autres solutions plus robustes (du moins au niveau local pour l'instant). Les solutions déterministes optimales fournies par la méthode LOUP ne sont toutefois pas aussi précises que celles obtenues grâce à l'optimisation déterministe, car le problème est plus difficile à résoudre avec trois objectifs qu'avec deux. Dans le front de Pareto robuste, plus on s'éloigne du front déterministe, plus la performance se dégrade pour les objectifs initiaux mais plus la variance locale $\rho_{\mathrm{loc}_{D_{1,1}}}$ est faible (ce qui correspond à une robustesse globale accrue). On retrouve donc des solutions de compromis entre performance et robustesse.

La résolution du problème (10.7) a donné un ensemble de solutions parmi lesquelles le concepteur peut choisir celle qui répond le mieux à ses attentes en termes de performance et de robustesse. Mais avant de pouvoir les exploiter, il faut tout d'abord vérifier l'applicabilité de la méthode LOUP au cas traité afin de valider ces résultats.

10.4 Expression d'un critère d'applicabilité de la méthode

Comme nous l'avons vu précédemment, le principe de la méthode LOUP n'est valable que dans les problèmes où les disciplines incertaines ont un effet visible sur les variations des objectifs. Nous proposons maintenant un critère pour vérifier cette hypothèse pour un problème donné. Ce critère est basé sur la corrélation qui peut exister entre les sorties des disciplines incertaines et les objectifs. Si le test d'applicabilité se révèle positif, il est possible de traiter l'incertitude localement à l'aide de la méthode LOUP. Sinon, la méthode n'est pas applicable et il faudra avoir recours à une propagation d'incertitudes complète pour obtenir des solutions robustes (voir la figure 10.6).

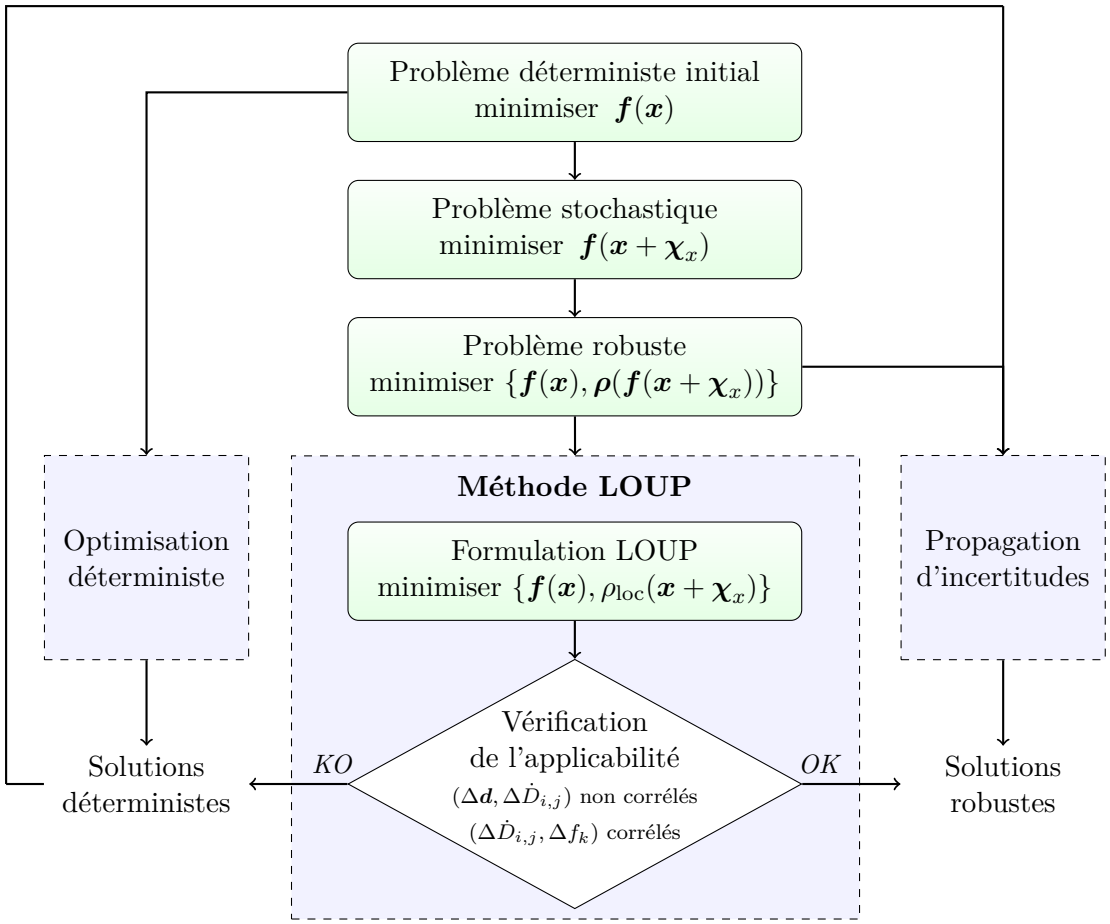


FIGURE 10.6 – Processus d'optimisation robuste multidisciplinaire par la méthode LOUP.

10.4.1 Évaluation de corrélations a posteriori

Le but du critère d'applicabilité de la méthode LOUP est d'estimer la corrélation entre les variations des sorties des disciplines incertaines et celles des objectifs. Si ces variations sont liées, alors on pourra en conclure que les disciplines incertaines ont effectivement un impact direct sur les variations des objectifs du problème. De larges variations des sorties disciplinaires correspondront alors à des variations importantes des objectifs, et c'est pourquoi il sera intéressant dans ce cas de minimiser les variations des disciplines incertaines pour réduire la variance des fonctions objectifs.

Afin d'éviter de lancer des calculs coûteux supplémentaires sur le système multidisciplinaire, nous proposons d'estimer les variations des disciplines et des objectifs par différences finies à partir des données récoltées lors de l'optimisation. En effet, l'optimiseur a exploré de nombreux points avant de trouver les solutions optimales, et ces points qui ont été évalués avec le système peuvent être sauvegardés pour être réutilisés a posteriori. Dans le cas d'un algorithme génétique, la population peut ainsi être copiée dans une archive à chaque itération sans entraver le bon déroulement de l'optimisation. La vérification de l'applicabilité de la méthode LOUP peut aussi être réalisée a priori si on a accès à des données issues par exemple d'une précédente optimisation déterministe réalisée sur le système.

On note $\boldsymbol{x} = (\boldsymbol{d}, \boldsymbol{e})$ l'ensemble des entrées du système multidisciplinaire. À partir de

deux points $\boldsymbol{x}$ et $\boldsymbol{x}'$ évalués durant l'optimisation, il est possible de calculer les variations correspondantes pour chaque sortie de discipline incertaine $D_{i,j}$ et chaque objectif f_k (nous utilisons ici la notation « $\dot{D}_{i,j}$ » car les valeurs des sorties disciplinaires sauvegardées sont celles obtenues à la convergence du système) :

$$\begin{aligned} \Delta\dot{D}_{i,j}(\boldsymbol{x}, \boldsymbol{x}') &= \dot{D}_{i,j}(\boldsymbol{x}) - \dot{D}_{i,j}(\boldsymbol{x}'), \\ \Delta f_k(\boldsymbol{x}, \boldsymbol{x}') &= f_k(\boldsymbol{x}) - f_k(\boldsymbol{x}'). \end{aligned} \tag{10.8}$$

Les variations que nous souhaitons observer ici devraient si possible être dues aux incertitudes du problème. Il faudrait donc sélectionner des points qui possèdent des valeurs identiques au niveau des variables déterministes et distinctes au niveau des variables incertaines : $\boldsymbol{x}$ et $\boldsymbol{x}'$ ne devraient différer que sur les variables incertaines. Cette sélection peut être réalisable lorsque l'incertitude n'affecte que des variables de décision, mais quand on a affaire à des incertitudes environnementales (comme dans la figure 10.3 par exemple) des valeurs différentes des variables incertaines $\boldsymbol{e}$ ne sont pas disponibles car le processus d'optimisation modifie uniquement les variables de décision $\boldsymbol{d}$. En effet, l'optimisation du problème (10.6) est menée comme une optimisation déterministe classique sans considérer de variations pour les paramètres incertains. Ceux-ci sont pris en compte localement dans l'estimation de la variance des sorties des disciplines incertaines mais ne sont pas visibles au niveau global. Les calculs de différences finies ne peuvent donc pas se baser sur des points possédant des valeurs de paramètres incertains distinctes. Dans ce cas, les variations $\Delta\dot{D}_{i,j}$ et Δf_k peuvent être estimées à partir de points avec une petite différence $\Delta\boldsymbol{d}$ sur les variables de décision : plutôt que de comparer les points $(\boldsymbol{d}, \boldsymbol{e})$ et $(\boldsymbol{d}, \boldsymbol{e}')$, nous utiliserons donc $(\boldsymbol{d}, \boldsymbol{e})$ et $(\boldsymbol{d}', \boldsymbol{e})$ car la valeur de $\boldsymbol{e}$ reste fixe durant l'optimisation. Les fluctuations des variables de décision affectent les sorties des disciplines incertaines et donc influencent le système tout entier de la même manière que le feraient les paramètres incertains. Si les valeurs des variables de décision des points sélectionnés sont assez proches l'une de l'autre, leur effet sur les autres disciplines non incertaines peut être négligé. Tous les couples $(\boldsymbol{x}, \boldsymbol{x}')$ possibles provenant des données de l'optimisation peuvent ainsi être triés afin de sélectionner ceux dont la distance entre $\boldsymbol{x}$ et $\boldsymbol{x}'$ ne dépasse pas un certain seuil $t \in \mathbb{R}$ (il est préférable que les données soient normalisées avant de calculer ces distances) :

$$\Delta\boldsymbol{d}(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{dist}(\boldsymbol{d}, \boldsymbol{d}') \leq t. \tag{10.9}$$

La valeur du seuil t doit être définie de telle sorte que la distance entre $\boldsymbol{x}$ et $\boldsymbol{x}'$ soit aussi faible que possible, mais tout en autorisant la sélection d'un nombre suffisant de couples pour les calculs de corrélation à venir. De plus, les variations des variables de décision $\boldsymbol{d}$ et celles des sorties disciplinaires $D_{i,j}$ ne doivent pas être corrélées sur les données sélectionnées afin que leurs effets sur f_k puissent être distingués (même si les fluctuations de $\boldsymbol{d}$ restent faibles). On peut ainsi tracer $\Delta\dot{D}_{i,j}$ en fonction de $\Delta\boldsymbol{d}$ pour voir si une corrélation existe entre les deux. Cela permet aussi d'avoir une aide visuelle pour fixer la valeur de t comme nous le verrons dans l'application de la section suivante. Finalement, les valeurs de $\Delta\dot{D}_{i,j}$ et Δf_k sont obtenues pour chaque couple $(\boldsymbol{x}, \boldsymbol{x}')$ sélectionné parmi les données d'optimisation disponibles.

La relation entre les disciplines incertaines et les objectifs peut être estimée en calculant un coefficient de corrélation $C_{i,j,k}$ pour chaque paire $(\Delta\dot{D}_{i,j}, \Delta f_k)$. Nous avons utilisé ici le coefficient standard de corrélation de Pearson. Il ne permet de mettre en évidence que des relations linéaires et il pourrait donc être intéressant d'utiliser d'autres coefficients plus

sophistiqués, mais celui-ci reste relativement simple et facile à évaluer :

$$C_{i,j,k} = \mathrm{corr}(\Delta\dot{D}_{i,j}, \Delta f_k) = \frac{\mathrm{cov}(\Delta\dot{D}_{i,j}, \Delta f_k)}{\mathrm{Var}(\Delta\dot{D}_{i,j})\,\mathrm{Var}(\Delta f_k)}. \tag{10.10}$$

Si le coefficient de corrélation $C_{i,j,k}$ est proche de ± 1, cela signifie que $\Delta f_k \simeq \beta\Delta\dot{D}_{i,j}$ avec $\beta \in \mathbb{R}$. La minimisation des variations de $D_{i,j}$ (ou leur maximisation, selon le signe de β) induira donc une minimisation des variations de f_k. β est la plupart du temps positif, et c'est pourquoi les variations des sorties disciplinaires $D_{i,j}$ sont minimisées dans notre méthode. Bien sûr, si β est faible il n'est pas très utile de chercher à minimiser les variations de $D_{i,j}$ car le gain sur f_k sera insignifiant, mais nous n'avons pas abordé ce cas particulier ici. On peut noter que l'approximation effectuée lors du calcul des variations $\Delta\dot{D}_{i,j}$ et Δf_k à partir de données avec une faible différence sur $\boldsymbol{d}$ au lieu de modifications sur les paramètres incertains $\boldsymbol{e}$ ne conduit qu'à une sous-estimation du coefficient de corrélation $C_{i,j,k}$. En effet, si les variations de $\boldsymbol{d}$ ne sont pas aussi infimes qu'escompté et ont un effet direct sur les variations de f_k par le biais des autres disciplines (en plus de leur effet sur $D_{i,j}$), cela aura plutôt tendance à réduire la corrélation entre les deux. Certaines disciplines corrélées avec les objectifs pourront ne pas être repérées à cause de cela, mais quand on trouve un coefficient $C_{i,j,k}$ élevé on peut conclure qu'une corrélation existe bel et bien. Cela n'est pas vrai lorsque $\Delta\boldsymbol{d}$ et $\Delta\dot{D}_{i,j}$ sont corrélés (ce qui a été vérifié auparavant) : dans ce cas, la corrélation entre $\Delta\dot{D}_{i,j}$ et Δf_k ne peut pas être estimée correctement et on considère donc que $C_{i,j,k}$ vaut zéro.

Nous venons de détailler une méthode de calcul a posteriori des corrélations $C_{i,j,k}$ des couples $(\Delta\dot{D}_{i,j}, \Delta f_k)$ d'un problème. Nous pouvons maintenant les utiliser pour vérifier l'applicabilité de la méthode LOUP. Cette méthode nécessite que les variations de chaque objectif f_k soient corrélées avec celles de toutes les disciplines incertaines $D_{i,j}$ afin que la minimisation de l'objectif de robustesse ρ_{loc} ajouté dans le problème (10.6) amène à une minimisation de la variance de f_k. Si un objectif donné n'est pas corrélé avec une des disciplines incertaines (au moins), rien ne peut être affirmé concernant la robustesse des solutions obtenues par rapport à cet objectif. Les paramètres incertains qui affectent la discipline non corrélée peuvent en effet venir perturber l'objectif de manière incontrôlable. La méthode LOUP produira donc plus facilement des résultats exploitables pour les problèmes où le nombre de disciplines incertaines est limité, car les chances de trouver des corrélations seront plus importantes. D'autres disciplines non incertaines dans le système peuvent aussi être corrélées aux objectifs, mais elles ne sont pas prises en compte ici car elles ne jouent pas de rôle dans le problème résolu vu qu'elles ne sont pas affectées directement par les variables incertaines. L'incertitude ne peut entrer dans le système qu'au travers des disciplines incertaines, et comme il n'y a pas de propagation elle n'atteint pas les autres disciplines. De plus, les effets des disciplines non incertaines sont pris en compte dans l'étude des variations des objectifs, car la valeur des objectifs dépend aussi de ces disciplines.

Lorsque la corrélation d'un objectif avec une des disciplines incertaines est faible, la méthode LOUP n'est pas capable de donner des solutions robustes relativement à cet objectif. Les solutions obtenues peuvent toutefois être intéressantes pour les autres objectifs du problème qui sont corrélés avec toutes les disciplines incertaines. Dans le cas où aucun des objectifs ne présente de corrélation élevée avec l'ensemble de ces disciplines, l'objectif de robustesse ρ_{loc} ajouté au problème (10.6) ne doit pas être pris en compte car son lien avec la robustesse des objectifs n'est pas vérifié. Mais comme les objectifs initiaux ont

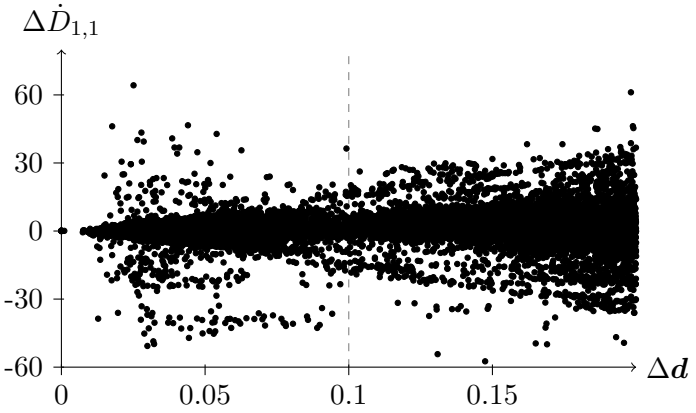


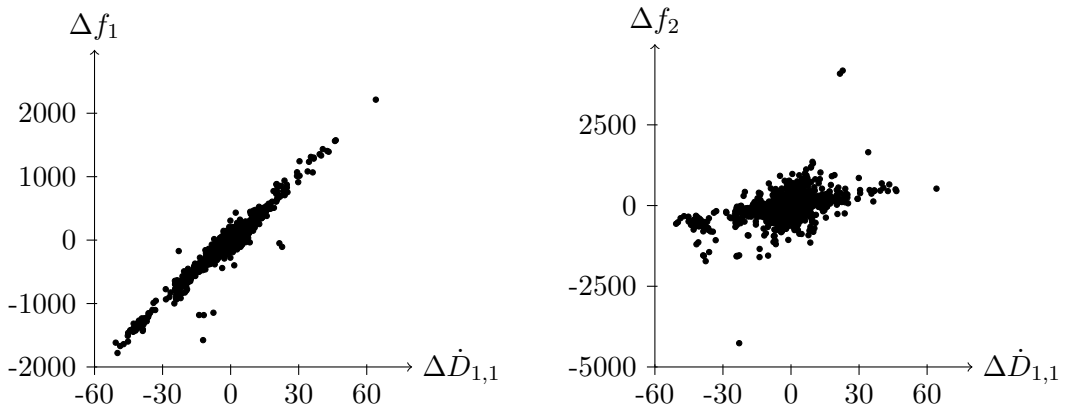
FIGURE 10.7 – Variations de la masse $D_{1,1}$ de l'empennage vertical par rapport à celles des variables de décision $\boldsymbol{d}$ normalisées. Le seuil $t = 0.1$ choisi pour sélectionner les couples considérés est représenté en pointillés.

été conservés dans le problème, les solutions déterministes standard peuvent toujours être extraites du front de Pareto obtenu. On peut donc récupérer des solutions optimales comme si on avait mené une optimisation déterministe du système. Pour trouver les solutions robustes d'un tel problème, la méthode LOUP ne peut pas être appliquée et une méthode de propagation d'incertitude devra être utilisée à la place.

10.4.2 Vérification de l'applicabilité pour le cas-test de conception avion

Revenons maintenant au problème d'optimisation d'un design d'avion, pour lequel la méthode LOUP a permis de trouver un ensemble de solutions optimales. Afin de valider ces résultats, il faut vérifier l'applicabilité de cette méthode dans le cadre de notre problème. On doit estimer pour cela la corrélation entre les variations $\Delta\dot{D}_{1,1}$ de la sortie de la discipline incertaine et celles Δf_1 et Δf_2 des deux objectifs. Ces corrélations peuvent être estimées à partir des 10000 points explorés par l'algorithme génétique durant la résolution du problème (10.7). Ces données définissent ainsi jusqu'à 10000^2 couples $(\boldsymbol{x}, \boldsymbol{x}')$ possibles, mais nous ne considèrerons pas ici les couples $(\boldsymbol{x}, \boldsymbol{x})$ composés d'un même point, ni les couples symétriques (si $(\boldsymbol{x}, \boldsymbol{x}')$ est pris en compte, alors $(\boldsymbol{x}', \boldsymbol{x})$ sera ignoré).

La première étape est de vérifier qu'il n'y a pas de corrélation entre les variations $\Delta\boldsymbol{d}$ des variables de décision et celles $\Delta\dot{D}_{1,1}$ de la discipline incertaine. Nous rappelons que $\Delta\boldsymbol{d}$ correspond à la distance entre deux points $\boldsymbol{x}$ et $\boldsymbol{x}'$ relativement aux variables de décision $\boldsymbol{d}$ normalisées. Prendre en compte l'ensemble des couples $(\boldsymbol{x}, \boldsymbol{x}')$ disponibles serait assez coûteux (et inutile). Nous avons tracé sur la figure 10.7 les variations observées pour les couples à une distance $\Delta\boldsymbol{d}$ maximale de 0.2 (dans l'espace normalisé). Grâce à ce graphe, nous avons finalement choisi de considérer les couples avec une distance $\Delta\boldsymbol{d}$ inférieure à $t = 0.1$. Nous aurons ainsi 8636 couples à prendre en compte dans les prochains calculs. Ce seuil de distance permet de s'assurer que les points des couples choisis sont relativement proches les uns des autres, et autorise la sélection de suffisamment de données pour pouvoir calculer correctement les corrélations sur les objectifs par la suite. Sur cet ensemble de points, la corrélation entre $\Delta\boldsymbol{d}$ et $\Delta\dot{D}_{1,1}$ est de 0.04. Les variations des variables de décision ne sont donc pas corrélées à celles de la masse de l'empennage vertical, et il sera ainsi



(a) $C_{1,1,1} = 0.94$ (b) $C_{1,1,2} = 0.42$

FIGURE 10.8 – Variations des objectifs f_1 (à gauche) et f_2 (à droite) par rapport à celles de la masse $D_{1,1}$ de l'empennage vertical.

possible de distinguer leur effet respectif sur les variations des objectifs du problème.

La figure 10.8 présente les variations Δf_1 et Δf_2 des objectifs par rapport à celles $\Delta \dot{D}_{1,1}$ de la masse de l'empennage vertical pour les 8636 couples sélectionnés. La corrélation entre la sortie $D_{1,1}$ de la discipline incertaine et l'objectif de masse de l'avion f_1 est relativement élevée (0.94), alors qu'elle est moindre pour l'objectif de consommation f_2 (0.42). Cela signifie que la masse de l'empennage est bien liée au premier objectif mais qu'elle ne joue pas un rôle prépondérant sur le second. Ainsi, la minimisation de la variance locale de cette masse d'empennage a résulté en une réduction de la variance de la masse à vide équipée de l'avion. Rien ne peut par contre être conclu concernant la variance de la masse de carburant nécessaire à la réalisation de la mission. Les solutions obtenues avec la méthode LOUP sont donc robustes en termes de masse de l'avion, mais pas nécessairement en termes de consommation. Si on recherche de la robustesse par rapport à ce second objectif, il faudra passer par une propagation d'incertitudes complète dans le système. Nous ne l'avons pas mise en place ici car il ne nous était pas possible d'effectuer de modifications au sein du système multidisciplinaire utilisé, et aussi car les solutions robustes que nous avons trouvées pour le premier objectif sont satisfaisantes. Nous en avons sélectionné quelques une pour une étude plus approfondie.

10.5 Analyse des solutions optimales obtenues

Afin d'analyser les résultats obtenus et de valider la démarche LOUP, nous avons évalué la robustesse de certaines solutions par des tirages de Monte-Carlo sur la simulation multidisciplinaire de l'avion. Cette étape n'est bien entendu pas obligatoire, mais elle permet de vérifier les résultats approchés fournis par la méthode LOUP et son critère d'applicabilité, et reste un bon moyen de préciser le comportement des objectifs autour des solutions optimales sélectionnées.

La meilleure façon de valider notre approche dans le cadre de ce cas-test aurait été de lancer une optimisation robuste sur le système tout entier considéré comme une boîte noire (en calculant la variance globale des objectifs par Monte-Carlo) puis de comparer

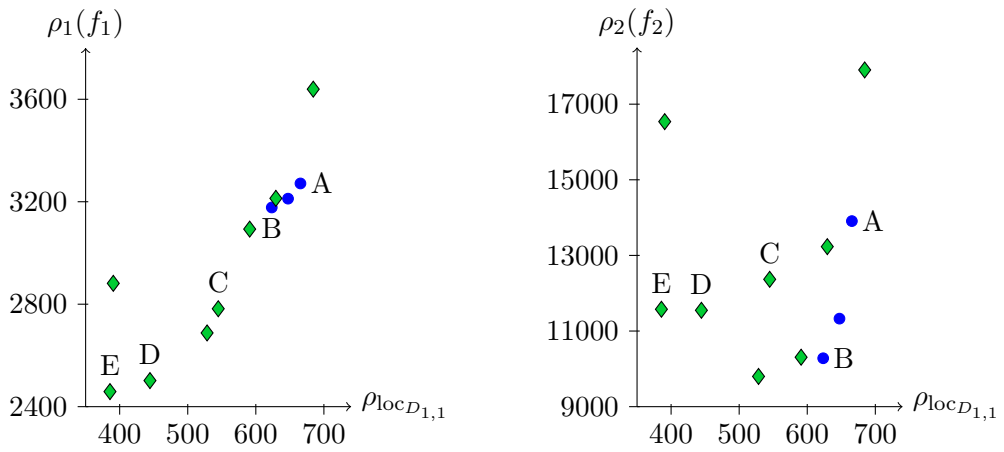


FIGURE 10.9 – Comparaison des variances locale $\rho_{\mathrm{loc}_{D_{1,1}}}$ et globales $\rho_i(f_i)$ des solutions sélectionnées au niveau de l'objectif de masse à vide équipé (à gauche) et de celui de masse de carburant emporté (à droite).

les résultats obtenus à ceux de la méthode LOUP. Mais l'estimation globale de variance par Monte-Carlo est très coûteuse, ce qui rend impraticable son utilisation au sein d'une boucle d'optimisation. C'est pourquoi nous avons simplement lancé quelques évaluations de variance sur un ensemble choisi de points d'intérêt. Trois points ont ainsi été extraits du front déterministe de la figure 10.5, et huit autres du front de Pareto robuste. Ces solutions ont été choisies de manière à être représentatives de l'ensemble des fronts (nous aurions pu réaliser une classification pour cela). La variance globale des objectifs en chacune de ces solutions a été estimée grâce à 1000 tirages de Monte-Carlo. Pour chaque tirage, une valeur de la surface de l'empennage vertical a été choisie au hasard relativement à la distribution de χ_{e_1} et les objectifs ont été évalués en utilisant cette valeur pour le paramètre incertain. Les calculs ont duré environ 3 heures pour chaque solution étudiée.

La figure 10.9 montre les valeurs obtenues pour les variances globales des objectifs f_1 et f_2 en fonction de la variance locale $\rho_{\mathrm{loc}_{D_{1,1}}}$ de chaque solution. On constate que l'optimisation robuste avec la méthode LOUP a permis de trouver des solutions avec une variance du premier objectif inférieure à celle des solutions déterministes. On peut aussi voir que la robustesse globale varie à peu près linéairement avec la robustesse locale pour f_1, pour lequel on avait trouvé une corrélation forte avec la discipline incertaine. La minimisation de la variance de cette discipline a donc bel et bien mené à une minimisation de la variance du premier objectif. Pour le second objectif, qui avait au contraire un coefficient de corrélation faible, la robustesse globale ne semble pas être clairement liée à la variance locale de la discipline incertaine. Ces résultats valident donc l'utilisation du critère de validation a posteriori de l'applicabilité de la méthode LOUP.

Nous avons finalement comparé les profils d'aile obtenus pour cinq de ces solutions. Le détail des valeurs des paramètres et des objectifs en ces points est donné dans le tableau 10.3 (ils sont aussi repérés par des lettres sur les figures précédentes). Les solutions A et B sont les solutions optimales extrêmes du problème déterministe. Elles sont tracées sur la figure 10.10. On peut voir qu'elles correspondent à des designs de voilure assez différents même si leur surface est similaire. La solution A correspond à une aile assez courte qui permet d'obtenir un avion de masse minimale, alors que la voilure de la solution B est

	λ	ε	φ	S	Y_k	E_e	E_k	E_r	$H_{\text{réf}}$	H_{opt}
A	7.67	0.18	29.59	124.0	5.43	0.15	0.13	0.11	10606	9448
B	9.92	0.27	28.08	129.8	5.60	0.14	0.10	0.11	10728	10166
C	6.83	0.17	26.32	146.2	5.36	0.14	0.11	0.11	10575	9974
D	7.13	0.17	28.98	171.6	7.41	0.15	0.12	0.11	11074	10297
E	6.58	0.21	29.94	199.9	6.99	0.15	0.12	0.11	11105	10290

	MTOW	W_{fuel}	$\rho_{\text{loc}_{D_{1,1}}}$	$\rho_1(f_1)$	$\rho_2(f_2)$
A	33084.72	22205.93	665.57	3271.38	13907.63
B	34620.37	20398.19	623.42	3177.59	10280.29
C	34068.55	22321.32	544.82	2782.27	12367.61
D	35153.91	22049.09	444.52	2501.82	11548.24
E	36457.28	22979.79	385.78	2458.81	11576.28

TABLE 10.3 – Valeurs des variables de décision, des objectifs ainsi que des variances locale et globales pour les cinq solutions sélectionnées.

plus aérodynamique afin de minimiser la consommation de l'appareil. Les solutions C, D et E sont quant à elles issues du front de Pareto robuste avec des valeurs croissantes de robustesse locale $\rho_{\text{loc}_{D_{1,1}}}$. Elles sont représentées sur la figure 10.11. Ces solutions emportent une masse de carburant globalement équivalent à celle de la solution A, et leur design est ainsi de même type. Ces trois solutions possèdent un allongement comparable, mais on peut par contre remarquer que leur surface augmente avec le degré de robustesse. Il semblerait donc que la diminution de la variance sur la masse à vide équipée de l'appareil soit réalisée grâce à une augmentation de la surface de la voilure.

L'ensemble des solutions que nous venons d'analyser sont fournies au concepteur afin que celui-ci puisse décider du compromis à effectuer entre les performances du futur appareil et leur robustesse vis-à-vis des incertitudes du problème.

10.6 Conclusion

Nous avons présenté dans ce chapitre une nouvelle méthode appelée LOUP qui permet de prendre en compte des incertitudes dans un problème d'optimisation multidisciplinaire de manière facilitée par rapport à une propagation d'incertitudes complète. Un certain degré de gestion des incertitudes peut ainsi être intégré aux processus d'optimisation multidisciplinaire existants sans avoir à effectuer de lourdes modifications. La méthode LOUP est basée sur un traitement local des incertitudes au sein des disciplines directement affectées par des paramètres incertains. La recherche des solutions robustes est réalisée en minimisant la variance des sorties de ces disciplines aux côtés des objectifs initiaux du problème déterministe. Elle mène à des résultats approchés qui peuvent être validés a posteriori grâce à un critère évaluant l'applicabilité de la méthode au problème. Ce critère est basé sur l'estimation de la corrélation entre les variations des sorties des disciplines incertaines et celles des objectifs, et peut être calculé grâce aux données récoltées lors de l'optimisation.

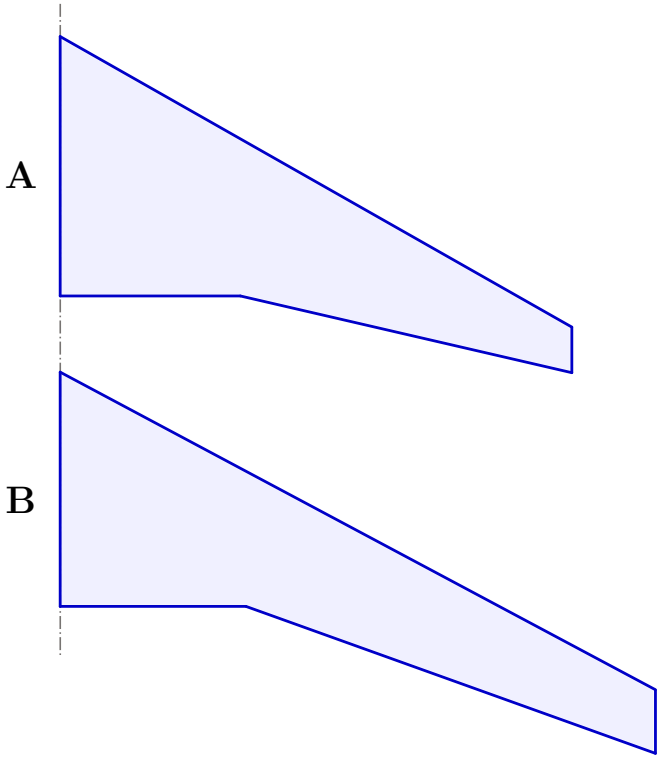


FIGURE 10.10 – Voilures des solutions déterministes sélectionnées.

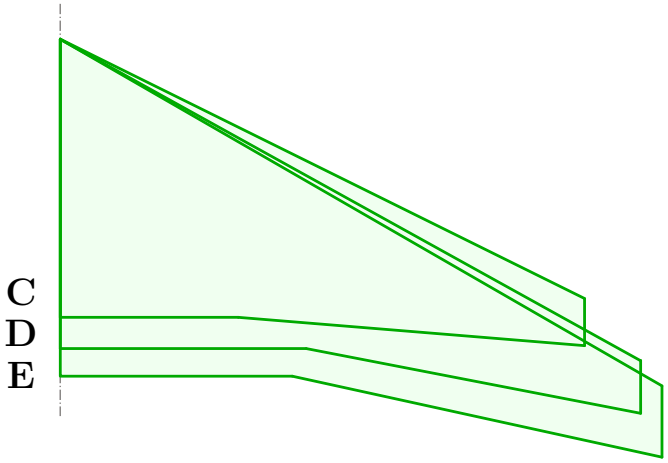


FIGURE 10.11 – Voilures des solutions robustes sélectionnées.

Nous avons appliqué cette méthode LOUP à un cas-test de conception avion pour lequel elle a permis de trouver des configurations plus robustes que les designs déterministes initiaux (du moins par rapport au premier objectif du problème pour lequel le critère était validé). Ces résultats ont été confirmés à l'aide d'estimations des variances globales des objectifs par Monte-Carlo au niveau de certaines solutions intéressantes.

L'inconvénient de la méthode LOUP est qu'il s'agit d'une approche heuristique qui ne fournit que des solutions approchées. Il est donc préférable de les valider a posteriori. Seule une propagation d'incertitudes complète à travers le système pourrait donner des résultats précis, mais il n'est pas toujours possible de la mettre en place. Notre méthode n'est de plus pas applicable à tous les problèmes d'optimisation robuste multidisciplinaire : son applicabilité doit être vérifiée avant d'exploiter ses résultats.

Dans le futur, la méthode LOUP pourrait être implémentée au sein d'un processus d'optimisation multiniveaux afin d'évaluer ses apports dans cet autre contexte (l'application présentée ici utilisant une stratégie de résolution mononiveau). Des critères de validation plus sophistiqués pourraient aussi être formulés en vue de dépasser les limitations du coefficient de corrélation linéaire que nous avons employé, et d'étendre ainsi l'applicabilité de la méthode à un plus grand nombre de problèmes. Enfin, la prise en compte de la robustesse relative aux contraintes pourrait être développée sur le même principe que celle des objectifs pour répondre aux attentes de fiabilité des systèmes.

Conclusion et perspectives

Cette thèse a permis de mieux appréhender les méthodes de prise en compte des incertitudes d'un problème d'optimisation. Nous avons ainsi présenté dans un premier temps une synthèse des processus de définition et de résolution d'un problème d'optimisation robuste. À partir d'un problème déterministe initial, l'intégration des incertitudes présentes sur les différents paramètres d'entrée permet d'exprimer le problème sous sa forme stochastique. On peut ensuite choisir des mesures de robustesse adaptées et formuler le problème d'optimisation robuste final en lien avec les possibilités de calcul de ces mesures. La résolution du problème en elle-même est réalisée à l'aide d'algorithmes d'optimisation déterministe classiques.

Les résultats principaux mis en évidence dans cette thèse sont les deux approches de résolution de problèmes d'optimisation robuste que nous avons proposées. La première est basée sur l'enrichissement adaptatif de plans d'expériences, et la seconde permet de prendre en compte les incertitudes dans le cadre des problèmes multidisciplinaires.

Nous avons développé tout d'abord une méthode d'optimisation multiobjectifs adaptative par modèles de substitution permettant d'optimiser des fonctions coûteuses. Cette approche localise les points les plus judicieux à ajouter au plan d'expériences afin de cerner au mieux les zones optimales de l'espace de recherche. Elle peut être utilisée aussi bien dans le cadre de l'optimisation déterministe (méthode PareBO) que de l'optimisation robuste (méthode PareBRO). L'approche est basée sur la recherche de la bande de Pareto d'un problème, qui est un ensemble de solutions possiblement optimales si on considère les intervalles de confiance des prédictions des modèles. Ces solutions déterminent différentes régions de l'espace possiblement optimales, que nous distinguons par classification. On recherche alors un échantillon améliorant la qualité des modèles sur chacune de ces zones. Comme on a affaire à un problème multiobjectifs, ces échantillons sont choisis comme des solutions de compromis entre les améliorations de tous les modèles établis. L'approche détermine ainsi plusieurs échantillons améliorants à chaque itération (un par classe), qu'il est ensuite possible d'évaluer en parallèle. Les méthodes PareBO et PareBRO permettent de ce fait de résoudre des problèmes d'optimisation multiobjectifs en limitant le nombre d'évaluations des fonctions de référence.

La méthode PareBRO a été appliquée avec succès à un problème industriel. Il consistait à optimiser une chambre de combustion de turbomachine en vue de réduire ses émissions polluantes, tout en considérant une incertitude sur la répartition du carburant entre ses différents injecteurs. Les résultats obtenus avec la méthode PareBRO sur les modèles de substitution ont été confirmés par des simulations numériques du système.

L'inconvénient majeur de la méthode PareBRO reste son coût, qui devient très vite prohibitif si on utilise d'autres mesures de robustesse que la mesure d'espérance. Diverses adaptations pourraient être envisagées pour remédier à cela, en simplifiant par exemple la recherche du point améliorant général ou en estimant les intervalles de confiance des mesures de robustesse de manière approchée. On pourrait aussi utiliser des modèles de substitution pour lesquels la complexité des évaluations de robustesse ne dépend pas du nombre d'échantillons d'apprentissage présents dans le plan d'expériences, contrairement au krigeage. Comme le nombre d'échantillons augmente au cours de l'enrichissement adaptatif, le coût de l'approche croît aussi dans la méthode actuelle. Si on se basait par exemple sur des fonctions à base radiales (RBF), la durée de la recherche des échantillons améliorants pourrait être la même à chaque itération. L'erreur de ces modèles devrait par contre être évaluée par bootstrap, ce qui peut représenter un coût supplémentaire non négligeable.

La seconde approche que nous avons développée est une méthode approchée nommée LOUP qui vise à faciliter la prise en compte d'incertitudes en optimisation multidisciplinaire. Elle permet d'éviter d'avoir à réaliser une propagation d'incertitudes coûteuse au sein du système, et est donc particulièrement adaptée au cas de systèmes multidisciplinaires existants pour lesquels toute modification est relativement lourde. Cette méthode s'appuie sur un traitement local de l'incertitude au niveau des disciplines directement affectées par les paramètres incertains. Les solutions robustes du problème sont ainsi obtenues en minimisant la variance des sorties de ces disciplines, dans le but de limiter l'impact des incertitudes sur le système. Le problème fait aussi intervenir les objectifs déterministes initiaux pour avoir accès aux solutions optimales déterministes si nécessaire. La méthode LOUP n'est en effet applicable que sous certaines hypothèses. Nous avons ainsi proposé un critère de vérification de son applicabilité à un problème donné. Ce critère est basé sur l'estimation de la corrélation entre les variations des sorties des disciplines incertaines et celles des objectifs, et peut être calculé a posteriori grâce aux données récoltées lors de l'optimisation. Si ce critère est vérifié, les solutions robustes renvoyées par la méthode peuvent être exploitées. Sinon, il faudra se contenter des solutions déterministes ou bien s'orienter vers une propagation d'incertitudes pour obtenir des solutions robustes.

La méthode LOUP a permis de résoudre un cas-test industriel visant à optimiser un design d'avion. L'incertitude provenait de l'estimation de la surface de l'empennage vertical. La méthode a fourni des solutions plus robustes que les solutions déterministes initiales, relativement au premier objectif de minimisation de la masse de l'appareil. La robustesse vis-à-vis du second objectif de consommation de carburant n'a par contre pas pu être attestée car le critère d'applicabilité n'était pas vérifié dans ce cas. Ces résultats ont été validés par la suite à l'aide de simulations complémentaires du système.

Cette méthode reste toutefois une heuristique permettant de trouver des solutions approchées à un problème d'optimisation robuste multidisciplinaire. Il est donc préférable de valider ses résultats par une étude approfondie des solutions optimales sélectionnées. Elle ne s'applique pas non plus à l'ensemble des problèmes qu'on peut rencontrer. La gamme de cas traitables pourrait néanmoins être étendue en définissant des critères d'applicabilité plus sophistiqués. Il faudrait enfin étudier l'intégration la méthode LOUP au sein d'une stratégie de résolution multiniveaux.

La prochaine étape consisterait à comparer ces approches à d'autres approches existantes dont le but est plus ou moins similaire. Dans le cadre de la méthode PareBO, il serait par exemple possible d'effectuer des comparaisons avec des méthodes d'optimisation adaptatives multiobjectifs inspirées de l'algorithme EGO, mais sans considérer de calculs de points en parallèle. On pourrait encore examiner les méthodes mono-objectif qui sélectionnent plusieurs points à la fois. Les méthodes existantes dédiées à l'optimisation robustes sont elles aussi limitées au cas mono-objectif, à notre connaissance. Concernant la méthode LOUP, on pourrait évaluer son efficacité sur un nombre plus important de cas en la comparant aux résultats obtenus par propagation d'incertitudes, voire par d'autres méthodes de résolution approchée semi-intrusives.

Dans le futur, il serait possible d'étendre ces méthodes à la gestion de la fiabilité des systèmes, en étudiant la robustesse vis-à-vis des contraintes des problèmes en plus de celle des objectifs. Les approches fiabilistes sont par contre presque exclusivement basées sur des mesures de quantile, qui permettent d'estimer la probabilité de violation d'une contrainte. Il serait donc intéressant d'établir des méthodes efficaces de calcul des quantiles, car ils demeurent pour l'instant les mesures les plus difficiles à évaluer. Nos recherches à ce sujet sont d'ailleurs restées vaines. Il existe par exemple des méthodes d'échantillonnage adaptatif qui ont été développées dans ce sens, mais elles sont généralement trop lourdes pour pouvoir être intégrées dans un processus d'optimisation du type de ceux que nous avons développés ici.

L'optimisation robuste est finalement un domaine relativement jeune, notamment en ce qui concerne les aspects multiobjectifs. La prise en compte d'incertitudes lors de la conception de nouveaux systèmes commence à peine aujourd'hui à intégrer les processus industriels. Il est donc certain que l'optimisation sous incertitude a encore de beaux jours devant elle !

Annexes

Annexe A

Mesure de l'espérance sur un modèle de krigeage

Cette annexe détaille le calcul de la mesure de l'espérance E_χ sur un modèle de krigeage ordinaire $\hat{f}$ doté d'une fonction de corrélation R gaussienne, vis-à-vis d'incertitudes gaussiennes définies sur ses variables d'entrées $\boldsymbol{x}$ par l'intermédiaire de variables aléatoires indépendantes $\boldsymbol{\chi_x}$. Nous allons exprimer l'espérance $\mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]$ et la variance $\mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)])$ de cette mesure de robustesse par rapport au processus stochastique $\hat{F}$ associé au modèle $\hat{f}$, d'après les résultats présentés dans [ALC06].

Nous rappelons qu'un modèle de krigeage ordinaire peut s'écrire sous la forme suivante (d'après l'équation (3.12)) :

$$\hat{f}(\boldsymbol{x}) = \hat{\mu} + \sum_{i=1}^{n_s} \gamma_i \sigma_K^2 R(\boldsymbol{x}, \boldsymbol{s}^i) = \hat{\mu} + \sum_{i=1}^{n_s} \gamma_i \sigma_K^2 \prod_{j=1}^{n_x} e^{-\theta_j (x_j - s_j^i)^2}. \tag{A.1}$$

Nous noterons μ_j la moyenne et σ_j^2 la variance de l'incertitude définie sur la j^{e} dimension de l'espace d'entrée, ainsi que $p_{x_j+\chi_{x_j}}$ la densité de probabilité de la variable aléatoire $(x_j + \chi_{x_j}) \sim \mathcal{N}(\mu_j, \sigma_j^2)$ correspondante :

$$p_{x_j+\chi_{x_j}}(z_j) = \frac{1}{\sigma_j\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{z_j-\mu_j}{\sigma_j}\right)^2}. \tag{A.2}$$

Ces variables aléatoires étant indépendantes, la densité de probabilité multivariée $p_{\boldsymbol{x}+\boldsymbol{\chi}_x}$ vaut :

$$p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) = \prod_{j=1}^{n_x} p_{x_j+\chi_{x_j}}(z_j). \tag{A.3}$$

Espérance

Intéressons-nous tout d'abord à l'espérance de la mesure d'espérance. Les deux opérateurs d'espérance E et E_χ pouvant être intervertis, on a :

$$\mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] = \mathrm{E}_\chi[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] = \mathrm{E}_\chi[\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)]. \tag{A.4}$$

D'où :

$$\begin{aligned}
\mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] &= \int_{\mathbb{R}^{n_x}} \hat{f}(\boldsymbol{z}) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} \\
&= \hat{\mu} + \int_{\mathbb{R}^{n_x}} \left(\sum_{i=1}^{n_s} \gamma_i \sigma_K^2 \prod_{j=1}^{n_x} e^{-\theta_j (z_j - s_j^i)^2} \right) \prod_{j=1}^{n_x} p_{x_j+\chi_{x_j}}(z_j) dz_j \\
&= \hat{\mu} + \sum_{i=1}^{n_s} \gamma_i \sigma_K^2 \prod_{j=1}^{n_x} \int_{-\infty}^{\infty} e^{-\theta_j (z_j - s_j^i)^2} p_{x_j+\chi_{x_j}}(z_j) dz_j \\
&= \hat{\mu} + \sum_{i=1}^{n_s} \gamma_i \sigma_K^2 \prod_{j=1}^{n_x} I_{ij},
\end{aligned} \tag{A.5}$$

avec :

$$\begin{aligned}
I_{ij} &= \int_{-\infty}^{\infty} e^{-\theta_j (z_j - s_j^i)^2} p_{x_j+\chi_{x_j}}(z_j) dz_j \\
&= \frac{1}{\sigma_j \sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\theta_j (z_j - s_j^i)^2} e^{-\frac{1}{2}\left(\frac{z_j - \mu_j}{\sigma_j}\right)^2} dz_j \\
&= \frac{1}{\sqrt{2\sigma_j^2 \theta_j + 1}} e^{-\frac{\theta_j (\mu_j - s_j^i)^2}{2\sigma_j^2 \theta_j + 1}}.
\end{aligned} \tag{A.6}$$

Variance

Afin de pouvoir estimer l'intervalle de confiance de la mesure d'espérance, il faut maintenant calculer sa variance :

$$\begin{aligned}
\mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) &= \mathrm{E}[(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]])^2] \\
&= \mathrm{E}[(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \mathrm{E}_\chi[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]])^2] \\
&= \mathrm{E}[\mathrm{E}_\chi^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x) - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]] \\
&= \mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x) - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\, \mathrm{E}_{\chi'}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x) - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]]] \\
&= \mathrm{E}[\mathrm{E}_\chi[\mathrm{E}_{\chi'}[(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x) - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)])(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x) - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)])]]] \\
&= \mathrm{E}_\chi[\mathrm{E}_{\chi'}[\mathrm{E}[(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x) - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)])(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x) - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)])]]] \\
&= \mathrm{E}_\chi[\mathrm{E}_{\chi'}[\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x), \hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&= \mathrm{E}_\chi[\mathrm{E}_{\chi'}[\sigma_K^2 R(\boldsymbol{x}+\boldsymbol{\chi}_x, \boldsymbol{x}+\boldsymbol{\chi}'_x) \\
&\quad - \sigma_K^4 \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x, s^i) R(\boldsymbol{x}+\boldsymbol{\chi}'_x, s^j)]] \\
&= \sigma_K^2\, \mathrm{E}_\chi[\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x, \boldsymbol{x}+\boldsymbol{\chi}'_x)]] \\
&\quad - \sigma_K^4\, \mathrm{E}_\chi[\mathrm{E}_{\chi'}[\sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x, s^i) R(\boldsymbol{x}+\boldsymbol{\chi}'_x, s^j)]],
\end{aligned} \tag{A.7}$$

d'après l'équation (3.13). On précise que les variables aléatoires $(x_j + \chi'_{x_j})$ qui ont été introduites ici possèdent la même densité de probabilité que les variables $(x_j + \chi_{x_j})$ correspondantes (donc la même moyenne et la même variance), mais tout en étant indépendantes de celles-ci.

On a :

$$\begin{aligned}
\mathrm{E}_\chi[\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x, \boldsymbol{x}+\boldsymbol{\chi}'_x)]] &= \int_{\mathbb{R}^{n_x}} \int_{\mathbb{R}^{n_x}} R(\boldsymbol{z}, \boldsymbol{z'}) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z'}) d\boldsymbol{z} d\boldsymbol{z'} \\
&= \int_{\mathbb{R}^{n_x}} \int_{\mathbb{R}^{n_x}} \prod_{j=1}^{n_x} e^{-\theta_j (z_j - z'_j)^2} p_{x_j+\chi_{x_j}}(z_j) p_{x_j+\chi'_{x_j}}(z'_j) dz_j dz'_j \\
&= \prod_{j=1}^{n_x} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-\theta_j (z_j - z'_j)^2} p_{x_j+\chi_{x_j}}(z_j) p_{x_j+\chi'_{x_j}}(z'_j) dz_j dz'_j \\
&= \prod_{j=1}^{n_x} J_j,
\end{aligned} \tag{A.8}$$

avec :

$$\begin{aligned}
J_j &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-\theta_j (z_j - z'_j)^2} p_{x_j+\chi_{x_j}}(z_j) p_{x_j+\chi'_{x_j}}(z'_j) dz_j dz'_j \\
&= \frac{1}{2\pi\sigma_j^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-\theta_j (z_j - z'_j)^2} e^{-\frac{1}{2}\left(\frac{z_j - \mu_j}{\sigma_j}\right)^2} e^{-\frac{1}{2}\left(\frac{z'_j - \mu_j}{\sigma_j}\right)^2} dz_j dz'_j \\
&= \frac{1}{\sqrt{4\sigma_j^2 \theta_j + 1}},
\end{aligned} \tag{A.9}$$

On remarque que J_j dépend uniquement de la variance σ_j^2 de l'incertitude, et il peut donc être calculé indépendamment de la valeur de x_j.

On a aussi :

$$\begin{aligned}
&\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x, s^i) R(\boldsymbol{x}+\boldsymbol{\chi}'_x, s^j)]] \\
&= \int_{\mathbb{R}^{n_x}} \int_{\mathbb{R}^{n_x}} \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{z}, s^i) R(\boldsymbol{z'}, s^j) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z'}) d\boldsymbol{z} d\boldsymbol{z'} \\
&= \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} \left(\int_{\mathbb{R}^{n_x}} R(\boldsymbol{z}, s^i) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} \right) \left(\int_{\mathbb{R}^{n_x}} R(\boldsymbol{z'}, s^j) p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z'}) d\boldsymbol{z'} \right) \\
&= \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} \left(\prod_{k=1}^{n_x} \int_{-\infty}^{\infty} e^{-\theta_k (z_k - s_k^i)^2} p_{x_k+\chi_{x_k}}(z_k) dz_k \right) \left(\prod_{k=1}^{n_x} \int_{-\infty}^{\infty} e^{-\theta_k (z_k - s_k^j)^2} p_{x_k+\chi_{x_k}}(z_k) dz_k \right) \\
&= \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} \prod_{k=1}^{n_x} I_{ik} I_{jk}.
\end{aligned} \tag{A.10}$$

On obtient donc finalement :

$$\mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) = \sigma_K^2 \prod_{j=1}^{n_x} J_j - \sigma_K^4 \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} C_{ij}^{-1} \prod_{k=1}^{n_x} I_{ik} I_{jk}. \tag{A.11}$$

La variance de la mesure d'espérance se calcule ainsi au prix d'une double boucle sur le nombre n_s d'échantillons d'apprentissage du modèle de krigeage. Sa complexité évolue donc relativement à n_s^2.

On peut noter que les intégrales simples I_{ij} et J_j peuvent intervenir plusieurs fois avec les mêmes indices au cours des calculs. Il est donc possible de stocker leur résultat afin d'éviter de les réévaluer, et d'accélérer ainsi l'estimation de robustesse.

Annexe B

Mesure de la variance sur un modèle de krigeage

Cette annexe détaille le calcul de la mesure de la variance Var_χ sur un modèle de krigeage ordinaire $\hat{f}$ doté d'une fonction de corrélation R gaussienne, vis-à-vis d'incertitudes gaussiennes définies sur ses variables d'entrées $\boldsymbol{x}$ par l'intermédiaire de variables aléatoires indépendantes $\boldsymbol{\chi_x}$. Nous allons exprimer l'espérance $\mathrm{E}[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))]$ et la variance $\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))$ de cette mesure de robustesse par rapport au processus stochastique $\hat{F}$ associé au modèle $\hat{f}$, d'après les résultats présentés dans [ALC06]. Les notations utilisées sont les mêmes que pour l'annexe précédente.

Espérance

Intéressons-nous tout d'abord à l'espérance de la mesure de variance :

$$
\begin{aligned}
&\mathrm{E}[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))]\\
&= \mathrm{E}[\mathrm{E}_\chi[\hat{F}^2(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \mathrm{E}^2_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{E}[\hat{F}^2(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{E}[\mathrm{E}^2_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{E}[\hat{F}^2(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{E}_\chi[\mathrm{E}^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] + \mathrm{E}_\chi[\mathrm{E}^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{E}[\mathrm{E}^2_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] + \mathrm{E}_\chi[\mathrm{E}^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{E}[\mathrm{E}^2_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] + \mathrm{E}_\chi[\mathrm{E}^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{E}[\mathrm{E}^2_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] + \mathrm{E}^2[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&\quad - \mathrm{E}^2[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] + \mathrm{E}_\chi[\mathrm{E}^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) - \mathrm{E}^2[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] - \mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) + \mathrm{E}_\chi[\mathrm{E}^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{E}^2[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] - \mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) + \mathrm{E}_\chi[\mathrm{E}^2[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \mathrm{E}^2_\chi[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\\
&= \mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] - \mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) + \mathrm{Var}_\chi(\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]).
\end{aligned}
\tag{B.1}
$$

Il faut donc évaluer ces trois différents termes. Le deuxième correspond à la variance de la mesure d'espérance exprimée dans l'équation (A.11).

D'après l'équation (3.14), on a :

$$\begin{aligned}
\mathrm{E}_\chi[\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] &= \mathrm{E}_\chi[\sigma_K^2 - \sigma_K^4 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x, s^i) R(\boldsymbol{x}+\boldsymbol{\chi}_x, s^j)] \\
&= \sigma_K^2 - \sigma_K^4 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} \int_{\mathbb{R}^{n_x}} R(\boldsymbol{z}, s^i) R(\boldsymbol{z}, s^j) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} \\
&= \sigma_K^2 - \sigma_K^4 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} \prod_{k=1}^{n_x} \int_{-\infty}^{\infty} e^{-\theta_k (z_k - s_k^i)^2 - \theta_k (z_k - s_k^j)^2} p_{x_k + \chi_{x_k}}(z_k) dz_k \\
&= \sigma_K^2 - \sigma_K^4 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} \prod_{k=1}^{n_x} I_{ijk},
\end{aligned} \tag{B.2}$$

avec :

$$\begin{aligned}
I_{ijk} &= \int_{-\infty}^{\infty} e^{-\theta_k (z_k - s_k^i)^2 - \theta_k (z_k - s_k^j)^2} p_{x_k + \chi_{x_k}}(z_k) dz_k \\
&= \frac{1}{\sigma_k \sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\theta_k (z_k - s_k^i)^2 - \theta_k (z_k - s_k^j)^2} e^{-\frac{1}{2}\left(\frac{z_k - \mu_k}{\sigma_k}\right)^2} dz_k \\
&= \frac{1}{\sqrt{4\sigma_k^2 \theta_k + 1}} e^{-\frac{\theta_k (\mu_k - s_k^i)^2 + \theta_k (\mu_k - s_k^j)^2 + 2\sigma_k^2 \theta_k^2 (s_k^i - s_k^j)^2}{4\sigma_k^2 \theta_k + 1}}.
\end{aligned} \tag{B.3}$$

D'autre part :

$$\begin{aligned}
\mathrm{Var}_\chi(\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) &= \mathrm{Var}_\chi(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)) \\
&= \mathrm{Var}_\chi(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x) - \hat{\mu}) \\
&= \mathrm{E}_\chi[(\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x) - \hat{\mu})^2] - \mathrm{E}_\chi^2[\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x) - \hat{\mu}] \\
&= \int_{\mathbb{R}^{n_x}} \left(\sum_{i=1}^{n_s} \gamma_i \sigma_K^2 \prod_{k=1}^{n_x} e^{-\theta_k (z_k - s_k^i)^2} \right)^2 p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} - (\mathrm{E}_\chi[\hat{f}(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \hat{\mu})^2 \\
&= \int_{\mathbb{R}^{n_x}} \left(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s} \gamma_i \gamma_j \sigma_K^4 \prod_{k=1}^{n_x} e^{-\theta_k (z_k - s_k^i)^2 - \theta_k (z_k - s_k^j)^2} \right) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) d\boldsymbol{z} \\
&\quad - (\mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] - \hat{\mu})^2 \\
&= \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} \gamma_i \gamma_j \sigma_K^4 \prod_{k=1}^{n_x} \int_{-\infty}^{\infty} e^{-\theta_k (z_k - s_k^i)^2 - \theta_k (z_k - s_k^j)^2} p_{x_k + \chi_{x_k}}(z_k) dz_k \\
&\quad - \left(\sum_{i=1}^{n_s} \gamma_i \sigma_K^2 \prod_{k=1}^{n_x} I_{ik} \right)^2, \text{ d'après l'équation (A.5)} \\
&= \left(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s} \gamma_i \gamma_j \sigma_K^4 \prod_{k=1}^{n_x} I_{ijk} \right) - \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} \gamma_i \gamma_j \sigma_K^4 \prod_{k=1}^{n_x} I_{ik} I_{jk} \\
&= \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} \gamma_i \gamma_j \sigma_K^4 \left(\left(\prod_{k=1}^{n_x} I_{ijk} \right) - \prod_{k=1}^{n_x} I_{ik} I_{jk} \right).
\end{aligned} \tag{B.4}$$

On peut ainsi évaluer $\mathrm{E}[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))]$. Cette expression fait intervenir des double sommes sur le nombre n_s d'échantillons d'apprentissage du modèle de krigeage. Sa complexité évolue donc relativement à n_s^2.

Variance

Afin de pouvoir estimer l'intervalle de confiance de la mesure de variance, il faut maintenant calculer sa variance. Il s'agit d'un développement complexe que nous n'avons pas détaillé ici. Nous adaptons simplement le résultat fourni dans [ALC06] :

$$\begin{aligned}\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) = {} & 2\,\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\mathrm{Cov}^2(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\ & - 4\,\mathrm{E}_\chi[\mathrm{E}^2_{\chi'}[\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\ & + 2\,\mathrm{Var}^2(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) \\ & + 4\,\mathrm{E}_\chi[\mathrm{E}_{\chi'}[(\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]) \\ & \quad \times (\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)] - \mathrm{E}[\mathrm{E}_{\chi'}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]]) \\ & \quad \times \mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]].\end{aligned} \tag{B.5}$$

Il y a donc quatre termes à calculer ici. Le troisième correspond au carré de la variance de l'espérance, donnée dans l'équation (A.11).

Pour le premier terme, on a :

$$\begin{aligned}&\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\mathrm{Cov}^2(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\ &= \mathrm{E}_\chi[\mathrm{E}_{\chi'}[(\sigma_K^2 R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x) - \sigma_K^4 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i) R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))^2]] \\ &= \sigma_K^4\,\mathrm{E}_\chi[\mathrm{E}_{\chi'}[R^2(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)]] \\ &\quad - 2\sigma_K^6\,\mathrm{E}_\chi[\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)\sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i) R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)]] \\ &\quad + \sigma_K^8\,\mathrm{E}_\chi[\mathrm{E}_{\chi'}[(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i) R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))^2]],\end{aligned} \tag{B.6}$$

avec :

$$\begin{aligned}\mathrm{E}_\chi[\mathrm{E}_{\chi'}[R^2(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)]] &= \int_{\mathbb{R}^{n_x}}\int_{\mathbb{R}^{n_x}} R^2(\boldsymbol{z},\boldsymbol{z'}) p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z}) p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z'}) d\boldsymbol{z} d\boldsymbol{z'} \\ &= \int_{\mathbb{R}^{n_x}}\int_{\mathbb{R}^{n_x}} \prod_{j=1}^{n_x} e^{-2\theta_j (z_j - z'_j)^2} p_{x_j+\chi_{x_j}}(z_j) p_{x_j+\chi'_{x_j}}(z'_j) dz_j dz'_j \\ &= \prod_{j=1}^{n_x} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-2\theta_j (z_j - z'_j)^2} p_{x_j+\chi_{x_j}}(z_j) p_{x_j+\chi'_{x_j}}(z'_j) dz_j dz'_j \\ &= \prod_{j=1}^{n_x} \frac{1}{\sqrt{8\sigma_j^2\theta_j + 1}}, \text{ d'après l'équation (A.9),}\end{aligned} \tag{B.7}$$

$$
\begin{aligned}
&\mathrm{E}_\chi[\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)]]\\
&=\int_{\mathbb{R}^{nx}}\int_{\mathbb{R}^{nx}}R(\boldsymbol{z},\boldsymbol{z}')\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{z},\boldsymbol{s}^i)R(\boldsymbol{z}',\boldsymbol{s}^j)p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z}')d\boldsymbol{z}d\boldsymbol{z}'\\
&=\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}\int_{\mathbb{R}^{nx}}\int_{\mathbb{R}^{nx}}R(\boldsymbol{z},\boldsymbol{z}')R(\boldsymbol{z},\boldsymbol{s}^i)R(\boldsymbol{z}',\boldsymbol{s}^j)p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z}')d\boldsymbol{z}d\boldsymbol{z}'\\
&=\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}\int_{\mathbb{R}^{nx}}\int_{\mathbb{R}^{nx}}\prod_{k=1}^{nx}e^{-\theta_k(z_k-z'_k)^2-\theta_k(z_k-s_k^i)^2-\theta_k(z'_k-s_k^j)^2}p_{x_k+\chi_{x_k}}(z_k)p_{x_k+\chi'_{x_k}}(z'_k)dz_kdz'_k\\
&=\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}\prod_{k=1}^{nx}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-\theta_k(z_k-z'_k)^2-\theta_k(z_k-s_k^i)^2-\theta_k(z'_k-s_k^j)^2}p_{x_k+\chi_{x_k}}(z_k)p_{x_k+\chi'_{x_k}}(z'_k)dz_kdz'_k\\
&=\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}\prod_{k=1}^{nx}J_{ijk},
\end{aligned}
\tag{B.8}
$$

avec :

$$
\begin{aligned}
J_{ijk}&=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-\theta_k(z_k-z'_k)^2-\theta_k(z_k-s_k^i)^2-\theta_k(z'_k-s_k^j)^2}p_{x_k+\chi_{x_k}}(z_k)p_{x_k+\chi'_{x_k}}(z'_k)dz_kdz'_k\\
&=\frac{1}{2\pi\sigma_k^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-\theta_k(z_k-z'_k)^2-\theta_k(z_k-s_k^i)^2-\theta_k(z'_k-s_k^j)^2}e^{-\frac{1}{2}\left(\frac{z_k-\mu_k}{\sigma_k}\right)^2}e^{-\frac{1}{2}\left(\frac{z'_k-\mu_k}{\sigma_k}\right)^2}dz_kdz'_k\\
&=\frac{1}{\sqrt{(2\theta_k\sigma_k^2+1)(6\theta_k\sigma_k^2+1)}}e^{-\theta_k\frac{4\theta_k^2\sigma_k^4(s_k^i-s_k^j)^2+(6\theta_k\sigma_k^2+1)((s_k^i-\mu_k)^2+(s_k^j-\mu_k)^2)}{(2\theta_k\sigma_k^2+1)(6\theta_k\sigma_k^2+1)}},
\end{aligned}
\tag{B.9}
$$

$$
\begin{aligned}
&\mathrm{E}_\chi[\mathrm{E}_{\chi'}[(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))^2]]\\
&=\int_{\mathbb{R}^{nx}}\int_{\mathbb{R}^{nx}}\left(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{z},\boldsymbol{s}^i)R(\boldsymbol{z}',\boldsymbol{s}^j)\right)^2p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z}')d\boldsymbol{z}d\boldsymbol{z}'\\
&=\int_{\mathbb{R}^{nx}}\int_{\mathbb{R}^{nx}}\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s}C_{ij}^{-1}C_{kl}^{-1}R(\boldsymbol{z},\boldsymbol{s}^i)R(\boldsymbol{z}',\boldsymbol{s}^j)R(\boldsymbol{z},\boldsymbol{s}^k)R(\boldsymbol{z}',\boldsymbol{s}^l)p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z}')d\boldsymbol{z}d\boldsymbol{z}'\\
&=\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s}C_{ij}^{-1}C_{kl}^{-1}\int_{\mathbb{R}^{nx}}R(\boldsymbol{z},\boldsymbol{s}^i)R(\boldsymbol{z},\boldsymbol{s}^k)p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})d\boldsymbol{z}\int_{\mathbb{R}^{nx}}R(\boldsymbol{z}',\boldsymbol{s}^j)R(\boldsymbol{z}',\boldsymbol{s}^l)p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z}')d\boldsymbol{z}'\\
&=\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s}C_{ij}^{-1}C_{kl}^{-1}\prod_{m=1}^{nx}\int_{-\infty}^{\infty}e^{-\theta_m(z_m-s_m^i)^2-\theta_m(z_m-s_m^k)^2}p_{x_m+\chi_{x_m}}(z_m)dz_m\\
&\quad\times\int_{-\infty}^{\infty}e^{-\theta_m(z'_m-s_m^j)^2-\theta_m(z'_m-s_m^l)^2}p_{x_m+\chi'_{x_m}}(z'_m)dz'_m\\
&=\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s}C_{ij}^{-1}C_{kl}^{-1}\prod_{m=1}^{nx}I_{ikm}I_{jlm}.
\end{aligned}
\tag{B.10}
$$

L'équation (B.6) peut donc s'écrire :

$$\begin{aligned}\mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{Cov}^2(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] &= \sigma_K^4 \prod_{j=1}^{n_x} \frac{1}{\sqrt{8\sigma_j^2\theta_j+1}} \\ &\quad - 2\sigma_K^6 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} \prod_{k=1}^{n_x} J_{ijk} \\ &\quad + \sigma_K^8 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s} C_{ij}^{-1}C_{kl}^{-1} \prod_{m=1}^{n_x} I_{ikm}I_{jlm}.\end{aligned} \tag{B.11}$$

Pour le deuxième terme de l'équation (B.5), on a :

$$\begin{aligned}&\mathrm{E}_{\chi}[\mathrm{E}^2_{\chi'}[\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\ &= \mathrm{E}_{\chi}[\mathrm{E}^2_{\chi'}[\sigma_K^2 R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x) - \sigma_K^4 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)]] \\ &= \mathrm{E}_{\chi}[(\sigma_K^2\,\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)] - \sigma_K^4 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)\,\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)])^2] \\ &= \mathrm{E}_{\chi}[\sigma_K^4\,\mathrm{E}^2_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)] \\ &\quad - 2\sigma_K^6\,\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)] \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)\,\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)] \\ &\quad + \sigma_K^8 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s} C_{ij}^{-1}C_{kl}^{-1} R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^k) \\ &\quad \times \mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)]\,\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^l)]] \\ &= \sigma_K^4\,\mathrm{E}_{\chi}[\mathrm{E}^2_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)]] \\ &\quad - 2\sigma_K^6 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1}\,\mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)]]\,\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)] \\ &\quad + \sigma_K^8 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s} C_{ij}^{-1}C_{kl}^{-1}\,\mathrm{E}_{\chi}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^k)] \\ &\quad \times \mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j)]\,\mathrm{E}_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^l)] \\ &= \sigma_K^4\,\mathrm{E}_{\chi}[\mathrm{E}^2_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)]] - 2\sigma_K^6 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} C_{ij}^{-1} \prod_{k=1}^{n_x} J_{ik}I_{jk} \\ &\quad + \sigma_K^8 \sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s} C_{ij}^{-1}C_{kl}^{-1} \prod_{m=1}^{n_x} I_{ikm}I_{jm}I_{lm},\end{aligned} \tag{B.12}$$

avec :

$$\begin{aligned}J_{ik} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-\theta_k(z_k-z'_k)^2-\theta_k(z_k-s_k^i)^2} p_{x_k+\chi_{x_k}}(z_k)\,p_{x_k+\chi'_{x_k}}(z'_k)\,dz_k\,dz'_k \\ &= \frac{1}{\sqrt{4\theta_k^2\sigma_k^4+6\theta_k\sigma_k^2+1}} e^{-\theta_k\frac{(s_k^i-\mu_k)^2(4\theta_k\sigma_k^2+1)}{4\theta_k^2\sigma_k^4+6\theta_k\sigma_k^2+1}},\end{aligned} \tag{B.13}$$

et :

$$
\begin{aligned}
\mathrm{E}_{\chi}[\mathrm{E}^2_{\chi'}[R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)]] &= \int_{\mathbb{R}^{n_x}} \left(\int_{\mathbb{R}^{n_x}} R(\boldsymbol{z},\boldsymbol{z}')p_{\boldsymbol{x}+\boldsymbol{\chi}'_x}(\boldsymbol{z}')d\boldsymbol{z}'\right)^2 p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})d\boldsymbol{z} \\
&= \int_{\mathbb{R}^{n_x}} \prod_{i=1}^{n_x} \left(\int_{-\infty}^{\infty} e^{-\theta_i(z_i-z'_i)^2} p_{x_i+\chi'_{x_i}}(z'_i)dz'_i\right)^2 p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})d\boldsymbol{z} \\
&= \int_{\mathbb{R}^{n_x}} \prod_{i=1}^{n_x} \left(\frac{e^{-\frac{\theta_i(z_i-\mu_i)^2}{2\sigma_i^2\theta_i+1}}}{\sqrt{2\sigma_i^2\theta_i+1}}\right)^2 p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})d\boldsymbol{z}, \text{ d'après le calcul de } I_{ij} \\
&= \int_{\mathbb{R}^{n_x}} \prod_{i=1}^{n_x} \frac{e^{-\frac{2\theta_i(z_i-\mu_i)^2}{2\sigma_i^2\theta_i+1}}}{2\sigma_i^2\theta_i+1} p_{\boldsymbol{x}+\boldsymbol{\chi}_x}(\boldsymbol{z})d\boldsymbol{z} \\
&= \prod_{i=1}^{n_x} \frac{1}{2\sigma_i^2\theta_i+1} \int_{-\infty}^{\infty} e^{-\frac{2\theta_i(z_i-\mu_i)^2}{2\sigma_i^2\theta_i+1}} p_{x_i+\chi_{x_i}}(z_i)dz_i \\
&= \prod_{i=1}^{n_x} \frac{1}{2\sigma_i^2\theta_i+1} \sqrt{\frac{2\sigma_i^2\theta_i+1}{6\sigma_i^2\theta_i+1}} \\
&= \prod_{i=1}^{n_x} \frac{1}{\sqrt{(2\sigma_i^2\theta_i+1)(6\sigma_i^2\theta_i+1)}}.
\end{aligned}
\tag{B.14}
$$

Pour le quatrième et dernier terme de l'équation (B.5), on a :

$$
\begin{aligned}
&\mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[(\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)] - \mathrm{E}[\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]])(\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)] - \mathrm{E}[\mathrm{E}_{\chi'}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]]) \\
&\quad \times \mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&= \mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)) \\
&\quad - \mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{E}[\mathrm{E}_{\chi'}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)) \\
&\quad - \mathrm{E}[\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\,\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)) \\
&\quad + \mathrm{E}[\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\,\mathrm{E}[\mathrm{E}_{\chi'}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&= \mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&\quad - \mathrm{E}[\mathrm{E}_{\chi'}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]]\,\mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&\quad - \mathrm{E}[\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\,\mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&\quad + \mathrm{E}[\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\,\mathrm{E}[\mathrm{E}_{\chi'}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]]\,\mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&= \mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&\quad - 2\,\mathrm{E}[\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\,\mathrm{E}_{\chi}[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]] \\
&\quad + \mathrm{E}^2[\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\,\mathrm{Var}(\mathrm{E}_{\chi}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]),
\end{aligned}
\tag{B.15}
$$

Il reste à calculer le premier terme de cette équation, ainsi que la deuxième partie du

deuxième terme (le reste étant déjà connu). On a ainsi :

$$
\begin{aligned}
&\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]]\\
&=\mathrm{E}_\chi[\mathrm{E}_{\chi'}[(\hat{\mu}+\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x},\boldsymbol{s}^i))(\hat{\mu}+\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x}',\boldsymbol{s}^i))\\
&\quad\times(\sigma_K^2R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)-\sigma_K^4\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))]]\\
&=\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}^2\sigma_K^2R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)]]\\
&\quad-\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}^2\sigma_K^4\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))]]\\
&\quad+\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}\sigma_K^2R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x},\boldsymbol{s}^i)]]\\
&\quad-\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}\sigma_K^4(\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x},\boldsymbol{s}^i))(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))]]\\
&\quad+\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}\sigma_K^2R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x}',\boldsymbol{s}^i)]]\\
&\quad-\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}\sigma_K^4(\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x}',\boldsymbol{s}^i))(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))]]\\
&\quad+\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\sigma_K^2R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)(\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x},\boldsymbol{s}^i))(\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x}',\boldsymbol{s}^i))]]\\
&\quad-\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\sigma_K^4(\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x},\boldsymbol{s}^i))(\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x}',\boldsymbol{s}^i))\\
&\quad\times(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{s}^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,\boldsymbol{s}^j))]]\\
&=\hat{\mu}^2\sigma_K^2\prod_{i=1}^{n_x}J_i\\
&\quad-\hat{\mu}^2\sigma_K^4\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}\prod_{k=1}^{n_x}I_{ik}I_{jk}\\
&\quad+2\hat{\mu}\sigma_K^4\sum_{i=1}^{n_s}\gamma_i\prod_{j=1}^{n_x}J_{ij}\\
&\quad-2\hat{\mu}\sigma_K^6\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}C_{ij}^{-1}\gamma_k\prod_{l=1}^{n_x}J_{ijkl}\\
&\quad+\sigma_K^2\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\gamma_i\gamma_j\prod_{k=1}^{n_x}J_{ijk}\\
&\quad-\sigma_K^4\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}\sum_{l=1}^{n_s}\gamma_i\gamma_jC_{kl}^{-1}\prod_{m=1}^{n_x}J_{ijklm},
\end{aligned}
\tag{B.16}
$$

avec :

$$J_{ijkl} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-\theta_l(z_l-s_l^i)^2-\theta_l(z'_l-s_l^j)^2-\theta_l(z_l-s_l^k)^2} p_{x_l+\chi_{x_l}}(z_l)p_{x_l+\chi'_{x_l}}(z'_l)dz_l dz'_l$$
$$= \frac{e^{-\theta_l \frac{4\theta_l^2\sigma_l^4(s_l^i-s_l^k)^2+(s_l^i-\mu_l)^2+(s_l^k-\mu_l)^2+(4\theta_l\sigma_l^2+1)(s_l^j-\mu_l)^2+4\theta_l\sigma_l^2((s_l^k)^2+(s_l^i)^2-s_l^is_l^k-s_l^i\mu_l-s_l^k\mu_l+\mu_l^2)}{(2\theta_l\sigma_l^2+1)(4\theta_l\sigma_l^2+1)}}}{\sqrt{(2\theta_l\sigma_l^2+1)(4\theta_l\sigma_l^2+1)}}, \tag{B.17}$$

et :

$$J_{ijklm} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-\theta_m(z_m-s_m^i)^2-\theta_m(z'_m-s_m^j)^2-\theta_m(z_m-s_m^k)^2-\theta_m(z'_m-s_m^l)^2}$$
$$\times\, p_{x_m+\chi_{x_m}}(z_m)p_{x_m+\chi'_{x_m}}(z'_m)dz_m dz'_m$$
$$= \frac{e^{-\theta_m \frac{2\theta_m\sigma_m^2((s_m^j)^2+(s_m^l)^2+(s_m^i)^2+(s_m^k)^2-2s_m^is_m^k-2s_m^js_m^l)+(s_m^i-\mu_m)^2+(s_m^j-\mu_m)^2+(s_m^k-\mu_m)^2+(s_m^l-\mu_m)^2}{4\theta_m\sigma_m^2+1}}}{4\theta_m\sigma_m^2+1}, \tag{B.18}$$

puis :

$$\begin{aligned}
&\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x),\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]]\\
&= \mathrm{E}_\chi[\mathrm{E}_{\chi'}[(\hat{\mu}+\sum_{i=1}^{n_s}\gamma_i\sigma_K^2R(\boldsymbol{x},\boldsymbol{s}^i))\\
&\quad\times(\sigma_K^2R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)-\sigma_K^4\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,s^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,s^j))]]\\
&= \mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}\sigma_K^2R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)]]-\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\hat{\mu}\sigma_K^4\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,s^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,s^j)]]\\
&\quad+\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\sigma_K^4R(\boldsymbol{x}+\boldsymbol{\chi}_x,\boldsymbol{x}+\boldsymbol{\chi}'_x)\sum_{i=1}^{n_s}\gamma_iR(\boldsymbol{x},\boldsymbol{s}^i)]]\\
&\quad-\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\sigma_K^6(\sum_{i=1}^{n_s}\gamma_iR(\boldsymbol{x},\boldsymbol{s}^i))(\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}R(\boldsymbol{x}+\boldsymbol{\chi}_x,s^i)R(\boldsymbol{x}+\boldsymbol{\chi}'_x,s^j))]]\\
&= \hat{\mu}\sigma_K^2\prod_{i=1}^{n_x}J_i-\hat{\mu}\sigma_K^4\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}C_{ij}^{-1}\prod_{i=1}^{n_x}I_{ik}I_{jk}+\sigma_K^4\sum_{i=1}^{n_s}\gamma_i\prod_{j=1}^{n_x}J_{ij}-\sigma_K^6\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\sum_{k=1}^{n_s}C_{ij}^{-1}\gamma_k\prod_{l=1}^{n_x}J_{ijkl}.
\end{aligned} \tag{B.19}$$

On peut ainsi calculer la valeur $\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))$ de la variance de la mesure de variance. Elle fait apparaître des sommes quadruples sur le nombre n_s d'échantillons du modèle, et a donc une complexité en n_s^4.

Ces expressions analytiques ont été validées expérimentalement en les comparant à des estimations de Monte-Carlo (en faisant des tirages pour évaluer la robustesse sur des réalisations du processus gaussien $\hat{F}$ comme présenté dans la section 5.8.5).

Mesure de l'écart-type

Pour calculer la mesure du quantile approché, on a besoin de connaître l'espérance et la variance de la mesure de l'écart-type σ_χ. Celles-ci peuvent se déduire facilement à

partir des données de la mesure de variance, si on suppose que l'écart-type suit une loi gaussienne (comme expliqué dans [ALC06]).

Pour une variable aléatoire χ qui suit une loi gaussienne $\chi \sim \mathcal{N}(\mu, \sigma^2)$, le moment d'ordre n s'écrit : $\mathrm{E}[\chi^n] = (n-1)\sigma^2\,\mathrm{E}[\chi^{n-2}] + \mu\,\mathrm{E}[\chi^{n-1}]$.

On a donc par exemple :

$$\mathrm{E}[\chi^2] = \sigma^2 + \mu^2, \tag{B.20}$$

et :

$$\begin{aligned}
\mathrm{Var}(\chi^2) &= \mathrm{E}[\chi^4] - \mathrm{E}^2[\chi^2] \\
&= 3\sigma^2\,\mathrm{E}[\chi^2] + \mu\,\mathrm{E}[\chi^3] - (\sigma^2+\mu^2)^2 \\
&= 3\sigma^2(\sigma^2+\mu^2) + \mu(2\sigma^2\,\mathrm{E}[\chi] + \mu\,\mathrm{E}[\chi^2]) - \sigma^4 - 2\sigma^2\mu^2 - \mu^4 \\
&= 2\sigma^4 + \sigma^2\mu^2 + \mu(2\sigma^2\mu + \mu(\sigma^2+\mu^2)) - \mu^4 \\
&= 2\sigma^4 + \sigma^2\mu^2 + 2\sigma^2\mu^2 + \mu^2\sigma^2 + \mu^4 - \mu^4 \\
&= 2\sigma^4 + 4\sigma^2\mu^2.
\end{aligned} \tag{B.21}$$

Comme $\mathrm{Var}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \sigma^2_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))$, alors en supposant que $\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))$ suit une loi gaussienne on a :

$$\begin{aligned}
\mathrm{E}[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] &= \mathrm{E}[\sigma^2_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] \\
&= \mathrm{Var}(\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)) + \mathrm{E}^2[\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))],
\end{aligned} \tag{B.22}$$

et :

$$\begin{aligned}
\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) &= \mathrm{Var}(\sigma^2_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) \\
&= 2\,\mathrm{Var}^2(\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) \\
&\quad + 4\,\mathrm{Var}(\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))\,\mathrm{E}^2[\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))].
\end{aligned} \tag{B.23}$$

On peut ainsi en déduire l'espérance et la variance de la mesure de l'écart-type :

$$\begin{aligned}
\mathrm{E}[\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] &= \sqrt[4]{\mathrm{E}^2[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] - \frac{\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))}{2}}, \\
\mathrm{Var}(\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) &= \mathrm{E}[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] \\
&\quad - \sqrt{\mathrm{E}^2[\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] - \frac{\mathrm{Var}(\mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))}{2}}.
\end{aligned} \tag{B.24}$$

Annexe C

Mesure du quantile approché sur un modèle de krigeage

Cette annexe présente le calcul de la mesure du quantile approché $\hat{\mathrm{Q}}_\chi$ sur un modèle de krigeage ordinaire $\hat{f}$ doté d'une fonction de corrélation R gaussienne, vis-à-vis d'incertitudes gaussiennes définies sur ses variables d'entrées $\boldsymbol{x}$ par l'intermédiaire de variables aléatoires indépendantes $\boldsymbol{\chi_x}$. Nous allons exprimer l'espérance $\mathrm{E}[\hat{\mathrm{Q}}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))]$ et la variance $\mathrm{Var}(\hat{\mathrm{Q}}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))$ de cette mesure de robustesse par rapport au processus stochastique $\hat{F}$ associé au modèle $\hat{f}$, d'après les résultats présentés dans [ALC06]. Les notations utilisées sont les mêmes que pour les annexes précédentes.

Le quantile approché est défini comme une agrégation de l'espérance et de l'écart-type (voir l'équation (7.3)) :

$$\hat{\mathrm{Q}}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)) = \mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)] + \alpha\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)),\ \text{avec } \alpha \in \mathbb{R}. \tag{C.1}$$

Son espérance vaut ainsi :

$$\mathrm{E}[\hat{\mathrm{Q}}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))] = \mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]] + \alpha\,\mathrm{E}[\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))], \tag{C.2}$$

et sa variance :

$$\begin{aligned}\mathrm{Var}(\hat{\mathrm{Q}}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) = \mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]) + \alpha^2\,\mathrm{Var}(\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))\\ + 2\alpha\,\mathrm{Cov}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)], \sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))).\end{aligned} \tag{C.3}$$

Il reste à calculer le dernier terme de covariance entre les mesures d'espérance et d'écart-type. L'article [ALC06] indique qu'elle peut être obtenue à partir de la covariance entre l'espérance et la variance :

$$\mathrm{Cov}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)], \sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) = \frac{\mathrm{Cov}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)], \mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)))}{2\,\mathrm{E}[\sigma_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))]}, \tag{C.4}$$

qui vaut :

$$\begin{aligned}\mathrm{Cov}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)], \mathrm{Var}_\chi(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x))) = 2\,\mathrm{E}_\chi[\mathrm{E}_{\chi'}[\mathrm{E}[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]\,\mathrm{Cov}(\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x), \hat{F}(\boldsymbol{x}+\boldsymbol{\chi}'_x))]]\\ - 2\,\mathrm{E}[\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]]\,\mathrm{Var}(\mathrm{E}_\chi[\hat{F}(\boldsymbol{x}+\boldsymbol{\chi}_x)]).\end{aligned} \tag{C.5}$$

Toutes ces formules ont déjà été exprimées précédemment. La complexité de l'estimation du quantile approché (aussi bien pour obtenir son espérance que sa variance) est relative au nombre d'échantillons du modèles à la puissance quatre.

Bibliographie

[ABB+99] E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, Jack J. Dongarra, J. Du Croz, S. Hammarling, A. Greenbaum, A. McKenney et D. Sorensen : *LAPACK Users' guide (third ed.)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.

[AL02] Natalia M. Alexandrov et Robert M. Lewis : Analytical and Computational Aspects of Collaborative Optimization for Multidisciplinary Design. *AIAA Journal*, 40(2):301–309, février 2002.

[ALC06] Daniel W. Apley, Jun Liu et Wei Chen : Understanding the Effects of Model Uncertainty in Robust Design With Computer Experiments. *Journal of Mechanical Design*, 128(4):945–958, 2006.

[AR04] Harish Agarwal et John Renaud : Reliability based design optimization using response surfaces in application to multidisciplinary systems. *Engineering Optimization*, 36(3):291–311, juin 2004.

[ASCM06] Janet K. Allen, Carolyn Seepersad, HaeJin Choi et Farrokh Mistree : Robust Design for Multiscale and Multidisciplinary Applications. *Journal of Mechanical Design*, 128(4):832–843, 2006.

[AW10] Hervé Abdi et Lynne J. Williams : Principal component analysis. *WIREs Comp Stat*, 2(4):433–459, 2010.

[Bar98] Peter Bartholomew : The role of MDO within aerospace design and progress towards an MDO capability. *In 7th AIAA/USAF/NASA/ ISSMO Symposium on Multidisciplinary Analysis and Optimization*. AIAA, 1998.

[Bar08] Séverine Barbosa : *Étude expérimentale de la dynamique de combustion d'un injecteur multipoint étagé de turbine à gaz*. Thèse de doctorat, Laboratoire d'Énergétique Moléculaire et Macroscopique, Combustion (EM2C) du CNRS et de l'ECP, décembre 2008.

[BBG+11] Dimitri Bettebghor, Nathalie Bartoli, Stéphane Grihon, Joseph Morlier et Manuel Samuelides : Surrogate modeling approximation using a mixture of experts based on EM joint estimation. *Structural and Multidisciplinary Optimization*, 43(2):243–259, février 2011.

[BDMG08] N. Bertier, F. Dupoirieux, L. Matuszewski et C. Guin : Simulation numérique d'une chambre de combustion multipoint. *In 2nd Colloque INCA*, octobre 2008.

[BKGB10] A. C. Berbecea, S. Kreuawan, F. Gillon et P. Brochet : A Parallel Multiobjective Efficient Global Optimization : The Finite Element Method

in Optimal Design and Model Development. *IEEE Transactions on Magnetics*, 46(8):2868–2871, août 2010.

[BKHU+12a] Vincent Baudoui, Patricia Klotz, Jean-Baptiste Hiriart-Urruty, Sophie Jan et Renaud Lecourt : Surrogate-based robust optimization method applied to injection system design under uncertainty. 2012.

[BKHU+12b] Vincent Baudoui, Patricia Klotz, Jean-Baptiste Hiriart-Urruty, Sophie Jan et Franck Morel : LOcal Uncertainty Processing (LOUP) method for multidisciplinary robust design optimization. *Structural and Multidisciplinary Optimization*, 2012.

[BL88] D. S. Broomhead et David Lowe : Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks. *Complex Systems*, 2:321–355, mars 1988.

[BS07] H. Beyer et B. Sendhoff : Robust optimization – A comprehensive survey. *Computer Methods in Applied Mechanics and Engineering*, 196(33-34):3190–3218, juillet 2007.

[BST04] Stuart J. Bates, Johann Sienz et Vassili V. Toropov : Formulation of the Optimal Latin Hypercube Design of Experiments Using a Permutation Genetic Algorithm. *In 45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, avril 2004.

[Caf98] Russel E. Caflisch : Monte Carlo and quasi-Monte Carlo methods. *Acta Numerica*, 7(-1):1–49, 1998.

[CAR+05] Hae J. Choi, Janet K. Allen, David Rosen, David L. McDowell et Farrokh Mistree : An Inductive Design Exploration Method for the Integrated Design of Multi-Scale Materials and Products. *ASME Conference Proceedings*, 2005(4739X):859–870, 2005.

[CB93] W. C. Carpenter et Barthelemy : A comparison of polynomial approximations and artificial neural nets as response surfaces. *Structural and Multidisciplinary Optimization*, 5(3):166–174, septembre 1993.

[CCD+05] P. Chevalier, B. Courbet, D. Dutoya, P. Klotz, E. Ruiz, J. Troyes et P. Villedieu : CEDRE : development and validation of a multiphysic computational software. *In European Conference for Aerospace Sciences (EUCASS)*, 2005.

[CDF+94] Evin J. Cramer, J. E. Dennis, Paul D. Frank, Robert M. Lewis et Gregory R. Shubin : Problem Formulation for Multidisciplinary Optimization. *SIAM Journal on Optimization*, 4(4):754+, 1994.

[Che00] Sophie Q. Chen : *Comparing Probabilistic and Fuzzy Set Approaches for Design in the Presence of Uncertainty*. Thèse de doctorat, Virginia Polytechnic Institute and State University, août 2000.

[Cle09] Joël Clement : *Optimisation multidisciplinaire : étude théorique et application à la conception des avions en phase d'avant projet*. Thèse de doctorat, Institut Supérieur de l'Aéronautique et de l'Espace, juin 2009.

[CMMY91] Carla Currin, Toby Mitchell, Max Morris et Don Ylvisaker : Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments. *Journal of the American Statistical Association*, 86(416):953–963, 1991.

[CS03] Yann Collette et Patrick Siarry : *Multiobjective Optimization : Principles and Case Studies*. Springer, 2003.

[Cyb89] G. Cybenko : Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, décembre 1989.

[DA94] Kalyanmoy Deb et Ram B. Agrawal : Simulated Binary Crossover for Continuous Search Space. Rapport technique, Departement of Mechanical Enginering, Indian Institute of Technology, Kanpur, India, 1994.

[Das00] Indraneel Das : Robustness optimization for constrained nonlinear programming problems. *Engineering Optimization*, 32(5):585–618, 2000.

[DC99] Xiaoping Du et Wei Chen : Towards a Better Understanding of Modeling Feasibility Robustness in Engineering Design. *In ASME Journal of Mechanical Design*, volume 122, pages 385–394, 1999.

[DC05] Xiaoping Du et Wei Chen : Collaborative Reliability Analysis under the Framework of Multidisciplinary Systems Design. *Optimization and Engineering*, 6(1):63–84, mars 2005.

[DD98] Indraneel Das et J. E. Dennis : Normal-Boundary Intersection : A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems. *SIAM Journal on Optimization*, 8(3):631+, 1998.

[DG96] Kalyanmoy Deb et Mayank Goyal : A Combined Genetic Adaptive Search (GeneAS) for Engineering Design. *Computer Science and Informatics*, 26: 30–45, 1996.

[DG06] Kalyanmoy Deb et Himanshu Gupta : Introducing robustness in multiobjective optimization. *Evolutionary Computation*, 14(4):463–494, décembre 2006.

[DKM09] G. Dellino, J. P. C. Kleijnen et C. Meloni : Robust simulation-optimization using metamodels. *In Proceedings of the 2009 Winter Simulation Conference (WSC)*, pages 540–550. IEEE, décembre 2009.

[DLP+08] Didier Dubois, M. Asuncin Lubiano, Henri Prade, Mara N. Gil, Przemyslaw Grzegorzewski et Olgierd Hryniewicz : *Soft Methods for Handling Variability and Imprecision*. Springer, 2008.

[DLR77] A. P. Dempster, N. M. Laird et D. B. Rubin : Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[DPAM02] K. Deb, A. Pratap, S. Agarwal et T. Meyarivan : A fast and elitist multiobjective genetic algorithm : NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, avril 2002.

[DS78] L. C. W. Dixon et G. P. Szego : The global optimization problem : an introduction. *Towards Global Optimization*, 2:1–15, 1978.

[DSC04] Xiaoping Du, Agus Sudjianto et Wei Chen : An Integrated Framework for Optimization Under Uncertainty Using Inverse Reliability Strategy. *Journal of Mechanical Design*, 126(4):562–570, 2004.

[DSM+04] Gérard Dreyfus, Manuel Samuelides, Jean-Marc Martinez, Mirta B. Gordon, Fouad Badran, Sylvie Thiria et Laurent Hérault : *Réseaux de neurones : méthodologies et applications*. Eyrolles, avril 2004.

[DTLZ01] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns et Eckart Zitzler : Scalable Test Problems for Evolutionary Multi-Objective Optimization. Rapport technique, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH Zurich), 2001.

[DWC00] Xiaoping Du, Yijun Wang et Wei Chen : Methods for robust multidisciplinary design. *In AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit, 41st*, avril 2000.

[EDK08] M. Emmerich, A. H. Deutz et J. W. Klinkenberg : The computation of the expected improvement in dominated hypervolume of Pareto front approximations. Rapport technique, Leiden University, septembre 2008.

[Efr81] Bradley Efron : Nonparametric estimates of standard error : The jackknife, the bootstrap and other methods. *Biometrika*, 68(3):589–599, décembre 1981.

[EGN06] M. T. M. Emmerich, K. C. Giannakoglou et B. Naujoks : Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439, août 2006.

[EGO+02] Michael Emmerich, Alexios Giotis, Mutlu Özdemir, Thomas Bäck et Kyriakos Giannakoglou : Metamodel-Assisted Evolution Strategies. *In* Juan Guervós, Panagiotis Adamidis, Hans-Georg Beyer, Hans-Paul Schwefel et José-Luis Fernández-Villacañas, éditeurs : *Parallel Problem Solving from Nature — PPSN VII*, volume 2439 de *Lecture Notes in Computer Science*, chapitre 35, pages 361–370. Springer Berlin / Heidelberg, Berlin, Heidelberg, octobre 2002.

[EL07] Daniel Egloff et Markus Leippold : Quantile Estimation with Adaptive Importance Sampling. *Social Science Research Network Working Paper Series*, juillet 2007.

[EN04] Michael Emmerich et Boris Naujoks : Metamodel Assisted Multiobjective Optimisation Strategies and their Application in Airfoil Design. *Proc. of ACDM VI*, 2004.

[EU10] Tohid Erfani et Sergei V. Utyuzhnikov : Directed search domain : a method for even generation of the Pareto frontier in multiobjective optimization. *Engineering Optimization*, 43(5):467–484, novembre 2010.

[FCLB11] Rajan Filomeno Coelho, Jérémy Lebon et Philippe Bouillard : Hierarchical stochastic metamodels based on moving least squares and polynomial chaos expansion. *Structural and Multidisciplinary Optimization*, 43(5):707–729, mai 2011.

[FF95] Carlos M. Fonseca et Peter J. Fleming : An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3(1):1–16, mars 1995.

[Fri91] Jerome H. Friedman : Multivariate Adaptive Regression Splines. *The Annals of Statistics*, 19(1):1–67, 1991.

[Fun89] K. Funahashi : On the approximate realization of continuous mappings by neural networks. *Neural Netw.*, 2(3):183–192, 1989.

[GBM04] A. Giassi, F. Bennis et J. J. Maisonneuve : Multidisciplinary design optimisation and robust design approaches applied to concurrent design. *Structural and Multidisciplinary Optimization*, 28(5):356–371, novembre 2004.

[GE00] Anthony A. Giunta et Michael S. Eldred : Implementation Of A Trust Region Model Management Strategy In The Dakota Optimization Toolkit. *In Proceedings of 8th AIAA/USAF/ NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, septembre 2000.

[Gir04] Agathe Girard : *Approximate methods for propagation of uncertainty with gaussian process models*. Thèse de doctorat, University of Glasgow, 2004.

[GLRC10] David Ginsbourger, Rodolphe Le Riche et Laurent Carraro : Kriging Is Well-Suited to Parallelize Optimization. *In* Yoel Tenne et Chi-Keong Goh, éditeurs : *Computational Intelligence in Expensive Optimization Problems*, volume 2 de *Springer series in Evolutionary Learning and Optimization*, chapitre 6, pages 131–162. Springer, Berlin, Heidelberg, mars 2010.

[GRB+00] X. Gu, J. E. Renaud, S. M. Batill, R. M. Brach et A. S. Budhiraja : Worst case propagated uncertainty of multidisciplinary systems in robust design optimization. *Structural and Multidisciplinary Optimization*, 20(3): 190–213, novembre 2000.

[GRMS06] Shawn Gano, John Renaud, Jay Martin et Timothy Simpson : Update strategies for kriging models used in variable fidelity optimization. *Structural and Multidisciplinary Optimization*, 32(4):287–298, octobre 2006.

[GRP06] Xiaoyu S. Gu, John E. Renaud et Charles L. Penninger : Implicit Uncertainty Propagation for Robust Collaborative Optimization. *Journal of Mechanical Design*, 128(4):1001–1013, 2006.

[GW98] Anthony A. Giunta et Layne T. Watson : A Comparison Of Approximation Modeling Techniques : Polynomial Versus Interpolating Models. *In Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, septembre 1998.

[HHBW06] S. Huband, P. Hingston, L. Barone et L. While : A review of multiobjective test problems and a scalable test problem toolkit. *Evolutionary Computation, IEEE Transactions on*, 10(5):477–506, 2006.

[HM00] Achintya Haldar et Sankaran Mahadevan : *Reliability Assessment Using Stochastic Finite Element Analysis*. John Wiley & Sons, New York, juin 2000.

[HS07] G. I. Hawe et J. K. Sykulski : An Enhanced Probability of Improvement Utility Function for Locating Pareto Optimal Solutions. *In 16th Conference on the Computation of Electromagnetic Fields COMPUMAG*, juin 2007.

[HSW89] Kurt Hornik, Maxwell Stinchcombe et Halbert White : Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, janvier 1989.

[HTF09] Trevor Hastie, Robert Tibshirani et Jerome Friedman : *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, 2 édition, 2009.

[HW79] J. A. Hartigan et M. A. Wong : Algorithm AS 136 : A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[Jae09] Félix Jaegle : *Large eddy simulation of evaporating sprays in complex geometries using Eulerian and Lagrangian methods.* Thèse de doctorat, Institut National Polytechnique de Toulouse, décembre 2009.

[JCS01] R. Jin, W. Chen et T. W. Simpson : Comparative studies of metamodelling techniques under multiple modelling criteria. *Structural and Multidisciplinary Optimization*, 23(1):1–13, décembre 2001.

[JCS02] Ruichen Jin, Wei Chen et Agus Sudjianto : On Sequential Sampling for Global Metamodeling in Engineering Design. *ASME Conference Proceedings*, 2002(36223):539–548, 2002.

[JDC03] R. Jin, X. Du et W. Chen : The use of metamodeling techniques for optimization under uncertainty. *Structural and Multidisciplinary Optimization*, 25(2):99–116, juillet 2003.

[Jin04] Ruichen Jin : *Enhancements of metamodeling techniques in engineering design.* Thèse de doctorat, University of Illinois at Chicago, 2004.

[JLR12] Janis Janusevskis et Rodolphe Le Riche : Simultaneous kriging-based estimation and optimization of mean response. *Journal of Global Optimization*, pages 1–24, janvier 2012.

[JO05] Shinkyu Jeong et S. Obayashi : Efficient Global Optimization (EGO) for Multi-Objective Problem and Data Mining. *In Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 3, pages 2138–2145, 2005.

[Jon01] Donald R. Jones : A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of Global Optimization*, 21(4):345–383, décembre 2001.

[JOS01] Yaochu Jin, Markus Olhofer et Bernhard Sendhoff : Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization : Why Does It Work and How ? *In* Lee Spector, Erik D. Goodman, Annie Wu, W. B. Langdon, Hans M. Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon et Edmund Burke, éditeurs : *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 1042–1049, San Francisco, California, 2001. Morgan Kaufmann Publishers.

[JSW98] Donald R. Jones, Matthias Schonlau et William J. Welch : Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13(4):455–492, décembre 1998.

[Jur07] Florian Jurecka : *Robust Design OptimizationBased on Metamodeling Techniques.* Thèse de doctorat, Technische Universität München, mars 2007.

[Jur09] Florian Jurecka : Automated metamodeling for efficient multi-disciplinary optimisation of complex automotive structures. *In 7th European LS-DYNA Conference*, 2009.

[KdW06] I. Kim et O. de Weck : Adaptive weighted sum method for multiobjective optimization : a new method for Pareto front generation. *Structural and Multidisciplinary Optimization*, 31(2):105–116, février 2006.

[Kea06] Andy Keane : Statistical Improvement Criteria for Use in Multiobjective Design Optimization. *AIAA Journal*, 44(4):879–891, avril 2006.

[KHL01] Monu Kalsi, Kurt Hacker et Kemper Lewis : A Comprehensive Robust Design Approach for Decision Trade-Offs in Complex Systems Design. *Journal of Mechanical Design*, 123(1):1–10, 2001.

[Kno06] J. Knowles : ParEGO : a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, février 2006.

[KO96] J. R. Koehler et A. B. Owen : Computer experiments. *Handbook of Statistics*, pages 261–308, 1996.

[Kri53] D. G. Krige : A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand. *OR*, 4(1), 1953.

[KS00] Jack P. C. Kleijnen et Robert G. Sargent : A methodology for fitting and validating metamodels in simulation. *European Journal of Operational Research*, 120(1):14–29, janvier 2000.

[KWGS02] Patrick N. Koch, Brett Wujek, Oleg Golovidov et Timothy W. Simpson : Facilitating probabilistic multidisciplinary design optimizationusing kriging approximation models. *In 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, septembre 2002.

[LA08] M. Li et S. Azarm : Multiobjective Collaborative Robust Optimization With Interval Uncertainty and Interdisciplinary Uncertainty Propagation. *Journal of Mechanical Design*, 130(8):081402+, 2008.

[LAFMD06] G. Li, S. Azarm, A. Farhang-Mehr et A. Diaz : Approximation of multiresponse deterministic engineering simulations : a dependent metamodeling approach. *Structural and Multidisciplinary Optimization*, 31(4):260–269, avril 2006.

[Lav08] Jacques Lavedrine : *Simulations aux grandes échelles de l'écoulement diphasique dans des modèles d'injecteur de moteurs aéronautiques.* Thèse de doctorat, Institut National Polytechnique de Toulouse, juin 2008.

[LCK+06] Huibin Liu, Wei Chen, Michael Kokkolaras, Panos Y. Papalambros et Harrison M. Kim : Probabilistic Analytical Target Cascading : A Moment Matching Formulation for Multilevel Optimization Under Uncertainty. *Journal of Mechanical Design*, 128(4):991–1000, 2006.

[Lev98] David Levin : The Approximation Power of Moving Least-Squares. *Mathematics of Computation*, 67:1517–1531, 1998.

[Li07] Mian Li : *Robust Optimization and Sensitivity Analysis with Multi-Objective Genetic Algorithms : Single- and Multi-Disciplinary Applications.* Thèse de doctorat, University of Maryland, novembre 2007.

[LRPG+09] Rodolphe Le Riche, Victor Picheny, David Ginsbourger, André Meyer et Nam-Ho Kim : Gears Design with Shape Uncertainties using Monte Carlo Simulations and Kriging. *In 50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, mai 2009.

[LSN04] Jeffrey S. Lehman, Thomas J. Santner et William I. Notz : Designing computer experiments to determine robust control variables. *Statistica Sinica*, 2004.

[LSS10] Ilya Loshchilov, Marc Schoenauer et Michèle Sebag : A mono surrogate for multiobjective optimization. *In Proceedings of the 12th annual conference on Genetic and evolutionary computation*, GECCO '10, pages 471–478, New York, NY, USA, 2010. ACM.

[Mat63] G. Matheron : *Principles of geostatistics*, volume 58. Society of Economic Geologists, 1963.

[MDGG08] Lionel Matuszewski, Francis Dupoirieux, Christian Guin et Frédéric Grisch : Numerical calculation of the NO formation in a multi-point combustion chamber and results of the associated validation experiments. *In 9th Onera-DLR Aerospace Symposium*, 2008.

[MGSH07] Yolanda Mack, Tushar Goel, Wei Shyy et Raphael Haftka : Surrogate Model-Based Optimization Framework : A Case Study in Aerospace Design. *In* Shengxiang Yang, Yew-Soon Ong et Yaochu Jin, éditeurs : *Evolutionary Computation in Dynamic and Uncertain Environments*, volume 51 de *Studies in Computational Intelligence*, chapitre 14, pages 323–342. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2007.

[Mic96] Zbigniew Michalewicz : *Genetic algorithms + data structures = evolution programs*. Springer-Verlag, London, UK, third édition, 1996.

[MS02] Jay D. Martin et Timothy W. Simpson : Use of adaptive metamodeling for design optimization. *In 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, septembre 2002.

[MS06] Jay D. Martin et Timothy W. Simpson : A Monte Carlo Method for Reliability-Based Design Optimization. Rapport technique, American Institute of Aeronautics and Astronautics, 2006.

[MTR00] H. Meunier, E. G. Talbi et P. Reininger : A multiobjective genetic algorithm for radio network optimization. *In Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, pages 317–324. IEEE, 2000.

[Nis03] Steffen Nissen : Implementation of a Fast Artificial Neural Network Library (fann). Rapport technique, Department of Computer Science University of Copenhagen (DIKU), octobre 2003.

[OBLP96] John C. Otto, Multidisciplinary O. Branch, Drew Landman et Anthony T. Patera : A Surrogate Approach To The Experimental Optimization Of Multielement Airfoils. Rapport technique, American Institute of Aeronautics and Astronautics, 1996.

[OGJ+09] Mikaël Orain, Frédéric Grisch, Eric Jourdanneau, Bjorn Rossow, Christian Guin et Brigitte Trétout : Simultaneous measurements of equivalence ratio and flame structure in multipoint injectors using PLIF. *Comptes Rendus Mécanique*, 337(6-7):373–384, juin 2009.

[OKN03] Yew S. Ong, Andrew J. Keane et Prasanth B. Nair : Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling. *AIAA Journal*, 41(4):687–696, avril 2003.

[OO04] Jeremy E. Oakley et Anthony O'Hagan : Probabilistic Sensitivity Analysis of Complex Models : A Bayesian Approach. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 66(3):751–769, 2004.

[OSF05] Akira Oyama, Koji Shimoyama et Kozo Fujii : New constraint-handling method for multi-objective multi-constraint evolutionary optimization and its application to space plane design. *In* R. Schilling, W. Haase, J. Periaux, H. Baier et G. Bugeda, éditeurs : *Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems (EUROGEN 2005)*. 2005.

[PBJ06] I. Paenke, J. Branke et Y. Jin : Efficient search for robust solutions by means of evolutionary algorithms and fitness approximation. *Evolutionary Computation, IEEE Transactions on*, 10(4):405–420, 2006.

[PLRBR09] G. Pujol, R. Le Riche, X. Bay et O. Roustant : Minimisation de quantiles – application en mécanique. *In 9e colloque national en calcul des structures*, mai 2009.

[PLRRB09] G. Pujol, R. Le Riche, O. Roustant et X. Bay : L'incertitude en conception : formalisation, estimation. *In* Rajan F. Coelho et Piotr Breitkopf, éditeurs : *Optimisation Multidisciplinaire en Mécanique*, volume 2 de *Mécanique et Ingénierie des Matériaux*, chapitre 3. Hermes Science Publications, avril 2009.

[Ray06] Daniel Raymer : *Aircraft Design : A Conceptual Approach*. AIAA education series. American Institute of Aeronautics and Astronautics, 2006.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton et Ronald J. Williams : Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, octobre 1986.

[Ros60] H. H. Rosenbrock : An Automatic Method for Finding the Greatest or Least Value of a Function. *The Computer Journal*, 3(3):175–184, janvier 1960.

[RS04] R. G. Regis et C. A. Shoemaker : Local function approximation in evolutionary algorithms for the optimization of costly functions. *Evolutionary Computation, IEEE Transactions on*, 8(5):490–505, 2004.

[RW06] C. E. Rasmussen et C. K. I. Williams : *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006.

[Sak03] S. Sakata : Structural optimization using Kriging approximation. *Computer Methods in Applied Mechanics and Engineering*, 192(7-8):923–939, février 2003.

[SBR96] R. S. Sellar, S. M. Batill et J. E. Renaud : Response Surface Based, Concurrent Subspace Optimization For Multidisciplinary System Design. *In 34th AIAA Aerospace Sciences Meeting and Exhibit*, pages 96–0714, 1996.

[Sch97] Matthias Schonlau : *Computer experiments and global optimization*. Thèse de doctorat, Waterloo, Ont., Canada, Canada, 1997.

[SGCD11] D. W. Stephens, D. Gorissen, K. Crombecq et T. Dhaene : Surrogate based sensitivity analysis of process equipment. *Applied Mathematical Modelling*, 35(4):1676–1687, avril 2011.

[SKMM98] Timothy W. Simpson, John J. Korte, Timothy M. Mauery et Farrokh Mistree : Comparison of response surface ans kriging models for multidisciplinary design optimization. *In 7th AIAA/USAF/NASA/ ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1998.

[SLK04] A. Sóbester, S. J. Leary et A. J. Keane : A parallel updating scheme for approximating and optimizing high fidelity computer simulations. *Structural and Multidisciplinary Optimization*, 27(5):371–383, juillet 2004.

[Soa08] Gustavo L. Soares : *Algoritmos Determinístico e Evolucionário Intervalares para Otimização Robusta Multi-Objetivo*. Thèse de doctorat, Universidade Federal de Minas Gerais, octobre 2008.

[Sob89] J. Sobieszczanski-Sobieski : Optimization by decomposition : A step from hierarchic to non-hierarchic systems. *NASA STI/Recon Technical Report N*, 892, avril 1989.

[SOF05] K. Shimoyama, A. Oyama et K. Fujii : A new efficient and useful robust optimization approach - design for multi-objective six sigma. *In Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 1, pages 950–957 Vol.1, 2005.

[SPKA01] T. W. Simpson, J. D. Poplinski, P. N. Koch et J. K. Allen : Metamodels for Computer-based Engineering Design : Survey and recommendations. *Engineering with Computers*, 17(2):129–150, juillet 2001.

[SRA+08] A. Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana et Stefano Tarantola : *Global Sensitivity Analysis : The Primer*. Wiley, 2008.

[SSAPS03] Jaroslaw Sobieszczanski-Sobieski, Troy D. Altus, Matthew Phillips et Robert Sandusky : Bilevel Integrated System Synthesis for Concurrent and Distributed Processing. *AIAA Journal*, 41(10):1996–2003, octobre 2003.

[SSAS98] Jaroslaw Sobieszczanski-Sobieski, Jeremy S. Agte et Robert R. Sandusky : Bi-Level Integrated System Synthesis (BLISS). Rapport technique, National Aeronautics and Space Administration, août 1998.

[SSH97] J. Sobieszczanski-Sobieski et R. T. Haftka : Multidisciplinary aerospace design optimization : survey of recent developments. *Structural and Multidisciplinary Optimization*, 14(1):1–23, août 1997.

[STBV08] Timothy W. Simpson, Vassili V. Toropov, Vladimir Balabanov et Felipe A. C. Viana : Design and analysis of computer experiments in multidisciplinary design optimization : a review of how far we have come – or not. *In 12th AIAAISSMO Multidisciplinary Analysis and Optimization Conference*. American Institute of Aeronautics and Astronautics, 2008.

[SWJ98] Matthias Schonlau, William J. Welch et Donald R. Jones : Global versus Local Search in Constrained Optimization of Computer Models. *Lecture Notes-Monograph Series*, 34, 1998.

[SWMW89] Jerome Sacks, William J. Welch, Toby J. Mitchell et Henry P. Wynn : Design and Analysis of Computer Experiments. *Statistical Science*, 4(4): 409–423, 1989.

[SWN03] Thomas J. Santner, Brian J. Williams et William Notz : *The Design and Analysis of Computer Experiments*. Springer, New York, 1 édition, juillet 2003.

[Tag89] G. Taguchi : Introduction to quality engineering. Rapport technique, Asian productivity organization, 1989.

[TBCW07] H. Taboada, F. Baheranwala, D. Coit et N. Wattanapongsakorn : Practical solutions for multi-objective optimization : An application to system reliability design problems. *Reliability Engineering & System Safety*, 92(3):314–322, mars 2007.

[TG97] S. Tsutsui et A. Ghosh : Genetic algorithms with a robust solution searching scheme. *Evolutionary Computation, IEEE Transactions on*, 1(3):201–208, 1997.

[TM06] Nathan P. Tedford et Joaquim R. R. A. Martins : On the Common Structure of MDO Problems : A Comparison of Architectures. *In 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, septembre 2006.

[Tro97] Michael W. Trosset : Taguchi and Robust Optimization. Rapport technique, Rice University, mars 1997.

[Vap00] Vladimir Vapnik : *The nature of statistical learning*. Springer, 2000.

[VH99] G. Venter et R. T. Haftka : Using response surface approximations in fuzzy set based design optimization. *Structural and Multidisciplinary Optimization*, 18(4):218–227, décembre 1999.

[Vil08] Julien Villemonteix : *Optimisation de fonctions coûteuses Modèles gaussiens pour une utilisation efficace du budget d'évaluations : théorie et pratique industrielle*. Thèse de doctorat, Université Paris Sud - Paris XI, décembre 2008.

[VVW08] Julien Villemonteix, Emmanuel Vazquez et Eric Walter : An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, septembre 2008.

[WCSF01] Benjamin Wilson, David Cappelleri, Timothy W. Simpson et Mary Frecker : Efficient Pareto Frontier Exploration using Surrogate Approximations. *Optimization and Engineering*, 2(1):31–50, mars 2001.

[WEDP11] Tobias Wagner, Michael Emmerich, André Deutz et Wolfgang Ponweiser : On Expected-Improvement Criteria for Model-based Multi-objective Optimization. *In* Robert Schaefer, Carlos Cotta, Joanna Kolodziej et Günter Rudolph, éditeurs : *Parallel Problem Solving from Nature – PPSN XI*, volume 6238 de *Lecture Notes in Computer Science*, chapitre 72, pages 718–727. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2011.

[Wie38] Norbert Wiener : The Homogeneous Chaos. *American Journal of Mathematics*, 60(4):897–936, 1938.

[WS07] Gary G. Wang et S. Shan : Review of Metamodeling Techniques in Support of Engineering Design Optimization. *Journal of Mechanical Design*, 129(4): 370–380, 2007.

[WSN00] Brian J. Williams, Thomas J. Santner et William I. Notz : Sequential design of computer experiments to minimize integrated response functions. *Statistica Sinica*, 2000.

[YCD05] Byeng Youn, Kyung Choi et Liu Du : Enriched Performance Measure Approach for Reliability-Based Design Optimization. *AIAA Journal*, 43(4), avril 2005.

[YN91] Wei N. Yang et Barry L. Nelson : Using Common Random Numbers and Control Variates in Multiple-Comparison Procedures. *Operations Research*, 39(4), 1991.

[ZDT00] Eckart Zitzler, Kalyanmoy Deb et Lothar Thiele : Comparison of Multiobjective Evolutionary Algorithms : Empirical Results. *Evolutionary Computation*, 8:173–195, 2000.

[ZKT08] Eckart Zitzler, Joshua Knowles et Lothar Thiele : Quality Assessment of Pareto Set Approximations Multiobjective Optimization. *In* Jürgen Branke, Kalyanmoy Deb, Kaisa Miettinen et Roman Slowinski, éditeurs : *Multiobjective Optimization*, volume 5252 de *Lecture Notes in Computer Science*, chapitre 14, pages 373–404. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2008.

[ZT99] E. Zitzler et L. Thiele : Multiobjective evolutionary algorithms : a comparative case study and the strength Pareto approach. *Evolutionary Computation, IEEE Transactions on*, 3(4):257–271, novembre 1999.

[ZTW09] Parviz Zadeh, Vassili Toropov et Alastair Wood : Metamodel-based collaborative optimization framework. *Structural and Multidisciplinary Optimization*, 38(2):103–115, avril 2009.

Je ne sais rien avec certitude,
mais la simple vue des étoiles me fait rêver.
– Vincent Van Gogh