Kriging for turbomachineries conception : high dimension and multi-objective robust optimization

Mélina Ribaud

ÉCOLE CENTRALE LYON

Numéro d'ordre NNT : 2018 LYSEC 026

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de l'École centrale de Lyon

École Doctorale 512
École Doctorale InfoMaths

Spécialité de doctorat : Mathématiques et applications
Discipline : Mathématiques

Soutenue publiquement le 17 Octobre 2018 par

Mélina Ribaud

Krigeage pour la conception de turbomachines : grande dimension et optimisation robuste

Devant le jury composé de :

M. Nicolas Gayton	Professeur des universités, SIGMA Clermont	Rapporteur
M. David Ginsbourger	Professeur des universités, Université de Berne	Rapporteur
M. Rodolphe Le Riche	Directeur de recherche, Mines Saint-Étienne	Examinateur
M. Jean-Marc Azaïs	Professeur des universités, Université Paul Sabatier	Examinateur
Mme Christophette Blanchet-Scalliet	Maître de conférences, École centrale de Lyon	Directeur
Mme Céline Helbert	Maître de conférences, École centrale de Lyon	Co-directeur
M. Frédéric Gillot	Maître de conférences, École centrale de Lyon	Co-directeur
M. Manuel Henner	Expert fan system, Valeo	Invité

Remerciements

Je voudrais remercier chaleureusement David Ginsbourger et Nicolas Gayton d'avoir accepté de rapporter le manuscrit de cette thèse ainsi que les membres du jury, Jean-Marc Azaïs, Rodolphe Le Riche et Manuel Henner.

Je remercie aussi tous les membres de l'ANR Pepito pour les échanges très intéressants lors des différentes réunions à Paris, Lyon et Toulouse. Ces conversations m'ont permis de confirmer mon attrait pour la recherche industrielle et ont conforté mon orientation professionnelle.

Je tiens à remercier Céline Vial pour ses enseignements lors de mon cursus à Polytech Lyon. C'est grâce à sa bienveillance et à ses conseils avisés que je me suis tournée vers une thèse et que je continue dans cette voie.

Je remercie mes directeurs de thèse, Céline Helbert, Christophette Blanchet-Scalliet et Frédéric Gillot. Frédéric pour toutes les conférences en France et à l'étranger auxquelles j'ai assisté et pour ce fantastique séjour de recherche effectué au Japon. Céline et Christophette qui m'ont guidée et épaulée dans mes travaux de recherche et dans les enseignements, qui ont toujours été présentes pour répondre à chacune de mes interrogations. Elles ont été de formidables exemples et repères tout au long de ces trois années. Et c'est grâce à elles si je garde un magnifique souvenir de cet apprentissage en recherche. En particulier, Céline qui m'a apporté beaucoup de soutien grâce à nos discussions, sa pédagogie et sa joie de vivre contagieuse.

Je souhaite aussi remercier tous les membres du laboratoire de l'ICJ de Centrale Lyon. Loïc, Alexis et Mona qui m'ont accueillie lors de mon arrivée et qui ont fait que le passage d'étudiant à doctorant ne soit pas trop douloureux. Isabelle qui a répondu à toutes mes questions concernant le fonctionnement de l'école. Tout ceux qui ont amené de si bonnes choses à grignoter, c'était un régal. François, Philippe, Hélène, Laurent et d'autres avec qui j'ai eu la chance de discuter de nombreux sujets intéressants lors de la pause thé-café. Mes camarades et amis Laura, Nicolas et Mathilde pour leur soutien, nos échanges sur des sujets divers et variés et pour bien d'autres choses. C'est grâce à eux si cet immense bureau des doctorants m'a semblé plein de vie et de bonheur.

Je remercie aussi tous les membres de ma famille ainsi que mes amis qui ont été tout simplement présents dans les moments de joie mais aussi de doute, Corentin, Florentin, Clelia, Laurine, Amaury, Florence, Alexandre, Julien, Victor, Lola et bien d'autres. Une pensée particulière pour mes grand-parents qui m'ont toujours soutenue dans mes études. Mes parents Christine et Jean-Philippe, mon frère Ardit et à mon amie Tatiana qui ont été présents et à l'écoute durant ces trois ans. Quitty, Jacqueline et Saucisse, mes fidèles compagnes à quatre pattes qui ont toujours été à mes côtés.

Enfin, je remercie tout particulièrement celui qui partage ma vie, Olivier, pour son soutien, son aide et sa joie de vivre quotidienne.

Résumé

Dans le secteur de l'automobile, les turbomachines sont des machines tournantes participant au refroidissement des moteurs de voitures. Leur performance dépend de multiples paramètres géométriques qui déterminent leur forme.

Cette thèse s'inscrit dans le projet ANR PEPITO réunissant industriels et académiques autour de l'optimisation de ces turbomachines. L'objectif du projet est de trouver la forme du ventilateur maximisant le rendement en certains points de fonctionnement.

Dans ce but, les industriels utilisent des codes CFD (Computational Fluid Dynamics) simulant le fonctionnement de la turbomachine. Ces codes sont très coûteux en temps de calcul. Il est donc impossible d'utiliser directement le résultat de ces simulations pour conduire une optimisation dans une durée compatible avec les contraintes industrielles.

Par ailleurs, lors de la construction des turbomachines, des perturbations sont observées sur les paramètres d'entrée. Elles sont le reflet de fluctuations des machines de production. Les écarts observés sur la forme géométrique finale de la turbomachine peuvent provoquer une perte de performance conséquente. Il est donc nécessaire de prendre en compte ces perturbations et de procéder à une optimisation qui soit robuste à ces fluctuations.

Dans ce travail de thèse, des méthodes basées sur du krigeage répondant aux deux principales problématiques liées à ce contexte de simulations coûteuses ont été proposées :

- Comment construire une bonne surface de réponse pour le rendement lorsqu'il y a beaucoup de paramètres géométriques ?
- Comment procéder à une optimisation du rendement efficace tout en prenant en compte les perturbations des entrées ?

Une réponse à la première problématique est donnée en proposant plusieurs algorithmes permettant de construire un noyau de covariance pour le krigeage, adapté à la grande dimension. Ce noyau est un produit tensoriel de noyaux isotropes où chacun de ces noyaux est lié à un sous groupe de variables d'entrée. Ces algorithmes sont testés sur des cas simulés et sur une fonction réelle. Les résultats montrent que l'utilisation de ce noyau permet d'améliorer la qualité de prédiction en grande dimension.

Concernant la seconde problématique, plusieurs stratégies itératives basées sur un co-krigeage avec dérivées pour réaliser l'optimisation robuste sont proposées. A chaque itération, un front de Pareto est obtenu par la minimisation de deux objectifs calculés à partir des prédictions de la fonction coûteuse. Le premier objectif représente la fonction elle-même et le second la robustesse. Cette robustesse est quantifiée par un critère estimant une variance locale et basée sur le développement de Taylor. Ces stratégies sont comparées sur deux cas tests : en petite et en grande dimension. Les résultats montrent que les meilleures stratégies permettent bien de trouver l'ensemble des solutions robustes.

Enfin, les méthodes proposées sont appliquées sur les cas industriels propres au projet PEPITO.

Mots clés : krigeage, algorithme, grande dimension, noyau de covariance, optimisation robuste

Abstract

The turbomachineries are rotary machines used to cool down the automotive engines. Their efficiency is depending on a high number of geometric parameters that describe the shape.

My thesis is a contribution of the ANR project PEPITO where industrials and academics collaborate. The aim of this project is to found the turbomachineries shape that maximizes the efficiency.

That is why, industrials use numerical CFD (Computational Fluid Dynamics) codes that simulate the work of turbomachineries. However, the simulations are time-consuming. We cannot directly use the simulations provided to perform the optimization.

In addition, during the production line, the input variables are subjected to perturbations. These perturbations are due to the production machineries fluctuations. The differences observed in the final shape of the turbomachinery can provoke a loss of efficiency. These perturbations have to be taken into account to conduct an optimization robust to fluctuations.

In this thesis, since the context is time consuming simulations we propose kriging based methods that meet the requirements of industrials. The issues are :

- How can we build a predictive kriging metamodel when the number of input variables is high ?
- How can we lead an efficient optimization, robust with regard to perturbations of some inputs ?

Several algorithms are proposed to answer to the first question. They construct a covariance kernel adapted to high dimension. This kernel is a tensor product of isotropic kernels in each subspace of input variables. These algorithms are benchmarked on some simulated cases and on a real function with fifteen inputs : the results show that the use of this kernel improved the prediction quality in high dimension.

For the second question, seven iterative strategies based on a co-kriging model are proposed to conduct the robust optimization. In each iteration, a Pareto front is obtained by the minimization of two objective computed from the kriging predictions. The first one represents the function and the second one the robustness. A criterion based on the Taylor theorem is used to estimate the local variance and quantifies the robustness. These strategies are compared in two test cases in small and high dimensions. The results show that the best strategies have well found the set of robust solutions.

Finally, the methods are applied on the industrial cases provided by the PEPITO project.

Keywords : kriging, algorithm, high dimension, covariance kernel, robust optimization

Table des matières

Introduction

I Présentation des problématiques liées au fonctionnement des turbomachines dans le cadre du projet ANR PEPITO

Cette thèse s'inscrit dans le projet ANR PEPITO pour Plan d'Expérience Pour l'Industrie du Transport et l'Optimisation. Le projet PEPITO regroupe des partenaires industriels et des partenaires académiques. Il a pour objectif d'élaborer des techniques d'optimisation de turbomachines par la combinaison de simulations numériques, de planification d'expériences, de construction de méta-modèles et de calcul inverse.

Le cas d'application choisi est celui d'un ventilateur automobile, destiné à forcer le passage de l'air (débit) dans un ensemble d'échangeurs thermiques. La plupart du temps cet ensemble est composé d'un radiateur pour le liquide de refroidissement, d'un condenseur pour le fluide frigorigène de la climatisation et d'un refroidisseur d'air de suralimentation pour l'air compressé du turbocompresseur. La forme générale du ventilateur automobile est le résultat d'évolutions constantes des besoins du marché de l'automobile. Il s'agit d'un compromis entre les performances aérodynamiques et acoustiques, avec des contraintes liées à l'encombrement et à la résistance mécanique dans un environnement difficile. En effet, les variations de températures peuvent être extrêmes (de -40° en hiver par grand froid, à $+110^\circ$ dans le compartiment lorsque le moteur est à pleine puissance). Le ventilateur est constitué d'une hélice et d'un moteur, parfois appelé GMV (Groupe Moto-Ventilateur). L'hélice est constituée d'un bol, sur lequel sont fixées des pales (ou aubes) dont les extrémités sont maintenues par une virole tournante. Ce composant est ensuite fixé directement sur un moteur d'entrainement. Ce dernier est fixé sur un support (cf Figure 1) qui assure une fonction mécanique (maintenir le ventilateur sur le véhicule) et aérodynamique (concentrer le flux d'air aspiré par l'hélice). D'autres architectures peuvent être trouvées, notamment pour les véhicules de transport, les engins agricoles ou de chantier. Il s'agit le plus souvent de variations au niveau du système d'entrainement de l'hélice mais le principe général reste le même. L'ensemble constitué des échangeurs thermiques et du ventilateur est appelé le module de refroidissement. Il se situe traditionnellement à l'avant du véhicule afin de bénéficier d'un effet de ventilation naturelle par la vitesse du véhicule (cf Figure 2).

Un premier "point de fonctionnement" intéressant du ventilateur correspond à un fonctionnement à vitesse élevée. Le débit d'air entrant est important, le delta de pression ΔP (sortie - entrée) est faible. Dans ce contexte le ventilateur sert à refroidir le moteur (groupe moto-propulseur). Un autre point de fonctionnement important est celui du véhicule à faible vitesse, ou même à l'arrêt. Dans ces conditions, la chaleur à dissiper provient à la fois du groupe moto-propulseur (peu sollicité) et du système de climatisation lorsqu'il est activé. Ces conditions sont caractérisées par un débit relativement faible et donc une sollicitation importante du ventilateur. L'objectif est d'obtenir un rendement maximal pour minimiser la consommation électrique. Ce point correspond au point de fonctionnement nominal du ventilateur. Les performances aérodynamiques de la turbomachine sont représentées par une courbe ΔP-Q (courbe rouge sur la Figure 3). A débit nul le différentiel de pression est maximal, ΔP s'annule

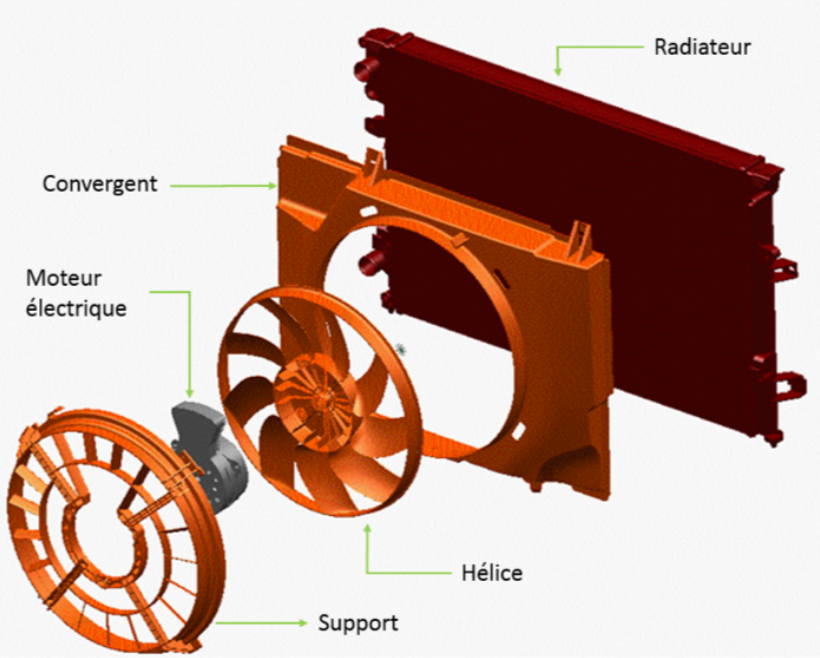

Figure 1 – Dessin d'une turbomachine complète.

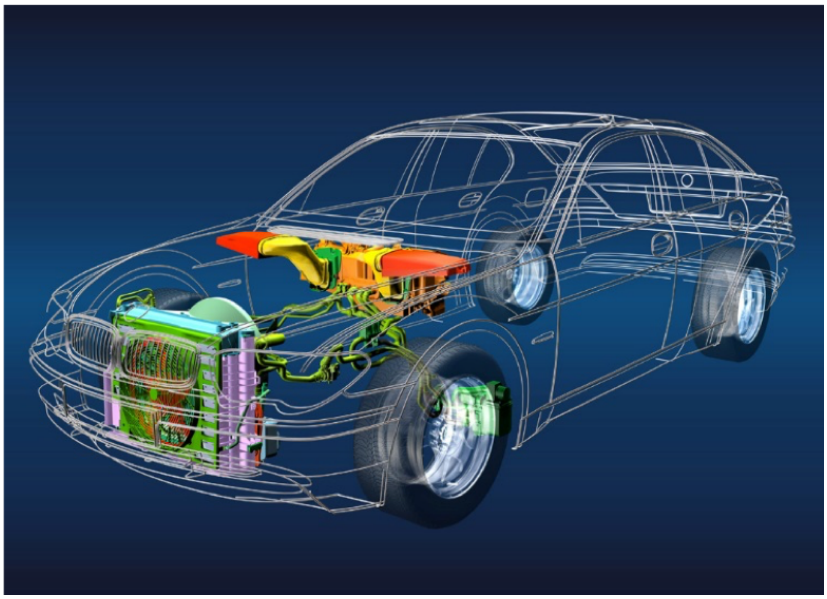

Figure 2 – Système de refroidissement d'une voiture.

pour un débit très élevé. Ce dernier point est parfois appelé "point de transparence", par analogie au fait qu'en ce point l'hélice n'est ni active, ni résistante.

La conception de ces turbomachines a suivi l'évolution des méthodes de développement dans l'industrie. Aujourd'hui, le processus repose en grande partie sur des méthodes numériques, soit de prédiction, soit d'optimisation des performances. L'utilisation de ces approches permet de gagner du temps et de donner des solutions de plus en plus optimales. Par exemple, la Figure 4 représente plusieurs générations d'hélice de ventilateur, depuis les années 1980 et par décennie. Au cours du temps, la géométrie s'est complexifiée pour tenir compte d'impératifs de plus en plus contraignant. Les pales simples se sont courbées pour créer des déphasages acoustiques (moins de bruit). Puis la surface des

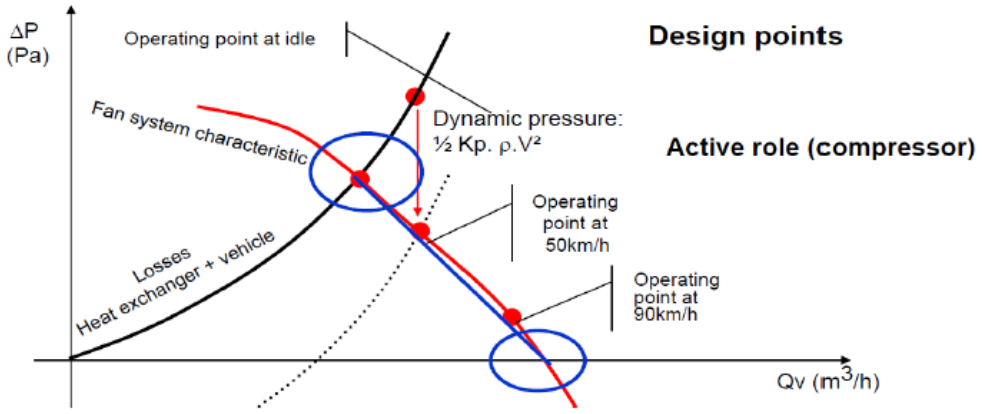

FIGURE 3 – Courbe caractéristique de pertes de charge et de pression de ventilation.

pales a augmenté pour produire plus de pression. Finalement, les formes se sont libérées jusqu'à ressembler à des ailes d'oiseaux. La complexification des formes est apparue avec la prise en compte d'un nombre croissant de paramètres. Pour l'hélice des années 1980, seulement deux ou trois paramètres comme le nombre de pales et leurs surfaces suffisaient à faire évoluer l'hélice. Actuellement, le nombre de facteurs a largement augmenté. Il est nécessaire par exemple de caractériser une pale à différentes envergures (sections) par la corde, la cambrure, l'épaisseur et le calage. En retenant ces 4 paramètres pour 4 sections, déjà 16 facteurs sont obtenus. Leur nombre n'est pas limité à cela. Selon l'investissement des développeurs dans les études, il est facilement envisageable d'étudier plus de trente paramètres. Le développement des hélices de ventilation nécessite la mise en place d'une méthodologie élaborée. De plus, les cycles d'étude sont de plus en plus courts, ce qui ne laisse pas le temps aux ingénieurs de procéder de façon séquentielle, avec des étapes dites "d'essais et d'erreurs". Lorsque l'industriel est sollicité, il doit, soit disposer d'une solution directement disponible, soit posséder un outil rapide lui permettant de concevoir l'hélice dans les temps.

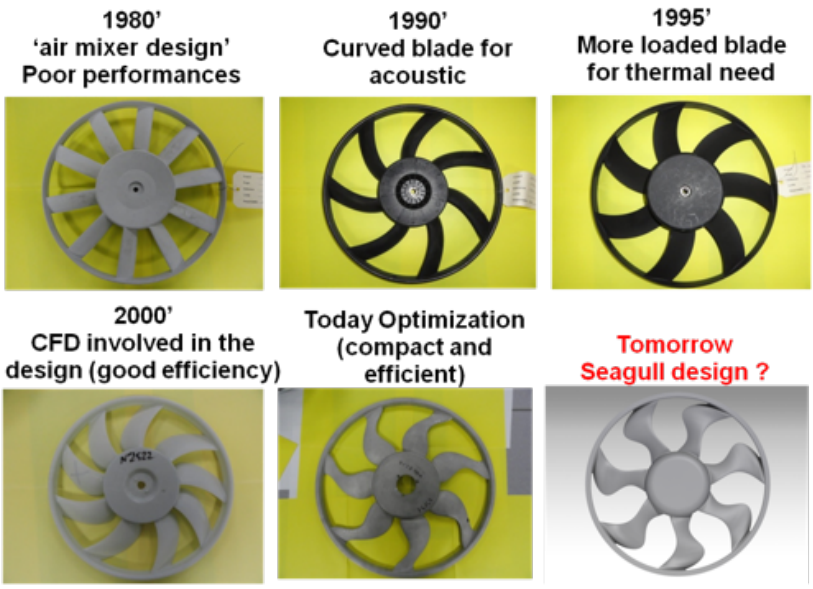

FIGURE 4 – Évolution des turbomachines au fil des années.

L'optimisation de l'hélice présente des objectifs multiples (plusieurs points de fonctionnement), plusieurs physiques sont à prendre en compte (aérodynamique, acoustique, mécanique). De plus, l'aspect économique peut orienter la solution dans une direction ou une autre (économie de matière ou économie d'énergie). Il faut donc répondre à chaque appel d'offre par une solution personnalisée. Par exemple, une voiture familiale ou une citadine n'ont pas les mêmes contraintes d'espace et la même utilisation. En effet, chaque constructeur développe sa propre stratégie de vente et donc de gestion thermique. L'approche choisie consiste à établir un méta-modèle facile à interroger pour sélectionner rapidement des hélices prometteuses. Des surfaces de réponse existent déjà pour 11 ou 15 paramètres d'entrée. Elles permettent d'optimiser une hélice, sans toutefois garantir que l'assemblage final soit le plus pertinent. C'est pourquoi, une surface de réponse prenant en compte plus de paramètres pour modéliser la géométrie du modèle complet est nécessaire. Les plans d'expérience numériques du projet ont pour objectif d'aller jusqu'à une dimension de l'ordre de 60 paramètres. De plus, la performance de l'hélice finale peut être altérée par un ensemble d'incertitudes ou de variations. Parmi elles, des incertitudes de fabrication sont observées. Elles correspondent à des perturbations sur les paramètres d'entrées. Si l'hélice n'est pas robuste à ces perturbations la perte de performance peut être considérable. L'objectif du projet est donc de mettre en place un métamodèle en grande dimension. Il fournira une surface de réponse pour chaque objectif (aérodynamique, acoustique, encombrement ...). Ces surfaces permettront de proposer des compromis entre les différents objectifs. L'optimalité des solutions pourra aussi être mesurée et ces solutions pourront être défendues lorsque les demandes constructeur seront irréalistes. Enfin, les surfaces de réponse sont ré-utilisables autant de fois que nécessaire puisque le coût d'exploitation est minime avec les moyens informatiques actuels. Cette économie peut de plus être mise à profit pour mesurer la robustesse des solutions proposées. Si les variations sont fortes et soudaines, le risque est réel d'être sur une solution "non robuste" qui verrait son fonctionnement très altéré avec les incertitudes de production. Il faudrait alors privilégier une solution très stable même si au départ elle semble moins intéressante.

Le projet PEPITO a pour objectif de développer de nouvelles techniques comprenant des méthodes statistiques (plan d'expérience, métamodélisation, optimisation multi-objectif et robuste ...) afin de les mettre en œuvre sur les cas industriels. VST (Valeo Systèmes Thermiques) et le LMFA (Laboratoire de Mécanique des Fluides et d'Acoustique) travaillent en amont sur la paramétrisation 3D pour Valeo et 1D pour le LMFA de la turbomachine. Valeo fournit des plans d'expériences avec le résultat des analyses numériques de dynamique de fluides. Le LMFA fournit le modèle 1D. L'ICJ (Institut Camille Jordan), l'UPS/IMT (Université Paul Sabatier / Institut de Mathématiques de Toulouse) et l'entreprise InModelia traitent la partie plan d'expérience et modélisation statistique. Le LTDS (Laboratoire de Tribologie et de Dynamique des Systèmes) et l'ICJ travaillent sur l'optimisation. L'entreprise Intes France effectue les simulations mécaniques (déformée dynamique sous l'effet des forces centrifuges, analyse modale).

Problématiques liées au fonctionnement des turbomachines

Le premier objectif de cette thèse consiste à construire des surfaces de réponse de qualité lorsque la dimension est élevée. Les modèles de régression linéaire, additif généralisé ou encore de krigeage sont des exemples de surfaces de réponse disponibles. Ce dernier est très répandu pour construire une surface de réponse à partir des codes numériques coûteux. Les études montrent que le méta-modèle de krigeage avec un noyau *anisotrope* (autant de paramètres estimés que de variables) et accompagné d'un design "space filling" est très performant lorsque le nombre de paramètres est peu élevé (cf [Marrel et al., 2008], [Sudret, 2012] et [Villa-Vialaneix et al., 2012]). Or, en grande dimension, estimer un modèle de krigeage avec un noyau anisotrope peut être difficile. Quand le nombre d'observations est faible devant la dimension, l'estimation des paramètres devient problématique. Cette mauvaise estimation peut altérer fortement la qualité prédictive. La première problématique de ce travail de thèse

est : **Comment obtenir une surface de réponse par krigeage lorsqu'il y a beaucoup de variables d'entrée ?**

Le second objectif de cette thèse consiste à prendre en compte les incertitudes de fabrication lors de l'optimisation. L'idée consiste à procéder à une optimisation robuste aux perturbations sur la surface de réponse. Les études montrent la pertinence du krigeage pour procéder à de l'optimisation séquentielle (appel au code coûteux à chaque itération). L'algorithme le plus connu EGO utilise le critère de l'EI (amélioration espérée) permettant d'explorer les zones inconnues tout en raffinant les zones à fort potentiel (cf [Jones et al., 1998]). La question qui se pose est celle de l'adaptation de cet algorithme existant à l'optimisation robuste. Tout d'abord, la quantification de la robustesse d'une solution reste une question difficile. Il existe, dans la littérature, une multitude de critères de robustesse. À partir de ces critères plusieurs algorithmes à base de krigeage ont été développés. Cependant, ils traitent d'incertitudes liées à l'environnement et non de celles dues au processus de fabrication. De plus, les stratégies développées consistent à améliorer de façon itérative une solution. Dans ce chapitre, l'objectif est d'observer l'évolution d'un front de solutions proposant différents compromis optimisation-robustesse. La seconde problématique de la thèse est donc : **Comment procéder à une optimisation efficace tout en prenant en compte les perturbations sur les variables d'entrée et par le bais d'une surface de réponse ?**

II Description des chapitres

Le manuscrit est organisé en quatre chapitres. Le chapitre 1 présente les pré-requis nécessaires à la compréhension des chapitres suivants ainsi que des méthodes de la littérature proposant des solutions aux problématiques traitées durant cette thèse. Les chapitres 2 et 3 correspondent à deux articles (le premier accepté et le second soumis) répondant d'une part aux problèmes liés à la grande dimension et d'autre part à l'optimisation robuste par métamodèle. Ils peuvent être lus séparément. Les méthodes développées aux chapitres 2 et 3 sont mises en oeuvre sur le cas industriel et les résultats font l'objet du chapitre 4. Une brève étude de comparaison des métamodèles est également présentée en début de chapitre. Un résumé des chapitres est donné ci-dessous.

II.1 Chapitre 1 : État de l'art : krigeage, co-krigeage, réduction de dimension et optimisation sur métamodèle

Les connaissances nécessaires à la compréhension de la thèse sont données dans ce chapitre. Pour commencer, les modèles de krigeage (cf [Santner et al., 2003] et [Rasmussen and Williams, 2006]) et de co-krigeage (cf [Le Gratiet, 2013]) sont introduits. Puis, trois méthodes de réduction de dimension dans le cadre du krigeage sont décrites : l'analyse de sensibilité, la méthode de pénalisation de la vraisemblance (cf [Yi, 2009]) et la méthode FANOVA (cf [Muehlenstaedt et al., 2012]). Enfin, des méthodes d'optimisation par métamodèle de krigeage sont présentées : optimisation mono-objectif (cf [Jones et al., 1998]), optimisation multi-objectif (cf [Wagner et al., 2010] et [Picheny, 2015]) et optimisation robuste (cf [Janusevskis and Le Riche, 2013], [Marzat et al., 2013] et [Ur Rehman and Langelaar, 2015]).

II.2 Chapitre 2 : Un nouveau noyau de covariance pour réduire la dimension et quatre algorithmes pour le construire

Dans ce chapitre, un nouveau noyau de covariance adapté à la grande dimension est introduit. Ce noyau nommé *isotrope par groupe* est construit à partir d'un noyau *anisotrope* (autant de paramètres

estimés que de variables) et d'un noyau *isotrope* (un seul paramètre estimé). En effet, les variables d'entrée sont réparties en plusieurs groupes et à chaque groupe est associé un paramètre. Ce procédé permet de réduire la dimension par rapport au noyau *anisotrope* et de gagner en flexibilité comparé au noyau *isotrope*. Les difficultés liées à la construction du noyau sont la répartition des variables dans les groupes et le nombre de groupes à choisir. Quatre algorithmes sont proposés pour déterminer le nombre de groupe et la composition de chacun. Ces algorithmes sont des extensions de celui introduit par [Welch et al., 1992]. Ces procédures partent d'un noyau *isotrope* pour terminer par un noyau *isotrope par groupe*. Au cours des algorithmes plusieurs modèles avec différents noyaux *isotrope par groupe* sont parcourus et comparés à l'aide d'un critère BIC (cf [Schwarz, 1978]). Les quatre algorithmes sont appliqués sur un cas simulé et le meilleur est conservé. Celui-ci est alors étudié en détail sur des cas simulés puis appliqué sur une fonction réelle.

II.3 Chapitre 3 : Optimisation robuste à l'aide d'un métamodèle de krigeage avec dérivées

Au cours de ce chapitre, plusieurs stratégies sont proposées pour répondre à la problématique d'optimisation robuste dans le contexte des simulations coûteuses. L'optimisation robuste consiste à trouver des optima de la fonction peu impactés par les perturbations sur les entrées. La première partie répond à la problématique de la quantification de la robustesse. La variance de la fonction approchée par le développement de Taylor à l'ordre 2 (cf [J. Darlington and Rustem, ated] et [Pronzato and Éric Thierry, 2003]) est utilisée. Ce critère nécessite la prédiction des dérivées et la prise en compte de l'observation de celles-ci quand elles sont disponibles. Il faut donc choisir un métamodèle adapté. Le modèle de co-krigeage (cf [Le Gratiet, 2013]) permet de prédire la fonction et ses dérivées. Sept stratégies d'optimisation multi-objectif (fonction et critère de robustesse) sont alors proposées. Elles permettent de mettre en œuvre une démarche d'optimisation globale comportant des phases d'exploitation des zones prometteuses mais aussi des zones d'exploration pour sortir d'éventuelles zones d'optima locaux. Cette exploration est possible grâce à l'information apportée par la prédiction par krigeage (loi conditionnelle du processus sachant les observations) en tenant compte ou non de la variance de krigeage. Cependant dans ce dernier cas un autre indicateur est utilisé pour repérer les zones où la prédiction est mauvaise par rapport à la vraie fonction. La procédure, commune à toutes ces méthodes est proche de l'algorithme EGO (cf [Jones et al., 1998]) et consiste à ajouter plusieurs points à chaque itération (batch). L'optimisation multi-objectif est opérée par l'algorithme génétique NSGA II (cf [Deb et al., 2002]). Puis, ces stratégies sont testées sur une fonction en dimension faible afin de sélectionner les trois méthodes les plus performantes. Le comportement de ces méthodes est étudié plus spécifiquement dans deux cas distincts : quand l'observation des dérivées est accessible et quand elles ne le sont pas. Enfin, ces stratégies sont appliquées sur une fonction en dimension supérieure.

II.4 Chapitre 4 : Application au cas industriel des turbomachines : paramétrisations proposées par Valeo et le LMFA

Au cours de ce chapitre, les nouvelles méthodes développées précédemment sont appliquées sur les cas industriels. Dans un premier temps, plusieurs études préliminaires sont effectuées sur le code 3D fourni par Valeo pour valider l'utilisation du krigeage et tester sa stabilité par rapport aux paramètres du modèle. Lors d'une seconde étude, une méthode d'analyse de sensibilité globale est utilisée pour repérer les variables influentes. La dernière étude consiste à réduire la dimension de façon naïve. Dans un second temps, la méthode de réduction de dimension développée dans le chapitre 2 est appliquée sur le code 3D. Pour chaque réponse, les groupes de portée obtenus sont analysés et comparés aux observations des experts industriels. Puis, la méthode est analysée afin de juger de sa capacité à réduire le nombre de paramètres de portée estimés tout en conservant une qualité de prédiction satis-

faisante. Dans un dernier temps, les trois meilleures stratégies d'optimisation robuste développées lors du chapitre 3 sont appliquées sur le code 1D. Le but est d'observer l'avancement du front de Pareto au cours de l'optimisation du rendement et de repérer la méthode la plus efficace.

Chapitre 1

État de l'art : krigeage, co-krigeage, réduction de dimension et optimisation sur métamodèle

Les deux grandes problématiques liées au cas industriel présentées dans l'introduction consistent à construire un métamodèle efficace en grande dimension pour remplacer les calculs trop coûteux et à optimiser la forme de la turbomachine. Un métamodèle est un modèle statistique peu coûteux ajusté sur le résultat de simulations du modèle numérique. [Villa-Vialaneix et al., 2012] font une comparaison de huit métamodèles pour la simulation de flux de N_20 et de N dans les épis de maïs. Ils montrent que le krigeage donne les meilleures performances pour les petits et grands jeux de données. Ils obtiennent aussi de bons résultats de prédiction avec les splines. Ce résultat n'est pas étonnant car [Maatouk, 2015] montre dans sa thèse la proximité entre ces deux méthodes. Les articles de [Booker et al., 1998], [Marrel et al., 2008] et [Sudret, 2012] présentent des exemples d'utilisation pertinente du krigeage sur des cas industriels. De plus, le krigeage est capable de modéliser des données complexes et en grande dimension (beaucoup de variables d'entrée) (cf [Santner et al., 2003] et [Rasmussen and Williams, 2006]). Ainsi le cadre de la thèse consiste à se placer dans le contexte d'une modélisation par krigeage encore appelée régression par processus Gaussien. Le chapitre est divisé en trois parties. Dans un premier temps, les modèles de krigeage et de co-krigeage sont présentés. Ensuite, les versions du krigeage traitant de la grande dimension sont introduites. Puis, des méthodes d'optimisation mono-objectif, multi-objectif et d'optimisation robuste à base de krigeage est effectué. Enfin, trois critères pour quantifier la qualité de prédiction d'un modèle sont présentés.

1 Krigeage

Soit p le nombre de variables d'entrée. Soient n observations $(\mathbf{x}^i, y^i)_{i=1,\dots,n}$ où $\mathbf{x}^i$ est le ième vecteur d'entrée à p coordonnées, $y^i = f(\mathbf{x}^i)$ ($y^i \in \mathbb{R}$) est la ième observation de la sortie correspondant à l'entrée $\mathbf{x}^i$. Le vecteur des sorties est noté $\mathbf{y} = (y^1, \dots, y^n)'$.

Hypothèse 1. $\mathbf{y}$ *est supposé être la réalisation d'un processus Gaussien* $(Y(\mathbf{x}))_{\mathbf{x}\in D}$ *aux points* $(\mathbf{x}^1, \dots, \mathbf{x}^n)'$ *tel que* $\forall \mathbf{x} \in D \subset \mathbb{R}^p$ *:*

$$Y(\mathbf{x}) = m + \epsilon(\mathbf{x}), \tag{1.1}$$

où $m \in \mathbb{R}$ *est la tendance,* $(\epsilon(\mathbf{x}))_{\mathbf{x}\in D}$ *est un processus Gaussien centré et stationnaire.*

Hypothèse 2. *La fonction de covariance du processus* $(Y(\mathbf{x}))_{\mathbf{x}\in D}$ *est stationnaire i.e.*

$$\begin{aligned} Cov(Y(\mathbf{x}), Y(\tilde{\mathbf{x}})) &= Cov(\epsilon(\mathbf{x}), \epsilon(\tilde{\mathbf{x}})) \\ &= \sigma^2 R(\mathbf{x}, \tilde{\mathbf{x}}) \\ &= \sigma^2 \mathbf{r}(\mathbf{x} - \tilde{\mathbf{x}}), \forall\, (\mathbf{x}, \tilde{\mathbf{x}}) \in D^2, \sigma^2 \in \mathbb{R} \end{aligned}$$

Proposition 1. *Dans le contexte des hypothèses 1 et 2, le prédicteur* $\widehat{Y}$ *linéaire en* $\mathbf{Y}$ *où* $\mathbf{Y} = (\mathbf{Y}(\mathbf{x}^1), \ldots, \mathbf{Y}(\mathbf{x}_n))'$ *et minimisant l'erreur quadratique moyenne (EQM) est :*

- *cas* m *connu*

$$\widehat{Y}(\mathbf{x}) = m + \mathbf{r}(\mathbf{x})' \boldsymbol{R}^{-1}(\mathbf{Y} - m\mathbb{1}_n),\ \widehat{Y}(\mathbf{x}) \in \mathbb{R} \tag{1.2}$$

- *cas* m *inconnu*

$$\widehat{Y}(\mathbf{x}) = \widehat{m} + \mathbf{r}(\mathbf{x})' \boldsymbol{R}^{-1}(\mathbf{Y} - \widehat{m}\mathbb{1}_n),\ \widehat{Y}(\mathbf{x}) \in \mathbb{R} \tag{1.3}$$

avec $\widehat{m} = (\mathbb{1}_n^t \mathbf{R}^{-1} \mathbb{1}_n)^{-1} (\mathbb{1}_n^t \mathbf{R}^{-1} \mathbb{1}_n)$

Dans les expressions 1.2 et 1.3, lorsque le vecteur aléatoire $\mathbf{Y}$ *est remplacé par le vecteur des réalisations* $\mathbf{y}$*, le prédicteur est déterministe et sera noté par la suite* $\widehat{y}$*.*
L'EQM $\widehat{s}^2$ *au point* $\mathbf{x} \in D$ *est (cf [Cressie, 1993]) :*

- *cas* m *connu*

$$\widehat{s}^2(\mathbf{x}) = \sigma^2(1 - \boldsymbol{r}(\mathbf{x})' \boldsymbol{R}^{-1} \boldsymbol{r}(\mathbf{x})),\ \widehat{s}^2(\mathbf{x}) \in \mathbb{R}^+$$

- *cas* m *inconnu*

$$\widehat{s}^2(\mathbf{x}) = \sigma^2 \left(1 - \mathbf{r}(\mathbf{x})^t \mathbf{R}^{-1} \mathbf{r}(\mathbf{x}) + \Gamma\right)\ \widehat{s}^2(\mathbf{x}) \in \mathbb{R}^+$$

Avec $\Gamma = (1 - \mathbb{1}_n' \mathbf{R}^{-1} \mathbf{r}(\mathbf{x}))(\mathbb{1}_n' \mathbf{R}^{-1} \mathbb{1}_n)^{-1} (1 - \mathbb{1}_n' \mathbf{R}^{-1} \mathbf{r}(\mathbf{x}))$

Où $\mathbb{1}_n = (1, \ldots, 1)' \in \mathbb{R}^n$*,* $\boldsymbol{R} \in \mathcal{M}_{n\times n}$ *est la matrice de corrélation du vecteur aléatoire* $(Y(\mathbf{x}^1), \ldots, Y(\mathbf{x}^n))'$*,* $\mathbf{r}(\mathbf{x}) \in \mathbb{R}^n$ *le vecteur des corrélations entre* $Y(\mathbf{x})$ *et le vecteur aléatoire* $(Y(\mathbf{x}^1), \ldots, Y(\mathbf{x}^n))'$*.*

Remarque : $\widehat{y}$ *est appelé le prédicteur de krigeage et* $\widehat{s}^2$ *la variance de krigeage.*

Démonstration. Soit $P(\mathbf{Y}, \mathbf{x})$ le prédicteur de $y(\mathbf{x})$ linéaire en $\mathbf{Y}$. $P(\mathbf{Y}, \mathbf{x})$ s'écrit :

$$P(\mathbf{Y}, \mathbf{x}) = \sum_{i=1}^{n} \lambda_i Y(\mathbf{x}^i) + \lambda_0$$

L'estimateur recherché est non biaisé i.e. $\mathbb{E}([P(\mathbf{Y}, \mathbf{x}) - Y(\mathbf{x})]) = \lambda_0 + m \times (\sum_{i=1}^n \lambda_i - 1)$ doit s'annuler.

Cas m **connu**

Soit le paramètre m considéré comme connu et sans perte de généralité m est prit nul. La contrainte de nulité du biais impose $\lambda_0 = 0$. Il reste ensuite à minimiser l'erreur quadratique moyenne (EQM) :

$$\mathbb{E}\left[(P(\mathbf{Y}, \mathbf{x}) - Y(\mathbf{x}))^2\right]$$

L'estimateur $\boldsymbol{\lambda}\mathbf{Y}$ est non biaisé donc minimiser $\mathbb{E}[(Y(\mathbf{x}) - \boldsymbol{\lambda}\mathbf{Y})^2]$ revient à minimiser :

$$\begin{aligned} Var[Y(\mathbf{x}) - \boldsymbol{\lambda}\mathbf{Y}] &= Var(Y(\mathbf{x})) - 2\boldsymbol{\lambda} Cov(\mathbf{Y}, Y(\mathbf{x})) + \boldsymbol{\lambda} Var(\mathbf{Y}) \boldsymbol{\lambda}' \\ &= f(\boldsymbol{\lambda}) \end{aligned} \tag{1.4}$$

f est minimale en $\widehat{\lambda}$ tel que :

$$\begin{aligned}\frac{\partial f}{\partial \boldsymbol{\lambda}} = 0 \Leftrightarrow \boldsymbol{\lambda} Var(\mathbf{Y}) - Cov(\mathbf{Y}, Y(\mathbf{x})) = 0 \\ \Rightarrow \widehat{\boldsymbol{\lambda}} = Cov(\mathbf{Y}, Y(\mathbf{x}))(Var(\mathbf{Y}))^{-1} \\ \Rightarrow \widehat{\boldsymbol{\lambda}} = \mathbf{r}(\mathbf{x})'\mathbf{R}^{-1}\end{aligned}$$

D'où

$$\widehat{y}(\mathbf{x}) = P(\mathbf{Y}, \mathbf{x}) = \mathbf{r}(\mathbf{x})'\boldsymbol{R}^{-1}\mathbf{y}.$$

et

$$\begin{aligned}\widehat{s}^2(\mathbf{x}) &= \mathbb{E}\left[(P(\mathbf{Y}, \mathbf{x}) - Y(\mathbf{x}))^2\right] \\ &= \sigma^2\left(1 - \mathbf{r}(\mathbf{x})'\mathbf{R}^{-1}\mathbf{r}(\mathbf{x})\right)\end{aligned}$$

Cas m inconnu

Dans ce cas aussi, l'estimateur ne doit pas avoir de biais i.e. $\mathbb{E}([P(\mathbf{Y}, \mathbf{x}) - Y(\mathbf{x})]) = \lambda_0 + m \times (\sum_{i=1}^n \lambda_i - 1)$ doit s'annuler. La seule solution pour l'annuler sans ajouter de contrainte sur m est d'imposer $\sum_{i=1}^n \lambda_i = 1$ et $\lambda_0 = 0$. Donc $P(\mathbf{Y}, \mathbf{x}) = \sum_{i=1}^n \lambda_i y^i$. Le multiplicateur de Lagrange est utilisé pour trouver $\lambda \in \mathbb{R}^n$ qui minimise l'EQM sous les contraintes précédentes :

$$Q(\lambda) = \mathbb{E}\left[(P(\mathbf{Y}, \mathbf{x}) - Y(\mathbf{x}))^2\right] + 2\mu\left(\sum_{i=1}^n \lambda_i - 1\right)$$

où μ est le multiplicateur de Lagrange.
L'Equation (1.4) donne :

$$\begin{aligned}\mathbb{E}\left[(P(\mathbf{Y}, \mathbf{x}) - Y(\mathbf{x}))^2\right] &= Var(Y(\mathbf{x})) - 2\boldsymbol{\lambda} Cov(\mathbf{Y}, Y(\mathbf{x})) + \boldsymbol{\lambda} Var(\mathbf{Y})\boldsymbol{\lambda}' \\ &= \sigma^2\left(1 + \boldsymbol{\lambda}'\mathbf{R}\boldsymbol{\lambda} - 2\boldsymbol{\lambda}'\mathbf{r}(\mathbf{x})\right)\end{aligned}$$

D'où

$$Q(\lambda) = \sigma^2\left(1 + \boldsymbol{\lambda}'\mathbf{R}\boldsymbol{\lambda} - 2\boldsymbol{\lambda}'\mathbf{r}(\mathbf{x})\right) + 2\mu\left(\boldsymbol{\lambda}'\mathbb{1}_n - 1\right)$$

Le système suivant est obtenu en dérivant cette fonction par rapport aux variables λ_i et μ et en annulant les dérivées :

$$\begin{cases} \sigma^2\mathbf{R}\boldsymbol{\lambda} - \sigma^2\mathbf{r}(\mathbf{x}) - \mu\mathbb{1}_n = 0 \\ \boldsymbol{\lambda}'\mathbb{1}_n - 1 = 0 \end{cases}$$

Le système précèdent peut être réécrit matriciellement avec $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_n)^t$. Soit :

$$\Sigma_0 = \begin{pmatrix} \sigma^2\mathbf{R} & \mathbb{1}_n \\ \mathbb{1}_n' & 0 \end{pmatrix} \quad \boldsymbol{\lambda}_0 = \begin{pmatrix} \boldsymbol{\lambda} \\ -\mu \end{pmatrix} \quad c_0 = \begin{pmatrix} \sigma^2\mathbf{r}(\mathbf{x}) \\ 1 \end{pmatrix}$$

et

$$\Sigma_0\boldsymbol{\lambda}_0 = c_0$$

Et donc $\boldsymbol{\lambda}_0 = \Sigma_0^{-1} c_0$, l'inversion de Σ_0^{-1} par bloc donne :

$$\Sigma_0^{-1} = \begin{pmatrix} (\sigma^2\mathbf{R})^{-1} + \frac{\mathbf{R}^{-1}\mathbb{1}_n'\mathbb{1}_n\Sigma^{-1}}{-\sigma^2\mathbb{1}_n'\Sigma^{-1}\mathbb{1}_n} & \frac{\mathbf{R}^{-1}\mathbb{1}_n}{\mathbb{1}_n'\mathbf{R}^{-1}\mathbb{1}_n} \\ \\ \frac{\mathbb{1}_n'\mathbf{R}^{-1}}{\mathbb{1}_n'\mathbf{R}^{-1}\mathbb{1}_n} & \frac{-\sigma^2}{\mathbb{1}_n'\mathbf{R}^{-1}\mathbb{1}_n} \end{pmatrix}$$

Par conséquent, $\boldsymbol{\lambda}^t$ et μ deviennent :

$$\boldsymbol{\lambda}' = \left(\mathbf{r}(\mathbf{x}) + \mathbb{1}_n \frac{1 - \mathbb{1}_n'\mathbf{R}^{-1}\mathbf{r}(\mathbf{x})}{\mathbb{1}_n'\mathbf{R}^{-1}\mathbb{1}_n}\right)' \mathbf{R}^{-1}$$

et

$$\mu = \sigma^2 \frac{1 - \mathbb{1}_n'\mathbf{R}^{-1}\mathbf{r}(\mathbf{x})}{\mathbb{1}_n'\mathbf{R}^{-1}\mathbb{1}_n}$$

D'où :

$$\widehat{y}(\mathbf{x}) = P(\mathbf{Y}, \mathbf{x}) = \widehat{m} + \mathbf{r}(\mathbf{x})'\boldsymbol{R}^{-1}(\mathbf{y} - \widehat{m}\mathbb{1}_n).$$

avec $\widehat{m} = (\mathbb{1}_n'\mathbf{R}^{-1}\mathbb{1}_n)^{-1}(\mathbb{1}_n'\mathbf{R}^{-1}\mathbf{y})$ et

$$\begin{aligned} \widehat{s}^2(\mathbf{x}) &= \mathbb{E}\left[(P(\mathbf{Y}, \mathbf{x}) - Y(\mathbf{x}))^2\right] \\ &= \sigma^2\left(1 - \mathbf{r}(\mathbf{x})'\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \Gamma\right) \end{aligned}$$

Avec $\Gamma = (1 - \mathbb{1}_n'\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}))'(\mathbb{1}_n'\mathbf{R}^{-1}\mathbb{1}_n)^{-1}(1 - \mathbb{1}_n'\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}))$ □

Le modèle est entièrement spécifié dès que l'on connait le noyau de covariance. Les deux structures les plus classiques sont :

- Noyau de covariance anisotrope avec un produit de tenseur :

$$\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{j=1}^{p} \rho_{\theta_j}\left(|x_j - x_j'|\right),\ \boldsymbol{\theta} = (\theta_1, \ldots, \theta_p) \in \mathbb{R}_+^p$$

  Dans la suite du manuscrit, ce noyau sera nommé "le noyau *anisotrope*".
- Noyau de covariance *isotrope* :

$$\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \rho_\theta\left(\|\mathbf{x} - \mathbf{x}'\|_2\right),\ \theta \in \mathbb{R}_+$$

  Dans la suite du manuscrit, ce noyau est nommé "le noyau *isotrope*".

où ρ_{θ_j} est une fonction de corrélation stationnaire qui dépend uniquement d'un paramètre de portée unidimensionnel θ_j, cf [Santner et al., 2003] et [Stein, 1999]. Le noyau *anisotrope* est le plus utilisé car il estime autant de paramètres de portée qu'il y a de variables d'entrée ce qui permet une plus grande flexibilité.

Les paramètres m, σ^2 et $\boldsymbol{\theta}$ sont estimés par maximum de vraisemblance lorsqu'ils ne sont pas connus.

Proposition 2. *Les estimateurs par maximum de vraisemblance de m et σ^2 sont des fonctions de $\boldsymbol{\theta}$, telles que :*

$$\begin{aligned} \hat{m}(\boldsymbol{\theta}) &= (\mathbb{1}_n'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\mathbb{1}_n)^{-1}\mathbb{1}_n'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\mathbf{y} \\ \hat{\sigma}^2(\boldsymbol{\theta}) &= \frac{1}{n}\left(\mathbf{y} - \widehat{m}(\boldsymbol{\theta})\mathbb{1}_n\right)^t\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}(\mathbf{y} - \widehat{m}(\boldsymbol{\theta})\mathbb{1}_n\Big) \end{aligned}$$

où $(\boldsymbol{R}_{\boldsymbol{\theta}}^{-1})_{i,j} = \mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x}_i - \mathbf{x}_j)$, $\forall i, j \in \{1, \ldots, n\}$ et $\boldsymbol{R}_{\boldsymbol{\theta}}^{-1} \in \mathcal{M}_{n,n}$.

Démonstration. Soit $\mathcal{L}$ la fonction de vraisemblance :

$$\mathcal{L}(m, \sigma^2, \boldsymbol{\theta}; \mathbf{x}^1, \ldots, \mathbf{x}^n, \mathbf{y}) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}} |\boldsymbol{R_\theta}|^{\frac{1}{2}}} e^{-\frac{1}{2\sigma^2}(\mathbf{y}-m\mathbb{1}_n)^t \boldsymbol{R_\theta}^{-1}(\mathbf{y}-m\mathbb{1}_n)}$$

$$\begin{aligned} l(m, \sigma^2, \boldsymbol{\theta}) &= -\log(\mathcal{L}) \\ &= \frac{n}{2}\log(2\pi\sigma^2) + \frac{1}{2}\log(|\mathbf{R}_{\boldsymbol{\theta}}|) + \frac{1}{2\sigma^2}(\mathbf{y} - m\mathbb{1}_n)^t \mathbf{R}_{\boldsymbol{\theta}}^{-1}(\mathbf{y} - m\mathbb{1}_n) \end{aligned}$$

Où $\mathcal{L}$ est la fonction de vraisemblance. L'annulation de la dérivée de cette fonction par rapport à m donne :

$$\begin{aligned} \nabla_m(l) = 0 \quad &\Leftrightarrow \tfrac{2}{2\sigma^2}\mathbb{1}_n' \mathbf{R}_{\boldsymbol{\theta}}^{-1}(\mathbf{y} - m\mathbb{1}_n) &&= 0 \\ &\Leftrightarrow \mathbb{1}_n' \mathbf{R}_{\boldsymbol{\theta}}^{-1}\mathbf{y} - \mathbb{1}_n' \mathbf{R}_{\boldsymbol{\theta}}^{-1} m\mathbb{1}_n &&= 0 \end{aligned}$$

d'où

$$\widehat{m} = (\mathbb{1}_n' \mathbf{R}_{\boldsymbol{\theta}}^{-1}\mathbb{1}_n)^{-1}\mathbb{1}_n' \mathbf{R}_{\boldsymbol{\theta}}^{-1}\mathbf{y}$$

L'annulation de la dérivée de cette fonction par rapport à σ^2 donne :

$$\begin{aligned} \tfrac{\partial l}{\partial \sigma^2} = 0 \quad &\Leftrightarrow \tfrac{n}{2}\tfrac{4\pi\sigma}{2\pi\sigma^2} - 2\tfrac{(y-\widehat{m}\mathbb{1}_n)^t \mathbf{R}_{\boldsymbol{\theta}}^{-1}(\mathbf{y}-\widehat{m}\mathbb{1}_n)}{2\sigma^3} &&= 0 \\ &\Leftrightarrow \tfrac{n}{\sigma} - \tfrac{(\mathbf{y}-\widehat{m}\mathbb{1}_n)^t \mathbf{R}_{\boldsymbol{\theta}}^{-1}(\mathbf{y}-\widehat{m}\mathbb{1}_n)}{\sigma^3} &&= 0 \end{aligned}$$

d'où

$$\hat{\sigma}^2 = \frac{(\mathbf{y} - \widehat{m}\mathbb{1}_n)^t \mathbf{R}_{\boldsymbol{\theta}}^{-1}(y - \widehat{m}\mathbb{1}_n)}{n}$$

□

L'estimation du paramètre $\boldsymbol{\theta}$ est donnée par :

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[l(\boldsymbol{\theta}, \hat{\sigma}^2(\boldsymbol{\theta}), \widehat{m}(\boldsymbol{\theta}); \mathbf{x}^1, \ldots, \mathbf{x}^n, \mathbf{y}) \right]$$

la fonction de vraisemblance devient :

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \hat{\sigma}^2(\boldsymbol{\theta}), \widehat{m}(\boldsymbol{\theta}); \mathbf{x}^1, \ldots, \mathbf{x}^n, \mathbf{y}) &= \frac{1}{(2\pi\hat{\sigma}^2(\boldsymbol{\theta}))^{\frac{n}{2}} |\boldsymbol{R_\theta}|^{\frac{1}{2}}} e^{-\frac{1}{2\widehat{\sigma}^2(\boldsymbol{\theta})}\left(\mathbf{y}-\widehat{m}(\boldsymbol{\theta})\mathbb{1}_n\right)^t \boldsymbol{R_\theta}^{-1}\left(\mathbf{y}-\widehat{m}(\boldsymbol{\theta})\mathbb{1}_n\right)} \\ &= \frac{1}{(2\pi\hat{\sigma}^2(\boldsymbol{\theta}))^{\frac{n}{2}} |\boldsymbol{R_\theta}|^{\frac{1}{2}}} e^{-\frac{n}{2}} \end{aligned}$$

et donc

$$l(m, \sigma^2, \boldsymbol{\theta}) = \frac{n}{2}\log 2\pi\hat{\sigma}^2(\boldsymbol{\theta}) + \frac{1}{2}\log \boldsymbol{R_\theta} + \frac{n}{2}$$

La minimisation de $-\log(\mathcal{L})$ permet d'estimer les paramètres de portée $\boldsymbol{\theta}$. Cette minimisation nécessite un algorithme d'optimisation. L'algorithme "L-BFGS-B" de type descente de gradient introduit par [Byrd et al., 1995] peut être utilisé pour maximiser la vraisemblance. Il nécessite le gradient de $-\log(\mathcal{L})$ dont l'expression est connue. Lorsque le gradient n'est pas connu, un algorithme génétique n'utilisant pas les dérivées peut être utilisé. L'estimation des paramètres ainsi que les routines permettant de construire le prédicteur de krigeage sont implémentées dans le package DiceKriging de R cf [Roustant et al., 2012]. Il existe plusieurs fonctions de corrélation ρ_θ possibles. Elles doivent toutes

Gaussien	$\rho_\theta(h) = \exp\left(-\frac{h^2}{2\theta^2}\right)$
Matern5_2	$\rho_\theta(h) = \left(1 + \frac{\sqrt{5}\lvert h\rvert}{\theta} + \frac{5h^2}{3\theta^2}\right)\exp\left(-\frac{\sqrt{5}\lvert h\rvert}{\theta}\right)$
Exponentiel	$\rho_\theta(h) = \exp\left(-\frac{\lvert h\rvert}{2\theta}\right)$

TABLE 1.1 – Fonction de corrélation.

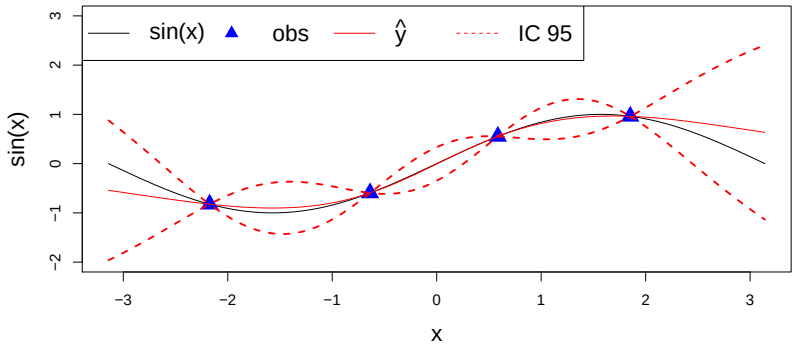

FIGURE 1.1 – La courbe noire représente la fonction sinus, les points représentés par des triangles bleus constituent l'ensemble d'apprentissage, la courbe rouge est la prédiction $\widehat{y}$ par krigeage et les courbes rouges pointillées sont les intervalles de confiance à 95%.

vérifier la condition de semi-définie positivité. Le Tableau 1.1 donne une liste non-exhaustive de fonctions de corrélation. Dans un contexte industriel où la fonction représente une sortie physique comme un rendement, le noyau Matern5_2 est en général le plus adapté. Il est moins lisse que le Gaussien et il est $\mathcal{C}^2$ contrairement à l'Exponentiel qui est $\mathcal{C}^0$.

Exemple : Soit la fonction $f(x) = \sin(x)$ avec $x \in [-\pi; \pi]$. Quatre points d'apprentissage sont choisis bien répartis dans l'espace de définition. Le modèle de krigeage avec une fonction de corrélation Matern5_2 (En 1D le noyau *anisotrope* et *isotrope* sont les mêmes) est estimé. Puis, de nouveaux points sont prédits. Les résultats sont représentés sur la Figure 1.1. La valeur des paramètres estimés se trouve dans le Tableau 1.2. La figure montre que la prédiction par krigeage en rouge passe par les points d'observation (triangles bleus). Les intervalles de confiance en rouges pointillés sont nuls en ces points. Plus le point prédit est loin d'un point d'apprentissage, par exemple en $x = 1.5$, plus les intervalles de confiance sont grands.

$\widehat{m}$	$\widehat{\sigma}^2$	$\widehat{\theta}$
0.03	1.9	0.65

TABLE 1.2 – Paramètres de krigeage estimés par maximum de vraisemblance pour la fonction sinus.

2 Co-Krigeage

Soient d fonctions $f_1, \ldots, f_d$, $d > 1$ coûteuses et corrélées. Le but de ce paragraphe consiste à construire un modèle de co-krigeage prenant en compte la corrélation entre les fonctions. Soit le

vecteur $\mathbf{z} = (z^1, \ldots, z^n)'$ tel que $z^i = (y_1^i, \ldots, y_d^i)$, $i = (i, \ldots, n)$. z^i est la ième observation des sorties correspondant à l'entrée $\mathbf{x}^i$.

Hypothèse 3. $\mathbf{z}$ *est supposé être la réalisation d'un processus Gaussien* $(Z(\mathbf{x}))_{\mathbf{x}\in D}$ *aux point* $\mathbf{x}^1, \ldots, \mathbf{x}^n$ *tel que :*

$$Z(\mathbf{x}) = \mathbf{m} + \epsilon(\mathbf{x}) \tag{1.5}$$

où $\mathbf{m} = (\mu_1, \ldots, \mu_d)' \in \mathbb{R}^d$ *le vecteur des tendances, le processus* $(\epsilon(\mathbf{x}))_{\mathbf{x}\in D}$ *est le vecteur de* d *processus Gaussiens centrés et stationnaires tels que :*

$$\epsilon(\mathbf{x}) = (\eta_1(\mathbf{x}), \ldots, \eta_d(\mathbf{x}))$$

Soient :

$$Y^\ell(\mathbf{x}) = \mu_\ell + \eta_\ell(\mathbf{x}),\ \ell = \{1, \ldots, d\} \tag{1.6}$$

La covariance entre les processus Gaussiens stationnaires Y_ℓ et Y_k avec $\ell, k = \{1, \ldots, d\}$ est :

$$\begin{aligned} Cov(Y_\ell(\mathbf{x}), Y_k(\tilde{\mathbf{x}}); \boldsymbol{\theta}_{\ell,k}) &= Cov(\eta_\ell(\mathbf{x}), \eta_k(\tilde{\mathbf{x}}); \boldsymbol{\theta}_{\ell,k}) \\ &= \Sigma_{Y_\ell, Y_k}(\mathbf{x}, \tilde{\mathbf{x}}), \forall\, (\mathbf{x}, \tilde{\mathbf{x}}) \in \mathbb{R}^p \times \mathbb{R}^p \end{aligned}$$

Lorsque $k = \ell$, la covariance est celle vu dans le cadre précédent du krigeage. Lorsque $k \neq \ell$, la structure de covariance est inconnue. La difficulté du co-krigeage réside dans la définition de cette covariance. En effet, il faut trouver une relation qui permet de garder les propriétés d'une covariance. [Le Gratiet, 2013] propose des structures de covariance admissibles comme le modèle auto régressif utilisé en multi fidélité ou le krigeage avec dérivées.

Proposition 3. *Sous l'Hypothèse 3, la prédiction linéaire non-biaisée qui minimise l'erreur quadratique moyenne au point* $\mathbf{x} \in D$ *est :*

$$\widehat{\mathbf{z}}(\mathbf{x}) = \widehat{\mathbf{m}}(\boldsymbol{\theta}) + \mathbf{c}_{\boldsymbol{\theta}}(\mathbf{x})'\boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1}(\mathbf{z} - \mathbb{1}_{dn}\widehat{\mathbf{m}}(\boldsymbol{\theta})),\ \widehat{\mathbf{z}}(\mathbf{x}) \in \mathbb{R}^d \tag{1.7}$$

avec $\widehat{\mathbf{m}}(\boldsymbol{\theta}) = (\mathbb{1}_{dn}'\boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1}\mathbb{1}_{dn})^{-1}\mathbb{1}_{dn}'\boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1}\mathbf{z}$ *et l'erreur quadratique moyenne (MSE) au point* $\mathbf{x} \in D$ *est :*

$$\widehat{\mathbf{s}}^2(\mathbf{x}) = \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}) - \begin{pmatrix} I_d' & \boldsymbol{c}_{\boldsymbol{\theta}}(\mathbf{x})' \end{pmatrix} \begin{pmatrix} 0 & \mathbb{1}_{dn}' \\ \mathbb{1}_{dn} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{pmatrix}^{-1} \begin{pmatrix} I_d \\ \boldsymbol{c}_{\boldsymbol{\theta}}(\mathbf{x}) \end{pmatrix},\ \widehat{\mathbf{s}}^2(\mathbf{x}) \in \mathcal{M}_{d\times d} \tag{1.8}$$

avec $\mathbb{1}_{dn} = (I_d, \ldots, I_d)' \in \mathcal{M}_{dn\times d}$, $I_d \in \mathcal{M}_{d\times d}$ *est la matrice identité,* $\boldsymbol{\Sigma}_{\boldsymbol{\theta}} \in \mathcal{M}_{dn\times dn}$ *est la matrice de covariance aux points d'observation telle que :*

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}} = \begin{pmatrix} \Sigma_{\mathbf{x}_1,\mathbf{x}_1} & \cdots & \Sigma_{\mathbf{x}_1,\mathbf{x}_n} \\ \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{x}_n,\mathbf{x}_1} & \cdots & \Sigma_{\mathbf{x}_n,\mathbf{x}_n} \end{pmatrix}$$

où

$$\Sigma_{\mathbf{x},\tilde{\mathbf{x}}} = \begin{pmatrix} \Sigma_{Y_1,Y_1}(\mathbf{x}, \tilde{\mathbf{x}}) & \cdots & \Sigma_{Y_1,Y_d}(\mathbf{x}, \tilde{\mathbf{x}}) \\ \vdots & \ddots & \vdots \\ \Sigma_{Y_d,Y_1}(\mathbf{x}, \tilde{\mathbf{x}}) & \cdots & \Sigma_{Y_d,Y_d}(\mathbf{x}, \tilde{\mathbf{x}}) \end{pmatrix},$$

$\mathbf{c}_{\boldsymbol{\theta}}(\mathbf{x}) \in \mathcal{M}_{dn\times d}$ *est la matrice de covariance entre* $\mathbf{x}$ *et les observations et* $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}) \in \mathcal{M}_{d\times d}$ *est la matrice de covariance au point* $\mathbf{x} \in D$.

Démonstration. La démonstration de la prédiction et du MSE suit la même démarche que celle de la **Proposition 1** en prenant le processus $(Z(\mathbf{x}))_{\mathbf{x}\in D}$ à la place du processus $(Y(\mathbf{x}))_{\mathbf{x}\in D}$. La démonstration de la forme de $\widehat{\mathbf{m}}(\boldsymbol{\theta})$ suit la même démarche que celle de la **Proposition 2**. □

Le paramètre $\boldsymbol{\theta}$ est estimé par maximum de vraisemblance, son expression est donnée par :

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[l(\boldsymbol{\theta}, \widehat{\mathbf{m}}(\boldsymbol{\theta}); \mathbf{x}^1, \ldots, \mathbf{x}^n, \mathbf{z}) \right]$$

avec $l = -\log(\mathcal{L})$ et $\mathcal{L}$ est la fonction de vraisemblance, tel que :

$$\mathcal{L}(\boldsymbol{\theta}, \widehat{\mathbf{m}}(\boldsymbol{\theta}); \mathbf{x}^1, \ldots, \mathbf{x}^n, \mathbf{z}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}_{\boldsymbol{\theta}}|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\mathbf{z}-\widehat{\mathbf{m}}(\boldsymbol{\theta})_n\right)' \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1} \left(\mathbf{z}-\widehat{\mathbf{m}}(\boldsymbol{\theta})_n\right)}$$

Exemple : Soit la fonction sinus $f(x) = \sin(x)$, $x \in [-\pi; \pi]$ une fois dérivable. Quatre points d'apprentissage pour la fonction et sa dérivée sont choisis répartis dans l'espace. Les points sont supposés être des réalisations des processus Gaussiens $(Y(x))_{x\in[-\pi;\pi]}$ et $\left(\frac{\partial Y(\tilde{\mathbf{x}})}{\partial \tilde{x}}\right)_{x\in[-\pi;\pi]}$. Le noyau de covariance associé au processus Yest noté k. La structure de covariance est donnée par (cf [Rasmussen and Williams, 2006]) :

$$cov\left(Y(\mathbf{x}), \frac{\partial Y(\tilde{\mathbf{x}})}{\partial \tilde{x}}\right) = \frac{\partial k(\mathbf{x}, \tilde{\mathbf{x}})}{\partial \tilde{x}}$$
$$cov\left(\frac{\partial Y(\mathbf{x})}{\partial x}, \frac{\partial Y(\tilde{\mathbf{x}})}{\partial \tilde{x}}\right) = \frac{\partial^2 k(\mathbf{x}, \tilde{\mathbf{x}})}{\partial x^2}$$

et la matrice de covariance s'écrit :

$$\Sigma_{\mathbf{x},\tilde{\mathbf{x}}} = \begin{pmatrix} \Sigma_{Y,Y}(\mathbf{x}, \tilde{\mathbf{x}}) & \Sigma_{Y,\frac{\partial Y}{\partial \tilde{x}}}(\mathbf{x}, \tilde{\mathbf{x}}) \\ \Sigma_{\frac{\partial Y}{\partial x},Y}(\mathbf{x}, \tilde{\mathbf{x}}) & \Sigma_{\frac{\partial Y}{\partial x},\frac{\partial Y}{\partial \tilde{x}}}(\mathbf{x}, \tilde{\mathbf{x}}) \end{pmatrix}$$

Un modèle de co-krigeage est estimé pour f et f' conjointement, un modèle de krigeage pour f et un modèle de co-krigeage pour f'. La fonction f est prédite en de nouveaux points. Les résultats sont représentés sur la Figure 1.2. La figure montre que la prédiction avec le co-krigeage est plus précise et les intervalles de confiance sont plus petits sur f.

3 Krigeage en grande dimension

Le fonctionnement d'une turbomachine dépend de beaucoup de paramètres géométriques. Le nombre de ces paramètres devient alors problématique pour la modélisation par krigeage que ce soit en terme d'estimation ou de prédiction. En effet, la dimension est directement liée au nombre de paramètres de portée estimés. Plus la dimension est élevée plus l'espace d'optimisation de la vraisemblance augmente ce qui peut empêcher la convergence. De plus, certaines variables n'ont parfois aucune influence sur la sortie. Trois méthodes permettant de traiter les problèmes liés au krigeage en grande dimension sont présentées.

3.1 Sélection de variables : analyse de sensibilité

Une première idée pour améliorer la qualité d'estimation et de prédiction du krigeage en grande dimension est de réduire le nombre de variables d'entrée et donc de faire une analyse de sensibilité (cf [Saltelli et al., 2000]) pour détecter les variables influentes et celles qui ne le sont pas. Lorsque le code

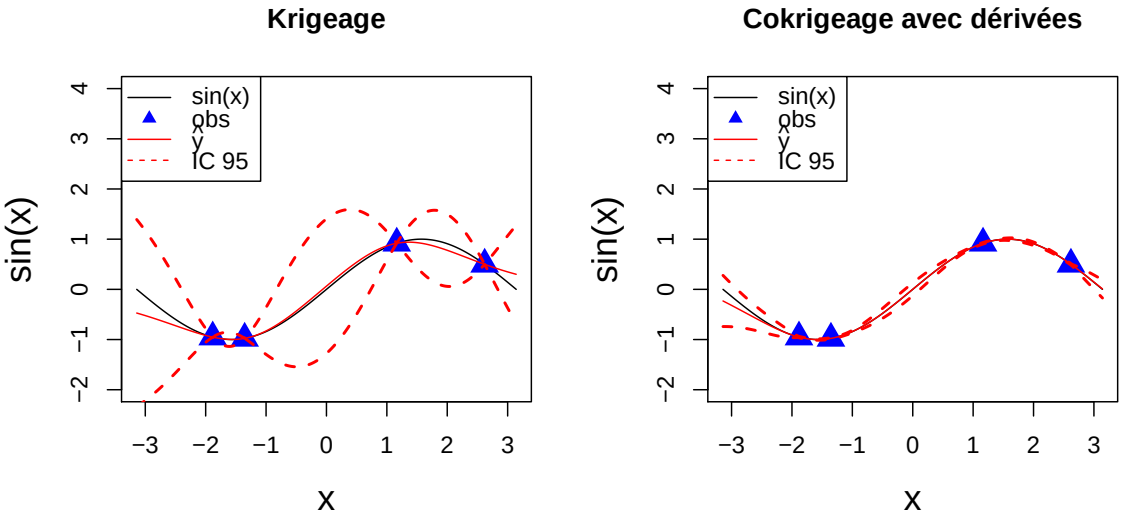

FIGURE 1.2 – La courbe noire représente la fonction sinus, les points représentés par des triangles bleus constituent l'ensemble d'apprentissage, la courbe rouge est la prédiction $\widehat{y}$ par krigeage sans dérivées à gauche et avec à droite. Les courbes rouges pointillées sont les intervalles de confiance à 95%.

est coûteux, [Iooss and Lemaître, 2015] proposent de faire une analyse de sensibilité par le calcul des indices de Sobol ([Sobol, 1993]) directement sur un métamodèle de krigeage. Cependant le résultat de l'analyse de sensibilité sera de mauvaise qualité si le krigeage n'est pas ajusté sur suffisamment de points.

Exemple : Soit la fonction Ishigami $f(\mathbf{x})$, $\mathbf{x} \in [-\pi;\pi]^3$:

$$f(\mathbf{x}) = \sin(x_1) + 7\sin(x_2)^2 + \frac{x_3^4 \sin(x_1)}{10}$$

La fonction Ishigami est considérée comme coûteuse. Deux plans d'apprentissage sont tirés bien répartis dans l'espace de taille 30 et 60. Le modèle de krigeage est ensuite estimé puis les indices de Sobol sont calculés à l'aide de la fonction "SobolEff" du package "sensitivity" de R cf [Monod et al., 2006]. La Figure 1.3 montre qu'avec seulement 30 points d'apprentissage (figure de gauche) les indices de Sobol estimés donnent une indication fausse de l'influence des paramètres. En effet, les résultats montre que la variable x_2 ne semble pas influente et la retirer n'engendrerait pas de perte de qualité alors qu'en réalité elle est très importante (cf indices de Sobol théoriques). Avec 60 points d'apprentissages, les indices de Sobol théoriques et estimés sont presque confondus.

Afin d'avoir une bonne idée de l'influence des paramètres l'estimation du krigeage doit être correcte et nécessite un nombre de points d'apprentissage conséquent. Cependant, cette approche demande une estimation par krigeage de bonne qualité pour obtenir des indices de Sobol estimés proches des théoriques.

3.2 Pénalisation de la vraisemblance

[Yi, 2009] présente dans sa thèse un autre type de méthode. Elles s'inspirent de la sélection de modèle effectuée dans le cadre de la régression linéaire qui consiste à pénaliser la fonction de vraisemblance. Par exemple le krigeage pénalisé de type Bridge revient à estimer les paramètres de portée

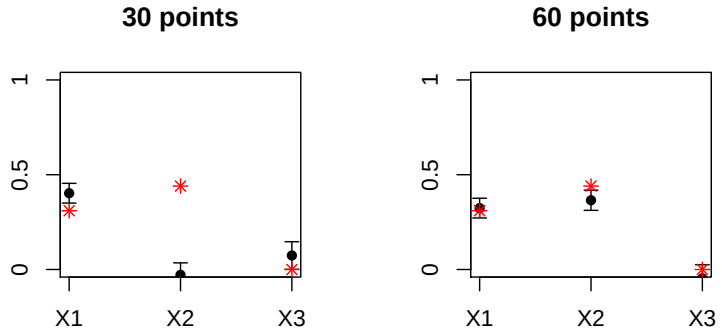

FIGURE 1.3 – Les indices de Sobol théoriques sont représentés par des étoiles rouges et les indices estimés par des points noirs.

en maximisant la fonction de vraisemblance pénalisée :

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[\frac{1}{n} l(\boldsymbol{\theta}; \mathbf{x}^1, \ldots, \mathbf{x}^n, \mathbf{y}) + \lambda \sum_{j=1}^{p} \frac{1}{(\theta_j)^\gamma} \right]$$

Le krigeage Bridge est le cas généralisé du Ridge $\gamma = 2$ et du Lasso $\gamma = 1$. La pénalisation est en $\frac{1}{\theta}$ car une variable non influente aura une valeur de portée très grande (cf [Ben Salem, 2018]). D'autres méthodes de sélection de modèles basées sur la pénalisation de la vraisemblance ont aussi été développées dans sa thèse. Cependant, elles demandent toutes l'estimation de paramètres de pénalisation λ et de régularité γ. [Yi, 2009] a donc construit des algorithmes permettant de les choisir au mieux de façon automatique. La méthode complète pour réduire la dimension est complexe et coûteuse car elle nécessite de multiples procédures d'optimisations imbriquées.

3.3 Modification du noyau de covariance : méthode Fanova

La méthode Fanova (Anova fonctionnelle cf [Muehlenstaedt et al., 2012]) est basée sur la décomposition Anova. Cette méthode gère la grande dimension en modifiant la forme du noyau de covariance ce qui permet de stabiliser l'optimisation de la vraisemblance et donc d'améliorer la prédiction. A partir d'un krigeage initial anisotrope la méthode consiste à construire ce que les auteurs appellent des Fanova Graphes. Il s'agit de graphes, les sommets correspondent à une variable d'entrée. Les poids des sommets correspondent aux indices de Sobol d'ordre 1 tandis que les poids des arêtes aux indices de Sobol d'ordre 2. Une fois seuillé le graphe fait apparaitre des cliques, i.e. des groupes de variables déconnectés les uns des autres. Ces groupes de variables permettent de redéfinir la forme du noyau de covariance. Un processus Gaussien $Y(\mathbf{x})$ représenté par un graphe contenant L cliques $C_1, \ldots, C_L$ s'écrit :

$$Y(\mathbf{x}) = \sum_{l=1}^{L} Y_{C_l}(\mathbf{x}_{C_l})$$

où Y_{C_l} sont des processus Gaussiens stationnaires et indépendants. Par conséquent :

$$Cov\left(Y(\mathbf{x}), Y(\tilde{\mathbf{x}})\right) = \sum_{l=1}^{L} Cov\left(Y_{C_l}(\mathbf{x}_{C_l}), Y_{C_l}(\tilde{\mathbf{x}}_{C_l})\right)$$

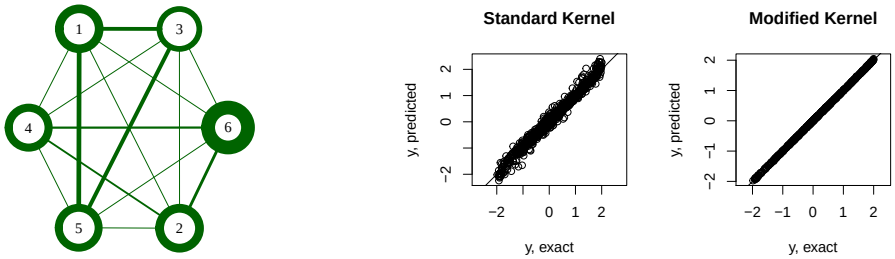

FIGURE 1.4 – La figure de gauche montre un Fanova graphe sur un exemple 6D. La figure de droite montre la prédiction par krigeage contre les vraies valeurs de la fonction avec un noyau standard (*anisotrope*) et un noyau modifié.

Exemple : Soit la fonction suivante :

$$f(\mathbf{x}) = \cos\left[\begin{pmatrix}1 & x_1 & x_2 & x_3\end{pmatrix}\begin{pmatrix}-0,8\\-1,1\\1,1\\1\end{pmatrix}\right] + \sin\left[\begin{pmatrix}1 & x_4 & x_5 & x_6\end{pmatrix}\begin{pmatrix}-0,8\\-1,1\\1,1\\1\end{pmatrix}\right]$$
$$+ (0,5 + 0,35x_3 - 0,6x_4)^2,\ \forall \mathbf{x} \in [-1,1]^6$$

Cent points d'apprentissage sont choisis bien répartis dans l'espace. Puis, un modèle de krigeage est estimé pour construire les graphes Fanova. Enfin, le noyau de covariance du krigeage est modifié à partir des graphes et à nouveau estimé. La Figure 1.4 montre un Fanova graphe à gauche sur un exemple d'une fonction avec six variables d'entrées. Le graphe construit contient deux cliques (x_1, x_3, x_5) et (x_2, x_4, x_6). La modification du noyau permet d'améliorer la qualité de prédiction.

Cette méthode permet donc de stabiliser l'optimisation du maximum de vraisemblance et d'améliorer la qualité de prédiction en modifiant la forme du noyau de covariance. Cependant, elle suppose que le krigeage sur lequel se base les graphes soit d'assez bonne qualité.

3.4 Modification du noyau de covariance : algorithmes "forward"

[Welch et al., 1992] proposent un algorithme itératif pour réduire le nombre de paramètres à estimer dans la fonction de covariance. Le noyau qu'ils proposent d'utiliser est de la forme suivante :

$$\mathbf{r}_{\boldsymbol{\theta},\boldsymbol{\alpha}}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{j=1}^{p} \exp\left(-\theta_j |x_j - \tilde{x}_j|^{\alpha_j}\right)$$

Les paramètres à estimer par maximum de vraisemblance sont $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)$ et $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)$. Au total, il y a donc $2p$ paramètres à estimer. Lorsque p est grand l'estimation par maximum de vraisemblance devient difficile. L'algorithme proposé consiste à initialiser la procédure avec un noyau simple tel que $\theta_1 = \ldots = \theta_p$ et $\alpha_1 = \ldots = \alpha_p$. A chaque étape, la contrainte d'égalité est relâchée sur un paramètre θ et un paramètre α. Le modèle est ré-estimé jusqu'à atteindre un critère d'arrêt basé sur un gain de vraisemblance. Le nombre final de paramètres sera plus petit que $2p$. L'algorithme est décrit en détail ci-après. Par exemple, le noyau suivant peut être obtenu :

$$\mathbf{r}_{\boldsymbol{\theta},\boldsymbol{\alpha}}(\mathbf{x} - \tilde{\mathbf{x}}) = \exp\left(-\theta_1 |x_1 - \tilde{x}_1|^{\alpha_1}\right) \exp\left(-\theta_2 |x_2 - \tilde{x}_2|^{\alpha_2}\right) \prod_{j=3}^{p} \exp\left(-\theta |x_j - \tilde{x}_j|^{\alpha}\right)$$

les contraintes sur les paramètres liés aux deux premières variables d'entrée sont relâchées.

Algorithm [Welch et al., 1992]

Maximize the log-likelihood subject to $\theta_1 = \ldots = \theta_p$ and $\alpha_1 = \ldots = \alpha_p$. The maximum is denoted by l_0 ;
Set $\mathcal{C} = \{1, \ldots, p\}$ and GoOn=TRUE ;
while GoOn **do**
    for each j in $\mathcal{C}$ **do**
        Maximize the log-likelihood subject to $\theta_i = \theta$ and $\alpha_i = \alpha \; \forall i \in \mathcal{C} - \{j\}$. The maximum is denoted by l_j ;
        Set $j^* = \text{argmax}_j \, l_j$;
        if $l_{j^*} - l_0$ is sufficiently large **then**
            $\mathcal{C} = \mathcal{C} - \{j^*\}$ and $l_0 = l_{j^*}$;
        else
            GoOn=FALSE
        end if
    end for
end while

[Marrel et al., 2008] proposent un second algorithme itératif permettant de réduire le nombre de paramètres à estimer. Le noyau de covariance est le même que celui utilisé par [Welch et al., 1992] mais ils ajoutent une fonction de régression au krigeage :

$$Y(\mathbf{x}) = f(\mathbf{x}) + \epsilon(\mathbf{x}),$$

avec $f(\mathbf{x}) = \sum_{j=0}^{p+1} \beta_j f_j(\mathbf{x})$, $f_0(\mathbf{x}) = 1$ et $f_j(\mathbf{x}) = x_j$ pour $j = \{1, \ldots, p\}$. L'algorithme proposé se construit de façon itérative et similaire à l'algorithme de Welch. Il vise à estimer conjointement les directions principales dans la fonction de tendance et dans le noyau de covariance.

Les deux algorithmes sont des premières approches pour construire un krigeage en grande dimension.

4 Optimisation

La dernière problématique liée au cas industriel est l'optimisation. En effet, le but est de trouver la forme géométrique de la turbomachine la plus performante en terme de rendement. Le contexte reste celui des simulations coûteuses. Dans cette section, différentes méthodes d'optimisation basées sur des métamodèles en mono-objectif, en multi-objectif et en optimisation robustes sont introduites.

4.1 Mono-objectif

Le but est de trouver $\mathbf{x}^* \in D \subset \mathbb{R}^p$ tel que :

$$\mathbf{x}^* = \underset{\mathbf{x} \in D}{\text{argmin}} \, f(\mathbf{x})$$

avec f coûteuse. L'optimisation directe (sur la vraie fonction) est considérée comme impossible ou trop chère. Il est donc nécessaire de passer par un métamodèle. L'idée la plus naïve est de remplacer la fonction par le prédicteur de krigeage $\widehat{y}$ et de le minimiser. En général, cette méthode ne fonctionne pas. Le prédicteur de krigeage est obtenu par la minimisation de l'erreur quadratique moyenne. Compte tenu des incertitudes sur la prédiction, les zones où le prédicteur est minimal ne correspondent

pas forcément aux zones où la fonction est minimale. [Jones, 2001] et [Ginsbourger et al., 2010] proposent plusieurs méthodes basées sur le krigeage permettant de procéder à une optimisation globale. Le krigeage permet de prédire la fonction en un point mais aussi de donner une variance en ce même point. Cette variance permet de localiser les zones de l'espace où il y a peu d'informations i.e. les zones non-explorées. L'idée générale est de faire un compromis entre la minimisation de la prédiction et l'exploration des zones où la variance est maximale. Une optimisation multi-objectif peut être conduite :

$$\begin{cases} \min_{\mathbf{x}\in\mathbb{R}^p}(\widehat{y}(\mathbf{x})) \\ \max_{\mathbf{x}\in\mathbb{R}^p}(\widehat{s}(\mathbf{x})). \end{cases}$$

L'optimisation multi-objectif est plus difficile à effectuer qu'en mono-objectif et il se pose un problème du choix du point. En effet, la solution n'est plus un unique point mais un ensemble de points non dominés. Plusieurs critères utilisant les propriétés du krigeage permettent de se passer du multi-objectif.

Définition 1. *La probabilité d'amélioration (PI) est la probabilité que pour tout* $\mathbf{x} \in D$ *la variable aléatoire* Y *soit plus petite que le minimum courant* $\min(y(\mathbb{X}))$ *conditionnellement aux observations* $\mathbf{y}$*.*

$$PI(\mathbf{x}) := \mathbb{P}\left(Y(\mathbf{x}) \leq \min(y(\mathbb{X}))|y(\mathbb{X}) = \mathbf{y}\right)$$

où $\mathbb{X}$ *est l'ensemble d'apprentissage.*

La PI favorise la minimisation de la moyenne au détriment de l'exploration. [Jones, 2001] propose de remplacer $min(y(\mathbb{X}))$ par un seuil pour améliorer la balance entre exploration et minimisation. Cependant, le choix du seuil dépend des cas étudiés. Il est compliqué d'automatiser ce choix et par difficile de généraliser cette méthode.
Le second critère nommé Expected Improvement permet un bon équilibre entre minimisation et exploration.

Définition 2. *Le critère de l'amélioration attendue ou Expected Improvement (EI) cf [Jones et al., 1998] est défini par :*

$$EI(\mathbf{x}) = \mathbb{E}\left[\left(\min(y(\mathbb{X})) - Y(\mathbf{x})\right)^+ |Y(\mathbb{X}) = \mathbf{y}\right]$$

où $\mathbb{X}$ *est l'ensemble d'apprentissage.*

Proposition 4. *La forme analytique de l'EI dans le cas Gaussien est :*

$$EI(\mathbf{x}) = (\min(y(\mathbb{X})) - \widehat{y})\Phi\left(\frac{\min(y(\mathbb{X})) - \widehat{y}}{\widehat{s}}\right) + \widehat{s}\phi\left(\frac{\min(y(\mathbb{X})) - \widehat{y}}{\widehat{s}}\right)$$

où Φ *est la fonction de répartition et* ϕ *la densité de la loi normale centrée réduite.*

Démonstration. Soit f_y la densité de Y et $y_{min} = \min(y(\mathbb{X}))$, l'EI s'écrit :

$$\begin{aligned} EI(\mathbf{x}) &= \mathbb{E}\left[\max\left(\left(y_{min} - Y(\mathbf{x})\right), 0\right)\right] \\ &= \int_{-\infty}^{y_{min}} (y_{min} - y) f_y(y) dy \\ &= \int_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}} (y_{min} - z\widehat{s} - \widehat{m})\phi(z) dz \\ &= \int_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}} (y_{min} - \widehat{m})\phi(z) dz - \widehat{s}\int_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}} z\phi(z) dz \end{aligned}$$

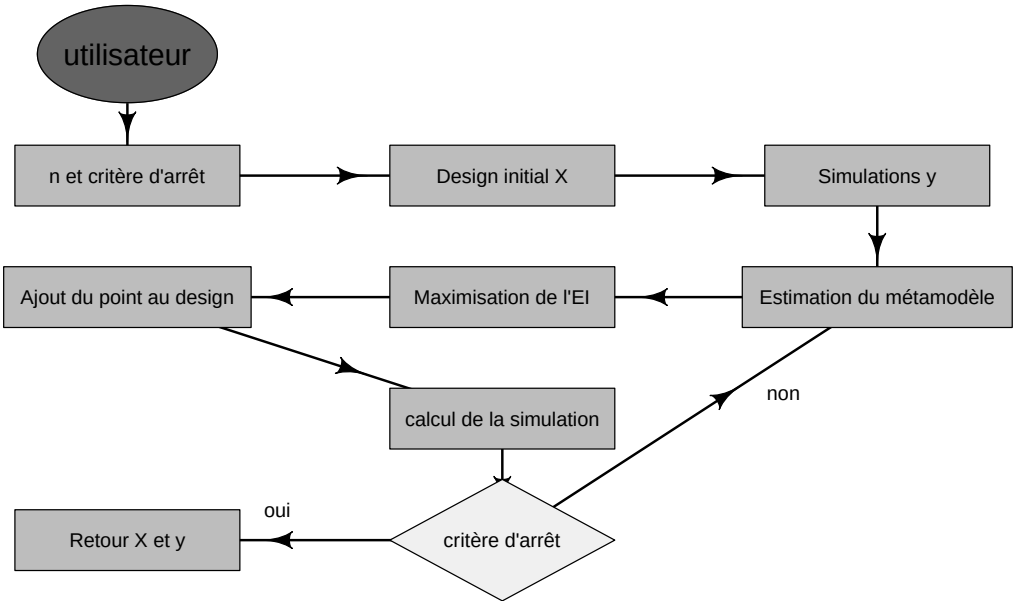

FIGURE 1.5 – Schéma de l'algorithme EGO.

L'écriture de la première intégrale se simplifie :

$$\begin{aligned}\int_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}}(y_{min}-\widehat{m})\phi(z)dz &= (y_{min}-\widehat{m})\int_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}}\phi(z)dz \\ &= (y_{min}-\widehat{m})\left[\Phi(z)\right]_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}} \\ &= (y_{min}-\widehat{m})\Phi\left(\frac{y_{min}-\widehat{m}}{\widehat{s}}\right)\end{aligned}$$

La seconde intégrale s'écrit :

$$\begin{aligned}-\widehat{s}\int_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}}z\phi(z)dz &= -\widehat{s}\int_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}}\frac{1}{\sqrt{2\pi}}ze^{-\frac{z^2}{2}}dz \\ &= -\widehat{s}\frac{1}{\sqrt{2\pi}}\left[-e^{-\frac{z^2}{2}}\right]_{-\infty}^{\frac{y_{min}-\widehat{m}}{\widehat{s}}} \\ &= \widehat{s}\frac{1}{\sqrt{2\pi}}e^{-\frac{\left(\frac{y_{min}-\widehat{m}}{\widehat{s}}\right)^2}{2}} \\ &= \widehat{s}\phi\left(\frac{y_{min}-\widehat{m}}{\widehat{s}}\right)\end{aligned}$$

Donc

$$EI(\mathbf{x}) = (\min(y(\mathbb{X}))-\widehat{y})\Phi\left(\frac{\min(y(\mathbb{X}))-\widehat{y}}{\widehat{s}}\right)+\widehat{s}\phi\left(\frac{\min(y(\mathbb{X}))-\widehat{y}}{\widehat{s}}\right)$$

□

[Jones et al., 1998] proposent une stratégie globale d'optimisation basée sur l'EI : l'algorithme EGO pour Efficient Global Optimization. L'utilisateur commence par choisir le nombre de points initiaux

et un critère d'arrêt qui se traduit souvent par un nombre d'appels maximal à la fonction f coûteuse. Ensuite, un plan d'expérience initial $\mathbb{X}$ est généré. Il correspond, en général, à un hypercube latin optimisé cf [Morris and Mitchell, 1995] et [Jin et al., 2005]. Ce plan permet une excellente répartition des points dans l'espace pour obtenir la meilleure estimation possible. Puis les paramètres du métamodèle sont estimés afin de construire les prédicteurs. L'EGO cherche à maximiser l'EI :

$$\mathbf{x}_{new} = \underset{\mathbf{x}\in\mathbb{R}^p}{\operatorname{argmax}}\, EI(\mathbf{x}).$$

Le point $\mathbf{x}^*$ correspondant au maximum de l'EI est ensuite ajouté au plan $\mathbb{X}$.
Puis, le critère d'arrêt est testé, si celui-ci n'est pas atteint l'algorithme reprend à partir de l'estimation du modèle avec le design enrichi. Le critère d'arrêt correspond souvent à des contraintes imposées par l'industriel comme un budget total ou un temps de calcul maximal alloué pour les évaluations de la fonction coûteuse. L'algorithme EGO est schématisé sur la Figure 1.5. [Vazquez and Bect, 2010] montrent que l'algorithme fournit une suite de points denses dans l'espace à tendance et fonction de covariance connues.

L'exemple suivant met en œuvre l'algorithme EGO sur une fonction réelle de $[0; 1]$.
Exemple : soit la fonction $f(x)$, $x \in [0; 1]$ tel que :

$$f(x) = 10x\sin(10x) + 10x\cos(20x)$$

Quatre points d'apprentissage sont choisis bien répartis dans l'espace puis cinq points sont ajoutés afin de trouver le minimum de la fonction. La Figure 1.6 montre que l'algorithme commence par explorer de nouvelles zones. Au départ, le processus Gaussien estimé est très lisse i.e. le paramètre de portée est grand. Puis l'ajout d'un nouveau point permet au krigeage d'estimer un processus moins lisse et plus proche de la fonction réelle. Le paramètre de portée estimé est plus petit pour permettre les irrégularités. L'ajout de cinq points a été nécessaire pour identifier le minimum global de la fonction. De plus, la figure 1.6 montre que l'EI est multi-modal. C'est pourquoi, dans le package DiceOptim de R (cf [Roustant et al., 2012]), l'EI est maximisé avec un algorithme génétique. Cet algorithme est nommé genoud pour Genetic Optimization Using Derivatives (cf [Mebane Jr et al., 2011]) et se sert des dérivées. Le gradient de l'EI est explicité dans la thèse de [Ginsbourger, 2009]. L'algorithme EGO fonctionne très bien pour l'ajout séquentiel de points. Lorsque les contraintes liés à l'industriel imposent l'ajout de points par batch (plusieurs points en même temps) l'EI ne peut plus être directement utilisé. C'est pourquoi un EI multi-point est introduit.

Définition 3. *L'EI multi-point (q-EI) étudié par [Ginsbourger et al., 2010] est défini par :*

$$\begin{aligned} qEI(\mathbf{X}) &= EI(\mathbf{x}^1, \ldots, \mathbf{x}^q) \\ &= \mathbb{E}\left[\Big(\min(y(\mathbb{X})) - \min(Y(\mathbf{x}^1), \ldots, Y(\mathbf{x}^q))\Big)^+ | Y(\mathbb{X}) = \mathbf{y}\right] \end{aligned}$$

où $\mathbb{X}$ est l'ensemble d'apprentissage et $q \geq 2$ candidats notés $\mathbf{X} = \mathbf{x}^1, \ldots, \mathbf{x}^q$.

[Ginsbourger et al., 2010] présentent une formule analytique pour le cas $q = 2$. [Chevalier and Ginsbourger, 2013] proposent une formule analytique pour le cas général. L'optimisation du critère passe par l'implémentation de son gradient numérique qui permet de traiter les cas raisonnables $q < 10$. Le qEI est maximisé à l'aide du même algorithme génétique que l'EI dans le package DiceOptim de R (cf [Roustant et al., 2012]). Cependant, [Marmin et al., 2015] ont développé un gradient analytique pour accélérer l'optimisation du critère et traité les cas où q est plus grand. L'algorithme associé au qEI est le même que l'EGO à la différence près que l'étape d'optimisation est donnée par :

$$(\mathbf{x}^1_{new}, \ldots, \mathbf{x}^q_{new}) = \underset{(\mathbf{x}^1,\ldots,\mathbf{x}^q)\in\mathbb{R}^{q\times p}}{\operatorname{argmax}} EI(\mathbf{x}^1, \ldots, \mathbf{x}^q)$$

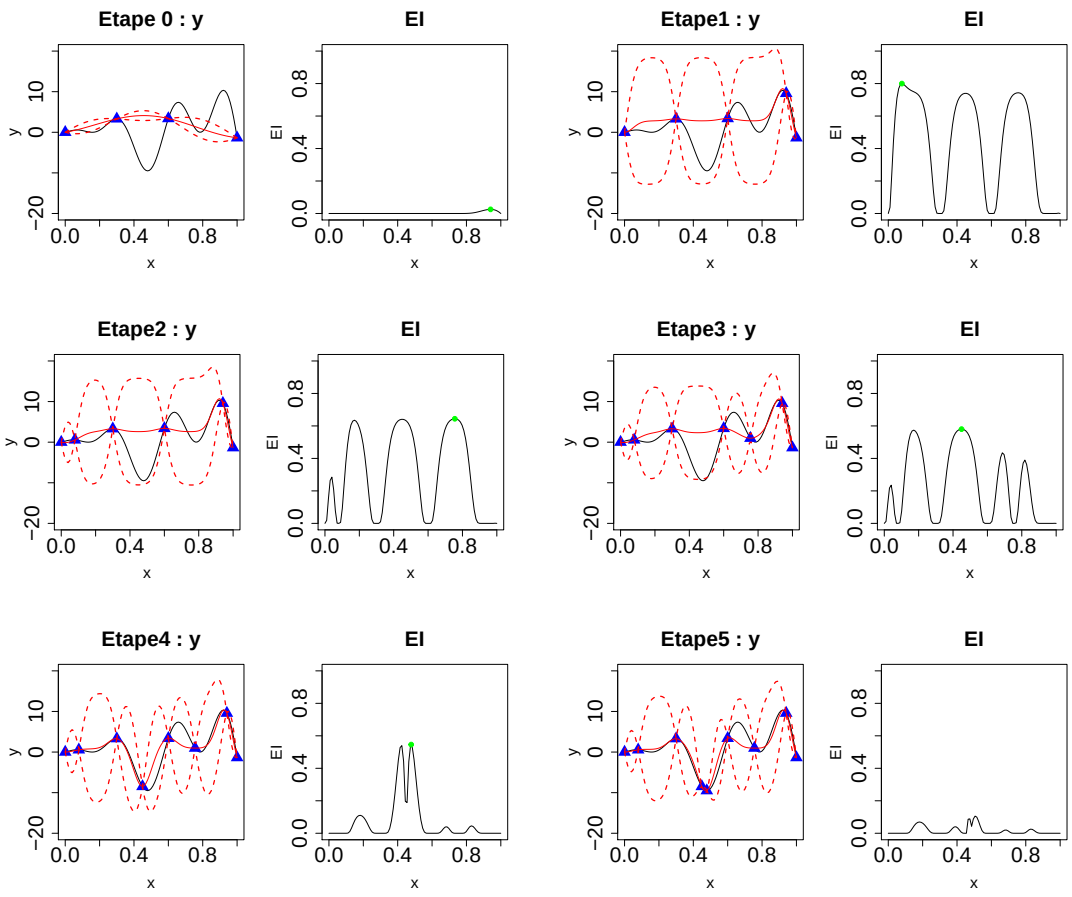

FIGURE 1.6 – Evolution de l'algorithme EGO. La courbe noire représente la vraie fonction, les points bleus les points d'observation et la courbe rouge la moyenne de krigeage.

Le problème d'optimisation est de dimension $p \times q$ au lieu de p ce qui peut rapidement donner des temps de calcul très grand. Deux stratégies utilisant l'EI sont proposées dans l'article de [Ginsbourger et al., 2010] pour permettre l'ajout de points par batch en conservant un temps de calcul raisonnable. Le constant liar **Algorithme CL** et le kriging believer **Algorithme KB** consistent à ajouter des points de façon artificielle au design sans faire appel au simulateur. L'algorithme commence comme celui de l'EGO. Une fois le point maximisant l'EI noté $\mathbf{x}_0^1$ trouvé, il est ajouté au design mais l'observation $y(\mathbf{x}_0^1)$ prendra la valeur du minimum courant $\min(y(\mathbb{X}))$ pour le constant liar ou de $\widehat{y}(\mathbf{x}_0^1)$ pour le kriging believer. Puis l'EI est maximisé jusqu'à atteindre q points $(\mathbf{x}_0^1, \ldots, \mathbf{x}_0^q)$. Enfin, $(f(\mathbf{x}_0^1), \ldots, f(\mathbf{x}_0^q))$ est calculé puis ajouté au design. L'algorithme reprend alors à la première minimisation de l'EI jusqu'à atteindre le budget total.

Algorithm CL

for i from 1 to q **do**
    $\mathbf{x}_0^i = \text{argmax}_{\mathbf{x}\in\mathbb{R}^p} EI(\mathbf{x})$;
    $\mathbb{X}$=$\mathbb{X} \cup \{\mathbf{x}_0^i\}$;
    y=**y** $\cup \{\min(y(\mathbb{X}))\}$;
end for

Algorithm KB

for i from 1 to q **do**
    $\mathbf{x}_0^i = \text{argmax}_{\mathbf{x}\in\mathbb{R}^p} EI(\mathbf{x})$;
    $\mathbb{X}$=$\mathbb{X} \cup \{\mathbf{x}_0^i\}$;
    y=**y** $\cup \{\widehat{y}(\mathbf{x}_0^1)\}$;
end for

4.2 Multi-objectif

Lors de la description du contexte industriel, il a été montré que la maximisation du rendement revenait à minimiser le couple et maximiser la pression. Ce problème peut donc être traité de façon multi-objectif. Soient $f_1, \ldots, f_m$ les m fonctions à minimiser. Les m fonctions à minimiser sont supposées antagonistes. La Figure 1.7 montre deux fonctions antagonistes à gauche (cf [Parr, 2013]) définies par :

$$\begin{cases} f_1 = & \left(x_2 - 5.1(\frac{x_1}{2\pi})^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 1\right) \\ f_2 = & -\sqrt{(10.5 - x_1)(x_1 + 5.5)(x_2 + 0.5)} - \frac{1}{30}\left(x_2 - 5.1(\frac{x_1}{2\pi})^2 - 6\right)^2 \\ & -\frac{1}{3}\left(\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 1\right) \end{cases}$$

et deux fonctions non-antagonistes à droite définies par

$$\begin{cases} f_1 = & x_1 + x_2 \\ f_2 = & \max(x_1, x_2) \end{cases}$$

En effet, la figure de droite montre que la minimisation de f_1 revient à minimiser f_2, elles évoluent dans le même sens et le problème peut facilement s'écrire en mono-objectif. La figure de gauche quant à elle montre que la minimisation de f_1 engendre la maximisation de f_2, elles évoluent dans un sens opposé.

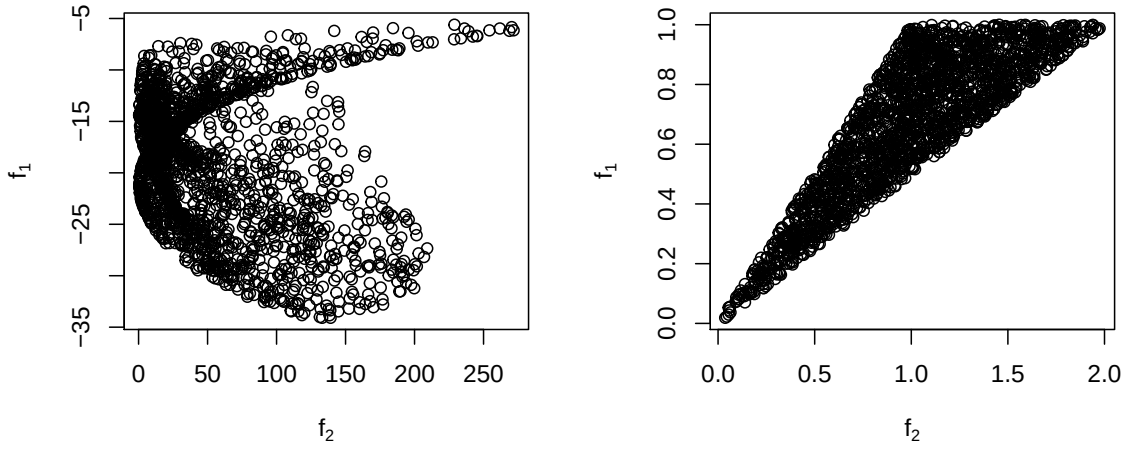

FIGURE 1.7 – Les fonctions f_1 et f_2 sont antagonistes à gauche et ne le sont pas à droite.

Le premier chapitre de [Jozefowiez, 2013] propose une bonne introduction aux outils d'optimisation multi-objectif. L'optimisation de plusieurs fonctions antagonistes donne comme solution un ensemble de points non-dominés qui forment un front de Pareto.

Définition 4. *$\mathbf{x}^1$ domine au sens de Pareto $\mathbf{x}^2$ si et seulement si $\forall i \in 1, \ldots, m$ $f_i(\mathbf{x}^1) \leq f_i(\mathbf{x}^2)$ et $\exists i \in 1, \ldots, m$ tel que $f_i(\mathbf{x}^1) < f_i(\mathbf{x}^2)$.*

Définition 5. *Le point idéal est défini par $f^I = (f_1^I, \ldots, f_m^I)$ avec $f_\ell^I = \min_{x \in D} f_\ell$, $\ell = 1, \ldots, \ell$.*

Définition 6. *Le point nadir est défini par $f^N = (f_1^N, \ldots, f_m^N)$ avec $f_\ell^N = \max_{x \in \mathcal{P}} f_\ell$, $\ell = 1, \ldots, \ell$ avec $\mathcal{P}$ l'ensemble des points non-dominés (le front de Pareto).*

La Figure 1.8 illustre les trois propriétés énoncées.

La littérature contient un grand nombre d'algorithmes d'optimisation multi-objectif, l'algorithme génétique NSGA II décrit et testé dans l'article de [Deb et al., 2002] sera choisi dans la suite de la thèse. Ce dernier est constamment testé et utilisé dans la communauté liée aux méta-heuristiques. Cependant, il est impossible de procéder à une optimisation directe avec cet algorithme sur les fonctions coûteuses. Chaque fonction est donc modélisée par m métamodèles de krigeage indépendants ou par un seul métamodèle de co-krigeage. [Svenson, 2011] montre que l'utilisation du co-krigeage (Nonseparable Linear Model of Coregionalization) apporte un bénéfice minime par rapport à la complexification du modèle. La corrélation éventuelle entre les sorties ne sera donc pas considérée. Cependant, l'algorithme EGO ne peut pas être directement utilisé. L'article de [Wagner et al., 2010] propose une étude des différents algorithmes développés dans le cadre de l'optimisation multi-objectif sur métamodèle de krigeage. Cette étude est sortie en 2011 et depuis, de nouvelles méthodes sont apparues. Elles peuvent être séparées en différentes catégories :

- *agrégation* : Par EGO [Knowles, 2006], WS-EI [Liu et al., 2007] et TA-EI [Zhang et al., 2010]
- *multi-objectif* : EI-EMO [Jeong and Obayashi, 2005]
- *hypervolume* : SMS-EGO [Ponweiser et al., 2008] et EHI [Emmerich et al., 2011]
- *maximin* : EMI [Svenson and Santner, 2016]
- *réduction d'incertitude* : SUR [Picheny, 2015]

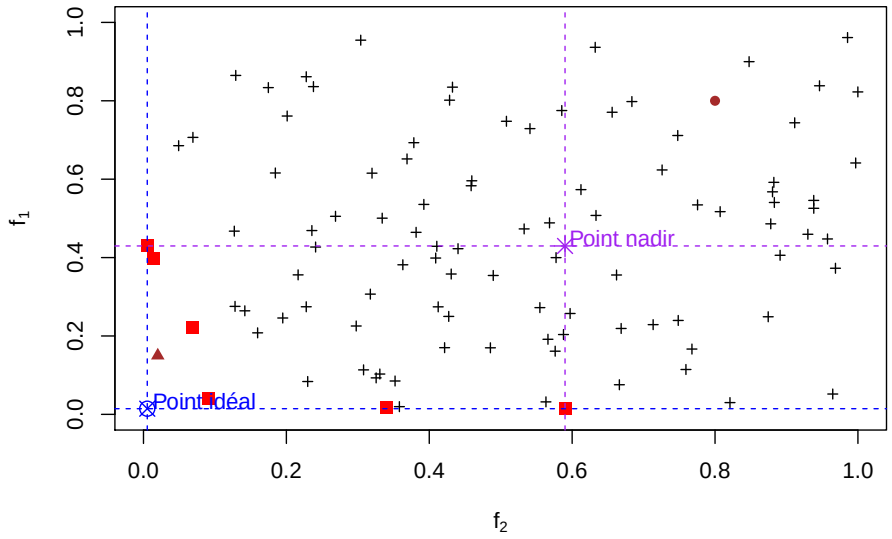

FIGURE 1.8 – Les carrés rouges représentent le font de Pareto (ie les points non-dominés). Le cercle barré bleu représente le point idéal et l'étoile violette le point nadir. Ces deux points sont hypothétiques. Le triangle marron est un point très intéressant dans l'avancée du front alors que le cercle marron ne l'est pas du tout.

Les méthodes d'*agrégation* consistent à transformer les m fonctions objectifs en un seul objectif. Les fonctions sont agrégées :

- soit en faisant une somme pondérée
- soit en utilisant l'approche de Tchebycheff (cf [Zhang et al., 2010]).

Le problème dans ces deux cas est le choix des paramètres de pondération qui peut être compliqué. De plus, lorsque le front de Pareto est complexe ces approches donnent des résultats peu satisfaisants, comme le montre [Henkenjohann and Kunert, 2007]. Les méthodes d'*hypervolume*, de *maximim* et de *réduction d'incertitude* proposent de nouveaux critère de type EI qui tiennent compte de l'optimisation multi-objectif. Une fois le critère d'EI modifié et calculé, l'optimisation multi-objectif basée sur du krigeage en maximisant un seul objectif (l'EI modifié) est lancée. Le package GPareto de R [Binois and Picheny,] permet de procéder à cette optimisation en utilisant ces différents critères. La dernière méthode consiste à maximiser les EIs multi-objectif de chaque fonction. L'EI multi-objectif est défini comme suit où l'on a changé la valeur de référence.

Définition 7. *Le critère de l'EI multi-objectif est (cf [Jeong and Obayashi, 2005]) :*

$$EI_{Y_\ell}(\mathbf{x}) = \mathbb{E}\left[\left(\max_{\mathbb{X}^*} Y_\ell - Y_\ell(\mathbf{x})\right)^+ | Y_\ell(\mathbb{X}) = y_\ell(\mathbb{X})\right],\ \ell = 1, \ldots, m$$

où $\mathbb{X}^*$ est l'ensemble des points non-dominés de l'ensemble d'apprentissage $\mathbb{X}$

La Figure 1.9 montre l'importance de prendre le point nadir comme seuil de référence dans l'optimisation multi-objectif à la place du minimum courant (point idéal en multi-objectif) pris dans le cas du mono-objectif. En effet, le point $\mathbf{x}$ (représenté par un triangle bleu) est un point très intéressant dans l'avancée du front et pourtant avec comme seuil de référence le point idéal, les EIs seraient nuls pour les deux fonctions car $\min(f_1(\mathbb{X})) < f_1(\mathbf{x})$ et $\min(f_2(\mathbb{X})) < f_2(\mathbf{x})$. Alors qu'en prenant le point nadir en référence les EIs seront non nuls et certainement élevés car $\max_{\mathbb{X}^*} f_1 > f_1(\mathbf{x})$, $\max_{\mathbb{X}^*} f_2 > f_2(\mathbf{x})$.

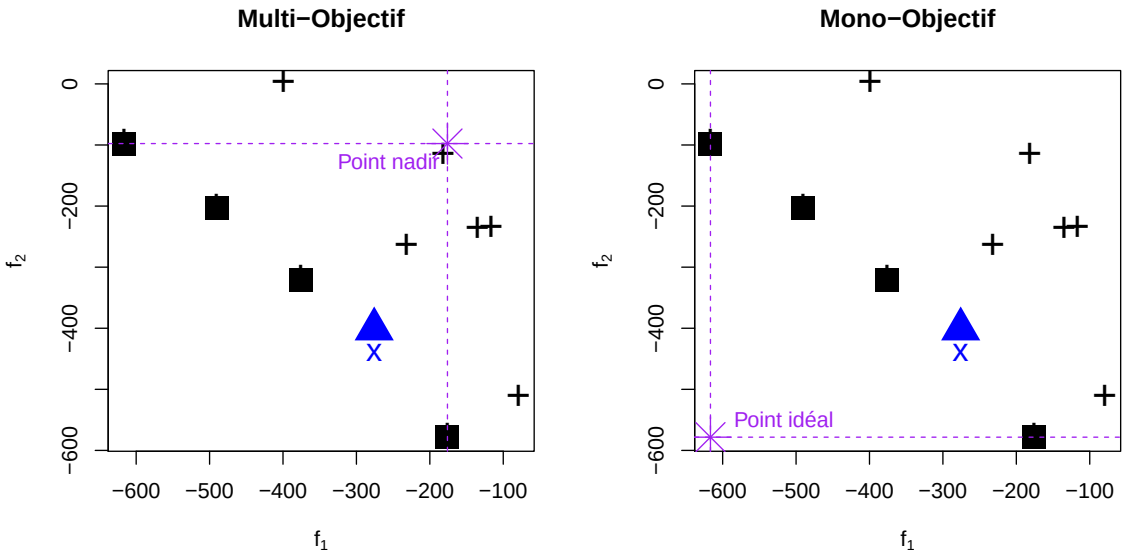

FIGURE 1.9 – Ces figures sont tracées dans l'espace des objectifs (f_1, f_2). Les croix et les carrés représentent les points d'observation de l'ensemble d'apprentissage $\mathbb{X}$. Les carrés sont les points non-dominés (front de Pareto). L'étoile violette représente le point nadir à gauche et le point idéal à droite.

4.3 Optimisation Robuste

Lors de la fabrication des turbomachines les aléas provoqués par les machines industrielles peuvent engendrer une perte conséquente du rendement. En effet, avant de lancer la production, les ventilateurs sont dessinés sur ordinateur puis construits. À la fin de la production les techniciens sélectionnent aléatoirement quelques turbomachines et mesurent la géométrie pour comparer celle du dessin technique initial avec celle du produit final (conformité au dessin de définition). La Figure 1.10 montre la localisation des erreurs de production, en rouge les erreurs importantes et en vert les erreurs minimes. Malheureusement, ces erreurs ne peuvent être évitées pour le moment, c'est pourquoi elles doivent être prises en compte lors de l'optimisation. Le but est de trouver un optimum peu impacté par les variations des entrées. La Figure 1.11 montre deux fonctions (1D à gauche et 2D à droite) avec deux minimums. Pour les deux fonctions, le minimum représenté par un carré est plus robuste que celui dessiné avec un cercle. Les formes rouges sont obtenues par une perturbation dans la première direction. La figure montre que le carré rouge reste proche de l'optimum en noir en terme d'objectif alors que le cercle rouge est loin de son optimum en noir. Notons, $\mathbf{x}^1 \in \mathbb{R}^2$ le point carré et $\mathbf{x}^2 \in \mathbb{R}^2$ le cercle, alors $|f(\mathbf{x}^1) - f(\mathbf{x}^1 + (\epsilon, 0)')| < |f(\mathbf{x}^2) - f(\mathbf{x}^2 + (\epsilon, 0)')|$ avec $\epsilon > 0$ la perturbation.

[Lelièvre et al., 2016] proposent une classification des approches permettant de gérer les incertitudes en optimisation, fiabilité et robustesse. Le cadre de la thèse est le cas où la fonction à optimiser de façon robuste n'a pas de contraintes liées à la fiabilité (section 4.3 de l'article). Ils classent aussi les incertitudes en deux types : celles liées à l'environnement et aux conditions d'utilisation et celles liées aux processus de fabrication. Dans la suite de la section, deux méthodes qui traitent du premier type d'incertitudes puis du second sont présentées. D'autre part, la quantification de la robustesse n'est pas évidente. Plusieurs critères ont été développés pour optimiser une fonction de façon robuste. Les articles de [Gabrel et al., 2014], [Göhler et al., 2016] et [Coco et al., 2014] en donnent un aperçu. Le contexte des simulations coûteuses impose l'utilisation d'un métamodèle lors de l'optimisation robuste.

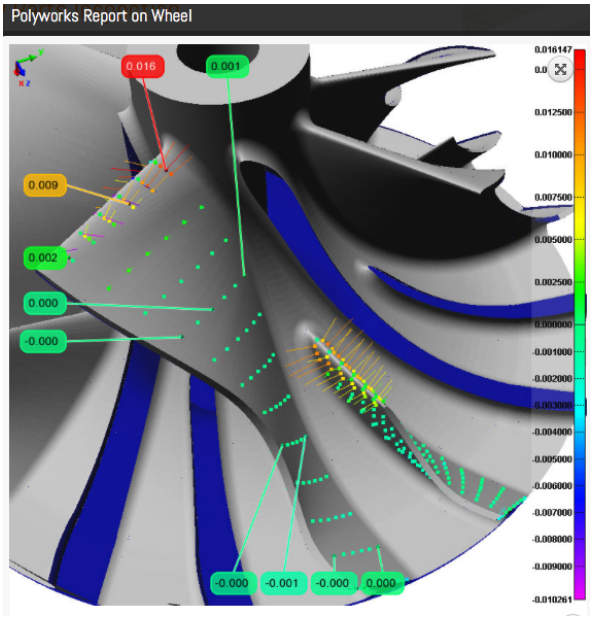

FIGURE 1.10 – Dessin par ordinateur d'une turbomachine quelconque. Les points rouges représentent les zones où les mesures sur la turbomachine construite (réelle) et celles faites par ordinateur sont très différentes. Plus les points se rapprochent de la couleur verte plus les différences de mesures sont insignifiantes.

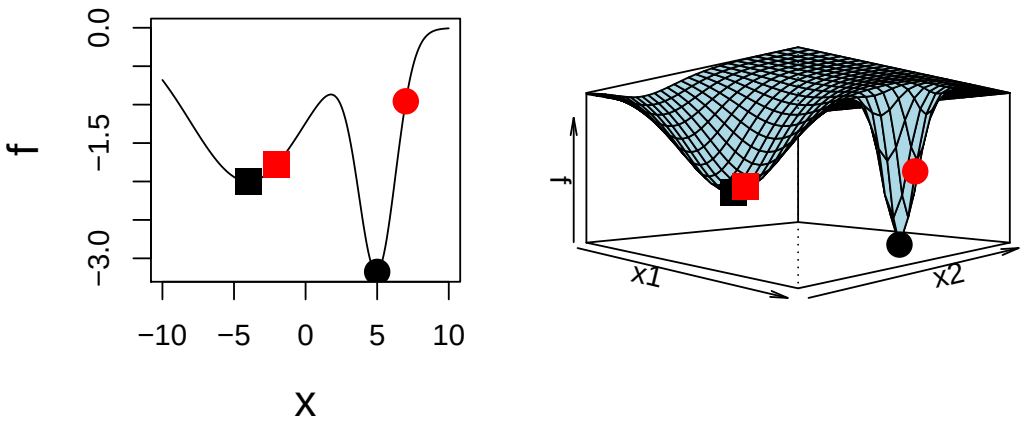

FIGURE 1.11 – Les deux figures illustrent la robustesse en 1D à gauche et 2D à droite. L'optimum carré est plus robuste que le cercle. Les points rouges sont obtenus après une perturbation en x pour la fonction en 1D et en x_1 pour la fonction en 2D.

4.3.1 Optimisation robuste en présence d'incertitude sur les variables d'environnement

[Janusevskis and Le Riche, 2013] proposent une modélisation permettant de minimiser une fonction prenant en entrée des variables de design incertaines sur tout le domaine. Le problème consiste à écrire :

$$\min_{\mathbf{x}\in D} \mathbb{E}_{\mathbf{U}} \left[f(\mathbf{x}, \mathbf{U}) \right]$$

où le vecteur des variables de design est noté $\mathbf{x} \in D$ et celui des variables incertaines est $\mathbf{U} \in \mathbb{R}^q$ avec $\mathbf{U}$ de loi μ. Soit $\mathbf{u}$ une réalisation de $\mathbf{U}$. $f(\mathbf{x}, \mathbf{u})$ est supposée être la réalisation du processus Gaussien $(Y_{(\mathbf{x},\mathbf{u})})$. Soit $\widetilde{Y}_{(\mathbf{x},\mathbf{u})}$ le GP conditionnel aux observations :

$$\widetilde{Y}_{(\mathbf{x},\mathbf{u})} = \left[Y_{(\mathbf{x},\mathbf{u})} | Y_{(\mathbb{X},\mathbb{U})} = \mathbf{Y} \right]$$

L'optimisation avec variables incertaines est menée sur le Processus Gaussien $Z_{(\mathbf{x})}$ défini par l'espérance par rapport à la loi de $\mathbf{U}$ du processus $\widetilde{Y}$:

$$Z_{(\mathbf{x})} = \mathbb{E}_{\mathbf{U}} \left[\widetilde{Y}_{(\mathbf{x},\mathbf{U})} \right] = \int_{\mathbb{R}^q} \widetilde{Y}_{(\mathbf{x},\mathbf{u})} d\mu(\mathbf{u})$$

L'algorithme de minimisation robuste est proche de celui de l'EGO à la différence près que l'EI est adapté à ce contexte d'incertitude. Le critère permettant de choisir le nouveau point $\mathbf{x}$ à ajouter au design est le suivant :

$$EI_z(\mathbf{x}) = \mathbb{E} \left[\min_{\mathbf{x}} \mathbb{E} \left[Z_{(\mathbf{x})} \right] - Z_{(\mathbf{x})} \right]^+$$

Le point $\mathbf{x}$ à ajouter étant fixé, le choix de la valeur $\mathbf{u}$ (réalisation de $\mathbf{U}$) se fait par un critère SUR.
Les articles de [Marzat et al., 2013] et [Ur Rehman and Langelaar, 2015] proposent aussi de procéder à une optimisation robuste globale en se basant sur un critère du "pire des cas". Ces approches ne correspondent pas au contexte de cette thèse. En effet, les incertitudes considérées dans le manuscrit sont liées au processus de fabrication plutôt qu'à celles liées à l'environnement.

4.3.2 Optimisation robuste en présence d'incertitudes sur les variables de design

L'article [Ur Rehman et al., 2014] propose une optimisation robuste plus locale qui se rapproche du contexte industriel de la thèse.

$$\min_{\mathbf{x}\in D} f(\mathbf{x})$$

où $f(\mathbf{x}) \in \mathbb{R}$ est la fonction objectif et $\mathbf{x} \in D$ sont les variables de design. Ils supposent que le code est affecté par une erreur d'implémentation $\mathbf{\Delta} \in \mathcal{U}$, $\mathcal{U}$ est l'espace incertain. $\mathbf{x}$ dévie de sa valeur nominale de $\mathbf{\Delta}$, telle que la nouvelle valeur est $\mathbf{x} + \mathbf{\Delta}$. La loi de l'espace incertain est supposée inconnue. Le problème robuste est :

$$\min_{\mathbf{x}} \max_{\mathbf{\Delta}} f(\mathbf{x} + \mathbf{\Delta})$$

Leur méthode consiste à modifier l'algorithme EGO en procédant à la maximisation d'un EI adapté EI_c. Cet EI nécessite une optimisation préalable afin de trouver la valeur de référence qui correspond à l'optimum robuste du métamodèle noté r tel que :

$$r = \min_{\mathbf{x}} \max_{\mathbf{\Delta}} \hat{y}(\mathbf{x} + \mathbf{\Delta})$$

Ensuite la maximisation de l' EI_c permet de trouver $\mathbf{x}_{new}$ à ajouter à l'ensemble d'apprentissage :

$$EI_c(\mathbf{x}) = (r - \widehat{y})\Phi\left(\frac{r - \widehat{y}}{\widehat{s}}\right) + \widehat{s}\phi\left(\frac{r - \widehat{y}}{\widehat{s}}\right)$$

En résumé cette méthode est très proche de l'algorithme EGO à la différence près que la valeur de référence n'est plus le minimum courant mais un minmax (r) calculé sur la surface de réponse. Elle nécessite une étape supplémentaire d'optimisation avant la maximisation de l'EI et ne prend pas en compte les incertitudes locales dans le choix du nouveau point.

5 Critères et métriques

5.1 Critères de qualité de prédiction

Cette section propose différents critères de qualités de prédiction permettant de comparer les modèles entre eux. Les modèles statistiques sont estimés sur un ensemble d'apprentissage et validés sur un ensemble test. C'est sur ce second ensemble que les critères de qualité de prédiction sont calculés. Soit la fonction f avec $f : D \subset \mathbb{R}^p \to \mathbb{R}$ quelconque, $p \geq 1$. Soit $\widehat{y}(\mathbf{x})$ la prédiction de la fonction f au point $\mathbf{x} \in \mathbb{R}^p$ obtenue par un modèle quelconque. Soit $\mathbf{x}^1, \ldots, \mathbf{x}^n$ les n points de l'ensemble test, $\mathbf{x}^i \in \mathbb{R}^p,\ i = 1, \ldots, n$. Dans la suite de cette section, trois critères de qualité de prédiction sont décrits.

Le premier critère est le RMSE pour Root Mean Square Error en anglais. Ce critère représente l'erreur standard de déviation des prédictions par rapport aux valeurs réelles. La forme du RMSE est la suivante :

$$\text{RMSE} = \frac{1}{n}\sqrt{\sum_{i=1}^{n}\left(f(\mathbf{x}^i) - \widehat{y}(\mathbf{x}^i)\right)^2}$$

Plus le RMSE est proche de zéro plus la qualité de prédiction est bonne. Ce critère dépend du domaine de variation de la fonction.

Le second critère mesure la corrélation multiple. Il représente la proportion de variance totale de la fonction f prise en compte par la fonction de prédiction $\widehat{y}$. Habituellement, ce critère est noté $R2$ car il est calculé sur la base d'apprentissage. Lorsqu'il est calculé sur une base de validation (ensemble test), il est noté $Q2$, sa forme est la suivante :

$$\text{Q2} = 1 - \frac{\sum_{i=1}^{n}(f(\mathbf{x}^i) - \widehat{y}(\mathbf{x}^i))^2}{\sum_{i=1}^{n}\left(f(\mathbf{x}^i) - \frac{1}{n}\sum_{j=1}^{n} f(\mathbf{x}^j)\right)^2}$$

Le critère Q2 est toujours inférieur à 1. Plus il est proche de 1 plus la qualité de prédiction est bonne.

Le dernier critère représente une erreur relative absolue (cf [Van Loan, 1996]). Elle mesure la distance des prédictions aux vraies valeurs en normalisant par rapport à la valeur de la réponse. Elle se nomme errRelat et sa forme est la suivante :

$$\text{errRelat} = \frac{1}{n}\sum_{i=1}^{n}\frac{|f(\mathbf{x}^i) - \widehat{y}(\mathbf{x}^i)|}{|f(\mathbf{x}^i)|}$$

L'erreur relative absolue varie dans l'intervalle $[0; 1]$. Plus elle est proche de 0 plus la qualité de prédiction est bonne.

5.2 Métriques de qualité des fronts de Pareto

La quantification de la qualité d'un front de Pareto peut passer par des distances au front théorique ou par des mesures de qualité directement sur le front de Pareto empirique. Soient $\mathbf{f} = (f_1, \ldots, f_m)$ les m fonctions objectif, $\mathcal{P}$ le front de Pareto théorique et $\mathbb{X}^*$ le front de Pareto empirique. La première métrique de performance utilisée est l'IGD (Inverted Generational distance) décrite en détail dans l'article de [Coello and Sierra, 2004]. Dans ce cas, le front théorique $\mathcal{P}$ est décrit par un ensemble de N points notés $(x^i, \ldots, x^N)$. La forme de la métrique est la suivante :

$$IGD(\mathbb{X}^*) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} d_i^2}$$

où $d_i = \min_{\mathbf{x}\in\mathbf{X}^*}(\|\mathbf{f}(\mathbf{x}^i) - \mathbf{f}(\mathbf{x})\|_2)$, $\mathbf{f}(\mathbf{x}^i) \in \mathcal{P}$. La figure 1.12 montre que plus le front de Pareto empirique est proche du théorique plus la mesure IGD est faible.

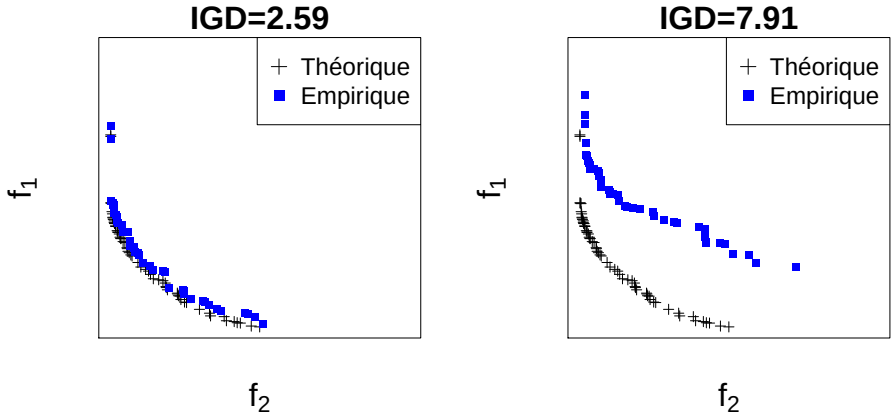

FIGURE 1.12 – Mesure de la distance IGD de deux fonts de Pareto empiriques (carrés bleus) par rapport au front de Pareto théorique (croix noires).

La seconde métrique utilisée mesure le volume du front de Pareto à partir d'un point de référence. Ce volume est nommé hypervolume (HV), il est décrit en détail dans l'article de [Zitzler and Thiele, 1999]. [Fonseca et al., 2006] ont introduit un algorithme pour le calculer. Il est codé en R dans la fonction "dominated_hypervolume" du package "emoa". La figure 1.13 montre que le front de Pareto théorique est celui qui a le plus grand hypervolume, le front empirique le plus avancé à un hypervolume proche du théorique. L'hypervolume dépend fortement de la valeur de référence souvent choisie en fonction des bornes supérieures des fonctions étudiées. Si celles-ci ne sont ne sont pas disponibles, le point de nadir peut alors être utilisé comme point de référence. Dans ce cas, l'hypervolume donnera une indication sur la dispersion du front de Pareto. Il sera possible d'observer des fronts de Pareto moins avancés avec un plus grand hypervolume comme le montre la figure 1.14. Si le front théorique ou les bornes supérieures de la fonction sont disponibles le point de référence sera alors fixe. La mesure du HV permettra de comparer deux fronts empiriques.

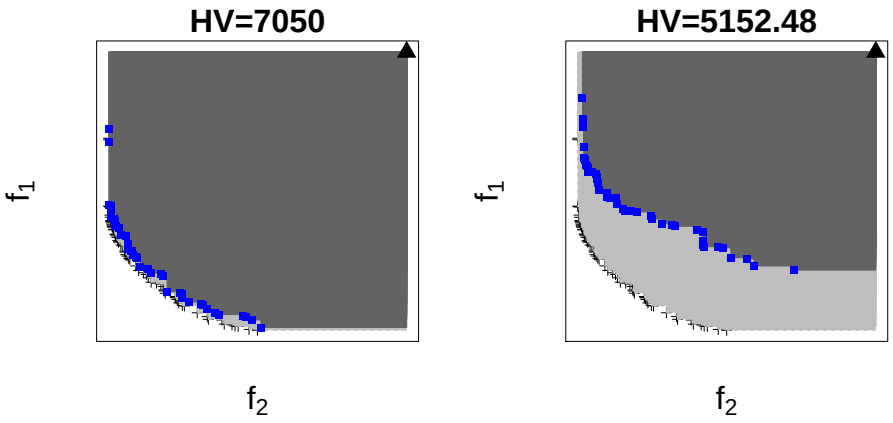

FIGURE 1.13 – Visualisation des hypervolumes du front théorique en gris clair (7214.31) et des deux fronts empiriques en gris foncé (7050 à gauche et 5152.48 à droite). Le point de référence est représenté par le triangle, il correspond aux bornes supérieures des fonctions f_1 et f_2.

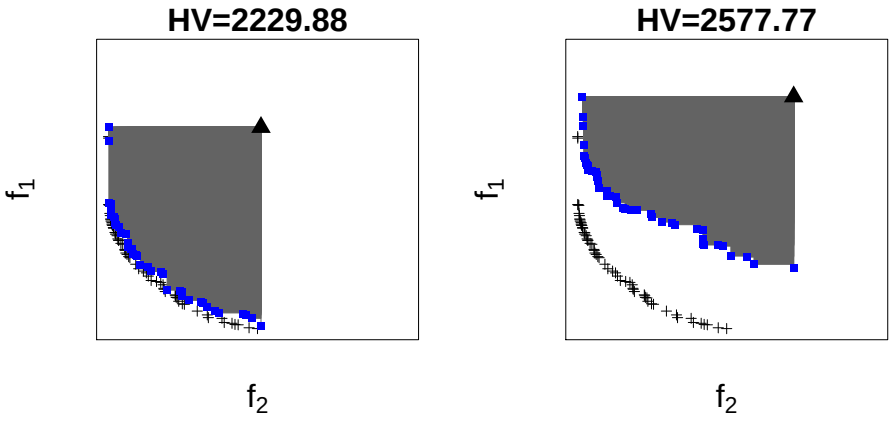

FIGURE 1.14 – Visualisation des hypervolumes des deux fronts empiriques en gris foncé (2229.88 à gauche et 2577.77 à droite). L'hypervolume théorique est de 2161.57. Les points de référence sont représentés par les triangles, ils correspondent aux point de nadir.

Chapitre 2

Un nouveau noyau de covariance pour réduire la dimension et quatre algorithmes pour sa construction

Introduction du chapitre

Les surfaces de réponse actuellement utilisées par Valeo pour répondre aux appels d'offre client sont en dimension 11 ou 15. Les paramètres considérés par ces surfaces sont uniquement liés à la géométrie de l'hélice et non au module thermodynamique complet. Il est possible que l'hélice trouvée ne convienne pas en terme d'acoustique, de mécanique ou même d'encombrement. C'est pourquoi, ils souhaitent passer à des surfaces en dimension 30 ou 60 qui prennent en considération les paramètres liés au module thermique complet. Il y a réellement un changement d'échelle dans le krigeage lorsque le nombre de paramètres d'entrée passe de 15 à 30 ou 60. Or, l'apprentissage d'un modèle statistique peut être difficile en grande dimension. Dans le cas du krigeage :

- soit le nombre d'observations et de paramètres est élevé ce qui rend l'inversion de la matrice de corrélation difficile.
- soit le nombre d'observations est inférieur à dix fois la dimension ce qui engendre des problèmes d'estimation des paramètres.

Cette mauvaise estimation peut altérer fortement la qualité prédictive. Dans ce chapitre, la seconde possibilité est considérée dans le cas du krigeage. Ce problème se pose souvent dans l'industrie lorsque les simulations sont coûteuses. Soit la fonction Six-Hump Camel définie par :

$$f(\mathbf{x}) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1x_2 + \left(-4 + 4x_2^2\right) x_2^2, \mathbf{x} \in [-2; 2] \times [-1; 1]$$

avec les entrées adimensionnées sur $[0; 1]^2$.

Étant donné que cette fonction a seulement deux entrées x_1 et x_2, peu de points seront observés pour se ramener à un contexte similaire à celui de la grande dimension. La Figure 2.1 de gauche montre les lignes de niveaux de la fonction et les points d'observations (croix noires). Le noyau de covariance choisi pour le krigeage est anisotrope. La forme de la vraisemblance sur la Figure 2.1 de droite montre que la fonction augmente jusqu'à atteindre une zone plate (zone grisée sur la figure). L'algorithme aurait pu s'arrêter en tout point de cette zone, la vraisemblance aurait eu la même valeur. L'estimation et la qualité de prédiction du krigeage sont très instables.

Les données ne sont pas assez nombreuses par rapport à la dimension pour obtenir une estimation précise des paramètres de portées $\boldsymbol{\theta}$. En augmentant le nombre d'observations de 9 à 54 comme sur

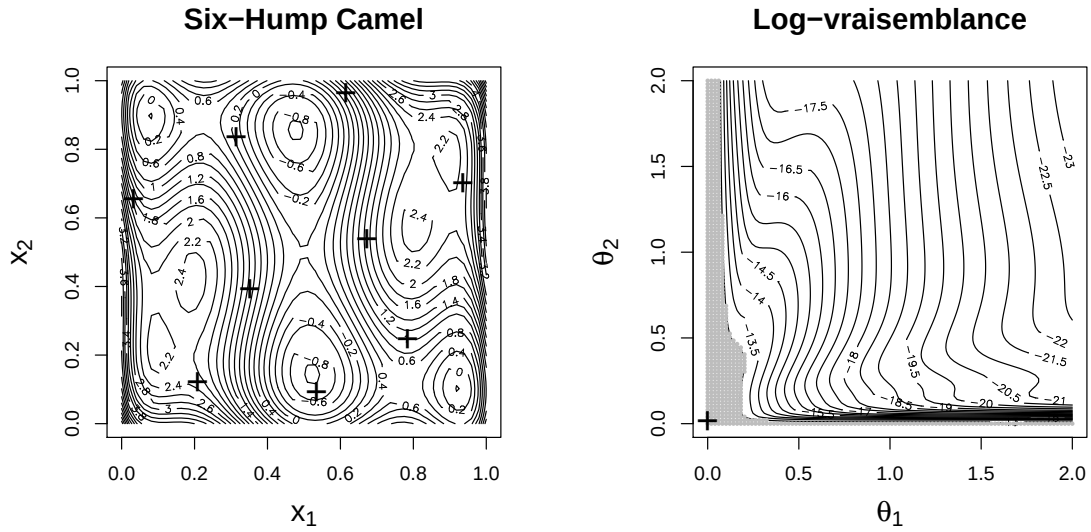

FIGURE 2.1 – La Figure de gauche représente les lignes de niveau de la fonction Six-Hump Camel avec les 9 points d'observations (croix noires). La figure de droite représente les lignes de niveau de la fonction de vraisemblance, la croix noire en bas à gauche est le $\widehat{\boldsymbol{\theta}}$ estimé et la zone grisée correspond à l'aire dans laquelle la vraisemblance est plate (elle a la même valeur).

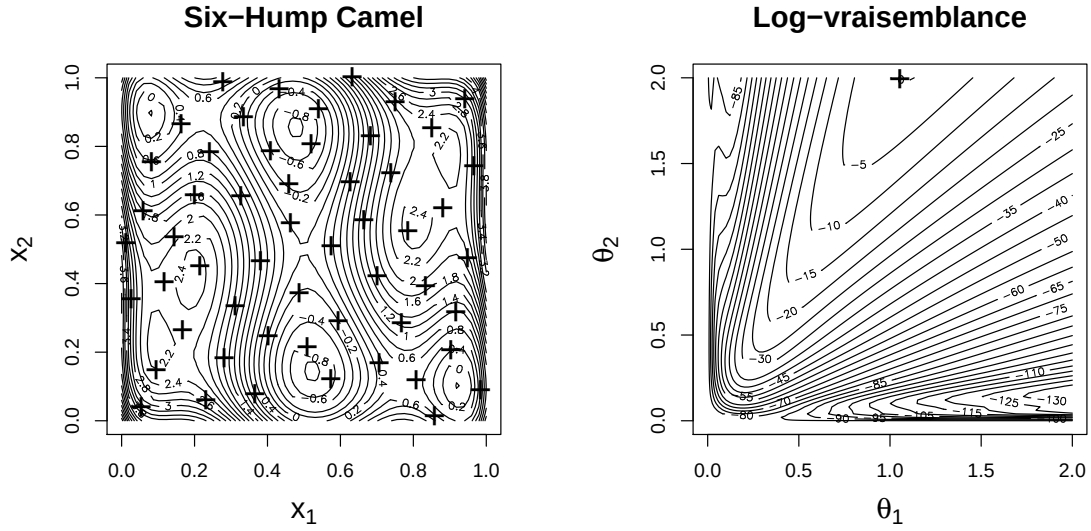

FIGURE 2.2 – La figure de gauche représente les lignes de niveau de la fonction Six-Hump Camel avec les 54 points d'observations (croix noires). La figure de droite représente les lignes de niveau de la fonction de vraisemblance, la croix noire est le $\widehat{\boldsymbol{\theta}}$ estimé.

la Figure 2.2, la zone où se trouve le maximum de vraisemblance n'est plus plate. L'estimation du krigeage est correcte et la prédiction est plus proche de la réalité. En grande dimension et avec peu de points d'observation, le même phénomène se produit lors de l'optimisation de la vraisemblance. Les paramètres de portée estimés sont loin de leur valeur optimale (celle obtenue avec un nombre conséquent de points d'observation). Les méthodes existantes de réduction de dimension du type analyse de sensibilité ou Fanova Graphes (cf Chapitre 1 Section 3) se basent sur un premier modèle de krigeage anisotrope où l'estimation des paramètres est supposée correcte. Or cette dernière hypothèse est

loin d'être garantie. Ces méthodes ne pourront pas être utilisées dans des cas similaires à l'exemple ci-dessus.

Dans ce chapitre, un nouveau noyau de krigeage défini à partir de peu de paramètres est proposé afin d'obtenir une bonne qualité prédictive. La forme du noyau permettant de regrouper les variables et de réduire le nombre de paramètres de portée à estimer est présenté. Aussi, plusieurs stratégies sont développées pour identifier efficacement et de façon automatique les groupes de variables.

Le chapitre suivant correspond à l'article "Four algorithms to construct a sparse kriging kernel for dimensionality reduction " accepté avec révisions mineures dans le journal " Computational Statistics". Les travaux de ce chapitre ont été présenté lors de la conférence internationale ENBIS 2017 à Naples.

Abstract

In the context of computer experiments, metamodels are largely used to represent the output of computer codes. Among these models, Gaussian process regression (kriging) is very efficient see e.g [Snelson, 2008]. In high dimension that is with a large number of input variables, but with few observations, the estimation of the parameters with a classical *anisotropic* kriging can be completely inaccurate. The number of ranges to estimate is the same as the number of inputs and it implies that the optimization space becomes too large compare to the available information. One way to overcome this drawback is to use an *isotropic* kernel which is more robust because it estimates only one parameter. However this model is too restrictive. The aim of this paper is twofold. Our first objective is to propose a smooth kernel with as few parameters as warranted. We introduce a kernel which is a tensor product of few isotropic kernels built on well-chosen subgroup of variables. The main difficulty is to find the number and the composition of the groups. Our second objective is to propose algorithmic strategies to overcome this difficult. Four forward strategies are proposed. They all start with the simplest isotropic kernel and stop when the best model according to BIC criterion is found. They all show very good accuracy results on simulation test cases. But one of them is the most efficient. Tested on a real data set, our kernel shows very good prediction results.

Keywords. metamodel, isotropic, anisotropic, clustering

1 Introduction

Complex physical phenomena are more and more studied through numerical simulations. These numerical models are able to mimic with a high accuracy the real experiments so they predict the physical measures of interest (outputs) very precisely. Then, we can use them as a replacement for real experiments because the numerical simulations are less costly in primary materials. However, these simulations stay often time-consuming. Only a few simulations are then affordable.

The idea to overcome this drawback is to replace the costly numerical model by a metamodel. A metamodel is a less expensive model adjusted on a few well-chosen simulations. It can be shown that among all the possible metamodels, Gaussian process regression (kriging) is a very efficient metamodel (see e.g. [Marrel et al., 2008], [Sudret, 2012] and [Villa-Vialaneix et al., 2012]). More precisely kriging is a spatial interpolation technique which aims at predicting the outputs using an adapted underlying correlation function between design points (see e.g [Santner et al., 2003] and [Rasmussen and Williams, 2006]).

One way to improve performance of kriging is to adapt the covariance structure to each specific case, see e.g. [Durrande, 2001] and [Ginsbourger et al., 2016]. For example [Paciorek and Schervish, 2006] create a new class of covariance functions (kernels) allowing the model to adapt itself to spatial surface whose variability changes with location. Likewise, [Padonou and Roustant, 2016] in a microelectronic framework define a GP model which inserts the geometry of the wafer in the kernel. That's why they introduce the *polar* GP defined with respect to polar coordinates : the covariance function is a sum of a product kernel of radius and a product kernel of polar angles. In the case of multiple outputs [Fricker et al., 2013] define a nonseparable covariance structure for GP with the aim of well

representing the different simulator outputs and the joint uncertainty.

One aim of this paper is to propose a covariance structure adapted to high dimension. In general, people use an *anisotropic* kernel that is a tensor product of as many 1D kernels as the number of inputs. Each kernel being parameterized by a spatial correlation length, called the *range parameter*. In high dimension with a very restricted number of data points the estimation of range parameters becomes quite difficult, the prediction quality suffering from this estimation uncertainty. That's why in this context, an *isotropic* kernel could be a good alternative. This kernel is a function of the euclidean distance defined on the entire input space, so it depends only on one range parameter. However *isotropic* kernels are too restrictive, given that spatial variations are controlled by only one range parameter. The idea of this paper is to automatically construct a data-driven kernel that is intermediate between these two extremal choices.

Contrary to [Yi, 2009] who proposed a penalizes kriging that mimics the penalization used in linear regression, our kernel takes into account all input variables. In order to solve the dimensionality of the problem, [Binois, 2015] proposes different sparse and specific covariance kernels based on expert information or on the work of [Muehlenstaedt et al., 2012] and [Durrande et al., 2012]. In these articles, they use the ANOVA kernels that are a sum of sub-groups of variables that interact together, see more details in [Stitson et al., 1997] and [Gao et al., 2002]. [Muehlenstaedt et al., 2012] found these groups through a sensitivity analysis computed from the prediction function of an *anisotropic* kriging. A first problem appears here. Because the number of points is often too restrictive, the quality of the anisotropic kriging can be poor and the result of the sensitivity analysis such as the proposed partition can be totally wrong. A second problem of the method proposed by Muehlenstaedt and coauthors is that the number of parameters of the final kernel can be higher than that of the *anisotropic* kriging. In [Binois et al., 2015] they test different FANOVA constraining them to have fewer range parameters than the *anisotropic* one and choosing the best kernel among them. This choice of the covariance structure is not done automaticall but lays on prior information.

Finally,[Welch et al., 1992] proposes a likelihood-based forward algorithm to determine the most important factors and to build the predictor. In this study a tensor product of power exponential kernels is chosen to catch the function regularity. The dimension of the kernel, i.e. two parameters (range and power) in each direction, is stepwise reduced by making some range parameters equal. In our paper the structure of the kernel is different. It is a tensor product of isotropic kernels by taking an euclidean norm for the distance in each subspace.

The article is structured as follows. In section 2, we introduce the kriging metamodel and the different kernels. In section 3, we present the four algorithms that automatically select the structure of the kernel according to the data. In section 4, we study the behavior of our algorithms on simulation test cases.

2 Statistical models

This section introduces kriging, *anisotropic* and *isotropic* kernels and finally the new general class of *isotropic by group* kernel.

2.1 Kriging

Let p be the number of input variables. We consider n observations $(\mathbf{x}^i, y^i)_{i=1,\dots,n}$ where $\mathbf{x}^i$ is the ith input vector with p coordinates, such that $\mathbf{x}^i \in D \subset \mathbb{R}^p$ and where y_i is the corresponding output ($y^i \in \mathbb{R}$). In the following we denote by $\mathbf{y} = (y^1, \dots, y^n)'$ the vector of the outputs. In kriging we assume that $\mathbf{y}$ is a realization of a Gaussian process $(Y(\mathbf{x}))_{\mathbf{x}\in D}$ at points $(\mathbf{x}^1, \dots, \mathbf{x}^n)'$ such that for each $x \in D$:

$$Y(\mathbf{x}) = m + \epsilon(\mathbf{x}), \tag{2.1}$$

where $m \in \mathbb{R}$ is the trend, the process $(\epsilon(\mathbf{x}))_{\mathbf{x}\in D}$ is a centered stationary Gaussian process with covariance function $Cov(\epsilon(\mathbf{x}),\epsilon(\mathbf{x}')) = \sigma^2 R(\mathbf{x}, \mathbf{x}') = \sigma^2 \mathbf{r}(\mathbf{x} - \mathbf{x}'), \forall\, (\mathbf{x}, \mathbf{x}') \in D \times D$. In this paper the trend m and the variance σ^2 are assumed to be constant.
In this context we want to construct a linear prediction that minimizes the mean-squared prediction error and that guarantees uniform unbiasedness. Under these two constraints, the prediction, see [Cressie, 1993], at a point $\mathbf{x} \in D$ is given by :

$$\widehat{Y}(\mathbf{x}) = m + \mathbf{r}(\mathbf{x})'\boldsymbol{R}^{-1}(\mathbf{y} - m\mathbb{1}_n) \tag{2.2}$$

and the Mean Square Error (MSE) at point $\mathbf{x} \in D$ is given by :

$$\widehat{s}(\mathbf{x}) = \sigma^2(1 - \boldsymbol{r}(\mathbf{x})'\boldsymbol{R}^{-1}\boldsymbol{r}(\mathbf{x})) \tag{2.3}$$

where $\mathbb{1}_n = (1, \dots, 1)' \in \mathbb{R}^n$, $\boldsymbol{R} \in \mathcal{M}_{n\times n}$ is the correlation matrix of the process Y at the observation points, $\mathbf{r}(\mathbf{x}) \in \mathcal{M}_{n\times 1}$ the correlation vector between $Y(\mathbf{x})$ and the vector $(Y(\mathbf{x}^1), \dots, Y(\mathbf{x}^n))'$.

In the following we will focus on the definition of the correlation function $\mathbf{r}$. First, we introduce the *anisotropic* kernel :

$$\mathcal{M}_a \qquad \mathbf{r}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{j=1}^{p} \rho_{\theta_j}\left(|x_j - \tilde{x}_j|\right),\ \boldsymbol{\theta} = (\theta_1, \dots, \theta_p) \in \mathbb{R}_+^p,\ \mathbf{x} \in D,\ \tilde{\mathbf{x}} \in D \tag{2.4}$$

where ρ_{θ_j} is a correlation function which only depends on the one dimensional range parameter θ_j, see e.g [Santner et al., 2003] and [Stein, 1999]. In the following, we denote the model (2.1) with kernel (2.4) as $\mathcal{M}_a$. The parameters m, σ^2 and $\boldsymbol{\theta}$ are unknown and will be estimated by maximum likelihood. In our industrial case study the output is supposed to be two times continuously differentiable this is why we use a Matern 5/2 kernel usual in that case, see e.g. [Cornford et al., 2002] . This kernel function is defined by :

$$\forall \theta \in \mathbb{R}^+, \forall h \in \mathbb{R}^+, \rho_\theta(h) = \left(1 + \frac{\sqrt{5}|h|}{\theta} + \frac{5h^2}{3\theta^2}\right)\exp\left(-\frac{\sqrt{5}|h|}{\theta}\right).$$

However, *anisotropic* kernels contain as many parameters as the number of variables p. In high dimension the estimation of these parameters becomes unstable. To stabilize the estimation an idea is to reduce the number of range parameters. The extremal case is the *isotropic* kernel, defined as follows for $(\mathbf{x}, \mathbf{x}') \in D \times D$:

$$\mathcal{M}_i \qquad r(\mathbf{x} - \tilde{\mathbf{x}}) = \rho_\theta\left(\|\mathbf{x} - \tilde{\mathbf{x}}\|_2\right),\ \theta \in \mathbb{R}_+\ \mathbf{x} \in D,\ \tilde{\mathbf{x}} \in D \tag{2.5}$$

where $\|.\|_2$ is the euclidian norm on $\mathbb{R}^p$. In the following, we denote the model (2.1) with kernel (2.5) as $\mathcal{M}_i$. In model $\mathcal{M}_i$, spatial variations are controlled by only one range parameter. If the value of that parameter is small the process Y fluctuates a lot in all directions. On the contrary if the value of θ is large, the process varies in a very slow way in all directions. That's why, one unique parameter is often too restrictive to characterize the underlying process. In many cases spatial behavior varies from a direction to another. Therefore we introduce in the section a kernel which is a compromise between the too restrictive *isotropic* kernel and the too flexible *anisotropic* one. We call it the *isotropic by group* kernel.

2.2 *Isotropic by group* kernel

The isotropic by group kernel is defined as follows :

$$\mathcal{M}_q \qquad r(\mathbf{x}-\tilde{\mathbf{x}}) = \prod_{\ell=1}^{q} \rho_{\theta_\ell}\left(\|\mathbf{x}-\tilde{\mathbf{x}}\|_{\mathcal{I}_\ell}\right),\ \boldsymbol{\theta} = (\theta_1,\ldots,\theta_q) \in \mathbb{R}_+^q\ \mathbf{x}\in D,\ \tilde{\mathbf{x}}\in D \tag{2.6}$$

where, $q < p$ is the number of groups. Let $\mathcal{I}_\ell$ be the set of indexes in the ℓth group with cardinal $|\mathcal{I}_\ell| = p_\ell$, $p_\ell \in \{1,\ldots,p\}$ such that $p_1 + \ldots + p_q = p$. We define the norm of the subvector $(x_j)_{j\in\mathcal{I}_\ell}$ of the vector $\mathbf{x} = (x_1,\ldots,x_p) \in D$ by $\|\mathbf{x}\|_{\mathcal{I}_\ell} = \sqrt{\left(\sum_{j\in\mathcal{I}_\ell} x_j^2\right)}$. In the following, we denote model (2.1) with kernel (2.6) as $\mathcal{M}_q$. The *anisotropic* and *isotropic* kernels are two particular cases of *isotropic by group* kernel with respectively p groups $\mathcal{I}_\ell = \{\ell\}$ for $\ell = \{1,\ldots,p\}$, and one unique group $\mathcal{I}_1 = \{1,\ldots,p\}$, ie $\mathcal{M}_p = \mathcal{M}_a$ and $\mathcal{M}_1 = \mathcal{M}_i$.

Rq : Comparisons of realizations with *anisotropic* and *isotropic* kernel are provided in appendix 2.A.

3 Methodology

In this article we propose four strategies to find simultaneously and automatically the number of groups and the composition of each group. In [Yi, 2009] and [Muehlenstaedt et al., 2012] the two proposed methodologies to reduce the number of parameters in kriging is starting with a fully *anisotropic* kernel. To gain in robustness in high dimensional problems, in the four procedures we introduce, we always start with an *isotropic* kernel that depends on one range parameter and we finish with an *isotropic by group* kernel. At each stage we compare different models and we choose the best model under a specific criterion. The model parameters are estimated by maximum likelihood, that is :

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[l(\boldsymbol{\theta}; \mathbf{x}^1,\ldots,\mathbf{x}^n,\mathbf{y}) \right] \tag{2.7}$$

where $l = -\log(\mathcal{L})$ and $\mathcal{L}$ is the concentrated likelihood function defined by :

$$\mathcal{L}(\boldsymbol{\theta}; \mathbf{x}^1,\ldots,\mathbf{x}^n,\mathbf{y}) = \frac{1}{(2\pi\hat{\sigma}^2)^{\frac{n}{2}}|\boldsymbol{R}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{y}-\hat{m}\mathbb{1}_n)^t\boldsymbol{R}^{-1}(\mathbf{y}-\hat{m}\mathbb{1}_n)} \tag{2.8}$$

where $\hat{m}$ and $\hat{\sigma}^2$ could be writen as functions of $\boldsymbol{\theta}$:

$$\hat{m} = (\mathbb{1}_n^t\boldsymbol{R}^{-1}\mathbb{1}_n)^{-1}\mathbb{1}_n^t\boldsymbol{R}^{-1}\mathbf{y}$$
$$\hat{\sigma}^2 = \frac{1}{n}\left(\mathbf{y}-m\mathbb{1}_n\right)^t\boldsymbol{R}^{-1}\left(\mathbf{y}-m\mathbb{1}_n\right)$$

As we estimate parameters by maximum likelihood, it could then be natural to select the best model under likelihood considerations. However the likelihood is an increasing function of the number of parameters. As a trade off between estimation qualities and sparsity, a penalized likelihood criterion is preferred. In our work, we consider the BIC (Bayesian Information Criterion see e.g. [Schwarz, 1978]), that is for a model $\mathcal{M}_q$ with q parameters to estimate $\boldsymbol{\theta}_q$:

$$BIC(\boldsymbol{\theta}_q, \mathcal{M}_q) = l_{\mathcal{M}_q}(\boldsymbol{\theta}_q; \mathbf{x}_1,\ldots,\mathbf{x}_n,\mathbf{y}) + q\log(n). \tag{2.9}$$

We propose four strategies based on the same algorithmic structure (see **Algorithm** G) that makes a sequential model selection on BIC criterion. The strategies begin with a classical multi-start procedure described in appendix 2.B (see **Algorithm** 0).

Algorithm G : General algorithmic structure

We choose θ_{opt} thanks to algorithm 0 for model $\mathcal{M}_1$.
Construct $\mathcal{M}_1$;
$k \leftarrow 1$;
while $BIC(\boldsymbol{\theta}_q, \mathcal{M}_{k+1}) < BIC(\boldsymbol{\theta}_q, \mathcal{M}_k)$ **do**
    Construct n_k submodels $\mathcal{M}_{k+1}^1, \ldots, \mathcal{M}_{k+1}^{n_k}$ from $\mathcal{M}_k$
    $\mathcal{M}_{k+1} \leftarrow \text{argmin}_{1 \leq i \leq n_k} BIC(\boldsymbol{\theta}_q, \mathcal{M}_{k+1}^i)$
    $k \leftarrow k + 1$
end while
return $\mathcal{M}_{k-1}$

3.1 Algorithm 1

General loop :
At each step k, if the previous model is composed of $k-1$ groups, the next one will have k groups. Let $\mathcal{M}_{k-1}$ be the model at step $k-1$, we note $\mathcal{I}_1 = \{i_{1,1}, \ldots, i_{1,p_1}\}; \ldots; \mathcal{I}_{k-1} = \{i_{k-1,1}, \ldots, i_{k-1,p_{k-1}}\}$, with $p_1 + \ldots + p_{k-1} = p$, the sets of indices of the variables included in each group. One of these $k-1$ groups is then divided into two groups. We have J_k ways of choosing this group, with $J_k = \sum_{\ell=1}^{k-1} J_{k,\ell}$ and $J_{k,\ell}$ is the number of ways of dividing into two groups the set of indexes $\mathcal{I}_\ell$, $\ell = 1, \ldots, k-1$. We note $\mathcal{M}_k^1, \ldots, \mathcal{M}_k^{J_k}$ the J_k new models. And among these J_k models we choose the one minimizing the BIC, says $\mathcal{M}_k^{\ell_{opt}}$.

Algorithm 1 :
We start with the estimation of the full *isotropic* model, noted $\mathcal{M}_1$. We follow the previous procedure and we obtain J_2 models noted $\mathcal{M}_2^\ell$, $\ell = 1, \ldots, J_2$. We estimate the parameter $\boldsymbol{\theta}$ of each model and we keep the model $\mathcal{M}_2^{\ell_{opt}}$ that minimizes the BIC. If the BIC of the model $\mathcal{M}_1$ is smaller than the one of $\mathcal{M}_2$, we stop the algorithm and we choose $\mathcal{M}_1$ as the best model. Otherwise, we set down $\mathcal{M}_2 = \mathcal{M}_2^{\ell_{opt}}$ and we build the models $\mathcal{M}_3^\ell$, $l = 1, \ldots, J_3$ and so forth at step $4, 5$... We stop the algorithm when $\mathcal{M}_k$ has a larger BIC than $\mathcal{M}_{k-1}$. At the end of the algorithm, we obtain the model $\mathcal{M}_{k-1}$.

Algorithm 1

We choose θ_{opt} thanks to **Algorithm** 0 for model $\mathcal{M}_1$.
$opt\_BIC \leftarrow BIC(\theta_{opt}, \mathcal{M}_1)$;
$k \leftarrow 2$;
while $k \leq p$ **do**
    $vect\_BIC \leftarrow \mathbf{0}_{J_k}$
    for ℓ from 1 to J_k **do**
        We choose θ_{opt} thanks to algorithm 0 with the model $\mathcal{M}_k^{\ell}$;
        $vect\_BIC(\ell) \leftarrow BIC(\boldsymbol{\theta}_{opt}, \mathcal{M}_{k+1}^{\ell})$;
    end for
    $\ell_{opt} \leftarrow \underset{\ell}{\operatorname{argmin}}\{vect\_BIC\}$;
    $\mathcal{M}_k \leftarrow \mathcal{M}_k^{\ell_{opt}}$;
    if $vect\_BIC(\ell_{opt}) \geq opt\_BIC$ **then**
        return $\mathcal{M}_{k-1}$;
    else
        $opt\_BIC \leftarrow vect\_BIC(\ell_{opt})$;
        $k \leftarrow k + 1$;
    end if
end while

Algorithm 1 is a quasi-exhaustive algorithm since at each step it browses all the possibilities to divide. But the calculation time grows really fast with the number of inputs p [1]. For example, the first separation, that consists in separating one group in two, estimates 500 models for $p = 10$ and 524287 for $p = 20$. That's why we propose several alternative algorithms that take much less time.

3.2 Algorithm 2

To accelerate the **Algorithm** 1, we propose to split variable one by one from the initial set of all variables. As this approach splits too much a merging step is added. The last isolated variable can be agglomerated to an existing subgroup. We note $\mathcal{M}_{s_{k-1}}$ the model at step $k-1$ and s_{k-1} the number of groups. Let $\mathcal{I}_1 = \{i_{1,1}, \ldots, i_{1,p_1}\}; \ldots; \mathcal{I}_{s_{k-1}} = \{i_{s_{k-1},1}, \ldots, i_{s_{k-1},p_{s_{k-1}}}\}$, with $p_1 + \ldots + p_{s_{k-1}} = p$, the variables' indexes of each group.

Checking step :
At step k, there are 2 possibilities :

1. If we have grouped some variables at the step $k-1$ or if the current step is $k < 3$ (aggregation is not allowed before $k = 3$), we create an additional group by taking an index $i_{1,m} \in \mathcal{I}_1$, $m = 1, \ldots, p_1$ out of the group 1. We have $J_k = J_k^+ = p_1$ possibilities and so we build J_k models.
2. If we have separated at the step $k-1$ and $k \geq 3$, we have 2 possibilities. The first one consists in taking an index $i_{1,s} \in \mathcal{I}_1$, $s = 1, \ldots, p_1$ out of the first group, we have $J_k^+ = p_1$ possible indexes. The second consists in grouping the last taken out variable $i_{s_{k-1},1}$ ($p_{s_{k-1}} = 1$) with another group, we have $J_k^- = s_{k-1} - 2$ possibilities. Then we build a total of $J_k = J_k^+ + J_k^- = p_1 + s_{k-1} - 2$ models.

1. The complexity is exponential (at each step $k > 1$, $\sum_{i=1}^{k-1}(2^{p_i - 1} - 1)$ models are estimated (see appendix 2.F).

In the two cases, we create J_k models noted $\mathcal{M}^1_{s_{k_1}}, \ldots, \mathcal{M}^{J_k}_{s_{k_{J_k}}}$.

General loop :
We follow the general procedure in introducing a boolean that represents the choice at step $k-1$ of grouping or separating. Then the value of the boolean will be TRUE if we grouped at the previous step and FALSE if not.

Algorithm 2

We choose θ_{opt} with **Algorithm** 0 in considering model $\mathcal{M}_1$.
$opt\_BIC \leftarrow BIC(\theta, \mathcal{M}_1)$;
$k \leftarrow 2$;
$p_1 \leftarrow p$;
while $p_1 > 1$ **do**
  if $reg_{bool} = false$ and $k \geq 3$ **then**
    We construct $J_k = p_1 + s_{k-1} - 2$ models $\mathcal{M}^\ell_{s_{k_\ell}}$, $\ell = 1, \ldots, J_k$;
    $reg_{bool} = true$;
  else
    We construct $J_k = p_1$ models $\mathcal{M}^\ell_{s_{k_\ell}}$, $\ell = 1, \ldots, J_k$;
    $reg_{bool} = false$;
  end if
  $vect\_BIC \leftarrow \mathbf{0}_J$
  for ℓ from 1 to J_k **do**
    We choose θ_{opt} with algorithm 0 in considering model $\mathcal{M}^\ell_{s_{k_\ell}}$;
    $vect\_BIC(k) \leftarrow BIC(\theta, \mathcal{M}^\ell_{s_{k_\ell}})$;
  end for
  $\ell_{opt} \leftarrow \underset{\ell}{\text{argmin}}\{vect\_BIC\}$;
  if $vect\_BIC(\ell_{opt}) \geq opt\_BIC$ **then**
    return $\mathcal{M}_{s_k}$;
  else
    $\mathcal{M}_{s_k} \leftarrow \mathcal{M}^{\ell_{opt}}_{s_{k_\ell}}$;
    $opt\_BIC \leftarrow vect\_BIC(\ell_{opt})$;
  end if
end while
return $\mathcal{M}_{s_k}$;

Algorithm 2 isolates variables more than **Algorithm** 1 and browses less possibilities. The complexity is linear (at each step k, p_1 or $p_1 + 1$ models are estimated) .This algorithm is largely identical to **Algorithm** W introduced by [Welch et al., 1992], that is described in appendix 2.C, except that we add a backward step.

3.3 Algorithm 3

Because **Algorithm** 2 generates a number of clusters often too high, we propose to add a clustering step at the end of the procedure to merge some subgroups. This step naturally reduces the complexity of the model.
General loop :

Algorithms	alg1	alg2	alg3	alg4	algW
Strategy	Quasi exhaustive	Separate	Separate + Cluster at the end	Separate + Cluster at each step	Separate
Direction	Forward	Both	Both	Both	Forward

TABLE 2.1 – Strategy and direction of the 4 algorithms and algorithm proposed by [Welch et al., 1992]

Let $\mathcal{M}_q$ be the best model found by **Algorithm** 2, such that $\mathcal{I}_1 = \{i_{1,1}, \ldots, i_{1,p_1}\}; \ldots; \mathcal{I}_q = \{i_{q,1} \ldots, i_{q,p_q}\}$, with $p_1 + \ldots + p_q = p$ the indexes of variables in each group. We build a new model noted $\mathcal{M}_{clust}$ with a clustering method that groups variables with a close value of range parameters (see [Everitt et al., 2011]). However, in the *isotropic by group* kernel the ranges are not comparable since they are linked to variable sets of different sizes. Only at the final step, we replace the *isotropic by group* kernel $r_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{\ell=1}^{q} \rho_{\theta_\ell}(\|\mathbf{x} - \mathbf{x}\|_{\mathcal{I}_\ell})$ by the *product* kernel $r_{\tilde{\boldsymbol{\theta}}}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{\ell=1}^{q} \prod_{j=1}^{p_\ell} \rho_{\tilde{\theta}_\ell}(|x_j - \tilde{x}_j|)$. We denote by $\mathcal{M}_q^{prod}$ the associated kriging model. We obtain 1D comparable ranges $\tilde{\theta}_1, \ldots, \tilde{\theta}_q$. Then a clustering of $\tilde{\theta}_1, \ldots, \tilde{\theta}_q$ is performed to obtain q' groups where $q' < q$. Finally we build the model $\mathcal{M}_{clust}$ using q' groups. Then, if the BIC of the model $\mathcal{M}_{clust}$ is smaller that the one of $\mathcal{M}_{s_k}$ we set down $\mathcal{M}_{s_k} = \mathcal{M}_{clust}$. We replace the step 19 in **Algorithm** 2 by :

Algorithm 3 Step 20bis :

We obtain $\mathcal{M}_{clust}$ from $\mathcal{M}_{s_k}$ with a clustering method ;
if $BIC(\theta, \mathcal{M}_{clust}) < BIC(\theta, \mathcal{M}_{s_k})$ **then**
    return $\mathcal{M}_{clust}$;
end if

3.4 Algorithm 4

Algorithm 4 combines **Algorithm** 2 with a clustering method at each step k of the algorithm.
General loop :
At each step k we compute $\mathcal{M}_{s_k}$ by **Algorithm** 2. Then, with the same groups composition we estimate $\mathcal{M}_{s_k}^{prod}$ to the ranges of which we apply a classification step to produce $\mathcal{M}_{k,clust}$. If the BIC of model $\mathcal{M}_{k,clust}$ is smaller that the one of $\mathcal{M}_{s_k}$ we set down $\mathcal{M}_{s_k} = \mathcal{M}_{k,clust}$. We replace the step 16 in **Algorithm** 2 by :

Algorithm 4 Step 16bis :

$\mathcal{M}_{s_k} \leftarrow \mathcal{M}_{s_{k_\ell}}^{\ell_{opt}}$;
We obtain $\mathcal{M}_{k,clust}$ from $\mathcal{M}_{s_k}$ with a clustering method ;
if $BIC(\theta, \mathcal{M}_{k,clust}) < BIC(\theta, \mathcal{M}_{s_k})$ **then**
    $vect\_BIC(\ell_{opt}) \leftarrow BIC(\theta, \mathcal{M}_{k,clust})$;
    $\mathcal{M}_{s_k} \leftarrow \mathcal{M}_{k,clust}$;
end if

3.5 Conclusion and summary

The Table 2.1 summarizes the general characteristics of the algorithms. The strategy of the first algorithm is Quasi-exhaustive, almost all the combinations of groups are tested. The other algorithms

isolate the variables from each others. A clustering step is added to gather variables in **Algorithm** 3 and 4. All the algorithms have a direction Both except the **Algorithm** 1 and W that are forward.

4 Application

4.1 Analytical examples

In this section we compare the algorithms on simulated data in term of performance and of calculation time. Then, we observe the behavior of the prediction quality of the best one according to the number of observations. We also investigate the impact of an error on range parameters of a nugget effect.

4.1.1 Algorithms' Comparison

To compare our four algorithms, we evaluate their performances on a simulated Gaussian process with *isotropic by group* kernel in dimension eight. We simulated the following model $(Y(\mathbf{x}))_{\mathbf{x}\in D}$:

$$Y(\mathbf{x}) = 1 + \epsilon(\mathbf{x})$$

where ϵ is a Gaussian process with a standard deviation $\sigma^2 = 1$ and an *isotropic by group kernel* :

$$r_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{\ell=1}^{3} \rho_{\theta_\ell}\left(\|x - \tilde{x}\|_{\mathcal{I}_\ell}\right)$$

where $\mathcal{I}_\ell = \{1\}, \mathcal{I}_2 = \{2, 3\}, \mathcal{I}_3 = \{4, 5, 6, 7, 8\}$ and ρ_θ is a Matern5_2. $\boldsymbol{\theta} = (0.5, 0.9, 0.8)$. The learning set is an optimized Latin Hypercube with n observations and the test set is a Sobol sequence of 1000 points.

We simulate 100 different trajectories of the model with $n = 800$ points of observations. For each trajectory we estimate the model with the algorithms. Table 2.2 shows that each algorithm finds the correct number and composition of groups for the 100 trajectories except **Algorithm** W. This result is not surprising because **Algorithm** W separates variable by variable and cannot propose several groups with more than one variable. Among the four algorithms, Table 2.3 shows that **Algorithm** 4 is

	good groups	too separated
alg1	100	0
alg2	100	0
alg3	100	0
alg4	100	0
algW	0	100

TABLE 2.2 – Gathering errors obtained by the algorithms on 100 trajectories with a learning set of size 800.

the fastest, it takes 1h21 to find the best model. In the next sections we only use **Algorithm** 4.

4.1.2 Evolution of the prediction quality with the learning set size

We test nine learning set sizes : $n = \{80, 280, 480, 680, 880, 1080, 1280, 1480, 1680\}$. For each size we simulate one learning set and 25 trajectories. For the high sizes, the algorithm could not converge due to the bad conditioning of the covariance matrix. There are too many observation points

alg	Calculation time by processor
alg1	6h38
alg2	1h52
alg3	1h50
alg4	1h21
algW	1h20

TABLE 2.3 – Calculation time for the four algorithms on 100 trajectories with a learning set of size 800.

Sample size	good groups (%)	bad gathering (%)	too separate (%)	Simulations
80	16	84	0	25
280	80	4	16	25
480	100	0	0	25
680	100	0	0	25
880	100	0	0	25
1080	100	0	0	23
1280	100	0	0	20
1480	100	0	0	17
1680	100	0	0	12

TABLE 2.4 – Gathering errors obtained by **Algorithm** 4 on 25 trajectories with different learning set size $\{80, 280, 480, 680, 880, 1080, 1280, 1480, 1680\}$

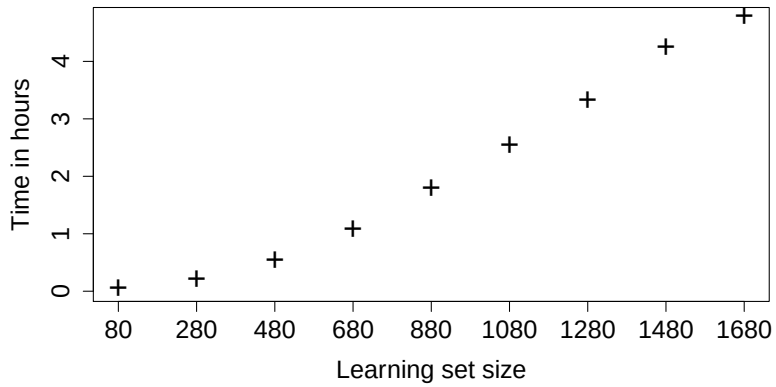

FIGURE 2.3 – Mean time's evolution in hours according to the sample size to find the best model with **Algorithm** 4. 25 trajectories are simulated for each size $\{80, 280, 480, 680, 880, 1080, 1280, 1480, 1680\}$.

for kriging. For a learning set of size 80, the algorithm makes lot of bad gathering (see Table 2.4). These errors are serious because they cause a loss of prediction quality. Figure 2.3 shows that the estimation time grows with the number of experiments. For a size of 1280 the estimation takes more than 3 hours (see Figure 2.3). The prediction quality measured through Q2 or RMSE ([Dupuy et al., 2015]) also increases with the number of experiments. From $n = 680$, the prediction quality becomes very good (see Figure 2.4).

Whatever the size of the learning set kriging with an *Isotropic by group* kernel stays the best

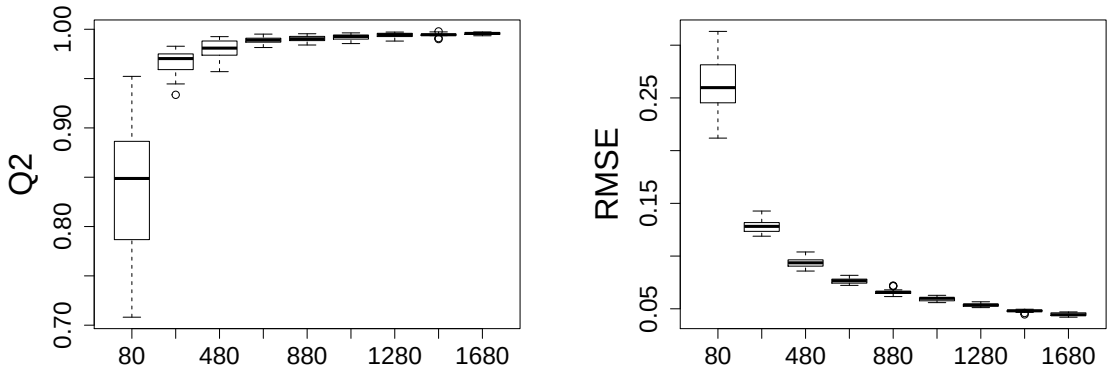

FIGURE 2.4 – Prediction quality of the model with an *isotropic by group* kernel found by **Algorithm** 4 according to the sample size. 25 trajectories are simulated for each size $\{80, 280, 480, 680, 880, 1080, 1280, 1480, 1680\}$. The criteria of prediction quality are Q2 and RMSE.

(see Figure 2.5). In addition, with a low number of points (80), we notice that the *anisotropic* give a completely wrong prediction in 20% of simulations. On the contrary kriging with an *isotropic by group* kernel gives robust predictions to estimation.

4.1.3 Evolution of the prediction quality with an error on simulation parameters

We simulated the model :

$$Y(\mathbf{x}) = 1 + \epsilon(\mathbf{x})$$

where ϵ is a Gaussian process with standard deviation $\sigma^2 = 1$ and an *anisotropic* kernel :

$$r_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{j=1}^{8} \rho_{\theta_j^*}(|x_j - \tilde{x}_j|)$$

where ρ_θ is a Matern5_2 and $\boldsymbol{\theta} = (0.5, 0.9, 0.9, 0.8, 0.8, 0.8, 0.8, 0.8)$. We simulate the range parameters with an error $\xi = (0, 2, 5, 10, 15, 40)$ such that $\theta_j^* \sim \mathcal{U}\left(\left[\theta_j - \frac{\xi_k}{100}\theta_j; \theta_j + \frac{\xi_k}{100}\theta_j\right]\right), k = 1, \ldots, 6$ and $j = 1, \ldots, 8$. The learning set is an optimized Latin Hypercube with $n = 400$ observations and the test set is a Sobol sequence of 1000 points. For each value of ξ we simulate 25 values of $\boldsymbol{\theta}^*$ and one trajectory for each..

Adding an error to range parameters doesn't influence the prediction quality of $\mathcal{M}_q$ and it stays the best compared to $\mathcal{M}_a$ and $\mathcal{M}_i$, see Figure 2.6.

4.1.4 Evolution of the prediction quality with a nugget effect

We consider the same Gaussian process as previously but with a nugget effect :

$$r_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \prod_{j=1}^{p} \rho_{\theta_j}(|x_j - \tilde{x}_j|) + \tau^2 \delta_0(\mathbf{x} - \tilde{\mathbf{x}})$$

where ρ_θ is a Matern5_2 with $\boldsymbol{\theta} = (0.5, 0.9, 0.9, 0.8, 0.8, 0.8, 0.8, 0.8)$. The learning set is an optimized Latin Hypercube with $n = 160$ observations and the test set is a Sobol sequence of 1000

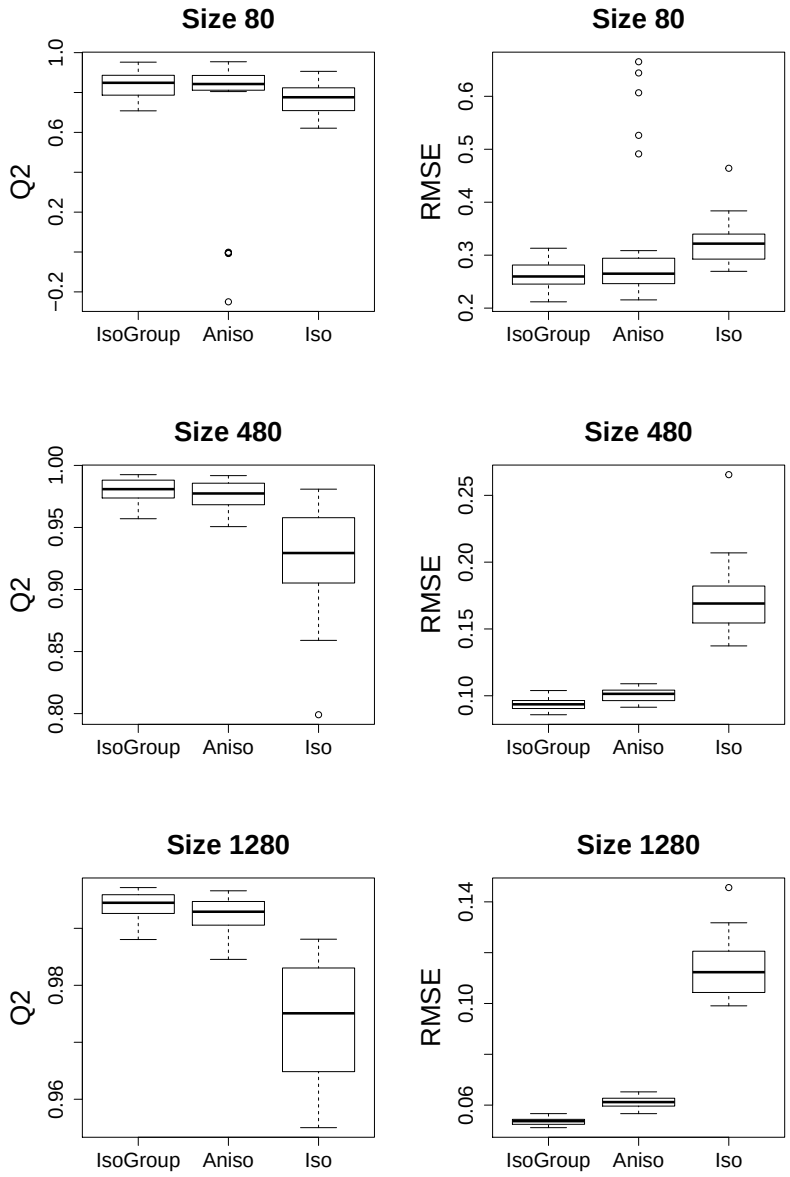

FIGURE 2.5 – Prediction quality of the model with an *isotropic by group* kernel found by **Algorithm** 4 on the left, *anisotropic* kernel in the middle and *isotropic* kernel on the right according to the sample size. 25 trajectories are simulated for each learning set size $\{80, 480, 1280\}$. The criteria of prediction quality are Q2 and RMSE.

points.

$$\delta_0(\mathbf{x} - \tilde{\mathbf{x}}) = \begin{cases} 1 & \text{if } \mathbf{x} = \tilde{\mathbf{x}} \\ 0 & \text{if not} \end{cases}$$

We test 6 different levels of nugget effect $\tau = (0, 0.02, 0.03, 0.05, 0.1, 0.3)$. Figure 2.7 shows that the prediction looses accuracy with the increasing of the nugget effect for the three kernels. The *anisotropic* kernel gives the worst prediction. The *isotropic by group* stays the best compared to the others. In this context, whatever the kernel, allowing the estimation of a nugget effect would probably improve predictions.

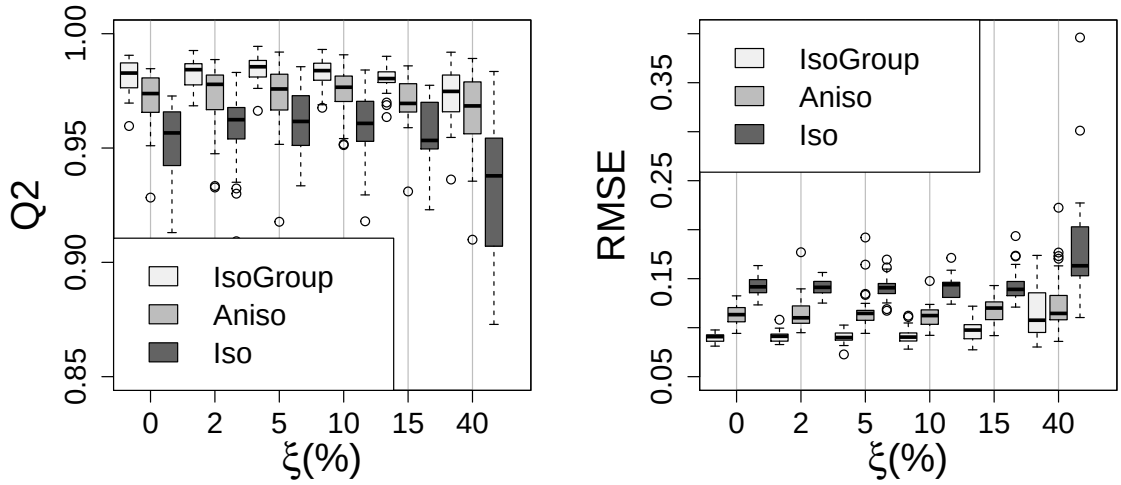

FIGURE 2.6 – Prediction quality of the model with an *isotropic by group* kernel found by **Algorithm** 4 (IsoGroup) *anisotropic* kernel (Aniso) and *isotropic* kernel (Iso) according to an error added on the range parameters in the simulated model. 25 trajectories are simulated for a learning set of size 400. The prediction quality criteria are Q2 and RMSE.

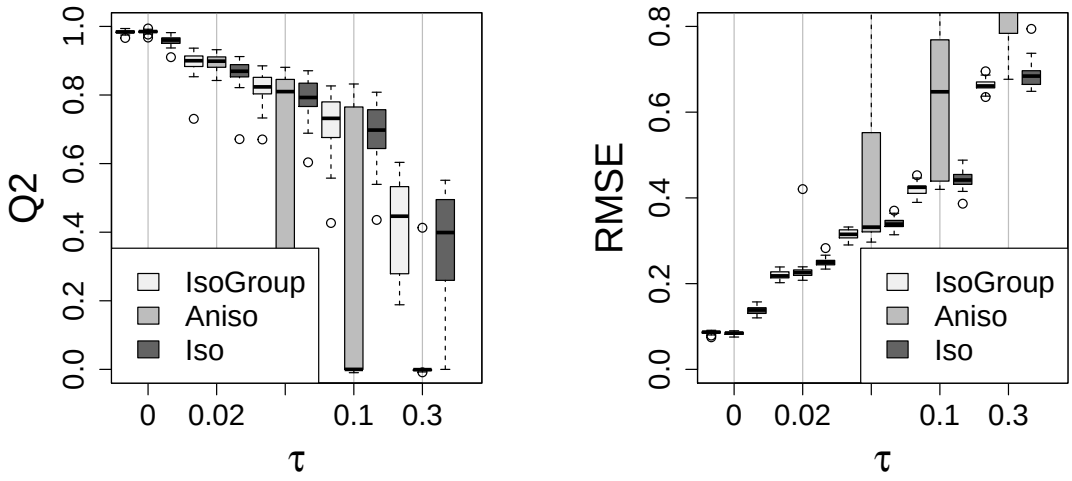

FIGURE 2.7 – Prediction quality of the model with an *isotropic by group* kernel found by **Algorithm** 4 (IsoGroup) *anisotropic* kernel (Aniso) and *isotropic* kernel (Iso) according to the value of a nugget effect added in the simulated model. 25 trajectories are simulated for a learning set of size 400. The prediction quality criteria are Q2 and RMSE.

4.2 Test function

To motivate the construction of our kriging model, let us consider the function :

$$f(\mathbf{x}) = \prod_{j=1}^{15} \frac{4x_j + a_j}{1 + a_j}$$

with $a_j = (0, 1, 1, 4.5, 10, 10, 99, 99, 99, 99, 99, 99, 99, 99, 99)$. $y^1, \ldots, y^n$ are the observations such that $y^i = f(\mathbf{x}^i)$ where f is the function and $\mathbf{x}^i \in [0; 1]^{15}$. Considering $250 \simeq 17 \times 15$ runs of the function, we aim at constructing a predictive model based on kriging, with three different kernels : model $\mathcal{M}_q$ with an *isotropic by group* kernel (see equation 2.6), model $\mathcal{M}_a$ with an *anisotropic* kernel (see equation 2.4) and model $\mathcal{M}_i$ with an *isotropic* kernel (see equation 2.5). We apply **Algorithm** 4 to find the number and the composition of groups of the model $\mathcal{M}_q$. For the three models the parameters $\boldsymbol{\theta}$, m and σ^2 are estimated from the sample. The correlation function ρ_θ is the Matern 5_2.

***Isotropic by group* kernel**

$\theta_1(x_1, x_3)$	$\theta_2(x_2)$	$\theta_3(x_4)$	$\theta_4(x_5, x_6)$	$\theta_5(x_7, \ldots, x_{15})$	m	σ^2
1.53	1.8	2.57	4.6	42.12	1.78	207.06

***Anisotropic* kernel**

θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9
0.63	0.91	0.87	1.42	2	2	1.99	1.99	2
θ_{10}	θ_{11}	θ_{12}	θ_{13}	θ_{14}	θ_{15}	m	σ^2	
2	2	1.99	1.99	1.99	1.99	6.67	9.78	

***Isotropic* kernel**

θ	m	σ^2
3.2	6.35	177.49

TABLE 2.5 – Kriging parameters for an *isotropic by group* kernel (top), an *anisotropic* kernel (middle) and an *isotropic* kernel (bottom). The learning set is an optimized Latin Hypercube of size 250. The test set is a Sobol sequence of 8000 points.

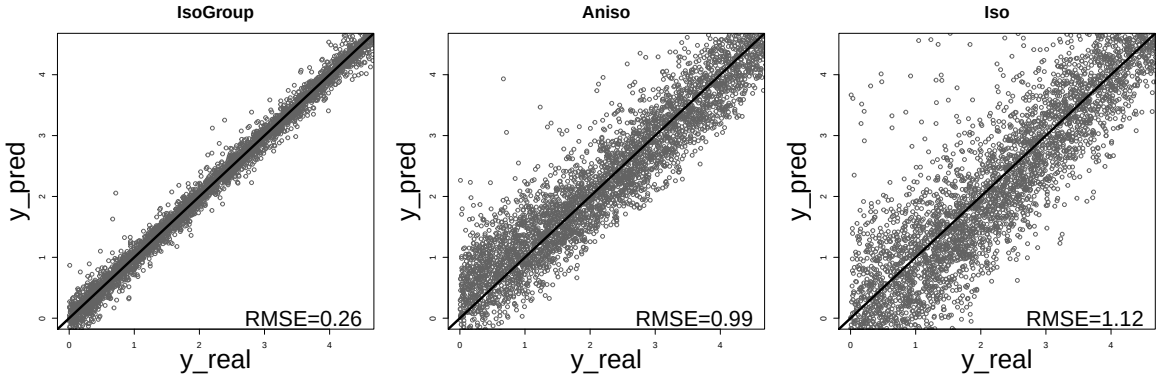

FIGURE 2.8 – Kriging prediction plots for the function : *isotropic by group kernel*(left), *anisotropic* kernel(middle), *isotropic* kernel (right).

Algorithm 4 found 5 groups : $\mathcal{I}_1 = \{1, 3\}$, $\mathcal{I}_2 = \{2\}$, $\mathcal{I}_3 = \{4\}$, $\mathcal{I}_3 = \{5, 6\}$ and $\mathcal{I}_1 = \{7, 8, 9, 10, 11, 12, 13, 14, 15\}$. We estimate 4 range parameters instead of 15 for model $\mathcal{M}_a$ and 1 for model $\mathcal{M}_i$. The estimate parameters are in Table 2.5. In Figure 2.8, we compare the predictive power of the three models. Kriging with an *isotropic by group* kernel improves prediction power compared to kriging with an *anisotropic* or an *isotropic* kernel.

5 Conclusion

In high dimension, kriging model with classical kernels provide poor predictions of the response. Yet, with a sparse kernel, predictions are much better. After a study of the algorithms' behavior on their prediction quality and their time of estimation. It results that **Algorithm** 4 is the most efficient. The estimation time does not grow too fast with the number of experiments and the quality of prediction is always the best. The prediction quality increases with the number of experiments. Adding an error on the simulated range parameters influences the groups composition but not the prediction quality of the *isotropic by group* model. On the other hand adding a nugget effect to the simulation model degrades the quality of the *isotropic by group* kernel predictor but it stays close to the quality of the *anisotropic*. The comparison of the predictive power on the test function shows that kriging with the *isotropic* kernel is poor. Kriging with an *isotropic by group* kernel improves the predictive power compared to an *anisotropic* kernel. To conclude, the proposed methods enable to improve the prediction quality in the context of time-consuming simulation in high dimension.

6 Acknowledgments

This work benefited from the financial support of the French ANR project "PEPITO" (ANR-14-CE23-0011)

Appendix

2.A Visualization of *isotropic* and *anisotropic* kernels

We simulate in the Figure 2.A.1 a Gaussian process with $p = 2$ parameters, the simulated model is :

$$Y(\mathbf{x}) = 1 + \epsilon(\mathbf{x})$$

where ϵ is a Gaussian process with a standard deviation $\sigma^2 = 1$.
With an *anisotropic* kernel :

$$r_{\boldsymbol{\theta}}(\mathbf{x} - \mathbf{x}') = \rho_\theta \left(|x_1 - x_1'|\right) \times \rho_\theta \left(|x_2 - x_2'|\right)$$

and an *isotropic* kernel :

$$r_{\boldsymbol{\theta}}(\mathbf{x} - \mathbf{x}') = \rho_\theta \left(\|\mathbf{x} - \mathbf{x}'\|_2\right)$$

where ρ_θ is a Matern5_2 and $\theta = 0.5$.

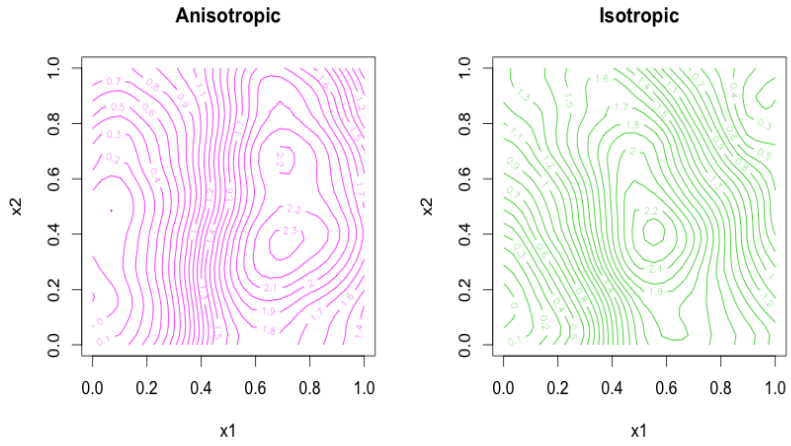

FIGURE 2.A.1 – Simulation of a Gaussian process trajectory with 2 different kernels. *Anisotropic* on the left and *isotropic* on the right.

2.B Multi-start algorithm

Equation (2.7) shows that the estimation of the range parameter $\boldsymbol{\theta}$ is done by minimizing l. The minimization of l is very difficult due to the nonlinearity of the objective function. We use a classical numerical method names Limited-memory BFGS to handle optimization. A well known drawback of the optimization algorithms is the dependence of the solution on the initialization point. That's why we create **Algorithm** 0 that finds a good initial value for $\boldsymbol{\theta}$ to minimize l and that gives an estimation of $\hat{\boldsymbol{\theta}}$. The principle of **Algorithm** 0 is to first select 10 vectors of parameters $\boldsymbol{\theta}$ ($\boldsymbol{\theta} \in \mathbb{R}^q$) that cover space as well as possible and to choose the set of parameters, let's say $\boldsymbol{\theta}_{opt\_init}$, that has the lowest l-value. Then, we minimize l using $\boldsymbol{\theta}_{opt\_init}$ as initial point and we obtain the estimation of $(\theta_\ell)_{\ell=1,\ldots,q}$, denoted by $\boldsymbol{\theta}_{opt}$.

Algorithm 0 Likelihood Optimization

$crit\_vrais = \mathbf{0}_{10}$;
$\boldsymbol{\theta}_{\boldsymbol{Mat},init} = \mathbf{0}_{q\times 10}$, where $\mathbf{0}_{q\times 10}$ is the zero matrix of size $q \times 10$;
for $k = 1$ to 10 **do**
    $\boldsymbol{\theta}_{\boldsymbol{Mat},init}(k) \rightsquigarrow \mathcal{U}([0;+\infty]^q)$;
    $crit\_vrais(k) \leftarrow l_{\mathcal{M}_2}(\boldsymbol{\theta}_{\boldsymbol{Mat},init}(k); \mathbf{x}_1, \ldots, \mathbf{x}_n, y_1, \ldots, y_n)$;
end for
$k_{opt} \leftarrow \underset{k}{\operatorname{argmin}}\{crit\_vrais(k)\}$;
$\boldsymbol{\theta}_{opt\_init} \leftarrow \boldsymbol{\theta}_{\boldsymbol{Mat},init}(k_{opt})$;
$\boldsymbol{\theta}_{opt} \leftarrow \underset{\boldsymbol{\theta}\in\mathcal{D}}{\operatorname{argmin}}\, l_{\mathcal{M}_2}(\boldsymbol{\theta}_{opt\_init}; \mathbf{x}_1, \ldots, \mathbf{x}_n, y_1, \ldots, y_n)$ with the initialization at $\boldsymbol{\theta}_{opt\_init}$.

2.C Algorithm W

This algorithm is inspired by [Welch et al., 1992]. At each step k, if the previous model has $s_{k-1} = k-1$ groups, the next estimated models will have $s_k = k$ groups.
We note $\mathcal{M}_{s_k}$ the model at step k with $s_k = k$ groups. Let $\mathcal{I}_1 = \{i_{1,1}, \ldots, i_{1,p_1}\}; \mathcal{I}_2 = \{i_{2,1}\}; \ldots; \mathcal{I}_{s_k} = \{i_{s_k,1}\}$, with $p_1 + s_k - 1 = p$ where $(s_k - 1 = k)$.

Checking step :
We create an additional group in taking an index $i_{1,m} \in \mathcal{I}_1, m = 1, \ldots, p_1$ out of the group 1. We have $J_k = p_1$ possibilities and we obtain $s_k = s_{k-1}+1$ groups. we create J_k models noted $\mathcal{M}^1_{s_k}, \ldots, \mathcal{M}^{J_k}_{s_k}$ and we follow the **General step**.
In the **Algorithm** 2 steps 5-9 are replaced by :

Algorithm W Find $\mathcal{M}_q$
Step 5bis-9bis :

We construct $J_k = p_1$ models $\mathcal{M}^\ell_{s_k}$, $\ell = 1, \ldots, J_k$;

2.D Code

The code of the algorithms is developed on R. It includes some C code completely integrate in the R code. The entire code and the examples used in the application section is available on the github
The package **kergp** is used to construct the C class of the *isotropic by group* kernel. All the functions needed to compute the four algorithms is on the same file named **fonctions.R**. The four algorithms are in four files **algo1Base.R**, **algo2Base.R**, **algo3Base.R**, **algo4Base.R** and the algorithm in appendix 2.C is in the file **algoWelch.R**. The functions are described in the table 2.D.2. The algorithms code are parallelized when it is possible. The hierarchical clustering method is done by the function *hclust* available in the package **stats**. The function used to cut the dendrogram to obtain the best composition of groups *best.cutree* is available in the package **JLutils**. Finally, the code to run the simulations of the applications is on the files given by the table 2.D.2. The file **DataVisu.R** is used to condition the data before visualization.

names	class	description	default value
x	data.frame	Design	NA
y	vector	Response	NA
crit	function	AIC, BIC or tune	NA
stockage	boolean	Data storage or not	FALSE
itmax	integer	maximum iteration number	500
ncores	integer	number of cores used	1

names	class	description
best	list	groups composition
critere	matrix (size 1×1)	Criterion value (BIC, AIC or tune)
model	gp	GP model
time	proc_time	Estimation time
Stock	list	storage of model estimate at each step
	character	"No storage"

TABLE 2.D.1 – Description of the functions : **algo1Base.R**, **algo2Base.R**, **algo3Base.R**, **algo4Base.R** and **algoWelch.R**

Applications	section	file R	parallelized
Comparison	4.1.1	Bench.R	Yes
Learning set size	4.1.2	Size.R	Yes
Error added on parameters	4.1.3	Err.R	Yes
nugget effect	4.1.4	Nugg.R	Yes
Sobol function	4.2	Sobol.R	No

TABLE 2.D.2 – Files to run the applications

Files to run the five applications done in this article. The first four examples need the file **DataVisu.R** to clean the data for the visualization.

Annexes supplémentaires

2.E Calcul du nombre de modèles estimés par l'Algorithme 1 (Section 3.1)

*Cette section présente la démonstration du nombre de modèles estimés à chaque étape de l'**Algorithme** 1.*

Soit $k > 1$ l'étape actuelle de l'algorithme, à l'étape précédente le modèle contient $k-1$ groupes. Le but est donc de compter le nombre N de modèles à estimer à l'étape k, sachant qu'un nouveau modèle est construit en séparant en deux un des $k-1$ groupes. Soit n_j le nombre de modèles crées en séparant en deux le groupe j, $j = 1, \ldots, k-1$. Le nombre de modèles à estimer à l'étape k est donné par :

$$N = \sum_{j=1}^{k-1} n_j$$

Il reste à calculer n_j le nombre de façons de séparer en deux le j-ème groupe composé de p_j éléments. Il faut étudier toutes les possibilités lorsque p_j pair et p_j impair. Afin de simplifier la lisibilité des calculs, p_j est noté p et n_j est noté n dans cette partie.

- p pair :

$$n = \sum_{i=1}^{\frac{p}{2}-1} \binom{p}{i} + \frac{1}{2}\binom{p}{\frac{p}{2}}$$

or $\sum_{i=1}^{\frac{p}{2}-1} \binom{p}{i} = \sum_{i=\frac{p}{2}+1}^{p} \binom{p}{i}$ d'où

$$\begin{aligned} n &= \frac{1}{2} \sum_{\substack{i=1 \\ i \setminus \{\frac{p}{2}\}}}^{p} \binom{p}{i} + \frac{1}{2}\binom{p}{\frac{p}{2}} \\ &= \frac{1}{2} \sum_{i=1}^{p} \binom{p}{i} \\ &= \frac{1}{2}(2^p - 2) \\ &= 2^{p-1} - 1 \end{aligned}$$

- p impair :

$$n = \sum_{i=1}^{\lfloor \frac{p}{2} \rfloor} \binom{p}{i}$$

avec $\lfloor . \rfloor$ et $\lceil . \rceil$ les fonctions partie entière inférieure et supérieure, de plus $\sum_{i=1}^{\lfloor \frac{p}{2} \rfloor} \binom{p}{i} =$

$\sum_{i=\lceil \frac{p}{2} \rceil}^{p} \binom{p}{i}$ d'où

$$= \frac{1}{2} \sum_{i=1}^{p} \binom{p}{i}$$
$$= 2^{p-1} - 1$$

Par conséquent, $n_j = 2^{p_j - 1} - 1$ et le nombre de modèle estimés à l'étape k est :

$$N = \sum_{j=1}^{k-1} (2^{p_j - 1} - 1)$$

2.F Simulation complémentaires pour la comparaison des quatre algorithmes (Section 4.1.1)

Cette section propose une étude complémentaire sur les quatre algorithmes présentés précédemment.

Afin de se placer dans un contexte plus proche de la problématique abordée au court de ce manuscrit, l'ensemble d'apprentissage choisi est de taille $40p = 320$ au lieu de 800. A part cette différence, les conditions de l'étude sont les mêmes que celle de la Section 4.1.1. La Table 2.F.1 montre que les algorithmes 2, 3 et 4 donnent les meilleurs résultats. Ces algo-

	bons groupes	mauvais groupes
alg1	44	56
alg2	47	53
alg3	47	53
alg4	47	53
algW	0	100

TABLE 2.F.1 – Erreur de regroupement pour les algorithmes sur 100 trajectoire avec un ensemble d'apprentissage de taille 320.

rithmes se trompent sur les mêmes simulations, baisser d'avantage le nombre d'observations ne permettra pas de les différencier .La seule option pour sélectionner l'algorithme le plus efficace est de comparer les temps de calcul. C'est le quatrième qui est le plus rapide (15 secondes au lieu de 21 pour les deux autres).

2.G Simulation complémentaires pour la fonction test Sobol (Section 4.2)

Cette section propose deux études complémentaires sur la fonction test de Sobol.

Afin de compléter la simulation effectuée sur la fonction de Sobol (Section 4.2), la même expérience est reproduite mais en augmentant la limite supérieure des paramètres de portée pour le noyau *anisotrope*. Dans l'étude précédente, les bornes étaient fixées par défaut

à 2 pour chaque direction ($2 \times |\max(x_j) - \min(x_j)|$,$i = 1, \ldots, p$). Dans cette étude, les bornes sont fixées à 100. Le but est de permettre au krigeage avec un noyau *anisotrope* d'aller chercher des valeurs de portée élevées pour les variables qui ne sont pas influentes. L'**Algorithme** 4 a trouvé 5 groupes : $\mathcal{I}_1 = \{1, 3\}$, $\mathcal{I}_2 = \{2\}$, $\mathcal{I}_3 = \{4\}$, $\mathcal{I}_4 = \{5, 6\}$ et $\mathcal{I}_5 = \{7, 8, 9, 10, 11, 12, 13, 14, 15\}$. Cinq paramètres de portée sont estimés au lieu de 15 pour le krigeage avec un noyau *anisotrope* et 1 pour un noyau *isotrope*. Les valeurs des paramètres estimés sont dans la Table 2.G.1. La Figure 2.G.1 montre la comparaison de la qualité de prédiction des trois modèles. Le krigeage avec un noyau *isotrope par groupe* n'améliore pas la qualité de prédiction comparé au noyau *anisotrope* mais la qualité de prédiction est très proche. Le krigeage avec un noyau *isotrope* donne de mauvais résultats.

***Isotropic by group* kernel**

$\theta_1(x_1, x_3)$	$\theta_2(x_2)$	$\theta_3(x_4)$	$\theta_4(x_5, x_6)$	$\theta_5(x_7, \ldots, x_{15})$	m	σ^2
1.53	1.8	2.57	4.6	42.12	1.78	207.06

Anisotropic kernel

θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9
1.39	2.18	1.84	3.42	4.94	4.91	68.24	47.79	77.55
θ_{10}	θ_{11}	θ_{12}	θ_{13}	θ_{14}	θ_{15}	m	σ^2	
67.99	48.34	41.18	78.68	59.43	56.6	2.02	492.67	

***Isotropic* kernel**

θ	m	σ^2
3.2	6.35	177.49

TABLE 2.G.1 – Paramètre de krigeage pour un noyau *isotrope par groupe* (en haut), *anisotrope* (au milieu) et *isotrope* (en bas). L'ensemble d'apprentissage est un Hypercube Latin optimisé de 250 points. L'ensemble test est construit à partir des suites de Sobol, il contient 8000 points.

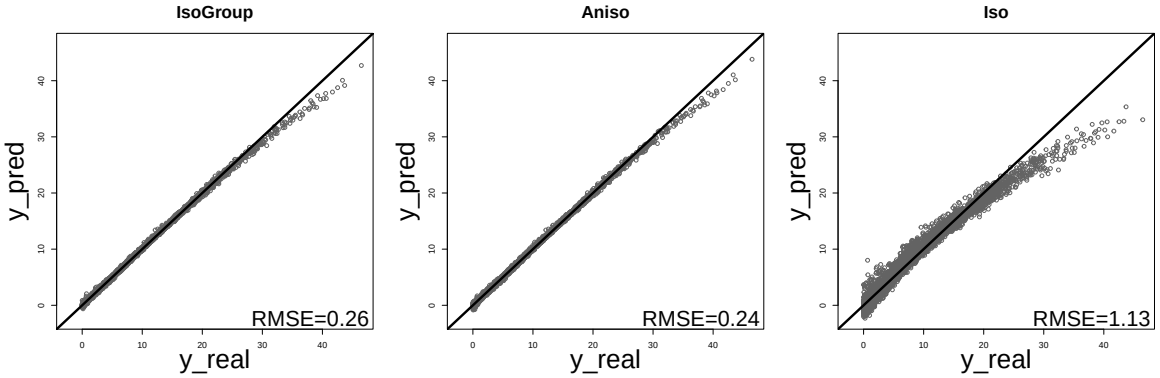

FIGURE 2.G.1 – Graphique de la qualité de prédiction par krigeage pour un noyau *isotrope par groupe* (à gauche), *anisotrope* (au milieu) et *isotrope* (à droite). L'ensemble d'apprentissage est un Hypercube Latin optimisé de 250 points. L'ensemble test est construit à partir des suites de Sobol, il contient 8000 points.

Dans cette dernière expérience, seulement la taille de l'ensemble d'apprentissage change. Au lieu de prendre 250 points, le design est composé de 45 observations. Le but est de se placer dans un contexte où le nombre de points d'apprentissage est très restreint par rapport à la dimension. L'**Algorithme** 4 a trouvé 3 groupes : $\mathcal{I}_1 = \{1, 2, 3\}$, $\mathcal{I}_2 = \{4, 5, 6\}$ et $\mathcal{I}_3 = \{7, 8, 9, 10, 11, 12, 13, 14,$
$15\}$. Trois paramètres de portée sont estimés au lieu de 15 pour le krigeage avec un noyau *anisotrope* et 1 pour un noyau *isotrope*. Les valeurs des paramètres estimés sont dans la Table 2.G.2. La figure 2.G.2 montre la comparaison de la qualité de prédiction des trois modèles. Le krigeage avec un noyau *isotrope par groupe* améliore significativement la qualité de prédiction par rapport aux deux autres modèles.

***Isotropic by group* kernel**

$\theta_1(x_1, x_2, x_3)$	$\theta_2(x_4, x_5, x_6)$	$\theta_3(x_7, \ldots, x_{15})$	m	σ^2
1.75	4.21	90	13.78	581.83

Anisotropic kernel

θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9
0.95	2.44	0.69	5.8	81.43	59.93	65.89	58.08	3.37
θ_{10}	θ_{11}	θ_{12}	θ_{13}	θ_{14}	θ_{15}	m	σ^2	
56.47	90.29	54.3	46.36	44.04	63.45	18.86	307.88	

***Isotropic* kernel**

θ	m	σ^2
1.66	8.06	102.74

TABLE 2.G.2 – Paramètre de krigeage pour un noyau *isotrope par groupe* (en haut), *anisotrope* (au milieu) et *isotrope* (en bas). L'ensemble d'apprentissage est un Hypercube Latin optimisé de 45 points. L'ensemble test est construit à partir des suites de Sobol, il contient 8000 points.

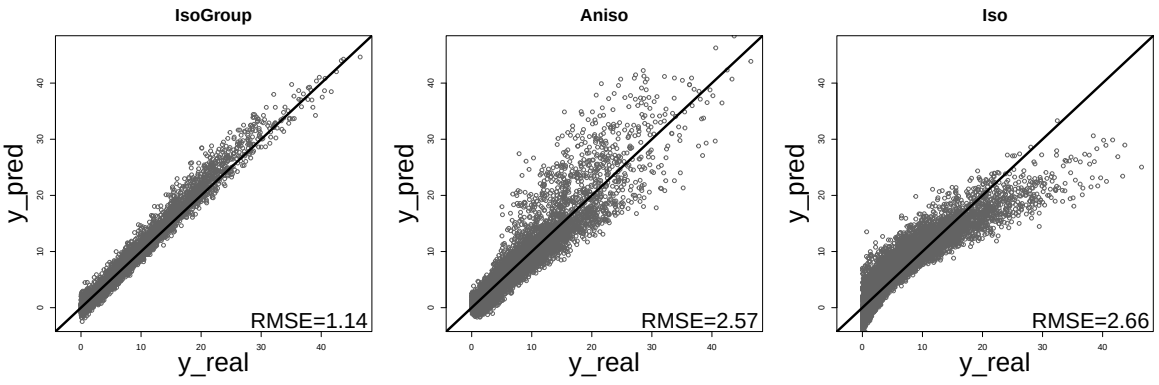

FIGURE 2.G.2 – Graphique de la qualité de prédiction par krigeage pour un noyau *isotrope par groupe* (à gauche), *anisotrope* (au milieu) et *isotrope* (à droite). L'ensemble d'apprentissage est un Hypercube Latin optimisé de 45 points. L'ensemble test est construit à partir des suites de Sobol, il contient 8000 points.

Conclusion du chapitre

Cette section propose un bref résumé de l'article. Elle présente les méthodes et donne un aperçu des résultats obtenus.

Les méthodes présentées dans ce chapitre proposent une solution aux problèmes liés à la grande dimension. Le noyau *isotrope par groupe* permettant de réduire le nombre de paramètres à estimer est construit à partir des noyaux *anisotrope* et *isotrope* avec une distance euclidienne. Par exemple, un noyau de covariance *isotrope par groupe* avec trois variables et deux groupes $\{x_1\}$, $\{x_2, x_3\}$ s'écrit :

$$\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \rho_{\theta_1}(|x_1 - \tilde{x}_1|) \times \rho_{\theta_2}\left(\sqrt{(x_2 - \tilde{x}_2)^2 + (x_3 - \tilde{x}_3)^2}\right)$$

avec ρ une fonction de covariance à choisir.
Quatre algorithmes sont développés pour déterminer la répartition des variables dans les groupes de façon automatique. Ils sont comparés à celui proposé par [Welch et al., 1992]. Le tableau 2.3 résume

Algorithmes	alg1	alg2	alg3	alg4	algW
Stratégie	Quasi exhaustive	Séparatrice	Séparatrice + Cluster à la fin	Séparatrice + Cluster à chaque étape	Séparatrice
Direction	Forward	Both	Both	Both	Forward
Temps	> 6h	< 2h	< 2h	< 1h30	< 1h30
Bons groupes	Oui	Oui	Oui	Oui	Non

TABLE 2.3 – Stratégies, directions et résultats des quatre algorithme et de l'algorithmes proposé par [Welch et al., 1992]

les caractéristiques ainsi que les résultats obtenus sur un exemple simulé d'un processus gaussien *isotrope par groupe* en dimension huit. Ces résultats sont détaillés dans la section 4.1.1. Le dernier algorithme (alg4) est le plus efficace.

L'**Algorithme** 4 est alors exécuté dans divers cas de processus Gaussien simulés avec le noyau *isotrope par groupe*. Pour commencer, la section 4.1.2 teste l'habileté de l'algorithme à retrouver les bons groupes en fonction du nombre de points. A partir d'environ soixante points par direction, l'algorithme retrouve à chaque fois les bons groupes. Lorsque le nombre de points est plus faible (10 par direction), il retrouve les bons groupes dans seulement 16% des cas. Il reste satisfaisant en prédiction et meilleur que le krigeage avec un noyau *isotrope*. Le noyau *anisotrope* quant à lui se trompe complètement dans 20% des cas. Le noyau *isotrope par groupe* permet donc bien de robustifier la prédiction par rapport à l'incertitude des estimations des portées. L'étude suivante (section 4.1.3) consiste à ajouter une perturbation sur les portées lors de la simulation du processus. Cette erreur ne change pas la capacité de l'algorithme à donner une composition de groupe permettant d'obtenir de bons résultats en prédiction. Contrairement à l'ajout d'un effet de pépite (section 4.1.4) qui engendre une perte de qualité de prédiction du krigeage avec un noyau *isotrope par groupe*. Ce noyau n'est pas la meilleure solution pour l'estimation d'un processus Gaussien avec effet de pépite. Il reste cependant convenable par rapport aux résultats obtenus avec les deux autres noyaux. Enfin, l'application de la méthode sur la fonction de Sobol (section 4.2 et applications supplémentaires) montre que l'utilisation de l'algorithme avec un noyau *isotrope par groupe* permet d'améliorer la qualité de prédiction.

Chapitre 3

Optimisation robuste à l'aide d'un métamodèle de krigeage avec dérivées

Introduction du chapitre

Au cours de ce chapitre, la problématique d'optimisation robuste est traité afin d'identifier la configuration géométrique de la turbomachine telle que le rendement soit maximal et peu affaibli par d'éventuelles perturbations sur la géométrie. La problématique revient donc à trouver des maxima (minima) stables. La sortie est considérée comme sensible aux perturbations, c'est à dire qu'un décalage même léger dans le domaine des entrées peut influencer la sortie. Les optima cherchés correspondent à des zones où le décalage aura le moins de conséquences. Dans un contexte industriel, des perturbations sont fréquemment observées en production, elles sont dues à des erreurs de fabrication. Ainsi dans le contexte des turbomachines, le dessin optimal recherché est celui dont le rendement n'est que faiblement impacté par des perturbations sur la géométrie.

Dans ce paragraphe, une machine quelconque dont l'objectif est de maximiser le rendement est prise comme exemple. La géométrie est supposée dépendante de deux variables d'entrées x et y. La Figure 3.1 montre la forme du rendement ainsi qu'un maximum robuste (carré noir) et un maximum moins robuste (cercle noir). Le rendement minimal est considéré comme acceptable lorsqu'il est supérieur à un certain seuil (ligne horizontale). Dans cet exemple, les disques gris représentes cent machines construites au point le moins robuste. La plupart de ces machines (disques gris) seront en dessous du seuil minimal, environ 80 sur la figure. Seulement 20% des machines seront utilisables, ce qui est inacceptable dans l'industrie. Au contraire, les machines construites à l'optimum le plus robuste (carré noir) ont un rendement acceptable dans 100% des cas.

L'optimisation robuste consiste à trouver des optima de la fonction objective peu influencés par les perturbations sur les entrées. Cependant, la quantification de la robustesse d'une solution est un véritable challenge. Plusieurs critères ont été développés dans la littérature, [Göhler et al., 2016] en propose une étude. Dans ce chapitre, le critère choisi est une approximation de la variance de la fonction. Ce critère est basé sur le développement de Taylor et utilise les dérivées premières et secondes de la fonction. Le but est de trouver les configurations géométriques ayant un rendement maximal et une variation du rendement minimale au voisinage du maximum. Souvent ces objectifs sont antagonistes, il faut trouver un équilibre entre les deux objectifs. Cet équilibre est réalisé sur le front de Pareto [Jozefowiez, 2013]. Le contexte des simulations coûteuses impose le développement d'une stratégie efficace permettant de procéder à l'optimisation en un nombre raisonnable d'appels au code numérique. Sept stratégies ont été développées. Elles suivent un schéma d'optimisation multi-objectif mixant un metamodèle de krigeage et l'algorithme NSGA II. Le métamodèle permet d'obtenir une

surface de réponse de qualité malgré le faible nombre d'appels au code. Le NSGA II permet d'explorer au mieux les deux objectifs et de trouver un front de Pareto.

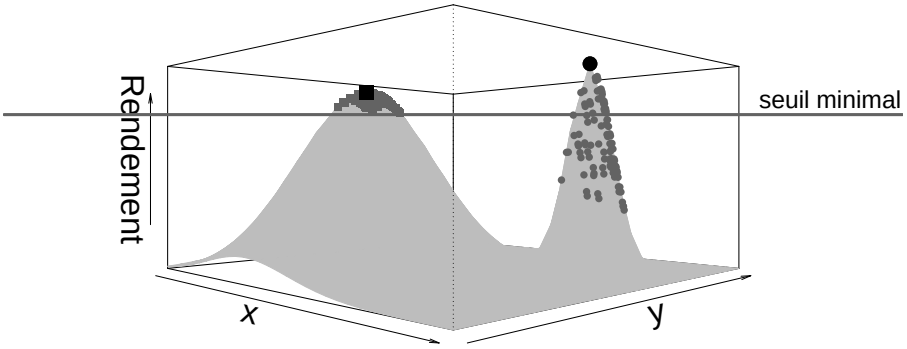

FIGURE 3.1 – Exemple d'une fonction de rendement en 2D avec deux maxima, un robuste (carré noir) un autre moins robuste (cercle noir). Le seuil minimal de rendement est fixé à une valeur quelconque (ligne horizontale). Cent points sont simulés autour des deux maximums avec la même loi (carrés et cercles gris).

Le critère de robustesse utilisé fait intervenir les dérivées premières et secondes du code car il se base sur le développement de Taylor. Le code 1D (cas industriel) permet d'observer les dérivées de la fonction en chaque point d'observation. Un modèle de co-krigeage est donc construit sur la fonction et les dérivées en définissant le processus Gaussien suivant :

$$(Z(\mathbf{x}))_{\mathbf{x}\in D} = \left(Y, \frac{\partial Y}{\partial x_1}, \ldots, \frac{\partial Y}{\partial x_p}, \frac{\partial^2 Y}{\partial x_1 \partial x_2}, \ldots, \frac{\partial^2 Y}{\partial x_{p-1}\partial x_p}, \frac{\partial^2 Y}{\partial x_1^2}, \ldots, \frac{\partial^2 Y}{\partial x_p^2}\right)$$

Le processus $(Y(\mathbf{x}))_{\mathbf{x}\in D}$ est un processus Gaussien de moyenne constante et de noyau de covariance $k(\mathbf{x}, \tilde{\mathbf{x}}), \forall(\mathbf{x}, \tilde{\mathbf{x}}) \in D \times D$ symétrique et dérivable deux fois à gauche (cf [Rasmussen and Williams, 2006]). Le noyau étant symétrique, il est donc aussi dérivable deux fois à droite. De plus, si celui-ci est stationnaire, il est alors dérivable quatre fois. Les processus dérivés sont aussi des processus Gaussiens. Leur moyenne est nulle et leur structure de covariance définie par rapport au noyau k est telle que :

$$\begin{aligned}
cov\left(Y(\mathbf{x}), \frac{\partial Y(\tilde{\mathbf{x}})}{\partial \tilde{x}_j}\right) &= \frac{\partial k(\mathbf{x}, \tilde{\mathbf{x}})}{\partial \tilde{x}_j} \\
cov\left(\frac{\partial Y(\mathbf{x})}{\partial x_i}, \frac{\partial Y(\tilde{\mathbf{x}})}{\partial \tilde{x}_j}\right) &= \frac{\partial^2 k(\mathbf{x}, \tilde{\mathbf{x}})}{\partial x_i \partial \tilde{x}_j} \\
cov\left(\frac{\partial Y(\mathbf{x})}{\partial x_i}, \frac{\partial^2 Y(\tilde{\mathbf{x}})}{\partial \tilde{x}_j^2}\right) &= \frac{\partial^3 k(\mathbf{x}, \tilde{\mathbf{x}})}{\partial x_i \partial \tilde{x}_j^2} \\
cov\left(\frac{\partial^2 Y(\mathbf{x})}{\partial x_i^2}, \frac{\partial^2 Y(\tilde{\mathbf{x}})}{\partial \tilde{x}_j^2}\right) &= \frac{\partial^4 k(\mathbf{x}, \tilde{\mathbf{x}})}{\partial x_i^2 \partial \tilde{x}_j^2}
\end{aligned}$$

L'utilisation des dérivées améliore de manière conséquente la qualité de prédiction comme l'illustre l'exemple sur la fonction sinus (cf Figure 3.2). En effet, pour le cas sans dérivée, quatre points d'observation de la fonction ont été choisis. Alors que, pour la seconde figure huit points sont observés (quatre pour la fonction et quatre pour sa dérivée).

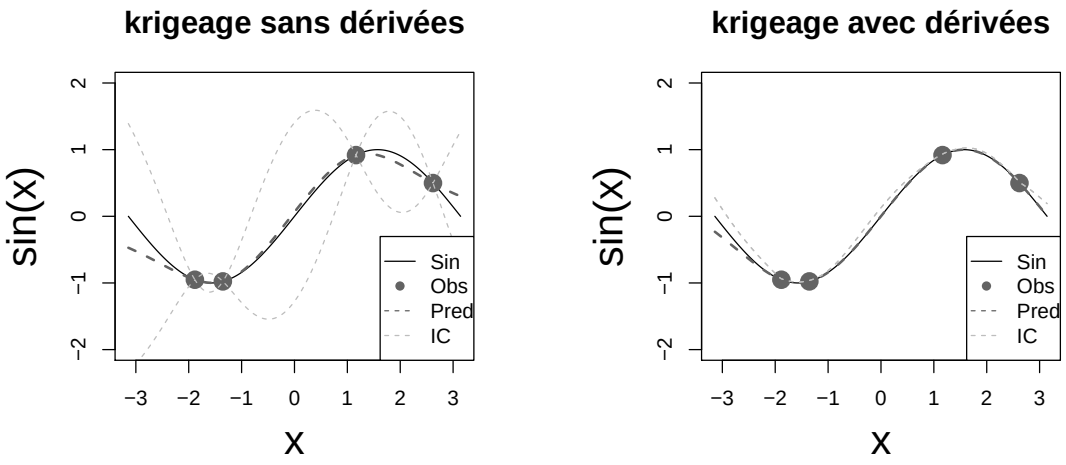

FIGURE 3.2 – Graphiques de prédiction de la fonction sinus. A gauche les résultats pour un krigeage classique et à droite pour un krigeage avec dérivées.

Tout d'abord, le critère de robustesse basé sur le métamodèle est introduit. Puis, le modèle de co-krigeage avec dérivées est présenté. Ensuite, le schéma d'optimisation robuste général est décrit. Puis, sept stratégies permettant la construction du front de Pareto sont développées. Enfin, les stratégies sont testées sur deux exemples.

Le chapitre suivant correspond à l'article "Robustness kriging-based optimization" soumis le 4 juillet 2018 dans le "Journal of Global Optimization". Les travaux de ce chapitre ont été présentés lors des conférences nationales CFM 2017 et JDS 2018 ainsi que lors des conférences internationales WCSMO12 (2017) en Allemagne et UseR 2018 en Australie.

Abstract

In the context of robust shape optimization, the estimation cost of some physical models is reduced by the use of a response surface. The multi objective methodology for robust optimization that requires the partitioning of the Pareto front (minimization of the function and the robustness criterion) has already been developed. However, the efficient estimation of the robustness criterion in the context of time-consuming simulation has not been much explored. We propose a robust optimization procedure based on the prediction of the function and its derivatives by a kriging. The second moment is replaced by an approximated version using Taylor theorem. A Pareto front of the robust solutions is generated by a genetic algorithm named NSGA-II. This algorithm gives a Pareto front in an reasonable time of calculation.
We detail seven relevant strategies and compare them for the same budget in two test functions (2D and 6D). In each case, we compare the results when the derivatives are observed and not.

Keywords. Robust Optimization, Kriging, Expected Improvement

1 Introduction

Complex physical phenomena are more and more studied through numerical simulations. These numerical models are able to mimic real experiments with a high accuracy. They predict the physical measures of interest (outputs) very precisely. Then, numerical simulations are used as a replacement for real experiments because they are less costly in primary materials. Sometimes, the solutions of the optimization problem could be sensitive to inputs' perturbations. For example, these perturbations are due to random fluctuations during production. A solution of a multi-objective optimization problem is then looked for where the first objective is the function itself and the second is a robustness criterion. These two objectives are assumed to be antagonistic. The issue of robust optimization (RO) is to find a Pareto front that makes a balance between the optimization of the function and the impact of input perturbations (uncertainties). As the simulations given by the numerical code are often time-consuming, only a few simulations are then affordable. So we cannot exploit intensively the computer code to provide the robust optimum.

In the context of costly simulations, the optimization procedure is often run on a kriging model that statistically approximates the computer code (kriging-based black-box optimization). Choosing where to sample the output in the input space to reach the optimum as fast as possible is a big issue. [Jones et al., 1998] developed the Efficient Global Optimization (EGO) algorithm that exploits the Expected Improvement (EI) criterion. However, the EGO algorithm is not an answer to the robust optimization problem because uncertainties are not taken into account.

Uncertainties can be of different kinds. They have to be well identified to make the robust optimization accurate. [Lelièvre et al., 2016] propose a classification of different approaches that deal with uncertainties in the context of reliability, optimization and robustness. In order to provide this survey, they sort uncertainties in two main groups : uncertainties that "are primitively linked to the environment and condition of use" and uncertainties that "are those connected with the production/manufacturing process".

In literature, we can find a sample of works that handle robust optimization with the first type of uncertainties. The aim is to find $\mathbf{x}$ such that $f(\mathbf{x}, \mathbf{u})$ is optimal with $\mathbf{u}$ the vector of the uncertain

variables (cf [Janusevskis and Le Riche, 2013], [Marzat et al., 2013], [Apley et al., 2006] and [Ur Rehman et al., 2014]). For example, [Janusevskis and Le Riche, 2013] propose a way to make a robust optimization based on a metamodel. They develop a Gaussian process (GP) model in the joint space $(\mathbf{x}, \mathbf{u})$ that takes into account the uncertainty of the $\mathbf{u}$-group of inputs. They define an adapted expected improvement and they maximize this criterion to enrich the design sequentially. [Marzat et al., 2013] propose an algorithm that make a Kriging Based Robust Optimization (KBRO) considering the worst-case. At each step i, they conduct two EGO. A first EGO is performed on the design space to found $\mathbf{x}^i$ that minimize $f(\mathbf{x}, \mathbf{u}^i)$. At the first iteration, they randomly choose $\mathbf{u}^i$ in the uncertain space. Then a second EGO is performed on the uncertain space to found $\mathbf{u}^i$ that minimizes $f(\mathbf{x}^i, \mathbf{u})$. They return $f(\mathbf{x}^i, \mathbf{u}^i)$ at the last iteration. In all these methods, the variables are clearly separated in two classes (design and uncertain) and the design is enriched sequentially.

In our context, we consider that the inputs we want to optimize deal with a little perturbation. The aim is to optimize the function $f(\mathbf{x} + \mathbf{H})$ where $\mathbf{x}$ are the design variables and $\mathbf{H}$ are the perturbations. This case is related to the second type of uncertainties described by [Lelièvre et al., 2016] (see section 4.3). [Ur Rehman et al., 2014] propose an algorithm close to the EGO to answer to this problem. They add a previous optimization that localizes the worst-case on the response surface, $\min_{\mathbf{x}} \max_{\mathbf{H}} \widehat{y}(\mathbf{x} + \mathbf{H})$ where $\widehat{y}$ is the kriging prediction. Then, they maximize the EI calculated with the worst case instead of the local minimum for the reference value. This solution is a first step to the robust optimization issue because the only difference with the EGO is the reference value on the EI. In addition, it provides only one point that makes a balance between the function to be optimized and the inputs' perturbations to be minimized. The entire Pareto front is not explored.

In our work we propose a multi-objective strategy to detect the whole set of robust solutions. The first objective is the function itself while the second objective is the robustness criterion which needs to be described. The robustness quantification of a solution is challenging, [Göhler et al., 2016], [Gabrel et al., 2014] and [Coco et al., 2014] give some overviews of different robustness criteria. Our industrial partners quantify the variability of a solution by the local variance of the output in the neighborhood of a solution(see e.g. [Apley et al., 2006] and [Troian et al., 2016]). One aim of this paper is to propose an accurate estimation of this local variance.

That is why, the robustness criterion we look for is based on Taylor development as proposed by [Darlington et al., 1999]. But [Darlington et al., 1999] do not provide a RO in the context of time-consuming simulations. In our article, the RO is coupled with a kriging. Once the criterion defined, we perform a kriging-based multi-objectif optimization on both the function and the robustness criterion. We choose the function instead of the mean because the inputs perturbation have already been taken into account in the robustness criterion. In addition, we are interested in the optimum and not the optimum in mean.

[Pronzato and Éric Thierry, 2003] study the behavior of the mean and the variance of the function computed with the Taylor Theorem. They prove that the kriging variance has a huge influence on the two moments and it is highly recommended to take into account this variance to make an efficient KBRO. All the KBRO strategies we develop take into account the kriging variance.

Since the Taylor theorem needs the values of derivatives, co-kriging is well adapted (see e.g [Le Gratiet, 2013]). This model is an extension of the kriging model. More precisely kriging is an interpolation technique which aims at predicting the output using an adapted underlying correlation function (see e.g [Santner et al., 2003]). The co-kriging method consists in exploiting the natural covariance structure between the GP model of the function and all the derivatives. This structure is described in [Rasmussen and Williams, 2006]. The observation of the derivatives are not necessary

to predict them, we only need observations of the function. However, all the observed derivatives are good to know to improve the prediction quality.

Then, the function and its robustness criterion are accessible through the co-kriging model. A mutiobtjective optimization is performed to provide solutions. [Wagner et al., 2010] make an overview of different multi-objective (MO) algorithms based on a kriging model : the aggregation methods (see [Knowles, 2006], [Liu et al., 2007] and [Zhang et al., 2010]), the Hypervolume methods (see [Ponweiser et al., 2008] and [Emmerich et al., 2011]), the maximin method (see [Svenson and Santner, 2016]), the uncertainty reduction method (see [Picheny, 2015]) and the MO method (see [Jeong and Obayashi, 2005]). [Henkenjohann and Kunert, 2007] shows that the aggregation methods are not efficient with a complex Pareto front. The hypervolume, maximin and uncertainty reduction algorithms need to make the multi-objective optimization on GP processes. As the robustness criterion we develop is not anymore Gaussian, it could be costly to adapt these methods in our case. We choose to develop some optimization procedures inspired by the MO EI introduced by [Jeong and Obayashi, 2005]. They propose to modify the reference value of the EI and they maximize the EI with a multi-objective algorithm. The MO EI is computed for each objective functions.

The article is structured as follows. Our robustness kriging-based criterion is introduced in section 2. In section 3, we introduce the estimation of our criterion in a context of a Gaussian process metamodeling. We present the general multi-objective scheme in section 4. The multi-objective optimization procedure is described in section 5. And finally, in section 6, we study the behavior of our methodology on two test cases.

2 Robustness criterion

The global aim of this article is to conduct a robust optimization of a two times differentiable function

$$\begin{array}{rccc} f: & D \subset \mathbb{R}^p & \longrightarrow & [a;b] \subset \mathbb{R} \\ & \mathbf{x} & \longmapsto & f(\mathbf{x}) \end{array} \tag{3.1}$$

To catch the robustness of f around a design point we consider a local variance, that's to say the variance of f in the neighborhood of the given point. However, we cannot compute the variance on the real function because it is too expensive. This section gives an approximation of this local variance that can easily be predicted in the context of a Gaussian process model of f.

Let $\mathbf{x} \in D$, an observation point. The variance of the function f around $\mathbf{x}$ is written $v_f(\mathbf{x}) = Var\left(f(\mathbf{x}+\mathbf{H})\right)$ where $\mathbf{H}$ represents fluctuations that can appear during fabrication. We consider that the production error $\mathbf{H}$ follows a Gaussian law. Then $\mathbf{H} \sim \mathcal{N}(0_{\mathbb{R}^d}, \Delta^2)$ where Δ^2 is defined by :

$$\Delta^2 = \begin{pmatrix} \delta_1^2 & 0 & \dots & 0 \\ 0 & \delta_2^2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \delta_p^2 \end{pmatrix}$$

Variances $\delta_1, \dots, \delta_p$ associated to each input are not necessary the same and are given by the experts.

A point $\mathbf{x}^1 \in D$ is considered less robust than a point $\mathbf{x}^2 \in D$ if $v_f(\mathbf{x}^1) > v_f(\mathbf{x}^2)$. In Figure 3.1, the minimum on the right (circles) is less robust than the one on the left (triangles). Let $\mathbf{h}^1, \dots, \mathbf{h}^N$,

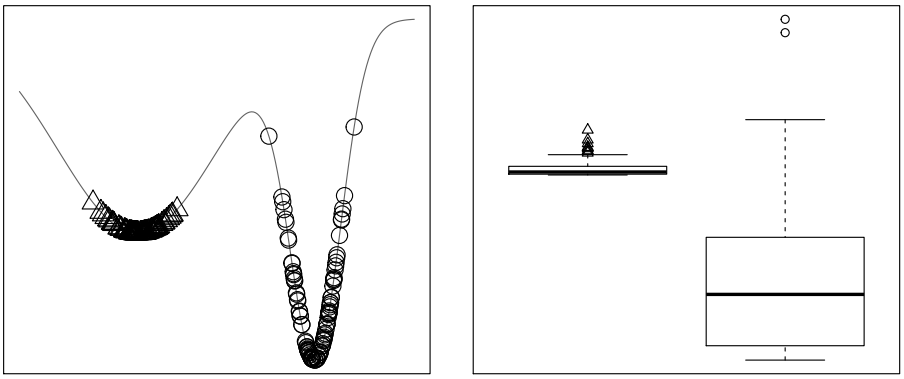

FIGURE 3.1 – Illustration of the robustness. The figure on the left shows a function in one dimension with two optima, the right one (circles) is the less robust. The figure on the right shows the variability of the points simulated by the same Gaussian law around the two optima.

$\mathbf{h}^j \in \mathbb{R}^p$, $j = 1, \ldots, N$ be N realizations of $\mathbf{H}$. The empirical estimation of the variance $v_f(\mathbf{x})$ is :

$$\widehat{v}_f(\mathbf{x}) = \frac{1}{N-1} \sum_{j=1}^{N} \left(f(\mathbf{x} + \mathbf{h}^j) - \bar{f}(\mathbf{x}) \right)^2 \tag{3.2}$$

where $\bar{f}(\mathbf{x}) = \frac{1}{N} \sum_{j=1}^{N} \left(f(\mathbf{x} + \mathbf{h}^j) \right)$ is the empirical mean (first moment). The estimation of the variance around only one point needs N calls to f.
N should satisfy the following inequality to obtain an estimation error lower than $\epsilon > 0$:

$$N \geq \left(z_{1-\alpha/2} \frac{\sqrt{\mu_4 - \left(S_f^2 \right)^2}}{\epsilon} \right)^2 \tag{3.3}$$

where $S_f^2 = \frac{1}{N} \sum_{j=1}^{N} \left(f(\mathbf{x} + \mathbf{h}^j) - \bar{f}(\mathbf{x}) \right)^2$, $\mu = \mathbb{E}[f(\mathbf{x}+\mathbf{H})]$, $\mu_4 = \mathbb{E}[(f(\mathbf{x}+\mathbf{H}) - \mu)^4]$, $z_{1-\alpha/2}$ is the quantile of risk α of the standard normal distribution and ϵ is the precision chosen by the user. The demonstration of the result (3.3) can be found in Appendix 3.A. This result implies that the number of calls to the function is too large to achieve a correct precision. To overcome this problem, we quantify the robustness using a Taylor approximation. This technique has been used for example in [Darlington et al., 1999].

For all $\mathbf{h} \in \mathbb{R}^p$, one has :

$$f(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + \nabla_f(\mathbf{x}) \cdot \mathbf{h} + \frac{1}{2} \mathbf{h}' \mathbb{H}_f(\mathbf{x}) \mathbf{h} + o(\|\mathbf{h}\|^2)$$

where ∇_f is the gradient of f and $\mathbb{H}_f$ the Hessian matrix of f. We introduce :

$$\widetilde{f}(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + \nabla_f(\mathbf{x}) \cdot \mathbf{h} + \frac{1}{2} \mathbf{h}' \mathbb{H}_f(\mathbf{x}) \mathbf{h}$$

Then, we define the robustness criterion by the following approximation of the local variance :

$$RC_f(\mathbf{x}) = Var\left(\widetilde{f}(\mathbf{x} + \mathbf{H}) \right)$$

An analytical form of this expression can be calculated (see [Beyer and Sendhoff, 2007] or the demonstration is in Appendix 3.B) and is given by the following expression :

$$RC_f(\mathbf{x}) = tr\left(\nabla_f(\mathbf{x}) \nabla_f(\mathbf{x})' \Delta^2 \right) + \frac{1}{2} tr\left(\mathbb{H}_f^2(\mathbf{x}) (\delta_1^2, \ldots, \delta_p^2)(\delta_1^2, \ldots, \delta_p^2)' \right) \tag{3.4}$$

where tr is the matrix trace. If the output of a simulation provides the results of the function and the first derivatives, RC_f criterion can be computed with only one call to the computer code. However in the context of costly simulations RO cannot be directly done on f and RC_f.
In the next section we present how with a kriging approach these quantities can be predicted.

3 Kriging prediction of the robustness criterion

As it can be seen in Equation (3.4), the robustness criterion depends on the first and second derivatives of f. A Gaussian process metamodel is well suited to this context in the sense that all derivatives can easily be predicted from that model. In this section, we present the co-kriging model and the predictions of the two objectives used in the robust optimization.

3.1 Co-kriging Model

This subsection is divided in two main parts. First, we present the general model for the function and its derivatives. Secondly, we introduce the kriging equations of prediction.

3.1.1 General model

We introduce a Gaussian Process (GP) to model the function, its first and second derivatives.

Let $(Y(\mathbf{x}))_{\mathbf{x}\in D}$ be a process with covariance function $k(\mathbf{x},\tilde{\mathbf{x}}), \forall(\mathbf{x},\tilde{\mathbf{x}}) \in D\times D$. This process is differentiable in mean square at point $(\mathbf{x},\tilde{\mathbf{x}})$ if and only if $\frac{\partial^2 \mathbf{k}}{\partial x_i \partial \tilde{x}_j}(\mathbf{x},\tilde{\mathbf{x}})$ exists $\forall i,j \in \{1,\dots,p\}$ and finite at point $(\mathbf{x},\tilde{\mathbf{x}}) = (\mathbf{t},\mathbf{t})$. In addition we have :

$$cov\left(Y(\mathbf{x}), \frac{\partial Y(\tilde{\mathbf{x}})}{\partial \tilde{x}_j}\right) = \frac{\partial k(\mathbf{x},\tilde{\mathbf{x}})}{\partial \tilde{x}_j}$$
$$cov\left(\frac{\partial Y(\mathbf{x})}{\partial x_i}, \frac{\partial Y(\tilde{\mathbf{x}})}{\partial \tilde{x}_j}\right) = \frac{\partial^2 k(\mathbf{x},\tilde{\mathbf{x}})}{\partial x_i \partial \tilde{x}_j}$$

We denote by $(Y_{x_i}(\mathbf{x}))_{\mathbf{x}\in D} = \left(\frac{\partial Y}{\partial x_i}(\mathbf{x})\right)_{\mathbf{x}\in D}$ the first-order partial derivative of $(Y(\mathbf{x}))_{\mathbf{x}\in D}$ with respect to x_i and by $\left(Y_{x_i,x_j}(\mathbf{x})\right)_{\mathbf{x}\in D} = \left(\frac{\partial^2 Y}{\partial x_i \partial x_j}(\mathbf{x})\right)_{\mathbf{x}\in D}$ the second-order partial derivative of $(Y(\mathbf{x}))_{\mathbf{x}\in D}$ with respect to x_i and x_j.

Let p be the number of input variables. Then each observation $\mathbf{x}$ is a vector with p coordinates, such that $\mathbf{x} = (x_1,\dots,x_p)$, $\mathbf{x} \in D$. The outputs (function and derivatives) at point $\mathbf{x}^k \in D$ are denoted by $y^k \in \mathbb{R}$, $y^k_{x_i} \in \mathbb{R}$ and $y^k_{x_i,x_j} \in \mathbb{R}$, where $i \in \{1,\dots,p\}$, $j \in \{i,\dots,p\}$ and $k \in \{1,\dots,n\}$. We note the collection of outputs $\mathbf{y}$, $\mathbf{y}_{x_i}$ and $\mathbf{y}_{x_i,x_j}$ such that :

$$\mathbf{y} = (y^1,\dots,y^n)'$$
$$\mathbf{y}_{x_i} = (y^1_{x_i},\dots,y^n_{x_i})'$$
$$\mathbf{y}_{x_i,x_j} = (y^1_{x_i,x_j},\dots,y^n_{x_i,x_j})'$$

In kriging context, $(y^k, y^k_{x_1},\dots,y^k_{x_p}, y^k_{x_1,x_1},\dots,y^k_{x_i,x_j},\dots,y^k_{x_p,x_p})$, $k \in \{1,\dots,n\}$ is assumed to be a realization of the following $d = 1 + \frac{3p}{2} + \frac{p^2}{2}$ dimensional GP :

$$Z(\mathbf{x}) = (Y(\mathbf{x}), Y_{x_1}(\mathbf{x}),\dots,Y_{x_p}(\mathbf{x}), Y_{x_1,x_1}(\mathbf{x}),\dots,Y_{x_i,x_j}(\mathbf{x}),\dots,Y_{x_p,x_p}(\mathbf{x})),\ 1 \le i \le p,\ i \le j \le p$$

at points $\mathbf{x}^1, \ldots, \mathbf{x}^n$ where $\mathbf{x}^\mathbf{k} \in D$, $k \in \{1, \ldots, n\}$, such that :

$$\begin{aligned} Y(\mathbf{x}) &= \mu + \eta(\mathbf{x}) \\ Y_{x_i}(\mathbf{x}) &= \eta_{x_i}(\mathbf{x}) \\ Y_{x_i,x_j}(\mathbf{x}) &= \eta_{x_i,x_j}(\mathbf{x}) \end{aligned}$$

where $\mu \in \mathbb{R}$ is the trend, the process $(\eta(\mathbf{x}))_{\mathbf{x}\in D}$ is a centered GP with a stationary covariance function that depends on a vector of range parameters $\boldsymbol{\theta} \in \mathbb{R}_+^p$ such that $Cov(\eta(\mathbf{x}), \eta(\mathbf{x})) = k_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \sigma^2 r_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}})$, $\forall(\mathbf{x}, \tilde{\mathbf{x}}) \in D \times D$. In this paper the trend μ and the variance σ^2 are assumed to be constants.
The process vector is then modeled as follow :

$$Z(\mathbf{x}) = \mathbf{m} + \epsilon(\mathbf{x}) \tag{3.5}$$

where $\mathbf{m} = (\mu, 0, \ldots, 0)' \in \mathbb{R}^d$ is the trend vector, the process $(\epsilon(\mathbf{x}))_{\mathbf{x}\in D}$ is the vector of d centered Gaussian processes i.e.

$$\epsilon(\mathbf{x}) = (\eta(\mathbf{x}), \eta(\mathbf{x})_{x_1}, \ldots, \eta(\mathbf{x})_{x_p}, \eta(\mathbf{x})_{x_1,x_1}, \ldots, \eta(\mathbf{x})_{x_i,x_j}, \ldots, \eta(\mathbf{x})_{x_p,x_p}),\ 1 \leq i \leq p,\ i \leq j \leq p$$

3.1.2 Kriging predictions

The co-kriging model presented by [Le Gratiet, 2013] is used to surrogate the function itself and its derivatives. The problem is to predict Z considering observations of z at points $x^1, ..., x^n$. But, the entire vector z is not always observable. Let $u_{obs} \subset \{1, \ldots, d\}$ be the components that are observable. For example, only the function and its first derivatives can be affordable. In the same way it is not always necessary to predict the whole vector z. Let $u_{pred} \subset \{1, \ldots, d\}$ be the components that are to be predicted.
The kriging mean is then given by the following equation :

$$\widehat{\mathbf{z}}_{u_{pred}}(\mathbf{x}) = \widehat{\mu} f_{pred} + \mathbf{c}_{\boldsymbol{\theta}}(\mathbf{x})' \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1} (\mathbf{z}_{u_{obs}} - \widehat{\mu}' f_{obs}), \widehat{\mathbf{z}}_{u_{pred}}(\mathbf{x}) \in \mathbb{R}^{d_{pred}} \tag{3.6}$$

where $\mathbf{z}_{u_{obs}} = (z^1_{u_{obs}}, \ldots, z^n_{u_{obs}})$ the observation vector, $1 \in u_{obs}$, $d_{obs} = \#u_{obs}$. $\widehat{\mathbf{z}}_{u_{pred}}(\mathbf{x})$ is the prediction vector and $d_{pred} = \#u_{pred}$ and $\widehat{\mu} = (f'_{obs}\Sigma_\theta^{-1} f_{obs})^{-1} f'_{obs}\Sigma_\theta^{-1}\mathbf{z}_{u_{obs}}$. The mean square error (MSE) at point $\mathbf{x} \in D$ is given by :

$$\widehat{\mathbf{s}}^2_{u_{pred}}(\mathbf{x}) = \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}) - \begin{pmatrix} f_{pred} & \boldsymbol{c}_{\boldsymbol{\theta}}(\mathbf{x}) \end{pmatrix} \begin{pmatrix} 0 & f'_{obs} \\ f_{obs} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{pmatrix}^{-1} \begin{pmatrix} f'_{pred} \\ \boldsymbol{c}_{\boldsymbol{\theta}}(\mathbf{x}) \end{pmatrix} \tag{3.7}$$

where $\widehat{\mathbf{s}}^2_{u_{pred}}(\mathbf{x}) \in \mathcal{M}_{d_{pred}\times d_{pred}}$, $f_{obs} = (1, 0_{\mathbb{R}^{d_{obs}-1}}, \ldots, 1, 0_{\mathbb{R}^{d_{obs}-1}})' \in \mathbb{R}^{nd_{obs}}$ and $f_{pred} = (1, 0_{\mathbb{R}^{d_{pred}-1}})' \in \mathbb{R}^{d_{pred}}$.
$\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ is the covariance matrix of size $nd_{obs} \times nd_{obs}$ given by :

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}} = \begin{pmatrix} \Sigma_{\mathbf{x}_1,\mathbf{x}_1}(u_{obs}, u_{obs}) & \ldots & \Sigma_{\mathbf{x}_1,\mathbf{x}_n}(u_{obs}, u_{obs}) \\ \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{x}_n,\mathbf{x}_1}(u_{obs}, u_{obs}) & \ldots & \Sigma_{\mathbf{x}_n,\mathbf{x}_n}(u_{obs}, u_{obs}) \end{pmatrix}$$

where

$$\Sigma_{\mathbf{x},\tilde{\mathbf{x}}} = \begin{pmatrix} \Sigma_{Y,Y} & \Sigma_{Y,Y_{\tilde{x}_j}} & \Sigma_{Y,Y_{\tilde{x}_j\tilde{x}_k}} & \Sigma_{Y,Y_{\tilde{x}^2_j}} \\ \Sigma_{Y_{x_i},Y} & \Sigma_{Y_{x_i},Y_{\tilde{x}_j}} & \Sigma_{Y_{x_i},Y_{\tilde{x}_j\tilde{x}_k}} & \Sigma_{Y_{x_i},Y_{\tilde{x}^2_j}} \\ \Sigma_{Y_{x_ix_l},Y} & \Sigma_{Y_{x_ix_l},Y_{\tilde{x}_j}} & \Sigma_{Y_{x_ix_l},Y_{\tilde{x}_j\tilde{x}_k}} & \Sigma_{Y_{x_ix_l},Y_{\tilde{x}^2_j}} \\ \Sigma_{Y_{x^2_i},Y} & \Sigma_{Y_{x^2_i},Y_{\tilde{x}_j}} & \Sigma_{Y_{x^2_i},Y_{\tilde{x}_j\tilde{x}_k}} & \Sigma_{Y_{x^2_i},Y_{\tilde{x}^2_j}} \end{pmatrix}$$

$i, j, k, l \in \{1, \dots p\}$ with $l > i$ and $k > j$. For instance $\Sigma_{Y_{x_i}, Y_{\tilde{x}_j}} = cov(Y_{x_i}, Y_{\tilde{x}_j}) = cov(\eta_{x_i}, \eta_{\tilde{x}_j}) = \frac{\partial^2 k(\mathbf{x}-\tilde{\mathbf{x}})}{\partial x_i \partial \tilde{x}_j}$. The matrix $\mathbf{c}_{\boldsymbol{\theta}}(\mathbf{x}) \in \mathcal{M}_{nd_{obs} \times d_{pred}}$ is the covariance matrix between $Z_{u_{pred}}(\mathbf{x})$ and the observations and the matrix $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}) \in \mathcal{M}_{d_{pred} \times d_{pred}}$ is the variance of $Z_{u_{pred}}(\mathbf{x})$.

3.2 Prediction of f

The prediction of the real function f is given by the co-kriging model and corresponds to the first coordinate of the vector $\widehat{z}_{u_{pred}}(\mathbf{x})$ in Equation (3.6) when $1 \in u_{pred}$ written :

$$\widehat{z}_{u_{pred}}(\mathbf{x}) = (\widehat{y}(\mathbf{x}), \dots, \widehat{y}_{x_p, x_p}(\mathbf{x}))$$

where $\widehat{y}$ is the prediction of the function f.

3.3 Prediction of RC_f

We propose to predict our robustness criterion by the co-kriging metamodel. The prediction $\widehat{z}_{u_{pred}}$ is used instead of the function to compute the criterion. The prediction of $RC_f(\mathbf{x})$ is given by :

$$RC_{\widehat{y}}(\mathbf{x}) = tr\left(\nabla_{\widehat{y}}(\mathbf{x}) \nabla_{\widehat{y}}(\mathbf{x})' \Delta^2\right) + \frac{1}{2} tr\left(\mathbb{H}^2_{\widehat{y}}(\mathbf{x})(\delta_1^2, \dots, \delta_p^2)'(\delta_1^2, \dots, \delta_p^2)\right) \tag{3.8}$$

where $\nabla_{\widehat{y}}$ is the vector $\begin{pmatrix} \widehat{y}_{x_1} \\ \vdots \\ \widehat{y}_{x_p} \end{pmatrix}$ and is the prediction of the gradient. $\mathbb{H}_{\widehat{y}}$ is the matrix $\begin{pmatrix} \widehat{y}_{x_1,x_1} & \dots & \widehat{y}_{x_1,x_p} \\ \vdots & \ddots & \vdots \\ \widehat{y}_{x_p,x_1} & \dots & \widehat{y}_{x_p,x_p} \end{pmatrix}$ and corresponds to the prediction of the hessian matrix. $\nabla_{\widehat{y}}$ and $\mathbb{H}_{\widehat{y}}$ are obtained from different components of $\widehat{z}_{upred}$.

3.4 Illustration with the six-hump Camel function

The studied function is the six-Hump Camel function, defined by :

$$f(\mathbf{x}) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1 x_2 + \left(-4 + 4x_2^2\right) x_2^2, \mathbf{x} \in [-2; 2] \times [-1; 1]$$

We consider for the kriging model a kernel which is *anisotropic* :

$$cov\left(Y(\mathbf{x}), Y(\tilde{\mathbf{x}})\right) = k(\mathbf{x} - \tilde{\mathbf{x}}) = \sigma^2 \prod_{j=1}^{p} \rho_{\theta_j}\left(|x_j - x'_j|\right), \boldsymbol{\theta} = (\theta_1, \dots, \theta_p) \in \mathbb{R}_+^p \tag{3.9}$$

where ρ_{θ_j} is a correlation function which only depends on the one dimensional range parameter θ_j, see e.g [Santner et al., 2003] and [Stein, 1999]. The *anisotropic* kernel contains as many parameters as the number of variables p. We use a Matern 5/2 kernel because the output is supposed to be two times continuously differentiable :

$$\forall \theta \in \mathbb{R}^+, \forall h \in \mathbb{R}^+, \rho_\theta(h) = \left(1 + \frac{\sqrt{5}|h|}{\theta} + \frac{5h^2}{3\theta^2}\right) \exp\left(-\frac{\sqrt{5}|h|}{\theta}\right).$$

We choose a maximin latin hypercube learning set of 10 points. The test set is a space filling of 1500 points. In the first application, without the derivatives, the observations are $y_1, \dots, y_{10}$ where $y_i = f(\mathbf{x}_i)$. In the second application, with the derivatives, the observations are $z_1, \dots, z_{10}$ where

$$z_i = \left(f(\mathbf{x}_i), \frac{\partial f(\mathbf{x}_i)}{\partial x_1}, \frac{\partial f(\mathbf{x}_i)}{\partial x_2}, \frac{\partial^2 f(\mathbf{x}_i)}{\partial x_1 \partial x_2}, \frac{\partial^2 f(\mathbf{x}_i)}{\partial x_1^2}, \frac{\partial^2 f(\mathbf{x}_i)}{\partial x_2^2}\right).$$

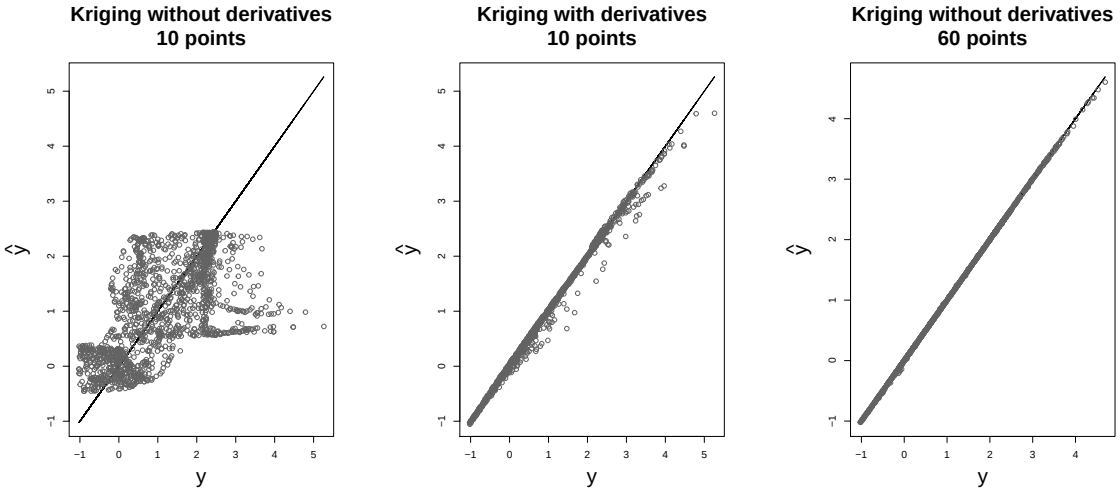

FIGURE 3.1 – Prediction plots for the six-hump Camel function : 10 points without observation of the derivatives (on the left), 10 with 5 derivatives (on the middle) and 60 points without observation of the derivatives (on the right).

In the third kriging, without the derivatives but with more observation points written $y_1, \ldots, y_{60}$ where $y_i = f(\mathbf{x}_i)$.

As expected, Figure 3.1 shows that kriging with derivatives does much better than without in the case of 10 points. If we consider that the computational cost of one derivative is the same as computing a new point, kriging without derivatives is better. In industrial application, computing all derivatives is cheaper than computing a new point.

4 Robust optimization procedure

In this section, we present the robust optimization procedure that uses our robustness criterion (cf Equation (3.8)). The approach to solve this optimization problem in the context of time consuming simulations is based on a classical black-box optimization scheme (see [Jones et al., 1998]). The robust optimization problem is written as :

$$\text{Find the Pareto set } \mathbb{X}_0, \text{ solution of the following multi-objective optimization}$$
$$\min_{\mathbf{x}\in\mathbb{R}^p} \{f(\mathbf{x}), RC_f(\mathbf{x})\} \tag{3.10}$$

The procedure is divided in two main parts : step 1, 2, 3 and step 4, 5, 6. The first is the initial part for the design and the Gaussian process. The second part solves the optimization problem using the metamodel. The procedure is adaptative : it consists in adding a set of q points (batch) at each iteration and it is repeated until the budget is reached. The size of the initial sample set and the size of the batches are given by the user. **Step 1** generates the initial sample $\mathbb{X}$ of n points spread in the p design variable space. An Optimized Latin Hypercube (OLH) is chosen. The observations of the function and its derivatives, when there are available, are computed in **Step 2**. In **Step 3** the co-kriging metamodel is estimated based on the observations. **Step 4** searches for non-dominated solutions of a kriging-based multi-objective problem using the optimizer NSGA II see e.g. [Deb et al., 2002]. **Step 5** aims at choosing the best set of q points among the Pareto front under a criterion described in next section. These points are added to the design $\mathbb{X}$ in **Step 6** and the simulation is run on these new points. The response surface $\widehat{\mathbf{y}}$ is then updated with the new observations. Figure 3.1 shows the general scheme.

Several choices are possible for **Step 4** and **Step 5**. The next section describes them.

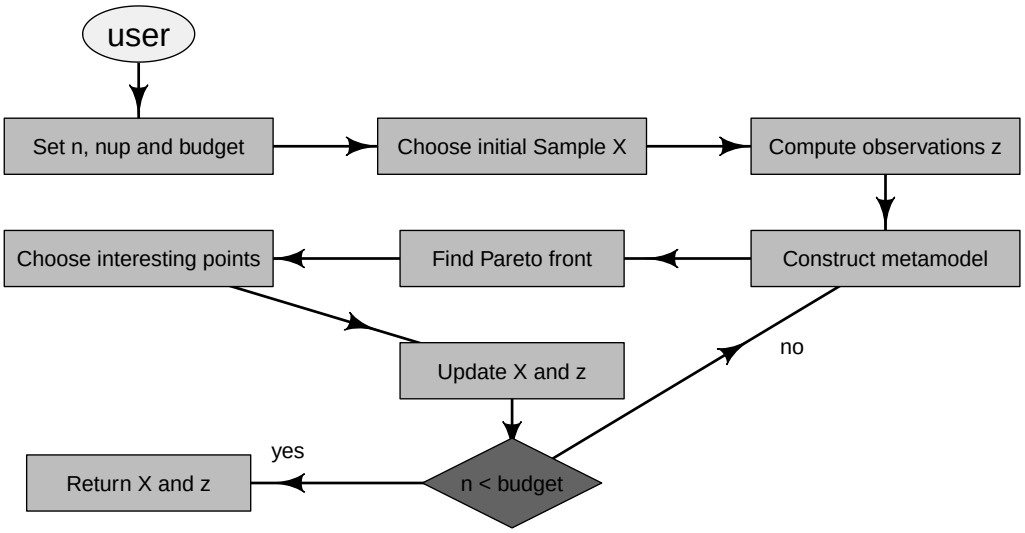

FIGURE 3.1 – The robust optimization procedure

5 Sequential procedure for the acquisition of new points

We have developed seven enrichment strategies based on three main MO optimization problems using the previous criterion. For the first problem, a NSGA II algorithm is applied to the prediction of the function and to the prediction of the robustness criterion (cf Equation (3.6) and (3.8)). Once the Pareto front is found, points are chosen using a random method or the Expected Improvement criterion. That's the enrichment step. A problem could appear during the Pareto front construction if kriging predictions turn out to be of poor quality. Some interesting areas can be missed. That is why, we introduce a second approach. The multiobjective optimization is conducted on the EI of the function and the robustness criterion. In this case the kriging variance is already taken into account during the optimization. Then several points are chosen among the Pareto front through different criteria. Selecting the good points from the Pareto front is not easy, so we introduce a last approach using qEI. This criterion measures the improvement of a batch of points. This strategy takes into account the kriging variance in the MO and add points by batch. The best point in the qEI space is selected in the Pareto front and corresponds to a set of q points in the design space. Before doing this, we recall some results on EI.

5.1 Background

In the EGO algorithm, the expected improvement (EI) criterion measures the improvement of a point $\mathbf{x}$ in the minimization of function f and is used to add new points to the learning set. The expression of the EI (cf [Jones et al., 1998]) at point $\mathbf{x}$ is :

$$EI(\mathbf{x}) = \mathbb{E}\left[(\min(y(\mathbb{X})) - Y(\mathbf{x}))^{+} \,|Y(\mathbb{X}) = \mathbf{y}\right]$$

where $\min(y(\mathbb{X})) = \min(y^1, \ldots, y^n)$.
The analytical expression of the EI for a Gaussian process is given by :

$$EI(\mathbf{x}) = (\min(y(\mathbb{X})) - \widehat{y}(\mathbf{x})\Phi\left(\frac{\min(y(\mathbb{X})) - \widehat{y}(\mathbf{x})}{\widehat{s}}\right) + \widehat{s}\phi\left(\frac{\min(y(\mathbb{X})) - \widehat{y}(\mathbf{x})}{\widehat{s}}\right)$$

where $\widehat{y}(\mathbf{x})$ is the kriging mean, $\widehat{s}(\mathbf{x})$ is the kriging standard deviation, Φ and ϕ are the cdf and pdf of the standard normal law.

In our case, we perform a multi-objective optimization on f and RC_f. Then, to evaluate an EI on RC_f we need to define the process $(RC_Y(\mathbf{x}))_{\mathbf{x}\in D}$. From Equation 3.4 the process is :

$$\begin{aligned} RC_Y(\mathbf{x}) &= tr\left(\begin{pmatrix} Y_{x_1}(\mathbf{x}) \\ \vdots \\ Y_{x_p}(\mathbf{x}) \end{pmatrix}\begin{pmatrix} Y_{x_1}(\mathbf{x}) & \dots & Y_{x_p}(\mathbf{x}) \end{pmatrix}\begin{pmatrix} \delta_1^2 & & \\ & \ddots & \\ & & \delta_p^2 \end{pmatrix}\right) \\ &+ \frac{1}{2}tr\left(\begin{pmatrix} Y^2_{x_1,x_1}(\mathbf{x}) & \dots & Y^2_{x_1,x_p}(\mathbf{x}) \\ \vdots & \ddots & \vdots \\ Y^2_{x_p,x_1}(\mathbf{x}) & \dots & Y^2_{x_p,x_p}(\mathbf{x}) \end{pmatrix}\begin{pmatrix} \delta_1^2 \\ \vdots \\ \delta_p^2 \end{pmatrix}\begin{pmatrix} \delta_1^2 & \dots & \delta_p^2 \end{pmatrix}\right) \\ &= \sum_{i=1}^{p} Y^2_{x_i}(\mathbf{x})\delta_i^2 + \frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{p} Y^2_{x_i,x_j}(\mathbf{x})\delta_i^2\delta_j^2 \end{aligned} \tag{3.11}$$

The EGO procedure in the context of multi-objective optimization can not be conducted without adapting the EI formula. That is why,in [Jeong and Obayashi, 2005] the authors change the reference value of the EI to adapt it to the multi-objective case. We recall that $\mathbf{y} = (y^1 = f(\mathbf{x}^1), \dots, y^n = f(\mathbf{x}^n))$. Let $RC_{\mathbf{y}} = \left(RC_{y^1} = RC_f(\mathbf{x}^1), \dots, RC_{y^n} = RC_f(\mathbf{x}^n)\right)$ be the evaluation of the robustness criterion on the design points. The multi-objective EI are :

$$\begin{aligned} EI_y(\mathbf{x}) &= \mathbb{E}\left[\left(\max(y(\mathbb{X}^*)) - Y(\mathbf{x})\right)^+ |\mathbf{z}_{u_{obs}}\right] \\ EI_{RC_y}(\mathbf{x}) &= \mathbb{E}\left[\left(\max(RC_y(\mathbb{X}^*)) - RC_Y(\mathbf{x})\right)^+ |\mathbf{z}_{u_{obs}}\right] \end{aligned}$$

where $\mathbb{X}^*$ is the set of non-dominated points for the objectives $\{y, RC_y\}$ of the learning set $\mathbb{X}$.

Remark :

- *A solution $\mathbf{x}^1$ dominates another solution $\mathbf{x}^2$ for the m objectives $g_1, \dots, g_m$ if and only if $\forall i \in \{1, \dots, m\}\ g_i(\mathbf{x}^1) \leq g_i(\mathbf{x}^2)$ and $\exists i \in \{1, \dots, m\}\ g_i(\mathbf{x}^1) < g_i(\mathbf{x}^2)$. Among a set of solution $\mathbb{X}$, the non-dominated set $\mathbb{X}^*$ (Pareto front) are those that are not dominated by any member of the set $\mathbb{X}$.*
- *When the derivatives used to compute the robustness criterion are not observed we replace them by the kriging prediction in $\max(RC_y(\mathbb{X}^*))$.*
- *The link between $RC_Y(\mathbf{x})$ and $Z(\mathbf{x})$ being not linear, the process $(RC_Y(\mathbf{x}))_{\mathbf{x}\in D}$ is not Gaussian anymore. EI_{RC_y} is then estimated by a Monte Carlo method.*

The EI makes a good balance between exploration and minimization but it computes the improvement of a single point. The multi-point EI (q-EI) developed by [Ginsbourger et al., 2010] is used to measure the improvement of q points $\mathbf{X} = (\mathbf{x}^{n+1}, \dots, \mathbf{x}^{n+q})'$.

$$\begin{aligned} qEI(\mathbf{X}) &= EI(\mathbf{x}^{n+1}, \dots, \mathbf{x}^{n+q}) \\ &= \mathbb{E}\left[\left(\min(y(\mathbb{X})) - \min(Y(\mathbf{x}^{n+1}), \dots, Y(\mathbf{x}^{n+q}))\right)^+ |\mathbf{z}_{u_{obs}}\right] \end{aligned}$$

In a multi-objective problem the q-EI are :

$$\begin{aligned} qEI_y(\mathbf{X}) &= \mathbb{E}\left[\left(\max(y(\mathbb{X}^*)) - \min(Y(\mathbf{x}^{n+1}), \dots, Y(\mathbf{x}^{n+q}))\right)^+ |\mathbf{z}_{u_{obs}}\right] \\ qEI_{RC_y}(\mathbf{X}) &= \mathbb{E}\left[\left(\max(RC_y(\mathbb{X}^*)) - \min(RC_Y(\mathbf{x}^{n+1}), \dots, RC_Y(\mathbf{x}^{n+q}))\right)^+ |\mathbf{z}_{u_{obs}}\right] \end{aligned}$$

5.2 Multi-objective optimization on the kriging predictor

The kriging prediction $\hat{y}$ and the robustness criterion $RC_{\hat{y}}$ are used instead of the real function f and the robustness criterion RC_f in the first group of strategies. The optimization problem of **Step 4** is written as :

Find the Pareto set $\mathbb{X}_0$, solution of the following multi-objective optimization

$$\min_{\mathbf{x}\in\mathbb{R}^p} \{\hat{y}(\mathbf{x}), RC_{\hat{y}}(\mathbf{x})\} \tag{3.12}$$

The NSGA II is used to compute the Pareto front. Five enrichment approaches to select q points from the Pareto front have been benchmarked and are described below (**Step 5**) :

1. MyAlea : $\lfloor \frac{q}{2} \rfloor$ [1] points are selected randomly on the Pareto front and $q - \lfloor \frac{q}{2} \rfloor$ points are randomly chosen in the parameter space.
2. MyEI : $-EI_y$ as well as $-EI_{RC_y}$ are computed for each point of the Pareto front. A k-means clustering using the method of [Hartigan and Wong, 1979] is applied to the non-dominated points of $\{-EI_y, -EI_{RC_y}\}$ to provide q clusters. Then the q clusters' medoids are added to the design.
3. MyqEI : a simulated annealing algorithm gives the set of q points among the Pareto front that minimizes the function $-qEI_y - qEI_{RC_y}$.

   Two sequential approaches presented in [Ginsbourger et al., 2010] can be used as the replacement of the q-EI to measure the improvement of q points : the Kriging Believer and the Constant Liar.

4. MyKB : q points are sequentially selected from the Pareto front based on the Kriging Beliver strategy. The $-EI_y$ and $-EI_{RC_y}$ are computed on the Pareto front, then a point $\mathbf{x}_0^1$ is randomly chosen from the EI Pareto front and added. $\widehat{y}(\mathbf{x}_0^1)$ is then considered known and is assumed to be equal to $\widehat{y}(\mathbf{x}_0^1)$. Another computation of $-EI_y$ and $-EI_{RC_y}$ provides one more point based on the same strategy up to the q requested points.
5. MyCL : q points are sequentially selected based on the Constant Liar strategy. The $-EI_y$ and $-EI_{RC_y}$ are computed on the Pareto front, then a point $\mathbf{x}_0^1$ is randomly chosen from the EI Pareto front and added. $y(\mathbf{x}_0^1)$ is then considered known and is assumed to be equal to $\min y(\mathbb{X}^*)$. Another computation of $-EI_y$ and $-EI_{RC_y}$ provides one more point based on the same strategy up to the q requested points.

   The problem with this group of strategies is that the kriging variance is not taken into account during the multi-objective optimization. Some interesting areas can be missed because the methods will always add points in the same place except for the MyAlea strategy. The second approach solves this issue by conducting the MO optimization directly on the EI.

5.3 Multi-objective optimization on the expected improvement criterion

In the second group of strategies, the multi-objective optimization is performed on the EI of the output and of the robustness criterion. This approach takes into account the kriging variance

1. $\lfloor . \rfloor$ is the floor function

Method	Minimization	Interesting points	Updates
MyAlea	y, RC_y	Random points on the Pareto front and the parameters space	Batch
MyEIClust	y, RC_y	Cluster on EIy and EIRCy	Batch
MyqEI	y, RC_y	annealing algorithm on qEIy and qEIRCy	Batch
MyKB	y, RC_y	Kriging believer	Batch
MyCL	y, RC_y	Constant liar	Batch
MEIyAlea	EI_y, EI_{RC_y}	Random point on the Pareto front	Seq
MqEIyAlea	qEI_y, qEI_{RC_y}	Random point on the Pareto front	Batch

TABLE 3.1 – Minimization problems and methods to choose the interesting points.

from the beginning of the procedure. The multi-objective problem that is computed in **Step 4** is the following :

Find the Pareto set $\mathbb{X}_0$, solution of the following multi-objective optimization

$$\min_{\mathbf{x}\in D}\{-EI_y(\mathbf{x}), -EI_{RC_y(\mathbf{x})}\}$$

The Pareto front is found by the NSGA II algorithm. For this approach, one enrichment strategy is proposed to add one point in **Step 5** and is the following :

6. MEIyAlea : a point is randomly chosen and sequentially added until the total budget is reached.

This strategy add point sequentially ($q = 1$) not anymore by batch ($q > 1$).

The last group is introduced to overcome this remark. The qEI is used instead of the EI to measure the improvement of a batch of points instead of one point.

5.4 Multi-objective optimization on the multi-point expected improvement criterion

In order to take into account the kriging variance in the optimization and to add points by batch, we modify the objectives. The multi-objective optimization is performed on the qEI of the outputs kriging prediction and of the robustness criterion. **Step 4** searches for non-dominated points of the following problem :

Find the Pareto set $\mathbb{X}_0$, solution of the following multi-objective optimization

$$\min_{\mathbf{x}\in D\subset\mathbb{R}^{p\times q}}\{-qEI_y(\mathbf{x}), -qEI_{RC_y}(\mathbf{x})\}$$

This optimization scheme is of size $p \times q$, which limits its use. One enrichment approach has been benchmarked and is described below to add q points in **Step 5**.

7. MqEIyAlea : for each Pareto front from **Step 4**, one point is randomly extracted from the qEI space, this point will provide q points in the parameter space for the next optimization step.

The seven methods to choose interesting points in **Step 5** are summarized in Table 3.1.

6 Applications

In order to compare the seven strategies, several measures exist to quantify the quality of a Pareto front (cf [Van Veldhuizen, 1999], [Schott, 1995], [Deb et al., 2002] and [Zitzler and Thiele, 1999]).

We decide to focus on two of them that are the most popular. Let $\mathbf{f} = (f_1, \ldots, f_m)$ be the objective functions, $\mathcal{P}$ the theoretical Pareto front and $\mathbb{X}^*$ the empirical Pareto front where $N = \#\mathcal{P}$. The chosen performance metrics are :

- Inverted Generational distance (IGD) see [Coello and Sierra, 2004] :

$$IGD(\mathbb{X}^*) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} d_i^2}$$

  where $d_i = \min_{\mathbf{x}\in\mathbf{X}^*}(\|\mathbf{f}(\mathbf{x}^i) - \mathbf{f}(\mathbf{x})\|_2)$, $\mathbf{f}(\mathbf{x}^i) \in \mathcal{P}$. This metric evaluates the distance between the empirical and the theoretical Pareto front. A small value is better.
- Hypervolume (HV) see [Zitzler and Thiele, 1999]. The Figure 3.1 shows the Hypervolume (HV) of a Pareto front. [Fonseca et al., 2006] introduce an algorithm to compute this volume. We compare the empirical HV to the theoretical.

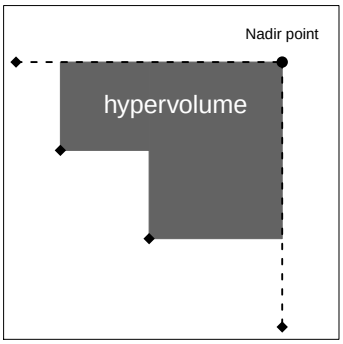

FIGURE 3.1 – Hypervolume : the diamonds represent the individuals of the empirical Pareto front $\mathbb{X}^*$. The black circle is the Nadir point of the set $\mathbb{X}^*$.

This section compares the strategies in two test functions. The first one is the six-hump Camel in two dimensions. We apply the seven strategies in two cases : the observations of the function and the derivatives are available and only the observations of the function are available. The second test function is the Hartmann in six dimensions. We divided the case in the same way as the previous function : observations of the functions and the derivatives then only observations of the function. In the same way as previously we consider that derivatives are affordable on one hand ; on the other hand that only the function is available. For the sake of efficiency only three of the best strategies are applied on Hartmann function.

6.1 Six-hump Camel function : 2D

In this application, we consider the six-hump Camel function. The two input variables are supposed to suffer uncertainties modeled with a Gaussian law with a standard deviation of $\delta_j = \frac{0.05}{4}(\max(x_j) - \min(x_j))$, $j = \{1, 2\}$. Then :

$$(\mathbf{x} + H) \sim \mathcal{N}\left(\mathbf{x}, \begin{pmatrix} \delta_1^2 & 0 \\ 0 & \delta_2^2 \end{pmatrix}\right)$$

The Figure 3.2 shows that the algorithms have to found four areas.
As we aim at performing a robust optimization, the function and all the first and second derivatives are

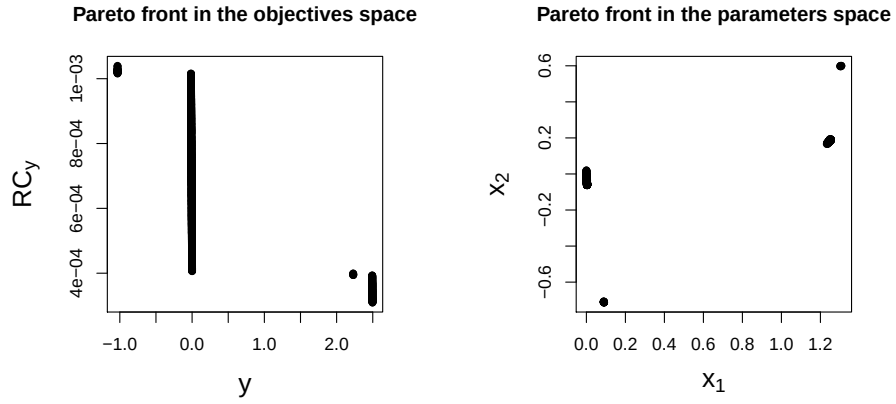

FIGURE 3.2 – Pareto front of the six-hump Camel function in the objectives space (left) and in the parameters space (right)

to be predicted. The set of predicted indexes is $u_{pred} = \{1, \ldots, 6\}$ that corresponds to the processes vector :

$$Z_{u_{pred}} = (Y, Y_{x_1}, Y_{x_2}, Y_{x_1,x_2}, Y_{x_1,x_1}, Y_{x_2,x_2})$$

6.1.1 Derivatives' observations

In this first part of the study we consider that the function and all derivatives are available at each evaluated point. $u_{obs} = \{1, \ldots, 6\}$ that corresponds to the vector of processes :

$$Z_{u_{obs}} = (Y, Y_{x_1}, Y_{x_2}, Y_{x_1,x_2}, Y_{x_1,x_1}, Y_{x_2,x_2})$$

The initial sample set is composed of 5 points. Nine updates of 5 points are added for a total budget of 54 points. The optimization scheme is performed 100 times with different initial learning sets to compare the seven strategies.

Method	Updates	Computation time	Nb areas
MyAlea	Batch	2 min	1.83
MyEIClust	Batch	2 min	2.73
MyqEI	Batch	6 min 30 sec	2.85
MyKB	Batch	3 min	3.77
MyCL	Batch	3 min	3.68
MEIyAlea	Seq	1 h	1.61
MqEIyAlea	Batch	3 h 30 min	3.06

TABLE 3.1 – Summary of the results obtained with the seven strategies on 100 simulation on the six-hump Camel function with derivatives observation. The true number of areas is 4.

Results are provided in Figure 3.3 and Table 3.1. In the table, we compare the methods with the computation time and the number of areas found after 54 evaluations. In the figure the methods are compared through two Pareto front performances metrics.

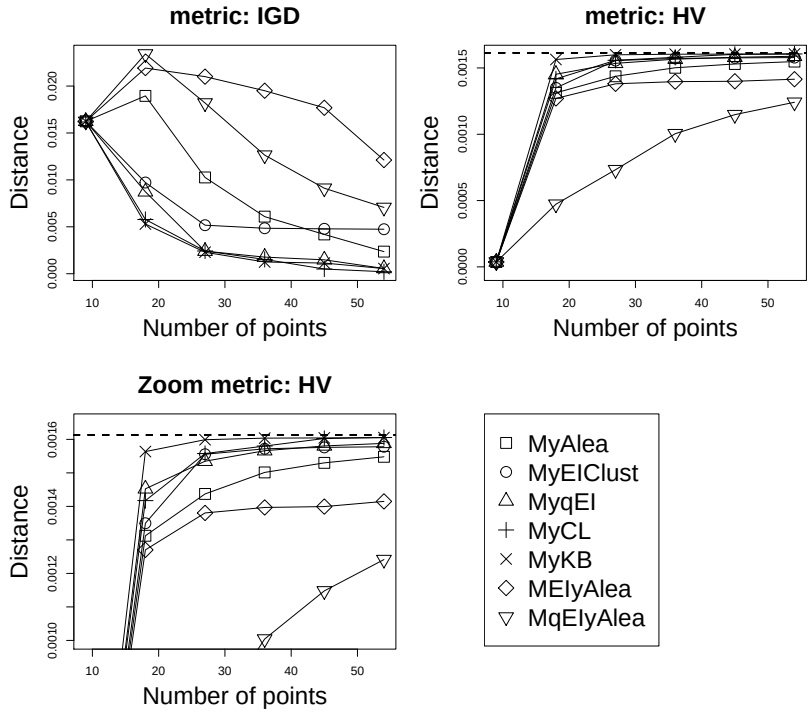

FIGURE 3.3 – Six-hump Camel function with derivatives observations. Evolution of the Pareto metrics with the number of points compute for all the methods over 100 different runs of the algorithm. The HV value of the theoretical front is represented by the dotted line.

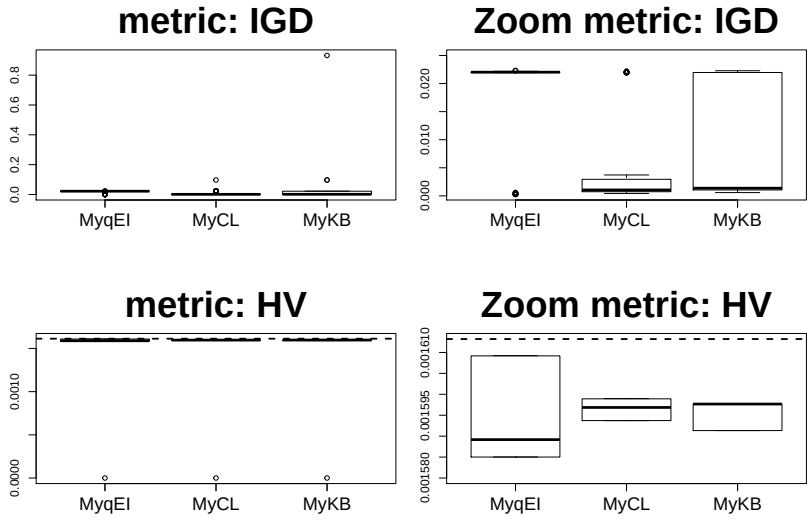

FIGURE 3.4 – Boxplots of the metrics computed for the three best methods over 100 simulations for the six-hump Camel function with derivative observations.

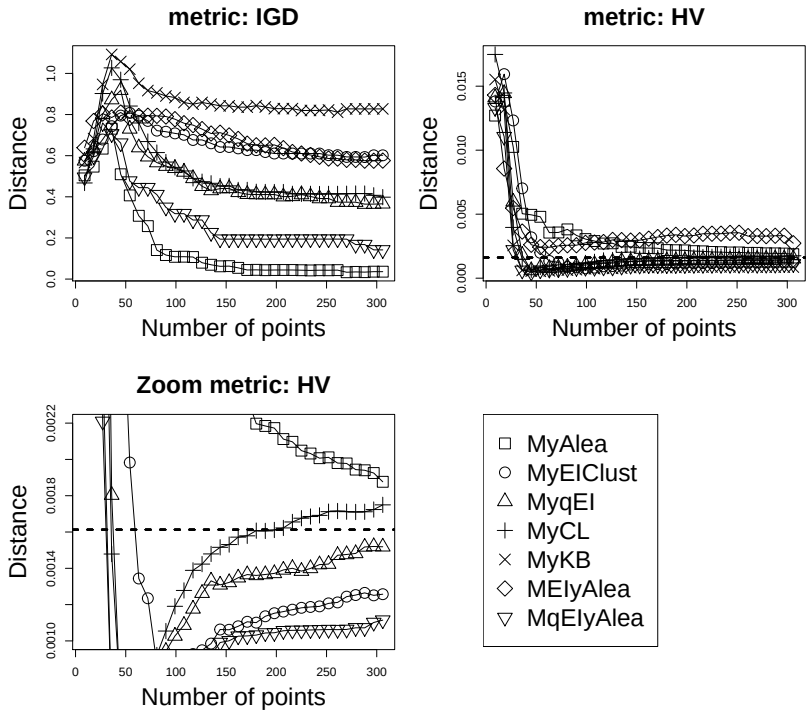

FIGURE 3.5 – Six-hump Camel function without derivatives observations. Evolution of the Pareto metrics with the number of points compute for all the methods over 100 different runs of the algorithm. The HV value of the theoretical front is represented by the dotted line.

Our analysis is as follow : the MyKB and MyCL are the two most efficient strategies in terms of metrics, areas found and computation times. Then MyqEI, MEIClust and MqEIyAlea gives good results for the metrics and the areas. Even if the MyqEI is quite better in metrics and MqEIyAlea in areas. Finally MyAlea and MEIyAlea are the worst efficient methods in areas and metrics. In addition, MEIyAlea and MqEIyAlea are really time consuming. Then, the best methods selected to be used for robust optimization of limited budget application are MyqEI, MyCL and MyKB, which fully exploit batch computation of EI without excessive computational cost. Figure 3.4 shows the boxplots of these three methods for each distance. We can see on this figure that the MyqEI method gives in mean the worst results. It comes from the annealing simulation of the strategy that is difficult to tune.

6.1.2 No derivatives' observations

The aim of this section is to analyze the behavior of the seven strategies when the derivatives observations are not available.
The indexes set is $u_{obs} = \{1\}$ and $u_{pred} = \{1, \ldots, 6\}$ that corresponds to the processes vectors :

$$Z_{u_{obs}} = Y$$
$$Z_{u_{pred}} = (Y, Y_{x_1}, Y_{x_2}, Y_{x_1,x_2}, Y_{x_1,x_1}, Y_{x_2,x_2})$$

The initial sample set is still a space filling of 5 points. Because available information is poorer than in the previous section more points need to be added to allow the detection of the front. That's why 35 updates of 5 points are performed until a total budget of 324 points. The optimization scheme is performed 100 times with different initial learning sets to compare the seven strategies.
Results are provided in Figure 3.5 and Table 3.2. Our analysis is as follow : the six hump Camel function is difficult to approximate without the information on derivatives. The MyAlea strategy that

Method	Updates	Computation time	Nb areas
MyAlea	Batch	18 min	2.98
MyEIClust	Batch	11 min	1.94
MyqEI	Batch	58 min	2.53
MyCL	Batch	15 min	2.58
MyKB	Batch	15 min	1.91
MEIyAlea	Seq	5 h 47 min	1.15
MqEIyAlea	Batch	15h17 min	3.57

TABLE 3.2 – Summary of the results obtained with the seven strategies on 100 simulation on the six-hump Camel function without derivatives observation. The true number of areas is 4.

does not used too much kriging informations to enrich the set gives the best results. In this context, it is a good thing to not entirely trust kriging. The MyqEI and MqEIyAlea strategies provide quite good results because they use the qEI criterion that takes into account the improvement provided by a batch of points of the front. However, MqEIyAlea is too time consuming. The MyCL strategy that does not trust the response surface gives quite good results too, contrary to the MyKB. Finally, the MyEIClust and MEIyAlea stategies that use the EI criterion provide poor results. Even, if the MyEIClust strategy is quite better thanks to the clustering used to enrich the set. The best strategy is MyAlea but we also retain MyqEI and MyCL in order to test them in higher dimension.

6.2 Hartmann function : 6D

In this section, we benchmark the three best strategies identified in Section 6.1.1 in higher dimension (six). The kriging model uses the anisotropic kernel with the Matern5_2 covariance function. The function studied is the Hartmann six-dimensional defined by :

$$f(\mathbf{x}) = -\sum_{i=1}^{4} \alpha_i exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - P_{ij})^2\right) x_1^2, \mathbf{x} \in [0;1]^2$$

with $\alpha = (1, 1.2, 3, 3.2)'$,

$$A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 18 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}$$

and

$$P = 10^{-4} \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}$$

We consider that the random variables are $\mathbf{x}_4$ and $\mathbf{x}_5$ and follow a Gaussian law with a standard deviation of $\delta_j = \frac{0.05}{4}(\max(x_j) - \min(x_j))$, $j = \{4, 5\}$.
We consider two cases : in the first one we have access to the observations of the function and the derivatives associated to the perturbations variables and in the second one we have only access to the observations of the function.

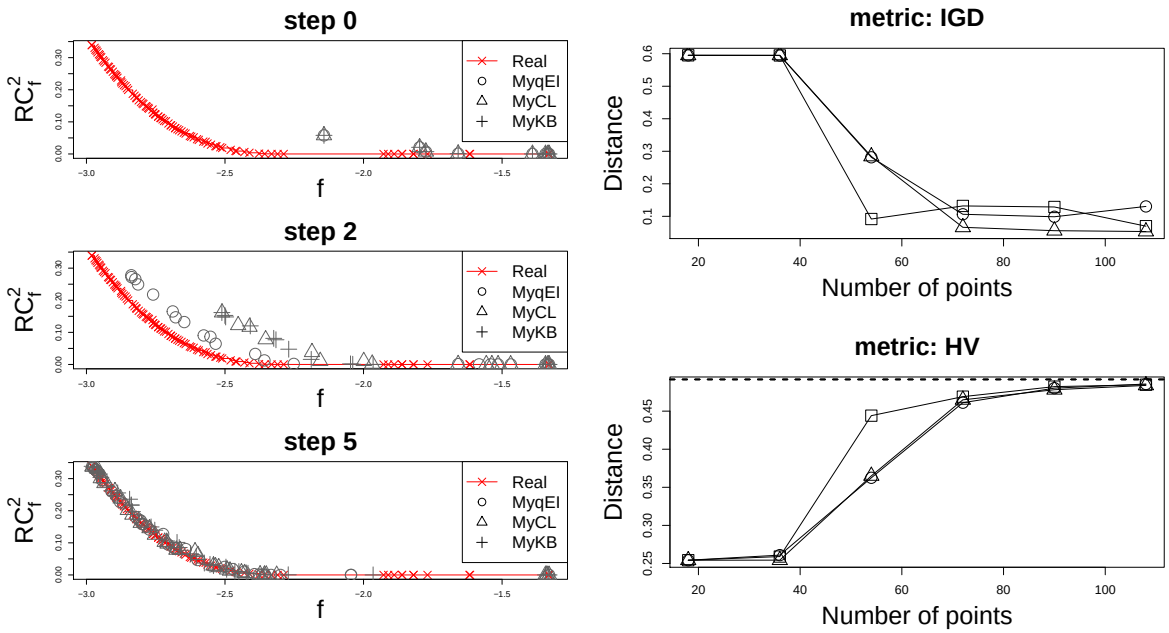

FIGURE 3.6 – On the left : Pareto fronts obtained during the optimization procedure of the three strategies at initial step (step 0), middle step (step 2) and final step (step 5). On the right : Evolution of the metrics computed during the algorithm for all the methods over 100 simulations for the Hartmann function with derivatives observations. The HV value of the theoretical front is represented by the dotted line.

6.2.1 Derivatives' observations

The indexes set is $u_{obs} = u_{pred} = \{1, 5, 6, 20, 26, 27\}$ that corresponds to the vector of processes :

$$Z_{u_{obs}} = Z_{u_{pred}} = (Y, Y_{x_4}, Y_{x_5}, Yx_4, x_5, Y_{x_4,x_4}, Y_{x_5,x_5})$$

The initial sample set is composed of 18 points. Five updates are made and 18 points are added by update for a total budget of 108 points. We apply the best methods found in the previous test case with derivatives informations : MyqEI, MyCL and MyKB strategies.

The left part of Figure 3.6 shows that the three methods converge to the real front. At step 2, MyqEI gives the more advanced front. At final step the three methods give the same good results (see the right part of Figure 3.6). MyKB and MyCL take 10 min for the five steps when MyqEI takes 12 minutes.

6.2.2 No derivatives' observations

The indexes set is $u_{obs} = \{1\}$ and $u_{pred} = \{1, 5, 6, 20, 26, 27\}$ that corresponds to the processes vector

$$\begin{aligned} Z_{u_{obs}} &= Y \\ Z_{u_{pred}} &= (Y, Y_{x_4}, Y_{x_5}, Yx_4, x_5, Y_{x_4,x_4}, Y_{x_5,x_5}) \end{aligned}$$

Like in previous section the initial design is composed of 18 points. In the same way as for the six-hump Camel more updates are added when derivatives are not affordable. Here 35 updates of 18 points are sequentially computed until a total budget of 648 points. We apply the best methods identified in Section 6.1.2 : MyAlea, MyqEI and MyCL strategies.

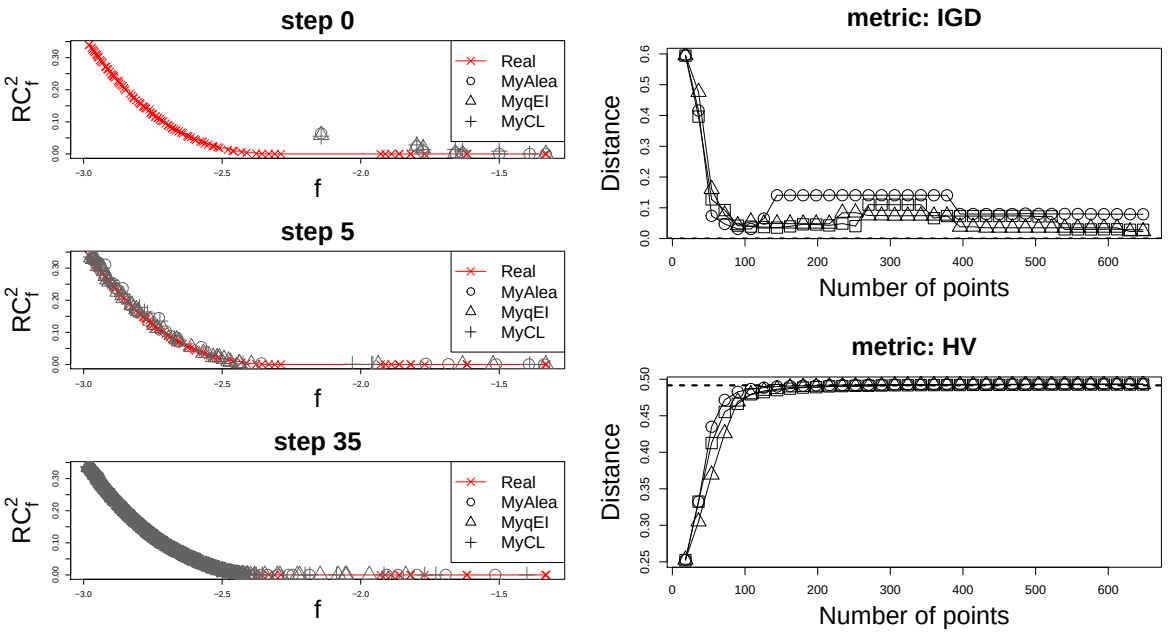

FIGURE 3.7 – On the left : Pareto fronts obtained during the optimization procedure of the three strategies at initial step (step 0), step 5 and final step (step 35). On the right : Evolution of the metrics during the algorithm compute for all the methods in 100 simulations for the six-hump Camel function with no derivatives observation. The HV value of the theoretical front is represented by the dotted line.

The left part of Figure 3.7 shows that the three methods converge to the true front. At step 5, all methods have almost found the entire front. The bottom part of the front is difficult to localize even with 578 additional points. The right part of Figure 3.7 shows that the distance starts to converge to the expected value in the 100 first points. Since then the distances are a little bit unstable. For the IGD metric, the values are subject to little perturbations around the expected value zero, that is a good thing. For the HV measure, the three methods have converged to the theoretical value with only 100 points that correspond to 6 updates. MyAlea takes 1h15min, MyqEI takes 1h40min and MyCL takes 1h04min for the 35 steps.

7 Conclusion

In this article, we propose a robust optimization procedure based on the prediction of the function and its derivatives by a co-kriging model. First of all, we describe the robustness criterion based on the Taylor theorem. Then, the co-kriging model is introduced. Finaly, the robust optimization is performed on both the function and the robustness criterion. A Pareto front of the robust solutions is generated by a genetic algorithm named NSGA-II. We detail seven strategies and compare them for the same budget in two test functions (2D and 6D). In each case, we compare the results when the derivatives are observed and not.
In conclusion, the results show that the efficiency of the strategies is linked to the regularity of the function. Indeed, if the function is easy, the derivative observations are not essential and the strategies like MyCL and MyqEI are relevant. Functions we find in the real life are often smooth. However, when the function is more complicated like the six-hump Camel a most exploratory strategy like MyAlea is recommended. This strategy is easy to use because the understanding of the complex criterion like EI or qEI is not necessary. The study we propose is a first work that reveal efficient strategies based on kriging prediction rather than EI approaches.

Appendix

3.A Number of points for the estimation of the empirical variance

Let $\mathbf{x} \in D \subset \mathbb{R}^p$, an observation point. Let $\mathbf{H} \sim \mathcal{N}(0_{\mathbb{R}^p}, \Delta^2)$ be the random variable such as $\mathbf{x} + \mathbf{H} \sim \mathcal{N}(\mathbf{x}, \Delta^2)$ where Δ^2 is defined by :

$$\Delta^2 = \begin{pmatrix} \delta_1^2 & 0 & \dots & 0 \\ 0 & \delta_2^2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \delta_p^2 \end{pmatrix}$$

Let

$$\begin{array}{rccc} f: & \mathbb{R}^p & \longrightarrow & [a;b] \\ & \mathbf{x} & \longmapsto & f(\mathbf{x}) \end{array}$$

be a 2 times differentiable bounded function, where $a \in \mathbb{R}$ and $b \in \mathbb{R}$. Then all the moments of $f(\mathbf{x} + \mathbf{H})$ exist. Let $\mu = \mathbb{E}(f(\mathbf{x} + \mathbf{H}))$ and $\bar{f} = \frac{1}{n}\sum_{i=1}^n f(\mathbf{x} + \mathbf{h}^i)$ be the empirical estimator of μ. Let $v_f = Var(f(\mathbf{x}+\mathbf{H}))$ and $S^2 = \frac{1}{n}\sum_{i=1}^n (f(\mathbf{x}+\mathbf{h}^i) - \bar{f}(\mathbf{x}+\mathbf{H}))^2$ the empirical estimator of σ^2. Where $\mathbf{h}^1, \dots, \mathbf{h}^n$ are n realizations of the random variable $\mathbf{X}$. The aim of this section is to find the asymptotic law of the empirical estimator S^2.
We recall that

$$\mathbb{E}(\bar{f}) = \mu$$
$$\mathbb{E}(S^2) = \frac{n-1}{n} v_f$$

In order to be as intelligible as possible $f(\mathbf{x} + \mathbf{h}^i)$ is written f_i.

We can notice that

$$S^2 = \frac{1}{n}\sum_{i=1}^n (f_i - \mu) - (\bar{f} - \mu)^2 \tag{3.13}$$

Consider the vector $(f_i - \mu)^2$, $1 \leq i \leq n$. It is a vector of iid random variables. Moreover $\mathbb{E}\left[(f_i - \mu)^2\right] = v_f$, $Var\left[(f_i - \mu)^2\right] = \mathbb{E}\left[(f - \mu)^4\right] - v_f^2 = \mu_4 - v_f^2$. Then the central limit theorem (CLT) implies :

$$\sqrt{n}\left(\frac{1}{n}\sum_{i=1}^n (f_i - \mu)^2 - v_f]\right) \to \mathcal{N}(0, \mu_4 - v_f^2)$$

With Equation (3.13), we obtain that :

$$\sqrt{n}(S^2 - v_f) = \sqrt{n}\left(\frac{1}{n}\sum_{i=1}^n (f_i - \mu) - v_f\right) - \sqrt{n}(\bar{f} - \mu)^2$$

The CLT applied to $\bar{f}$ gives :

$$\sqrt{n}\left(\bar{f} - \mu\right) \to \mathcal{N}(0, v_f)$$

Law of large number, $\bar{f} \to^{p.s} \mu \Rightarrow \bar{f} \to^{\mathbb{P}} \mu$ then $\sqrt{n}\left(\bar{f}-\mu\right)^2 \to^{\mathbb{P}} 0$ and $\sqrt{n}\left(\frac{1}{n}\sum_{i=1}^{n}(f_i-\mu)^2 - v_f\right) \to \mathcal{N}(0, \mu_4 - v_f^2)$ then by the theorem of Slutski $\sqrt{n}\left(S^2 - v_f\right) \to \mathcal{N}(0, \mu_4 - v_f^2) - 0$.
We obtain :

$$\sqrt{(n)}(S^2 - v_f) \longrightarrow^{\mathcal{L}} \mathcal{N}(0, \mu_4 - v_f^2)$$

Asymptotically

$$\frac{S^2 - v_f}{\sqrt{(\mu_4 - v_f^2)/n}} \sim \mathcal{N}(0,1)$$

Let z the quantile of the standard normal distribution of a risk α, then :

$$\mathbb{P}\left(\left|\frac{S^2 - v_f}{\sqrt{(\mu_4 - v_f)^2/n}}\right| \leq z\right) = 1 - \alpha$$

The empirical estimator $\hat{\mu}_4 = \frac{1}{n}\sum_{i=1}^{n}(f_i - \bar{f})^4$ of μ_4 is plugged in the equation :

$$\mathbb{P}\left(\left|\frac{S^2 - v_f}{\sqrt{(\hat{\mu}_4 - v_f^2)/n}}\right| \leq z\right) = 1 - \alpha$$

$$\begin{aligned}
\left|\frac{S^2 - v_f}{\sqrt{(\hat{\mu}_4 - v_f^2)/n}}\right| \leq z &\Leftrightarrow \frac{(S^2 - v_f)^2}{(\hat{\mu}_4 - v_f^2)/n} \leq z \\
&\Leftrightarrow ((S^2)^2 - 2S^2 v_f + v_f^2) - \frac{z}{n}(\hat{\mu}_4 - v_f^2) \leq 0 \\
&\Leftrightarrow \left(1 + \frac{z^2}{n}\right) v_f^2 - 2S^2 v_f + \left((S^2)^2 - \frac{z^2\hat{\mu}_4}{n}\right) \leq 0
\end{aligned}$$

$$\begin{aligned}
\Delta &= (-2S^2)^2 - 4\left(1 + \frac{z^2}{n}\right)\left(S^4 - \frac{z^2\hat{\mu}_4}{n}\right) \\
&= 4(S^2)^2 - 4(S^2)^2 + \frac{4\hat{\mu}_4 z^2}{n} - \frac{4z^2(S^2)^2}{n} + \frac{4z^4\hat{\mu}_4}{n^2} \\
&= \frac{4z^2}{n}\left(\hat{\mu}_4\left(1 + \frac{z^2}{n}\right) - (S^2)^2\right)
\end{aligned}$$

$\Delta > 0$ if $\hat{\mu}_4\left(1 + \frac{z^2}{n}\right) > (S^2)^2$.
The square function is convex, $\left((f_1 - \bar{f})^2, \ldots, (f_n - \bar{f})^2\right)$ is a real n-uplet and $\sum_{i=1}^{n}\frac{1}{n} = 1$. Thanks to the Jensen Inequality (convexity) :

$$\frac{1}{n}\sum_{i=1}^{n}(f_i - \bar{f})^4 \geq \left(\frac{1}{n}\sum_{i=1}^{n}(f_i - \bar{f})^2\right)^2 \Leftrightarrow \hat{\mu}_4 \geq (S^2)^2$$

Then $\Delta > 0$ and

$$v_f \in \left[\frac{2S^2 - \sqrt{\Delta}}{2\left(1 + \frac{z^2}{n}\right)}; \frac{2S^2 + \sqrt{\Delta}}{2\left(1 + \frac{z^2}{n}\right)}\right]$$

$$\frac{2S^2 - \sqrt{\Delta}}{2\left(1+\frac{z^2}{n}\right)} = \frac{2S^2 - \sqrt{\frac{4z^2}{n}\left(\hat{\mu}_4\left(1+\frac{z^2}{n}\right) - (S^2)^2\right)}}{2\left(1+\frac{z^2}{n}\right)}$$
$$= S^2 - \frac{z}{\sqrt{n}}\sqrt{\hat{\mu}_4 - (S^2)^2} + o\left(\frac{1}{n}\right)$$

$$\frac{2S^2 + \sqrt{\Delta}}{2\left(1+\frac{z^2}{n}\right)} = S^2 + \frac{z}{\sqrt{n}}\sqrt{\hat{\mu}_4 - (S^2)^2} + o\left(\frac{1}{n}\right)$$

Then approximatively,

$$v_f \in \left[S^2 - \frac{z}{\sqrt{n}}\sqrt{\hat{\mu}_4 - (S^2)^2}; S^2 + \frac{z}{\sqrt{n}}\sqrt{\hat{\mu}_4 - (S^2)^2}\right]$$

In order to obtain an approximation error lower or equal to ϵ, we choose :

$$n > \frac{z^2}{\epsilon^2}(\hat{\mu}_4 - (S^2)^2)$$

3.B Taylor

Proposition 5. *Let* $\mathbf{H} \sim \mathcal{N}(0_{\mathbb{R}^d}, \Delta^2)$ *where* $\Delta^2 = diag\{\delta_1, \ldots, \delta_p\} \in \mathcal{M}_{p\times p}$ *and a function* $f : D \subset \mathbb{R}^p \to \mathbb{R}$ *to times differentiable.* $\tilde{f}$ *is the Taylor approximation trunked at the order two then :*

$$Var\left(\tilde{f}(x+H)\right) = tr\left(\nabla_f \nabla_f' \Delta^2\right) + \frac{1}{2} tr\left(\mathbb{H}_f^2(\delta_1^2, \ldots, \delta_p^2)(\delta_1^2, \ldots, \delta_p^2)'\right)$$

where $\nabla_f \in \mathbb{R}^p$ *the vector of the gradient of* f *and* $\mathbb{H}_f \in \mathcal{M}_{p,p}$ *the Hessian matrix and* tr *the matrix trace.*

Démonstration. The Taylor approximation trunked at the order two is :

$$\tilde{f}(\mathbf{x}+H) = f(\mathbf{x}) + \nabla_f'(\mathbf{x})H + \frac{1}{2}H'\mathbb{H}_f(\mathbf{x})H$$

and

$$\begin{aligned}
Var\left(\tilde{f}(\mathbf{x}+\mathbf{H})\right) &= Var\left(f(\mathbf{x}) + \nabla_f'\mathbf{H} + \frac{1}{2}\mathbf{H}'\mathbb{H}_f\mathbf{H}\right) \\
&= Var\left(\nabla_f'\mathbf{H} + \frac{1}{2}\mathbf{H}'\mathbb{H}_f\mathbf{H}\right) \\
&= \mathbb{E}\left(\left(\nabla_f'\mathbf{H} + \frac{1}{2}\mathbf{H}'\mathbb{H}_f\mathbf{H}\right)^2\right) - \left[\mathbb{E}\left(\nabla_f'\mathbf{H} + \frac{1}{2}\mathbf{H}'\mathbb{H}_f\mathbf{H}\right)\right]^2 \\
&= \mathbb{E}\left((\nabla_f'\mathbf{H})^2\right) + \mathbb{E}\left(\nabla_f'\mathbf{H}\mathbf{H}'\mathbb{H}_f\mathbf{H}\right) + \frac{1}{4}\mathbb{E}\left((\mathbf{H}'\mathbb{H}_f\mathbf{H})^2\right) \\
&\quad - \left[\mathbb{E}\left(\nabla_f'\mathbf{H}\right)\right]^2 - \mathbb{E}\left(\mathbf{H}'\mathbb{H}_f\mathbf{H}\right)\mathbb{E}\left(\nabla_f'\mathbf{H}\right) - \frac{1}{4}\left[\mathbb{E}\left(\mathbf{H}'\mathbb{H}_f\mathbf{H}\right)\right]^2
\end{aligned}$$

Let's calculate each terms

1. $\mathbb{E}\left[\nabla_f'\mathbf{H}\right] = \sum_i (\nabla_f)_i \mathbb{E}\left[\mathbf{h}_i\right] = 0$
2. $\mathbb{E}\left[\mathbf{H}'\mathbb{H}_f\mathbf{H}\right] = \sum_i \sum_j (\mathbb{H}_f)_{i,j}\mathbb{E}\left[\mathbf{h}_i\mathbf{h}_j\right] = \sum_i (\mathbb{H}_f)_{i,i}\delta_i^2 = tr\left(\mathbb{H}_f\Delta^2\right)$

3. $\mathbb{E}\left[\left(\nabla_f' H\right)^2\right] = \sum_i \sum_j (\nabla_f)_i (\nabla_f)_j \mathbb{E}\left[\mathbf{h}_i \mathbf{h}_j\right] = \sum_i (\nabla_f)_i (\nabla_f)_i \delta_i^2 = tr\left(\nabla_f \nabla_f' \Delta^2\right)$
4. $\mathbb{E}\left[\left((\nabla_f' H)' H' \mathbb{H}_f H\right)^2\right] = \sum_i \sum_j \sum_k (\nabla_f)_i (\mathbb{H}_f)_{jk} \mathbb{E}\left[\mathbf{h}_i \mathbf{h}_j \mathbf{h}_k\right] = \sum_i (\nabla_f)_i (\mathbb{H}_f)_{ii} \mathbb{E}\left[\mathbf{h}_i^3\right] = 0$
5.

$$\begin{aligned}
\mathbb{E}\left[(H' \mathbb{H}_f H)^2\right] &= \sum_i \sum_j \sum_k \sum_l (\mathbb{H}_f)_{ij} (\mathbb{H}_f)_{kl} \mathbb{E}\left[\mathbf{h}_i \mathbf{h}_j \mathbf{h}_k \mathbf{h}_l\right] \\
&= \sum_i \sum_{k \neq i} (\mathbb{H}_f)_{ii} (\mathbb{H}_f)_{kk} \mathbb{E}\left[\mathbf{h}_i^2 \mathbf{h}_k^2\right] + 2 \sum_i \sum_{j \neq i} (\mathbb{H}_f)_{ij} (\mathbb{H}_f)_{ij} \mathbb{E}\left[\mathbf{h}_i^2 \mathbf{h}_j^2\right] + \sum_i (\mathbb{H}_f)_{ii}^2 \mathbb{E}\left[\mathbf{h}_i^4\right] \\
&= \sum_i \sum_{k \neq i} (\mathbb{H}_f)_{ii} (\mathbb{H}_f)_{kk} \delta_i^2 \delta_k^2 + 2 \sum_i \sum_{j \neq i} (\mathbb{H}_f)_{ij} (\mathbb{H}_f)_{ij} \delta_i^2 \delta_j^2 + 3 \sum_i (\mathbb{H}_f)_{ii}^2 \delta_i^2 \\
&= \sum_i \sum_k (\mathbb{H}_f)_{ii} (\mathbb{H}_f)_{kk} \delta_i^2 \delta_k^2 - \sum_i (\mathbb{H}_f)_{ii}^2 \delta_i^2 + 2 \sum_i \sum_j (\mathbb{H}_f)_{ij} (\mathbb{H}_f)_{ij} \delta_i^2 \delta_j^2 \\
&- 2 \sum_i (\mathbb{H}_f)_{ii}^2 \delta_i^2 + 3 \sum_i (\mathbb{H}_f)_{ii}^2 \delta_i^2 \\
&= tr\left(\mathbb{H}_f \Delta^2\right)^2 + 2tr\left(\mathbb{H}_f^{\cdot 2} (\delta_1^2, \ldots, \delta_p^2)(\delta_1^2, \ldots, \delta_p^2)'\right)
\end{aligned}$$

Finally

$$\begin{aligned}
Var\left(\tilde{f}(\mathbf{x} + H)\right) &= tr\left(\nabla_f \nabla_f' \Delta^2\right) + 0 + \frac{1}{4} tr\left(\mathbb{H}_f \Delta^2\right)^2 - 0 - tr\left(\mathbb{H}_f \Delta^2\right) \times 0 - \frac{1}{4} tr\left(\mathbb{H}_f \Delta^2\right)^2 \\
&- \frac{2}{4} tr\left(\mathbb{H}_f^{\cdot 2} (\delta_1^2, \ldots, \delta_p^2)(\delta_1^2, \ldots, \delta_p^2)'\right) \\
&= tr\left(\nabla_f \nabla_f' \Delta^2\right) + \frac{1}{2} tr\left(\mathbb{H}_f^{\cdot 2} (\delta_1^2, \ldots, \delta_p^2)(\delta_1^2, \ldots, \delta_p^2)'\right)
\end{aligned}$$

□

Conclusion du chapitre

Les stratégies développées dans ce chapitre proposent une solution à l'optimisation robuste dans un contexte où les simulations sont coûteuses. L'optimisation robuste est traitée de façon multi-objectif avec la minimisation conjointe de la fonction et du critère de robustesse. Ce critère est une approximation de la variance locale de la fonction. L'accès aux dérivées permet de construire un critère de robustesse peu coûteux basé sur le développement de Taylor.

Différentes stratégies basées sur le métamodèle de krigeage avec dérivées ont été développées. Elles consistent à minimiser trois sous problèmes d'optimisation multi-objectif à base de krigeage. Le premier problème consiste à remplacer la fonction et le critère de robustesse par la prédiction par krigeage de la fonction et du critère (minimisation de y et RC_y). Le second problème maximise les améliorations attendues (minimisation de $-EI_y$ et $-EI_{RC_y}$). Le dernier problème consiste à maximiser l'EI multi-points (minimisation de $-qEI_y$ et $-qEI_{RC_y}$). A la fin de la minimisation, il faut sélectionner plusieurs points du front de Pareto pour enrichir le metamodèle. Pour le premier problème d'optimisation, cinq stratégies sont développées : sélection aléatoire de points (MyALea), clustering k-means (MyEIClust), algorithme de recuit simulé (MyqEI), Kriging Believer (MyKB) et Constant Liar (MyCL). Pour les deuxième et troisième problèmes une seule stratégie consistant à sélectionner des points de façon aléatoire a été mise en place (MEIyAlea et MqEIyAlea). Les sept stratégies sont résumées dans le Tableau 3.1.

Ces stratégies ont été appliquées sur deux cas test en dimension faible (six-hump Camel) et en dimension plus élevée (Hartmann). Dans ces deux cas, les performances des méthodes ont été comparées lorsque l'accès aux dérivées était possible et lorsque cette information était absente. L'étude sur la fonction six-hump Camel a consiste à réaliser 100 optimisations robustes en considérant la fonction coûteuse. Les variables d'entrée sont considérées comme incertaines et suivent une loi Gaussienne locale centrée aux points d'observations et de variance fixe. Dans le cas avec dérivées, les 100 designs initiaux ont tous 9 points puis 5 mises à jour de 9 points sont effectuées pour atteindre un budget total de 54 points. Il en ressort que les méthodes les plus prometteuses sont : MyqEI, MyKB et MyCL. Dans le cas sans dérivées, les 100 designs initiaux ont tous 9 points puis 35 mises à jour de 9 points sont effectuées pour atteindre un budget total de 324 points. Il en ressort que les méthodes les plus prometteuses sont MyAlea, MyqEI et MyCL. Les trois méthodes les plus prometteuses dans chacun des cas sont ensuite testées sur la fonction Hartmann en dimension six. Seulement deux variables d'entrées sont considérées comme incertaines de loi gaussiennes centrées aux points d'observation et de variance fixe. Dans le premier cas, l'ensemble de départ est de 18 points et 5 mises à jour de 18 points sont effectuées pour atteindre un budget total de 108 points. Uniquement les dérivées des variables incertaines sont accessibles. Les résultats obtenus sont très satisfaisants car le front de Pareto théorique est retrouvé avec les trois méthodes même si MyKB et MyCL sont légèrement meilleures que MyqEI. Dans le second cas, l'ensemble de départ est de 18 points et 35 mises à jour de 18 points sont effectuées pour atteindre un budget total de 648 points. Les trois méthodes testées retrouvent le front de Pareto.

En conclusion, les résultats des méthodes dépendent de la régularité de la fonction. Plus la fonction est complexe plus les méthodes avec des stratégies exploratoire fortes sont conseillées (MyAlea). Au contraire, si la fonction est régulière, comme c'est le cas souvent pour des fonctions physiques, l'accès aux dérivées n'est pas essentielle et des méthodes comme MyqEI ou MyCL sont très efficaces. Cependant, il est possible d'améliorer l'efficacité des méthodes basées sur l'EI (MEIyAlea et MqEIyAlea).

La première remarque porte sur le critère de robustesse. Il peut s'écrire comme une fonction du

gradient et de la matrice hessienne :

$$RC_f(\mathbf{x}) = g\left(f(\mathbf{x}), f_{x_j}(\mathbf{x}), f_{x_j,x_k}(\mathbf{x})\right)$$

Une méthode "plug-in" a été choisie dans ce chapitre :

$$RC_{\widehat{y}}(\mathbf{x}) = g(\widehat{y}(\mathbf{x}), \widehat{y}_{x_j}(\mathbf{x}), \widehat{y}_{x_j,x_k}(\mathbf{x}))$$

Une autre façon de le faire serait d'estimer :

$$\mathbb{E}[RC_y(\mathbf{x})|\mathbf{z}_{u_{obs}}] = \mathbb{E}[g(y(\mathbf{x}), y_{x_j}(\mathbf{x}), y_{x_j,x_k}(\mathbf{x}))|\mathbf{z}_{u_{obs}}]$$

Cependant, il faudrait trouver une forme analytique de cette expression car elle est utilisée lors de l'optimisation avec le NSGA II. La seconde remarque concerne la stratégie peu efficace proposée dans le second groupe. L'idée serait de trouver un autre moyen pour sélectionner les points, cette sélection est pour l'instant aléatoire. Une seconde solution pourrait être d'adapter les algorithmes KB et CL pour que l'ajout de points par batch soit directement pris en compte lors de la formation du front. La dernière remarque porte sur l'algorithme génétique utilisé lors de la troisième stratégie. Il permet de minimiser $-qEI_y - qEI_{RC_y}$. Cette somme n'est pas forcément pertinente car il est possible que les deux termes ne varient pas dans le même espace. Il faudrait construire un critère normalisé.

Chapitre 4

Application au cas industriel des turbomachines : paramétrisations proposées par Valeo et le LMFA

1 Introduction

Le but du projet ANR PEPITO est de construire une turbomachine de forme géométrique optimale pour obtenir le plus grand rendement possible. Dans un premier temps, l'idée est de construire une surface de réponse représentative de la fonction réelle avec un grand nombre de variables d'entrée. En effet, Valeo a commencé par utiliser un code prenant en entrée 11 paramètres puis l'a raffiné en ajoutant 4 paramètres. L'étude effectuée dans ce chapitre porte sur ce plan à 15 paramètres. L'idée est ensuite de travailler sur des codes avec 30 puis 60 variables d'entrées. Tout d'abord, le métamodèle de krigeage est comparé à d'autres modèles puis sa stabilité est testée à l'aide de quatre études préliminaires. Cependant, le modèle de krigeage classique comporte beaucoup de paramètres à estimer. Ce nombre de paramètres peut provoquer une instabilité lors de l'estimation de la vraisemblance et une perte de qualité de prédiction. C'est pourquoi, les méthodes classiques de réductions de modèle ou de sélection de variables sont appliquées. La méthode sur les groupes de portée développée dans le Chapitre 2 a été appliquée pour tester l'habileté de cette méthode à réduire le nombre de paramètres de portée estimés sans perdre en qualité de prédiction. Dans un dernier temps, l'objectif est, à débit fixé, de mettre en place une démarche itérative d'optimisation : ajout de quelques simulations pour trouver la géométrie optimale robuste. L'algorithme d'optimisation robuste est appliqué sur le code 1D fourni par le LMFA. L'intérêt de ce code par rapport au code 3D est la rapidité d'exécution et le fait qu'il puisse être installé directement sur les machines de l'ICJ. Dans ce travail de thèse deux aspects sont intéressant : d'une part, la construction d'une surface de réponse pertinente et d'autre part, l'optimisation robuste de la turbomachine. Pour ce faire, les résultats des simulations numériques de Valeo et du LMFA sont utilisés. Les codes sont présentés ci-dessous.

1.1 Description du code 3D (Valeo)

Le simulateur de Valeo est construit à partir d'une paramétrisation 3D de la turbomachine. La paramétrisation de la lame prise sur une section est dessinée à gauche sur la Figure 4.1. Valeo a fait 5 sections uniformément réparties sur la longueur de la lame. Pour chaque section, les variables étudiées sont le calage (stagger), la hauteur maximale de la cambrure (Max camber height), la taille de la corde (Chord length) et le décalage des sections dans la direction azimutale (sweep). La figure de droite montre une vue de face de la machine tournante ainsi que les cinq sections. Les paramètres sélectionnés pour l'étude grâce aux connaissances métier sont : la longueur de corde aux sections $1, 2, 4, 5$, le calage

aux sections $1, 2, 4, 5$, la hauteur maximale de la cambrure aux section $1, 2$ et le sweep aux sections $3, 5$. Les deux paramètres restants sont la distance à la back-plate et le rayon. Le modèle mécanique de Valeo contient en résumé 3 objectifs (sorties) : le rendement, le couple et le différentiel de pression et 15 variables d'entrées : le débit et les 14 variables géométriques. Les variables sont résumées dans la Table 4.1. Le rendement η est une fonction du différentiel (en amont et en aval du ventilateur) de pression ΔP (Pa) et du couple C (N/m) fourni par le moteur électrique :

$$\eta = \frac{\Delta P Q}{C\Omega}$$

où Q (m^3/h) représente le débit d'air et Ω (rpm) la vitesse de rotation. Une fois la paramétrisation effectuée et le maillage 3D choisi, le calcul CFD avec StarCCM+ est lancé, il prend environ dix heures pour une simulation en parallèle sur les machines de Valeo. C'est un code RANS très optimisé.

Entrées

	Noms
x_1	Débit (Q)
x_2, x_3, x_4, x_5	Lcorde 1,2,4,5
x_6, x_7, x_8, x_9	Calage 1,2,4,5
x_{10}, x_{11}	Hmax 1,2
x_{12},x_{13}	Sweep 3,5
x_{14}	Distance_BP
x_{15}	R_BP

Sorties

	Noms
ΔP	DeltaP
C	Couple
η	Rendement

Table 4.1 – Variables d'entrée et de sortie du code 3D de Valeo.

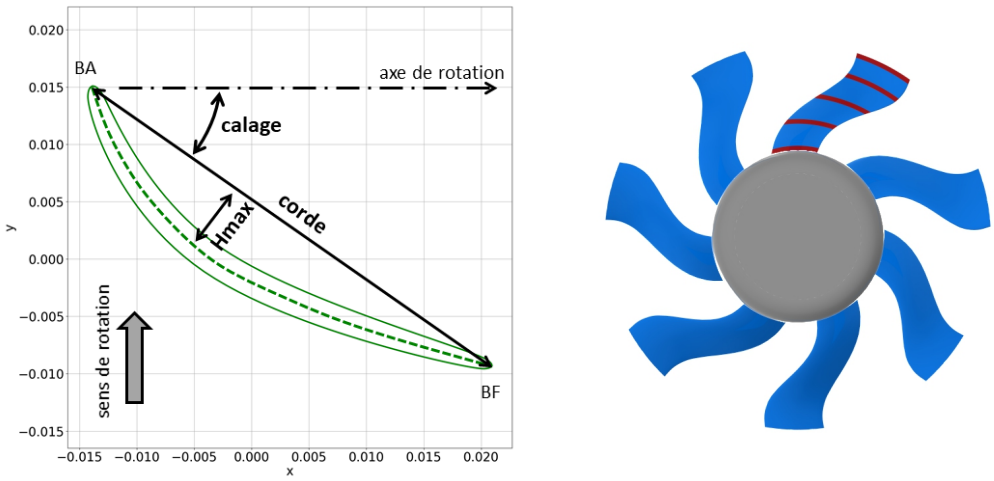

Figure 4.1 – Section d'une lame à gauche et vue de face du ventilateur. Les section sont représentées par les lignes rouge sur la hauteur de la lame. La section 1 correspond à celle la plus proche du disque et la section 5 à celle la plus éloignée.

1.2 Description du code 1D ou TurboConcept (LMFA)

Le simulateur du LMFA est basé sur une paramétrisation 1D (code 1D) de la turbomachine. Ce code permet d'avoir des paramètres comparables à ceux du code 3D de Valeo. Le Tableau 4.2 montre les différents types de paramètres sélectionnés pour le code 1D. La différence avec le code 3D de Valeo

est la façon de calculer les réponses. En effet le domaine de variation des paramètres géométriques dépend des 3 paramètres de point de fonctionnement à fixer en amont (cf Table 4.2). Il est possible de lancer autant de simulations que souhaitées dans le domaine de variation. Cependant, des simulations peuvent être lancées en dehors du domaine. Dans ce cas, la convergence de l'algorithme n'est pas assurée et la sortie peut être erronée ou manquante. Le code 1D permet d'obtenir les simulations des dérivées des sorties par rapport aux différents paramètres géométriques de design et au débit. Le code 1D est très rapide par rapport au code 3D, l'obtention d'une simulation de la fonction et de ses dérivées prend environ trois minutes.

Paramètres de point de fonctionnement

	Noms
Q	Débit
P_e	Pression entrante
Ω	Vitesse de rotation

Entrées

	Noms
x_1, x_2, x_3, x_4, x_5	Lcorde 1,2,3,4,5
$x_6, x_7, x_8, x_9, x_{10}$	Calage 1,2,3,4,5
$x_{11}, x_{12}, x_{13}, x_{14}, x_{15}$	Hmax 1,2,3,4,5

Sorties

	Noms
ΔP	DeltaP
C	Couple
η	Rendement

TABLE 4.2 – Variables de point de fonctionnement, d'entrée et de sortie du code 1D du LMFA.

2 Apport et limites du krigeage sur le code 3D

Plusieurs études préliminaires sont effectuées pour valider l'utilisation du krigeage sur le cas industriel. La première étude consiste à comparer le krigeage aux modèles linéaire et additif généralisé. La seconde permet de valider la stabilité du krigeage (choix du noyau, méthode d'estimation des paramètres et méthode d'optimisation de la vraisemblance). La troisième effectue une analyse de sensibilité de la sortie par rapport aux variables d'entrée et la dernière étudie l'influence des entrées en fonction des paramètres de portées.

2.1 Comparaison des modèles

Le plan d'expérience fourni par Valeo contient 301 simulations. Les 15 variables explicatives sont adimensionnées sur l'intervalle $[-1; 1]$. Le plan est issu d'un OLH Optimal Latin Hypercube (du logiciel Isight DS) sur 300 simulations + 1 (la simulation au centre du domaine). La distribution des points est uniforme sur les marges. Le Tableau 4.2 montre les propriétés de répartition des différents plans. Tout d'abord, le plan de Valeo doit être séparé en un ensemble d'apprentissage et de validation pour comparer les différents modèles. Les 301 simulations sont réparties de façon aléatoire en un échantillon d'apprentissage de taille 250 et un échantillon de validation de taille 51. Le Tableau 4.2

Variable	Débit	Lcorde				Calage				Hmax		Sweep		Distance	R
Section		1	2	4	5	1	2	4	5	1	2	3	5	BP	BP
Notation	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}

TABLE 4.1 – Récapitulatif des variables d'entrées du code 3D, les variables sont adimensionnées sur l'intervalle $[-1; 1]$.

montre que le plan complet à 301 point de Valeo à de très bonnes propriétés de répartition par rapport à des plans uniformes. De plus, les propriétés de répartition restent satisfaisantes en sélectionnant les plans de façon aléatoire par rapport à un plan optimisé à 250 points et sont proches du plan de Valeo. La comparaison des modèles va donc être possible avec cette méthode de construction des plans.

	Mindist	DisC2
Valeo (301)	1.1	0.185
Uniforme $\mathcal{U}\left([-1; 1]^{15}\right)$ (301)	0.567 (0.05)	0.288 (0.012)
OLHS sur $[-1; 1]^{15}$ (250)	0.706	0.215
Aléatoire parmi Valeo (250)	1.12 (0.036)	0.223 (0.0026)

TABLE 4.2 – Critères de qualité du design (cf [Dupuy et al., 2015]) pour différents design. Pour les plans à 301 points : le design complet de Valeo et moyenne (écart type) calculé(e)s sur 100 design uniformes sur $[-1; 1]^{15}$. Pour les plans à 250 points : Un OLHS sur $[-1; 1]^{15}$ et moyenne (écart type) calculé(e)s sur 100 design tirés aléatoirement sur le plan complet de Valeo.

Le modèle de krigeage (km) est comparé aux modèles linéaire (lm) et additif généralisé (gam) décrits dans les livres de [Chambers et al., 1992] et [Friedman et al., 2001]. La fonction de régression du modèle linéaire est obtenue de façon automatique avec un algorithme stepwise. Pour le modèle gam les fonctions de base sont des splines naturelles cubiques. Les interactions auraient pu être prise en compte néanmoins le nombre élevé de variables entrainerait une combinatoire des interactions trop importante. Contrairement au modèle de krigeage qui ne nécessite pas la spécification des effets d'interaction. Ils sont pris en compte automatiquement. Pour le krigeage, la tendance est constante, le noyau *anisotrope* stationnaire avec une fonction de corrélation Matern 5_2. Cent échantillons d'apprentissage sont tirés aléatoirement. Les échantillons de validation sont les points complémentaires à chaque échantillon d'apprentissage. Les trois modèles apprennent sur ces échantillons d'apprentissage et la qualité des modèles est représentée par le Q2 calculé sur les échantillons de test. Le Q2 varie sur $[0; 1]$ (cf Chapitre 1 Section 5.1). Plus il est proche de 1 plus la qualité de prédiction est bonne. Les Figures 4.1 et 4.2 montrent que le meilleur modèle pour les données et quelque soit la variable de réponse est le krigeage. Le rendement est cependant plus difficile à prédire que les autres réponses. La qualité du plan influence les résultats.

2.2 Stabilité du krigeage

L'étude préliminaire suivante consiste à tester la stabilité du krigeage par rapport :

- au choix du noyau de corrélation
- à la méthode d'estimation des paramètres de portée
- au choix de la méthode d'optimisation de la vraisemblance
- à la borne supérieure des paramètres de portée
- à l'initialisation de l'algorithme d'optimisation.

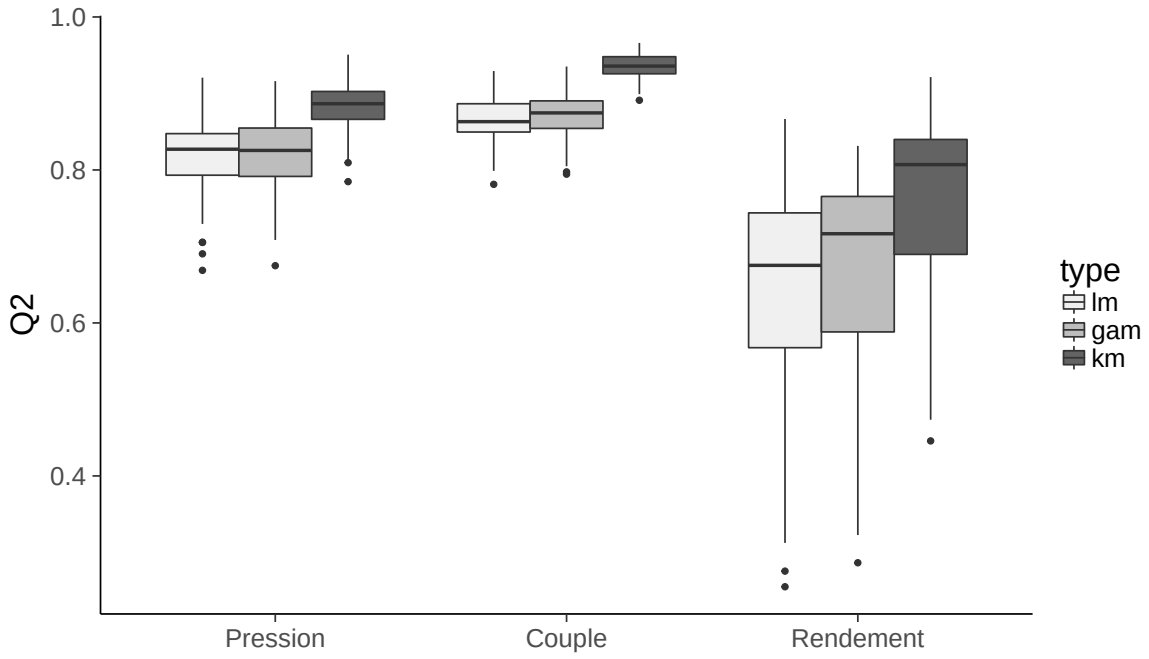

FIGURE 4.1 – Comparaison du Q2 pour les modèles de krigeage (km), linéaire (lm) et additif généralisé (gam) sur les 3 sorties. Le Q2 est calculé sur les cent échantillons de test. Les modèles sont construits sur les cent échantillons d'apprentissage.

Les plans OLH à 301 et 600 simulations fournis par Valeo sont utilisés pour l'étude. Le plan à 301 points est d'abord choisi comme échantillon d'apprentissage et celui à 600 points comme échantillon de test et inversement. Le plan à 600 points contient trois points où le rendement est négatif, la turbomachine fonctionne en mode turbine. Étant donné que ce fonctionnement est très différent de celui étudié, les trois points sont retirés et le plan ne contient plus que 597 points. Le krigeage est choisi avec une tendance constante et un noyau *anisotrope*. Afin de tester la stabilité du krigeage, la fonction de covariance (Matern5_2 ou Matern3_2), la méthode d'estimation des portées (Maximum de vraisemblance (MLE) ou Leave One Out (LOO)), l'algorithme d'optimisation des paramètres (descente de gradient (BFGS) ou génétique (gen)), la borne supérieure des paramètres de portée (défaut ou 3) et la valeur d'initialisation des portées pour l'algorithme BFGS (défaut, 4 ou 1) sont changés à tour de rôle. Le paramètre par défaut de la borne supérieure correspond à deux fois le domaine de variation des variables d'entrées. Ici, les variables sont adimensionnées sur l'intervalle $[-1; 1]$ donc ce paramètre vaut environ 4 pour toutes les variables. Pour la méthode "BFGS", certains points sont générées uniformément dans le domaine de variation des portées, soit $[10^{-10}; 4]$. Ensuite, le meilleur point par rapport au critère de vraisemblance est choisi. La configuration de référence est donnée par la première ligne du Tableau 4.3 et les lignes 2 à 8 montrent les sept autres configurations testées. Le critère de qualité de prédiction Q2 est tracé sur la Figure 4.3. Elle montre que la qualité de prédiction reste très satisfaisante et stable pour les trois réponses et quelque soit la configuration. Sur ce cas industriel, le krigeage est stable par rapport aux différentes configurations testées.

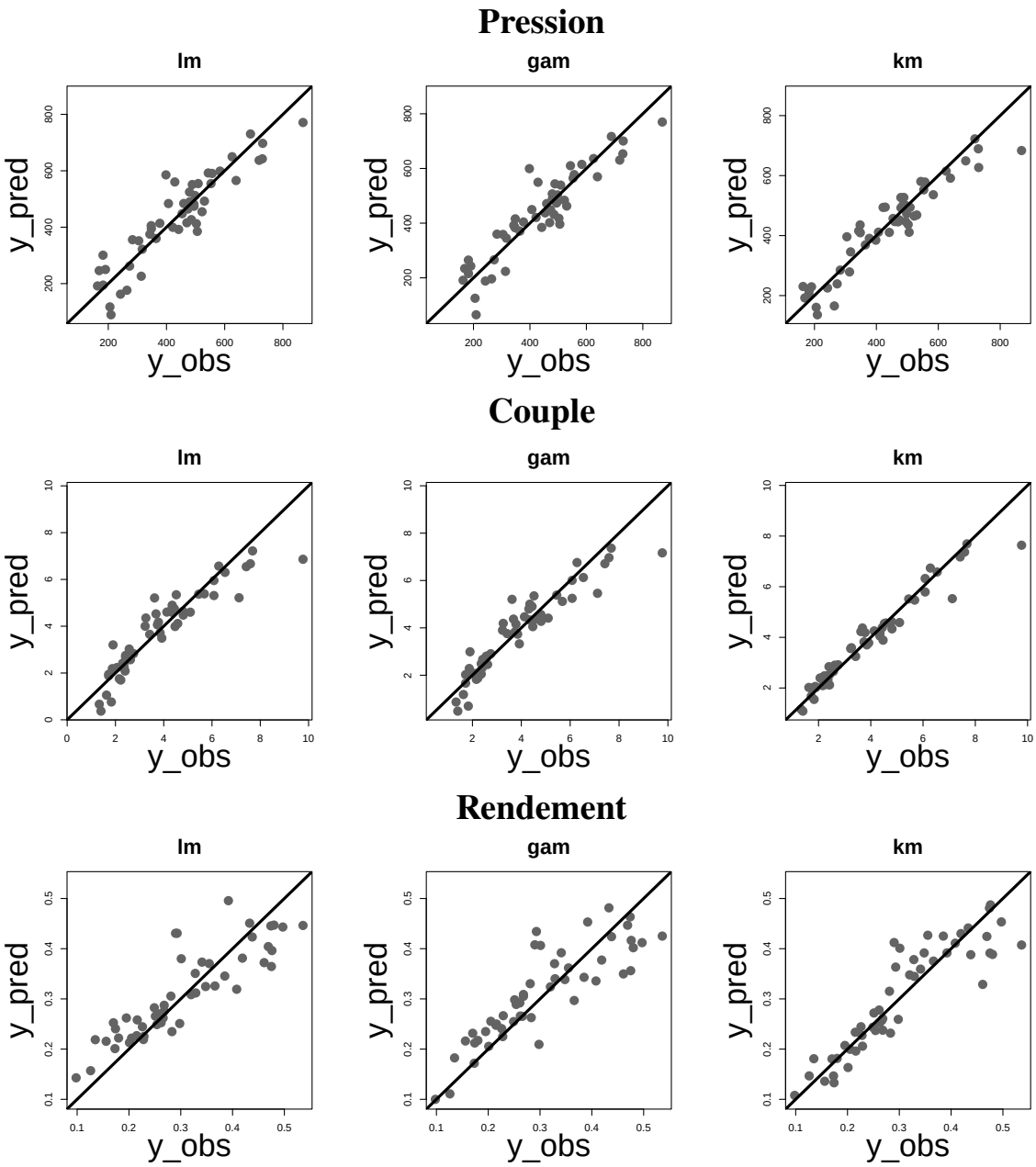

FIGURE 4.2 – Comparaison du modèle de krigeage (km) aux modèles linéaire (lm) et additif généralisé (gam). Valeurs prédites contre valeurs théoriques pour les trois sorties pour un échantillon pris au hasard parmi les cent.

	noyau	estimation	optimisation	borne sup	initialisation
1	Matern5_2	MLE	BFGS	défaut	défaut
2	Matern3_2	MLE	BFGS	défaut	défaut
3	Matern5_2	LOO	BFGS	défaut	défaut
4	Matern5_2	MLE	gen	défaut	
5	Matern5_2	MLE	BFGS	3	défaut
6	Matern5_2	MLE	BFGS	5	défaut
7	Matern5_2	MLE	BFGS	défaut	4
8	Matern5_2	MLE	BFGS	défaut	1

TABLE 4.3 – Configurations testées pour évaluer la stabilité du krigeage. La première ligne correspond à la configuration de référence.

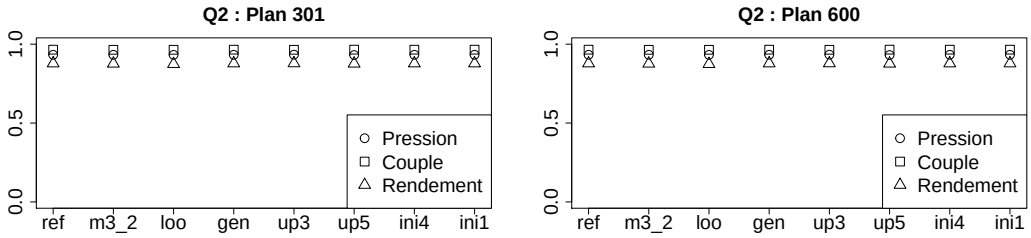

FIGURE 4.3 – Critères Q2 calculé sur les 3 réponses : pression, couple et rendement. Échantillon d'apprentissage de taille 301 et test de taille 597 à gauche. Échantillon d'apprentissage de taille 597 et test de taille 301 à droite.

3 Krigeage avec un noyau *isotrope par groupe* sur le code 3D

3.1 Études préliminaires

3.1.1 Analyse de sensibilité

L'analyse de sensibilité est utilisée dans le but d'analyser l'influence des variables. Les indices de Sobol sont calculés sur la fonction de prédiction du krigeage avec la routine "Sobol2007" de R (cf [Tarantola et al., 2007]). Les résultats sont représentés sur la Figure 4.1. Elle montre l'influence de chaque facteur sur les réponses ainsi que sa part d'interaction et d'effet principal. Les variables x_7, x_8 et x_9 sont influentes pour les trois réponses alors que les variables $x_2, x_6, x_{10}, \ldots, x_{14}$ ne le sont pas. La variable x_1 a beaucoup d'importance pour la pression et le rendement. Les résultats obtenus avec l'analyse de sensibilité sont à utiliser avec précaution lors de l'optimisation des sorties. En effet, une variable peut être peu influente sur l'ensemble du domaine de variation mais très influente autour d'un optimum. Les variables considérées comme influentes par l'analyse de sensibilité ont un paramètre de portée faible contrairement aux variables peu influentes. Dans l'étude suivante, une sélection de variables basée sur la valeur des paramètres de portée est effectuée.

3.1.2 Sélection de variables avec les portées

Dans cette partie, un modèle de krigeage sur le rendement avec la même configuration que précédemment est construit. Il est nommé "modèle complet" car toutes les variables d'entrée sont conservées. L'ensemble d'apprentissage est le plan à 597 points et l'ensemble test celui à 301 points. Le

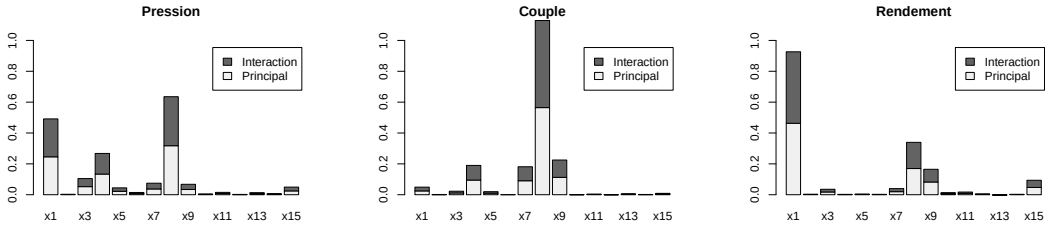

FIGURE 4.1 – Résultats de l'analyse de sensibilité basée sur la prédiction par krigeage avec les indices de Sobol (fonction "Sobol2007" de R).

même krigeage est ensuite estimé en enlevant les variables d'entrée avec une forte portée ($\theta_j \simeq 4$, $j = 1, \ldots, p$). Ce modèle est appelé "modèle réduit". Les valeurs des hyperparamètres estimés ainsi que les critères de qualité de prédiction pour les deux modèles se trouvent dans la Table 4.1. Cette Table montre que les variables avec une portée forte correspondent aux variables non-influentes repérées par l'analyse de sensibilité (cf Figure 4.1). Cependant, le retrait de ces variables engendre une perte de 10% de qualité de prédiction alors que les paramètres de tendance et de variance sont du même ordre. La sélection de variable basée sur l'analyse de sensibilité et la valeur de la portée n'est donc pas une bonne solution pour robustifier le krigeage en grande dimension.

	$\theta(x_1)$	$\theta(x_2)$	$\theta(x_3)$	$\theta(x_4)$	$\theta(x_5)$	$\theta(x_6)$	$\theta(x_7)$	$\theta(x_8)$
Complet	2.23	4	4	2.34	2.51	3.20	3.75	2.09
Réduit	1.44			1.23	0.89	2.92	2.32	0.61
	$\theta(x_9)$	$\theta(x_{10})$	$\theta(x_{11})$	$\theta(x_{12})$	$\theta(x_{13})$	$\theta(x_{14})$	$\theta(x_{15})$	
Complet	3.93	3.97	3.97	2.75	2.32	4	2.6	
Réduit	3.15			2.64	1.87		1.95	
	m	σ^2	errRelat					
Complet	372.01	19616.73	0.16					
Réduit	382.48	19640.88	0.27					

TABLE 4.1 – Valeurs des paramètres de portée, de variance et de tendance estimés et valeurs des critères de qualité de prédiction pour le modèle complet et réduit.

3.2 Application du krigeage avec un noyau *isotrope par groupe*

Les études préliminaires montrent que le modèle de krigeage donne de bons résultats sur les données de Valeo. De plus, l'étude sur l'analyse de sensibilité montre que pour chaque réponse seulement quelques variables semblent influentes, ces variables ont un paramètre de portée faible. La valeur du paramètre de portée semble liée à l'influence de la variable. Cependant, sélectionner directement les variables avec une petite portée est impossible car la perte de qualité de prédiction est trop importante. Il faut donc réduire le nombre de paramètres à estimer afin de gagner en robustesse pour monter en dimension tout en conservant une qualité prédictive acceptable. Dans cette section, la méthodologie proposée dans le Chapitre 2 permet de construire automatiquement un noyau de covariance *isotrope par groupe* pour estimer moins de paramètres de portée. Elle permet aussi de gagner en robustesse lorsque la dimension est élevée comparée au nombre d'observations. Bien que pour l'instant, le nombre de variables d'entrée reste raisonnable (15 paramètres), l'efficacité de la méthode doit néanmoins être

testée pour monter en dimension. Dans un premier temps, les groupes formés sont étudiés pour les confronter à la physique. Dans un deuxième temps, les valeurs des paramètres de portée des groupes sont mis en relation avec les résultats de l'analyse de sensibilité précédente. Dans un dernier temps, la qualité de prédiction obtenue avec le noyau *isotrope par groupe* est comparée à celle du noyau *anisotrope*.

Le noyau *isotrope par groupe* est construit à partir des noyaux *anisotrope* et *isotrope* avec une distance euclidienne. Par exemple, un noyau *isotrope par groupe* avec trois variables et deux groupes $\{x_1\}$, $\{x_2, x_3\}$ s'écrit :

$$\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \rho_{\theta_1}(|x_1 - \tilde{x}_1|) \times \rho_{\theta_2}\left(\sqrt{(x_2 - \tilde{x}_2)^2 + (x_3 - \tilde{x}_3)^2}\right)$$

avec ρ la fonction de covariance Matern5_2. Quatre algorithmes ont été développés pour déterminer la répartition des variables dans les groupes de façon automatique. L'**Algorithme** 4 qui donne les meilleurs résultats sur les cas test du Chapitre 2 est appliqué sur le cas industriel. Cet algorithme a une stratégie séparatrice mais une méthode de clustering est effectuée à chaque itération pour regrouper les variables.

Le noyau *isotrope par groupe* est estimé sur le plan à 597 points avec l'**Algorithme** 4 . L'algorithme est lancé sur les trois réponses : la pression, le couple et le rendement. Les groupes obtenus sont ensuite analysés. La composition des groupes peut changer suivant les estimations. Cette instabilité est due à l'optimisation du maximum de vraisemblance. La Figure 4.2 et le Tableau 4.2 montrent la composition des groupes obtenus à l'aide de l'**Algorithme** 4 pour le krigeage avec un noyau *isotrope par groupe*.

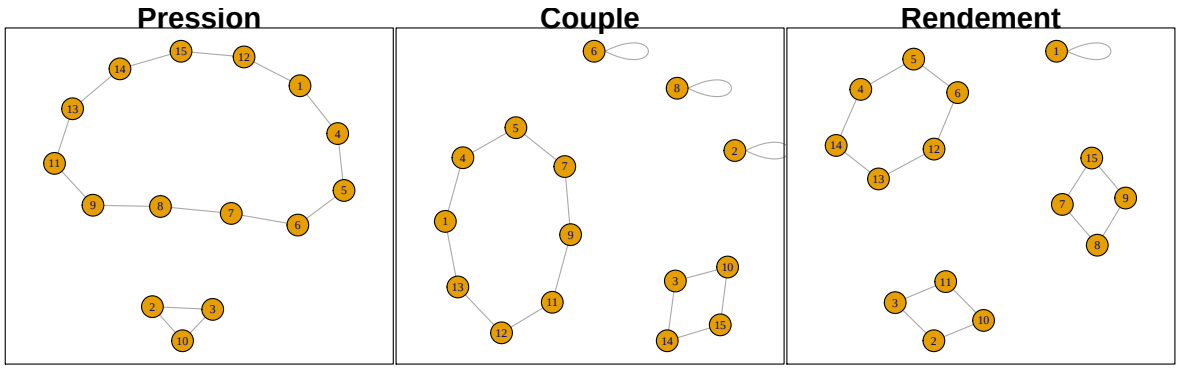

FIGURE 4.2 – Composition des groupes dans le cadre de l'estimation du noyau *isotrope par groupe* avec l'**Algorithme** 4 pour la pression a gauche, le couple au centre et le rendement à droite.

Pression

L'algorithme a détecté deux groupes, le premier contient le plus de variables et a la portée la plus faible, le dernier a une valeur de portée plus élevée et est composé de x_2, x_3 et x_{10}.

Couple

L'algorithme a détecté cinq groupes, le premier contient le plus de variables et a une valeur de portée moyenne, le second a une portée faible et contient uniquement x_8. Les trois derniers groupes ont de fortes valeurs de portée et contiennent les variables $x_2, x_3, x_{10}, x_{14}, x_{15}$ et x_6.

Pression

Groupe	$\theta_1(x_1, x_4, \ldots, x_9, x_{11}, \ldots, x_{15})$	$\theta_2(x_2, x_3, x_{10})$
Valeur	3.03	5.81

Couple

Groupe	$\theta_1(x_1, x_4, x_5, x_7, x_9, x_{11}, x_{12}, x_{13})$	$\theta_2(x_8)$	$\theta_3(x_2)$	$\theta_4(x_3, x_{10}, x_{14}, x_{15})$	$\theta_5(x_6)$
Valeur	3.25	1.21	10	8.27	4.48

Rendement

Groupe	$\theta_1(x_4, x_5, x_6, x_{12}, x_{13}, x_{14})$	$\theta_2(x_2, x_3, x_{10}, x_{11})$	$\theta_3(x_1)$	$\theta_3(x_7, x_8, x_9, x_{15})$
Valeurs	2.77	5.45	0.74	1.5

TABLE 4.2 – Paramètres de portées estimés avec un krigeage *isotrope par groupe* sur le plan à 597 points pour les trois réponses : pression, couple et rendement.

Rendement

L'algorithme a détecté quatre groupes, les variables avec des portées faibles sont réparties en deux groupes x_1 puis x_7, x_8, x_9 et x_{15}. Le groupe avec une portée moyenne est composée de $x_4, x_5, x_6, x_{12}, x_{13}$ et x_{14}. Le dernier groupe avec une valeur de portée élevée contient les variables x_2, x_3, x_{10} et x_{11}.

Pour les trois réponses, les variables x_2, x_3 et x_{10} ont une forte portée. Ces variables correspondent aux longueurs de corde aux sections 1 et 2 et à la hauteur maximale à la section 1. Les experts industriels confirment que ces variables n'influencent pas beaucoup le fonctionnement de la turbomachine. Contrairement aux variables x_1, x_7, x_8 et x_9 correspondant au débit et aux calages aux sections $2, 4$ et 5 qui sont très influentes. Ces observations se retrouvent dans l'analyse de sensibilité de l'étude préliminaire. De façon globale les variables avec une portée faible sont très influentes et celles avec une portée forte le sont beaucoup moins. En conclusion, les groupes de variables formés correspondent à la physique de la turbomachine.

La seconde partie consiste à regarder la qualité de prédiction du noyau *isotrope par groupe* sur l'ensemble test à 301 points. Le but est d'observer la conservation de la qualité de prédiction obtenue avec le noyau *isotrope par groupe* par rapport à celle obtenue avec le noyau *anisotrope*. La Figure 4.3 et le Tableau 4.3 montrent qu'en simplifiant le krigeage, estimation de 3 ou 5 portées pour le noyau *isotrope par groupe* au lieu de 15 pour l'*anisotrope*, la qualité prédictive est conservée pour les trois réponses. Un léger gain est observé pour la pression. Ces résultats sont très encourageants pour la suite. En effet, le fonctionnement de la turbomachine sera étudié en prenant en compte non plus 15 mais 30 et 60 paramètres. Dans ce contexte, le krigeage avec un noyau *isotrope par groupe* permettra de traiter ces cas avec moins de perte de qualité de prédiction par rapport à un krigeage avec un noyau *anisotrope* ou *isotrope*.

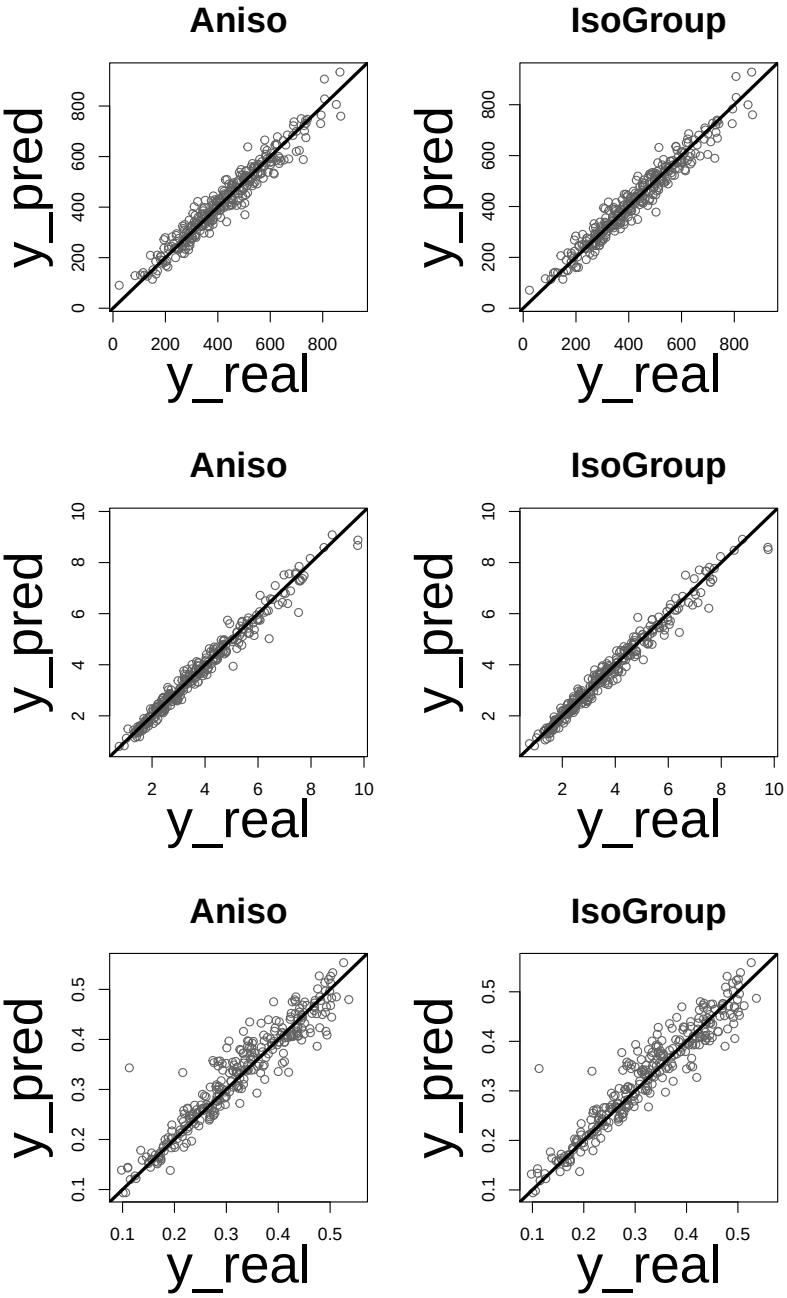

FIGURE 4.3 – Graphiques des valeurs prédites contre les valeurs réelles avec un noyau *anisotrope* (à gauche) et *isotrope par groupe* (à droite) pour les trois réponses : pression (en haut), couple (au milieu) et rendement (en bas).

Pression

Noyau	Q2	RMSE	errRelat
Anisotrope	0.939	39.0	0.090
Isotrope par groupe	0.943	37.8	0.085

Couple

Noyau	Q2	RMSE	errRelat
Anisotrope	0.973	0.287	0.059
Isotrope par groupe	0.973	0.28	0.060

Rendement

Noyau	Q2	RMSE	errRelat
Anisotrope	0.896	0.034	0.082
Isotrope par groupe	0.894	0.034	0.083

TABLE 4.3 – Critères de qualité de prédiction calculés pour les deux noyaux et sur les trois réponses (cf Chapitre 1 Section 5.1).

4 Optimisation robuste sur le code 1D

Le but de cette section est de procéder à une optimisation robuste du rendement sur le code 1D fourni par le LMFA. Ce code numérique, décrit en détail dans l'introduction de ce manuscrit (cf Chapitre Introduction), permet d'obtenir les observations du rendement et de ses dérivées par rapport aux quinze variables d'entrées : la longueur des cordes, les calages et les hauteurs maximales aux sections $1, 2, 3, 4, 5$ notées $\mathbf{x} = (x_1, \ldots, x_{15}) \in D$. Parmi ces variables, seuls les calages aux sections $2, 3, 4$ sont susceptibles de subir des perturbations qui ont une incidence sur la sortie. Ce sont aussi les variables les plus influentes. Ces informations sont données par les experts de Valeo. Les calages $2, 3, 4$ notés x_7, x_8, x_9 sont supposés suivre une loi normale tel que $X_i \sim \mathcal{N}(x_i, \delta_i^2)$, $i = 7, 8, 9$. Le choix des variances δ_i^2 dépend des perturbations subies. A un point de l'espace $\mathbf{x}$ fixé les erreurs commises par les machines pour les variables de calage $2, 3$ et 4 varient dans 99% des cas dans l'intervalle $\left[x_i\left(1-\frac{1}{20}\right); x_i(1+\frac{1}{20})\right]$, $i = \{7, 8, 9\}$ qui correspond à 5% de l'erreur relative. Cet intervalle est centré en $\mathbf{x}$, plus les valeurs sont excentrées plus la probabilité que les calages prennent ces valeurs est faible. C'est pourquoi, la loi normale est une bonne modélisation de ces perturbations. Cet intervalle dépend de la valeur de $\mathbf{x}$ alors que les variances sont fixes quelque soit le point. La variance est calculée à partir du pire des cas i.e. le plus grand intervalle donné par :

$$I_{max} = \left[\max|x_i|\left(1-\frac{1}{20}\right); \max|x_i|\left(1+\frac{1}{20}\right)\right],\ i = \{7, 8, 9\}$$

Les variables doivent être comprise dans l'intervalle I_{max} dans 99% des cas. Soit $q = \Phi^{-1}(0.995)$ le quantile d'ordre 0.995 de la loi normale standardisée, δ^2 est défini tel que (pour simplifier les notations : $x_i = x$ et $\delta_i = \delta$) :

$$\max|x|\left(1-\frac{1}{20}\right) = \max|x| - q\times\delta \Leftrightarrow q\times\delta = \frac{1}{20}\max|x|$$
$$\Leftrightarrow \delta = \frac{\max|x|}{q\times 20}$$

D'où $\delta_i = \frac{\max|x_i|}{q\times 20}$ pour $i = \{7, 8, 9\}$ sinon $\delta_i = 0$. Les variables étudiées, leur domaine de variation

Variable	Lcorde					Calage					Hmax				
Section	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
Notation	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
Min	0.04	0.06	0.08	0.09	0.11	-50.67	-59.68	-65.87	-70.29	-73.58	3.82	3.82	3.82	2.86	1.91
Max	0.07	0.09	0.11	0.14	0.16	-45.85	-54	-59.59	-63.6	-66.57	5.73	5.73	5.73	4.29	2.86
δ	0	0	0	0	0	0	1.16	1.28	1.36	0	0	0	0	0	0

TABLE 4.1 – Récapitulatif des variables d'entrées du code 1D. L désigne le paramètre de longueur de corde, C celui de calage et H celui de la hauteur maximale. Ces trois paramètres sont pris aux sections 1 à 5.

et la valeur de l'écart-type correspondant aux perturbations sont décrits dans le Tableau 4.1.

L'ensemble d'apprentissage initial est un plan de type LHS Optimisé par un algorithme Maximin cf [Dupuy et al., 2015]. Il contient 46 observations. La Figure 4.1 représente les points du plan d'apprentissage initial dans l'espace des objectifs $\{-\eta, RC_\eta\}$. η représente la fonction de rendement réelle et coûteuse. Le budget total est de 136 points, 90 points sont ajoutés au design initial par les différentes méthodes. Dix-huit points sont ajoutés au maximum par mise à jour. Les trois méthodes les plus prometteuses proposées dans le chapitre précédent, MyKB, MyCL et MyqEI sont testées. A chaque étape, le problème d'optimisation initial est remplacé par un problème multi-objectif sur les prédictions de

la fonction ($\widehat{y}$) et du critère de robustesse ($RC_{\widehat{y}}$)par krigeage. Le critère de robustesse est une approximation de la variance locale de la fonction obtenue par le développement de Taylor. A la fin de la minimisation, plusieurs points du front de Pareto sont sélectionnés pour enrichir le metamodèle. Cette sélection s'opère des trois façons suivantes : algorithme de recuit simulé (MyqEI), Kriging Believer (MyKB) et Constant Liar (MyCL). L'algorithme NSGA II (cf [Deb et al., 2002]) est utilisé sur les objectifs $\{-\widehat{y}, RC_{\widehat{y}}\}$ pour trouver un front de Pareto constitué d'une population de 100 individus. Cent générations de population sont faites avec une probabilité de croisement de 1 et de mutation de $\frac{1}{15}$.

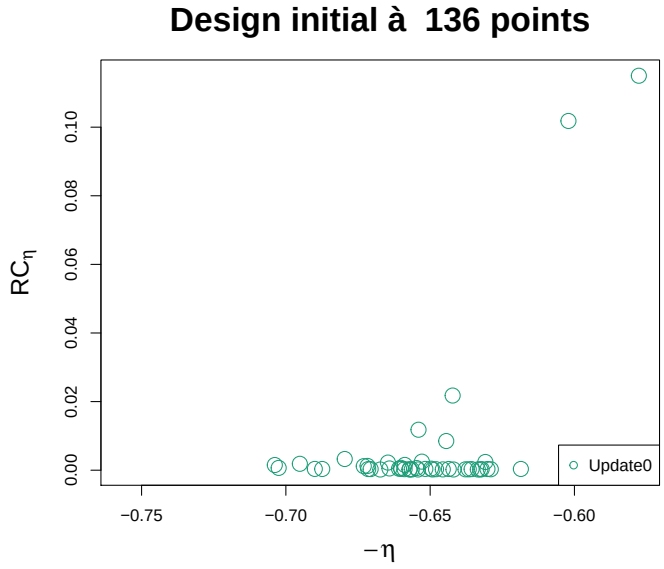

FIGURE 4.1 – Répartition des 46 points du design initial dans l'espace des objectifs réels : -rendement ($-\eta$) et critère de robustesse calculé sur le rendement (RC_η).

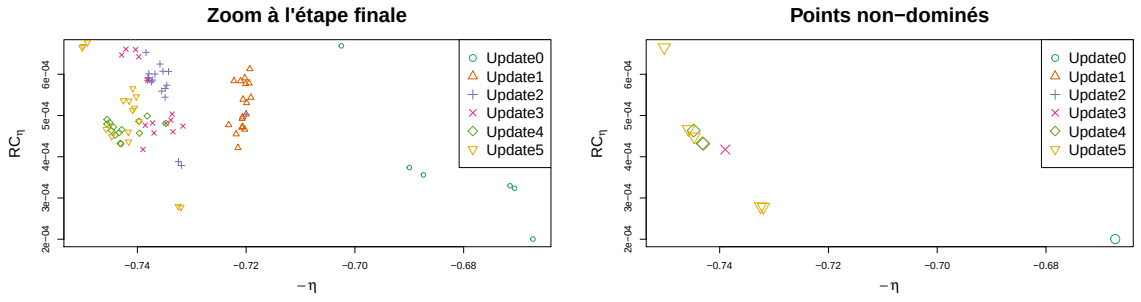

FIGURE 4.2 – Répartition des points du design final pour la méthode MyCL dans l'espace des objectifs réels : -rendement ($-\eta$) et critère de robustesse calculé sur le rendement (RC_η). A chaque étape (update) est associée une forme et une couleur. La figure de droite représente uniquement les points non-dominés.

Les Figures 4.2, 4.3 et 4.4 montrent les points ajoutés par la méthode MyCL, MyKB et MyqEI aux différentes étapes de l'algorithme dans la zone intéressante. Pour les trois méthodes l'algorithme commence par une phase d'exploration dans différentes zones de l'espace des objectifs. Dans le cas de MyCL, la formation d'un front n'apparait pas clairement contrairement à MyKB et MyqEI où un front est dessiné à la dernière étape. Le Tableau 4.2 montre que les trois algorithmes ont le même

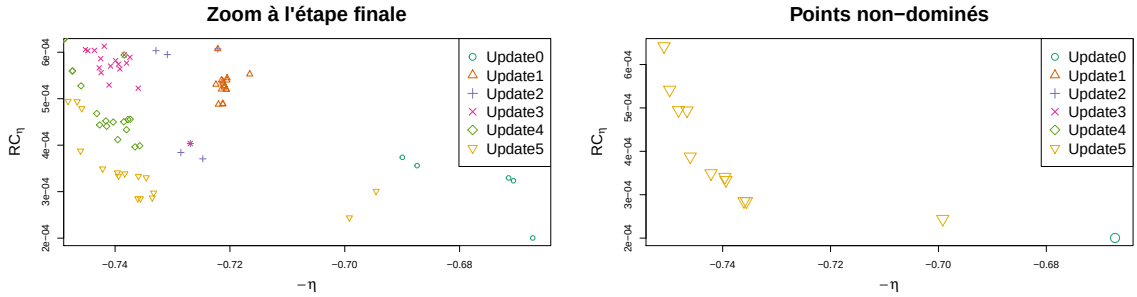

FIGURE 4.3 – Répartition des points du design final pour la méthode MyKB dans l'espace des objectifs réels : -rendement (-η) et critère de robustesse calculé sur le rendement (RC_η). A chaque étape (update) est associée une forme et une couleur. La figure de droite représente uniquement les points non-dominés.

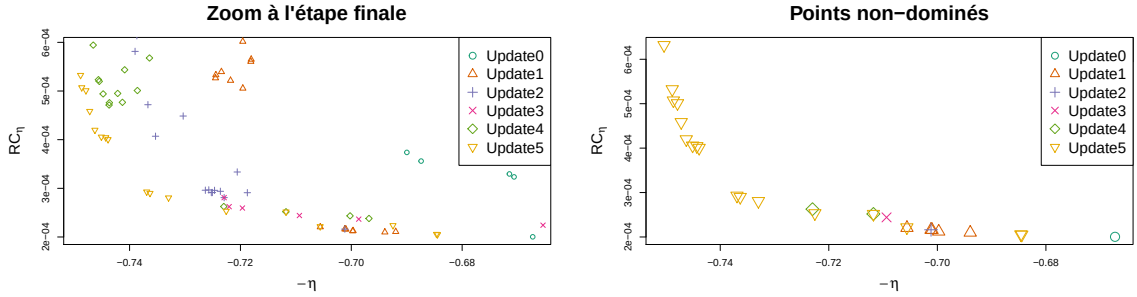

FIGURE 4.4 – Répartition des points du design final pour la méthode MyqEI dans l'espace des objectifs réels : -rendement -(η) et critère de robustesse calculé sur le rendement (RC_η). A chaque étape (update) est associée une forme et une couleur. La figure de droite représente uniquement les points non-dominés.

	Update	0	1	2	3	4	5	Total
MyCL	NB points	46	18	18	18	18	18	136
	Temps		0h18	0h30	0h40	1h00	1h00	3 h 28
MyKB	NB points	46	18	18	18	18	18	136
	Temps		0h18	0h31	0h44	1h00	1h01	3h34
MyqEI	NB points	46	18	18	18	18	18	136
	Temps		0h16	0h25	0h36	0h48	1h00	3 h05

TABLE 4.2 – Récapitulatif des mises à jours (updates) et du temps d'estimation par mise à jour pour les trois méthodes : MyCL, MyKB et MyqEI.

nombre de mise à jour. La méthode MyqEI est la plus rapide en temps de calcul. La Figure 4.5 montre que la méthode MyCL est la moins avancée sur le front. MyqEI est celle qui a exploré la plus grande zone et qui a construit le front le plus complet. MyKB est légèrement plus avancée sur la zone centrale du front.

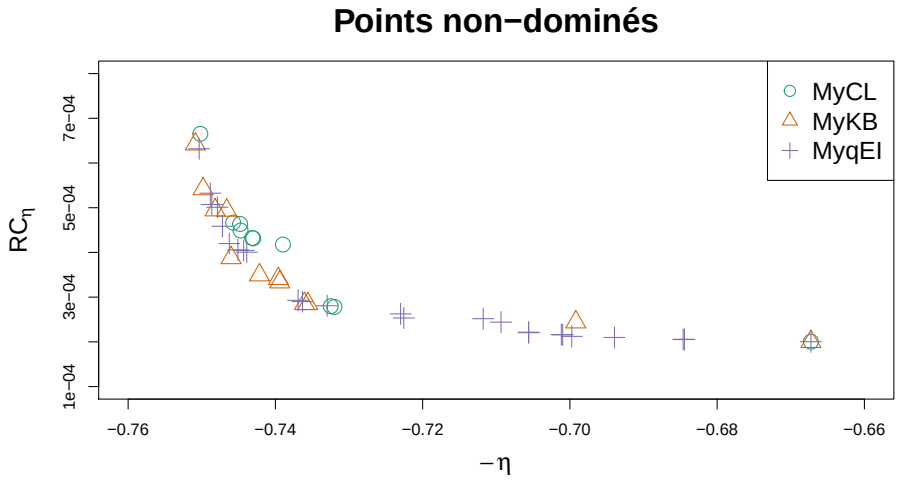

FIGURE 4.5 – Répartition des points non dominés des designs finaux pour les trois méthodes (MyCL, MyKB et MyqEI) dans l'espace des objectifs réels : rendement (η) et critère de robustesse calculé sur le rendement (RC_η).

Conclusion du chapitre

Tout d'abord, Les études préliminaires montrent que le krigeage donne de meilleurs résultats sur le code 3D comparé aux modèles linéaire et additif généralisé. De plus, l'estimation par krigeage est stable. L'analyse de sensibilité et la sélection des variables avec les portées montrent que la construction d'une méthode permettant de réduire la dimension sans sélection de variables est recommandée pour conserver la qualité de prédiction. Puis, l'application de l'algorithme des groupes de portée sur le code 3D montre que le nombre de paramètres estimés peut être réduit en conservant la qualité de prédiction. De plus, les groupes formés ont une interprétation physique. Enfin, l'optimisation robuste appliquée sur le code 1D prouve qu'il existe un front de Pareto des solutions robustes en rendement. Les trois méthodes testées ont proposé la même zone de solutions. La méthode MyqEI donne le front le plus complet en un temps de calcul plus faible.

Chapitre 5

Conclusions et perspectives

Ce chapitre est organisé en trois sections. Les conclusions et perspectives des trois chapitres précédents qui correspondent aux propositions originales de ce manuscrit sont données ici.

1 Chapitre 2 : Un nouveau noyau de covariance pour réduire la dimension et quatre algorithmes pour sa construction

1.1 Conclusions

Ce chapitre propose une solution aux problèmes liés au krigeage en grande dimension. Lorsqu'il y a beaucoup de variables d'entrée et peu d'observations, l'estimation du krigeage avec un noyau *anisotrope* devient difficile. L'espace d'optimisation est trop grand. Le noyau *isotrope*, quant à lui, permet de réduire l'espace d'optimisation à un seul paramètre mais il est trop restrictif et donc pauvre en prédiction. C'est pourquoi, un noyau *isotrope par groupe* est introduit. Ce noyau est construit comme un produit de noyaux *isotrope* où chacun de ces noyaux est lié à un sous groupe de variables d'entrée. Le nombre de paramètres à estimer correspond au nombre de sous groupes. L'espace d'optimisation est réduit tout en conservant une certaine flexibilité. La difficulté est de trouver le nombre de groupes et leur composition. Quatre algorithmes ont été développés pour trouver la meilleure partition des variables d'entrée. Ils partent tous d'un noyau composé d'un unique groupe contenant toutes les variables (noyau *isotrope*). Puis, ils parcourent différents noyaux *isotrope par groupe* selon leur stratégie respective. Le modèle retenu est celui qui a le critère BIC minimal. Ces algorithmes ont été appliqués sur un cas test simulé et l'**Algorithme** 4 a été retenu. L'algorithme alterne des étapes forward et des étapes backward, l'étape backward étant rendue possible par une procédure de classification sur les valeurs des portées. Des procédures de modélisation par krigeage avec différents noyaux (*isotrope*, *anisotrope* et *isotrope par groupe*) ont été comparées sur diverses applications. Le krigeage avec un noyau *isotrope par groupe* s'est révélé le plus prédictif. Plus précisément, le premier travail réalisé portant sur la qualité de prédiction en fonction du nombre de points dans l'ensemble d'apprentissage a montré que le noyau *isotrope par groupe* donne les meilleurs résultats. Pour les échantillons avec peu de points, la qualité de prédiction obtenue avec notre noyau est plus élevée que le noyau *isotrope*. Le noyau *anisotrope*, quant à lui, donne des résultats complètement erronés dans un nombre non négligeable de cas. L'ajout d'une perturbation sur les paramètres de portée lors de la simulation du processus n'engendre pas de perte de qualité de prédiction pour le noyau *isotrope par groupe* et reste plus performant que les deux autres. Ce n'est cependant pas une solution pour traiter les cas où il y a un effet de pépite. Les applications sur la fonction réelle montre que le noyau *isotrope par groupe* améliore de façon conséquente la qualité de prédiction par rapport aux noyaux *anisotrope* et *isotrope*.

En conclusion, le modèle de krigeage avec un noyau *isotrope par groupe* permet d'améliorer la qualité de prédiction en grande dimension.

1.2 Perspectives

Les quatre algorithmes présentés au cours de ce chapitre sont génériques, ils peuvent être utilisés pour construire séquentiellement d'autres noyaux creux en nombre de portée. Par exemple, pour trois variables et deux groupes $\{x_1\}$, $\{x_2, x_3\}$, la forme du noyau *isotrope par groupe* est la suivante :

$$\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \rho_{\theta_1}(|x_1 - \tilde{x}_1|) \times \rho_{\theta_2}\left(\sqrt{(x_2 - \tilde{x}_2)^2 + (x_3 - \tilde{x}_3)^2}\right)$$

D'autres exemples de forme de noyaux creux :

- le noyau *produit par groupe* :

$$\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \rho_{\theta_1}(|x_1 - \tilde{x}_1|) \times \rho_{\theta_2}(|x_2 - \tilde{x}_2|) \times \rho_{\theta_2}(|x_3 - \tilde{x}_3|)$$

- le noyau *additif* (cf [Muehlenstaedt et al., 2012]) :

$$\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x} - \tilde{\mathbf{x}}) = \rho_{\theta_1}(|x_1 - \tilde{x}_1|) + (\rho_{\theta_2}(|x_2 - \tilde{x}_2|) \times \rho_{\theta_2}(|x_3 - \tilde{x}_3|))$$

Ces algorithmes peuvent aussi être utilisé comme une méthode d'initialisation à l'optimisation de la vraisemblance pour un krigeage avec un noyau *anisotrope*.

[Welch et al., 1992] introduit un algorithme similaire au notre et il en propose une extension qui consiste à optimiser un seul paramètre de portée en fixant tous les autres. L'appliquer aux algorithmes introduits dans la thèse serait judicieux pour accélérer l'estimation. Il suffit de conserver la valeur du paramètre de portée pour les groupes qui ne subissent pas de modification et d'estimer seulement les paramètres de portée associés aux nouveaux groupes.

Enfin, l'étape de regroupement des variables (clustering) de l'**Algorithme** 3 et 4 peut être améliorée en développant des méthodes adaptées au type de noyau de covariance. Par exemple, pour le noyaux *isotrope par groupe*, il faudrait créer une distance pondérée en fonction du nombre de variables contenues dans chaque groupe. Et, à partir de cette distance construire un critère permettant de juger de la proximité de deux groupes.

2 Chapitre 3 : Optimisation robuste

2.1 Conclusions

Ce chapitre répond à la problématique de l'optimisation robuste avec simulations coûteuses. Dans le contexte de cette thèse, les variables d'entrées subissent des perturbations liées à la chaîne de production. En effet, pour un point du design $\mathbf{x}$ donné, la sortie peut prendre les valeurs $f(\mathbf{x} + \mathbf{H})$ où $\mathbf{H}$ est une perturbation de loi Gaussienne. Le challenge est de quantifier l'impact des perturbations sur la sortie en chaque point du design. La variance locale est utilisée pour approcher la robustesse d'une solution. Cette variance n'a pas de forme analytique, elle doit être estimée. Le critère de robustesse choisi est basé sur le développement de Taylor afin d'approximer cette variance. Le problème d'optimisation robuste devient alors un problème multi-objectif qui consiste à optimiser la fonction ainsi que le critère de robustesse. Le métamodèle de co-krigeage est utilisé ici pour prédire la fonction coûteuse ainsi que ses dérivées. Puis, le critère de Taylor est calculé. Cependant, le métamodèle engendre une incertitude de prédiction. Il faut donc mettre en place une méthode pour la prendre en compte lors

de la procédure d'optimisation. Pour ce faire, sept stratégies itératives basées sur les prédictions de la fonction et du critère de robustesse ont été développées. A chaque itération, un front de Pareto est obtenu grâce à ces prédictions et de nouvelles observations de la fonction coûteuse sont demandées pour améliorer le krigeage dans les zones intéressantes. Certaines stratégies ne construisent le front que sur la prédiction du krigeage, l'exploration se fait par le choix optimal des points sur le front. D'autres stratégies intègrent la variance de krigeage dès la construction du front assurant ainsi l'exploration de la méthode. Enfin, ces méthodes sont comparées sur deux cas tests avec deux possibilités : accès aux dérivées liées aux perturbations ou non. Plusieurs optimisations robustes sont menées avec chaque stratégie sur un cas en 2D. Dans le cas avec dérivées, il en ressort que trois stratégies MyKB, MyCL et MyqEI donnent les meilleures performances. Dans le cas sans dérivée, ce sont les stratégies MyAlea, MyqEI et MyCL qui donnent les meilleurs résultats. Dans les deux cas, les trois meilleures stratégies ont ensuite été appliquées sur un cas en dimension six. Deux variables subissaient des perturbations. Les résultats montrent qu'à chaque fois toutes les stratégies se valent tant en terme de construction du front qu'en temps de calcul. En conclusion, les résultats des méthodes dépendent de la régularité de la fonction. Lorsque la fonction est complexe il faut soit observer les dérivées soit adopter des stratégies exploratoire fortes (MyAlea). Au contraire, si la fonction est régulière, comme c'est le cas souvent pour des fonctions physiques, l'accès aux dérivées n'est pas essentielle et des méthodes comme MyqEI ou MyCL sont très efficaces. Cependant, il est possible d'améliorer l'efficacité des méthodes basées sur l'EI (MEIyAlea et MqEIyAlea).

2.2 Perspectives

Dans ce chapitre, les perturbations sont considérées comme des variables gaussiennes. Dans l'industrie, les variables sont parfois modélisées par d'autres lois. Cependant, la forme analytique du critère de robustesse est valable uniquement dans le cas Gaussien. Une possibilité, dans le cas non Gaussien, est de calculer le critère de façon empirique. Le problème est le coup élevé de calcul pour obtenir une bonne approximation. L'idéal serait d'expliciter une forme analytique du critère pour chaque loi modélisant les perturbations observées dans l'industrie.

De plus, le critère de robustesse peu s'écrire comme une fonction du gradient et de la matrice hessienne (cf Equation 3.11) :

$$RC_f(\mathbf{x}) = g\left(f(\mathbf{x}), f_{x_j}(\mathbf{x}), f_{x_j,x_k}(\mathbf{x})\right)$$

Dans cette thèse, le critère est prédit avec une méthode "plug-in" :

$$RC_{\widehat{y}}(\mathbf{x}) = g(\widehat{y}(\mathbf{x}), \widehat{y}_{x_j}(\mathbf{x}), \widehat{y}_{x_j,x_k}(\mathbf{x}))$$

Une autre façon de le faire serait d'estimer :

$$\mathbb{E}[RC_y(\mathbf{x})|\mathbf{z}_{u_{obs}}] = \mathbb{E}[g(y(\mathbf{x}), y_{x_j}(\mathbf{x}), y_{x_j,x_k}(\mathbf{x}))|\mathbf{z}_{u_{obs}}]$$

Cependant, il faudrait trouver une forme analytique de cette expression car le critère est utilisé lors de l'optimisation avec le NSGA II.

Par ailleurs, la stratégie proposée dans le second groupe est lente et peu efficace. L'idée serait de trouver un autre moyen pour sélectionner les points, cette sélection est pour l'instant aléatoire. Une seconde solution pourrait être d'adapter les algorithmes KB et CL pour que l'ajout de points par batch soit directement pris en compte lors de la formation du front. De plus, cette stratégie fait appel au critère de l'amélioration espérée (EI) calculé sur la fonction et sur le critère de robustesse. Cependant, ces deux EIs ne sont pas comparables car ils ne varient pas dans le même domaine. Les points sont donc sélectionnes aléatoirement sur le front. Cette méthode n'est pas optimale. L'idée serait de créer

un critère en pourcentage d'amélioration ce qui permettrait d'avoir des EIs à la même échelle et de sélectionner le point sur le front le plus proche de l'origine. Cette remarque s'applique aussi à l'algorithme génétique permettant de minimiser $-qEI_y - qEI_{RC_y}$. Cette somme n'est pas tout le temps pertinente car il est possible que les deux termes ne varient pas dans le même espace. Il faudrait aussi construire un critère normalisé.

Enfin, lors des applications, seules les dérivées nécessaires au calcul du critère de robustesse (celles liées aux variables perturbées) ont été utilisées. Les autres ne sont pas utilisées car l'estimation du modèle et la prédiction deviennent coûteuses en grande dimensions. En effet, la matrice de covariance qui est inversée pendant l'estimation des paramètres de krigeage est de taille $n \times d$ où n est le nombre d'observations et d le nombre de fonctions (fonction et dérivées de la fonction). Par exemple, pour un cas en dimension 15 avec seulement 50 points d'observations, la matrice à inverser serait de taille 6800, ce qui est très coûteux. Il est impératif de réduire la taille de cette matrice pour procéder à l'optimisation robuste par krigeage. Il serait donc intéressant de développer une procédure de sélection automatique des dérivées influentes. Cela permettrait de réduire la dimension dans l'espace des sorties du co-krigeage de façon judicieuse.

3 Chapitre 4 : Application au cas industriel

Les deux principales problématiques liées au cas industriel proposé par Valeo sont :

- Comment construire une bonne surface de réponse pour les sorties (pression, couple et rendement) lorsqu'il y a beaucoup de paramètres géométriques ?
- Comment procéder à une optimisation du rendement efficace tout en prenant en compte les perturbations sur variables d'entrée ?

Le code 3D de Valeo est utilisé pour répondre à la première question. Il est très coûteux en temps de calcul et donne uniquement les observations des sorties en chaque point du design. Les variables d'entrées sont le débit et quinze variables géométriques. Le code 1D est utilisé pour répondre à la seconde question,. Ce code permet de tester facilement les méthodes dans un contexte séquentiel (acquisition des données séquentiellement par un algorithme de type EGO). Il fournit les observations des sorties et de leurs dérivées par rapport au débit et aux variables géométriques, à moindre coût.

3.1 Conclusions

Afin de traiter ces deux problématiques, les méthodes développées dans les chapitres précédents sont appliquées. Elles utilisent un modèle de krigeage. Dans un premier temps, plusieurs études préliminaires sur le code 3D ont été effectuées pour valider le modèle choisi. Les plans d'expériences contenant les observations des sorties en différents points de l'espace et fourni par Valeo ont été choisis. La première étude montre que le krigeage donne de meilleurs résultats que le modèle linéaire et le modèle additif généralisé. De plus, l'estimation par krigeage est stable par rapport au choix du noyau, à la méthode d'estimation des paramètres et à la méthode d'optimisation de la vraisemblance. Cependant, la sélection de variable basée sur les valeurs des portées et l'analyse de sensibilité ne suffit pas à gérer la grande dimension. En effet, la construction d'une méthode permettant de réduire la dimension des paramètres du modèle sans sélection de variable est recommandée pour conserver une bonne qualité de prédiction. C'est pourquoi, la méthode proposée dans le chapitre 2 est appliquée sur le plus grand plan d'expérience obtenu avec le code 3D. Le second plan est utilisé pour évaluer la qualité de prédiction. L'**Algorithme** 4 est lancé sur chaque sortie. Les résultats montrent qu'en regroupant les variables, le nombre de paramètres à estimer est réduit et la qualité de prédiction est conservée. Les groupes formés par l'algorithme sont en accord avec l'intuition des experts de Valeo. L'application des trois stratégies les plus performantes du chapitre "optimisation robuste" sur le code 1D montre la

formation d'un front des solutions robustes pour le rendement. Seuls les trois calages intermédiaires subissent des perturbations. Les méthodes ont donné des fronts de Pareto proches. De plus, la méthode MyqEI ressort en donnant le front le plus complet en un temps de calcul le plus faible.

3.2 Perspectives

Au cours de ce chapitre, l'optimisation robuste a uniquement été conduite sur le code 1D. Il faudrait tester l'optimisation robuste sur le code 3D. Pour le code 3D, les observations des dérivées ne sont pas accessibles. Néanmoins, le co-krigeage permet d'obtenir les prédictions des dérivées sans en avoir les observations. Une autre solution est d'utiliser un modèle d'agrégation de code pour conduire l'optimisation robuste en utilisant les informations données par les deux simulateurs. En effet, les dérivées du code 1D sont observables et sont de bonnes approximations des dérivées du code 3D. Il faudrait donc créer un modèle capable de prédire le code 3D en se servant de l'observation du code 3D et des dérivées du code 1D.

De plus, l'optimisation robuste à uniquement été menée sur le rendement, l'idée serait de procéder à une optimisation multi-objectif robuste sur la pression et le couple. Dans ce cas, il faudrait utiliser l'algorithme NSGA III (cf [Mkaouer et al., 2014]) qui permet de traiter plus d'objectifs. D'autre part, le noyau *isotrope par groupe* présenté dans la section "groupe de portée" pourrait être utilisé pour conduire une optimisation robuste avec beaucoup de variables d'entrées. Ce noyaux permet de stabiliser l'estimation par krigeage en conservant toutes les variables.

Dans le contexte industriel, il arrive que les simulations fournies n'aient pas convergées. Il faudrait introduire un effet de pépite dans le modèle de co-krigeage avec dérivées pour prendre en compte cette erreur. Aussi, les codes industriels prennent en entrée des variables géométriques ainsi que le débit. Cette dernière variable est différente des autres car les sorties s'écrivent comme des fonctions du débit. L'idée serait de construire un krigeage fonctionnel pour prédire directement le rendement en tant que sortie fonctionnelle (fonction du débit) et non plus en tant que sortie scalaire.

Bibliographie

[Apley et al., 2006] Apley, D. W., Liu, J., and Chen, W. (2006). Understanding the effects of model uncertainty in robust design with computer experiments. *Journal of Mechanical Design*, 128(4) :945–958.

[Ben Salem, 2018] Ben Salem, M. (2018). *Model selection and adaptive sampling in surrogate modeling : Kriging and beyong*. PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne.

[Beyer and Sendhoff, 2007] Beyer, H.-G. and Sendhoff, B. (2007). Robust optimization – a comprehensive survey. *Computer Methods in Applied Mechanics and Engineering*, 196(33) :3190 – 3218.

[Binois, 2015] Binois, M. (2015). *Uncertainty quantification on pareto fronts and high-dimensional strategies in bayesian optimization, with applications in multi-objective automotive design*. PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne.

[Binois et al., 2015] Binois, M., Ginsbourger, D., and Roustant, O. (2015). Quantifying uncertainty on Pareto fronts with Gaussian process conditional simulations. *European J. Oper. Res.*, 243(2) :386–394.

[Binois and Picheny,] Binois, M. and Picheny, V. Gpareto : An r package for gaussian-process based multi-objective optimization and analysis.

[Booker et al., 1998] Booker, A. J., Dennis, Jr., J. E., Frank, P. D., Serafini, D. B., and Torczon, V. (1998). Optimization using surrogate objectives on a helicopter test example. In *Computational methods for optimal design and control (Arlington, VA, 1997)*, volume 24 of *Progr. Systems Control Theory*, pages 49–58. Birkhäuser Boston, Boston, MA.

[Byrd et al., 1995] Byrd, R., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5) :1190–1208.

[Chambers et al., 1992] Chambers, J. M., Hastie, T. J., et al. (1992). *Statistical models in S*, volume 251. Wadsworth & Brooks/Cole Advanced Books & Software Pacific Grove, CA.

[Chevalier and Ginsbourger, 2013] Chevalier, C. and Ginsbourger, D. (2013). Fast computation of the multi-points expected improvement with applications in batch selection. In Nicosia, G. and Pardalos, P., editors, *Learning and Intelligent Optimization*, pages 59–69, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Coco et al., 2014] Coco, A. A., Solano-Charris, E. L., Santos, A. C., Prins, C., and de Noronha, T. F. (2014). Robust optimization criteria : state-of-the-art and new issues. *Technical Report UTT-LOSI-14001, ISSN : 2266-5064*.

[Coello and Sierra, 2004] Coello, C. A. C. and Sierra, M. R. (2004). A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In *Mexican International Conference on Artificial Intelligence*, pages 688–697. Springer.

[Cornford et al., 2002] Cornford, D., Nabney, I. T., and Williams, C. K. I. (2002). Modelling frontal discontinuities in wind fields. *J. Nonparametr. Stat.*, 14(1-2) :43–58. Statistical models and methods for discontinuous phenomena (Oslo, 1998).

[Cressie, 1993] Cressie, N. A. C. (1993). *Statistics for spatial data*. Wiley Series in Probability and Mathematical Statistics : Applied Probability and Statistics. John Wiley & Sons, Inc., New York. Revised reprint of the 1991 edition, A Wiley-Interscience Publication.

[Darlington et al., 1999] Darlington, J., Pantelides, C., Rustem, B., and Tanyi, B. (1999). An algorithm for constrained nonlinear optimization under uncertainty. *Automatica*, 35(2) :217 – 228.

[Deb et al., 2002] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm : Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2) :182–197.

[Dupuy et al., 2015] Dupuy, D., Helbert, C., Franco, J., et al. (2015). Dicedesign and diceeval : Two r packages for design and analysis of computer experiments. *Journal of Statistical Software*, 65(11) :1–38.

[Durrande, 2001] Durrande, N. (2001). *Étude de classes de noyaux adaptées à la simplification et à l'interprétation des modèles d'approximation. Une approche fonctionnelle et probabiliste.* PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne.

[Durrande et al., 2012] Durrande, N., Ginsbourger, D., and Roustant, O. (2012). Additive covariance kernels for high-dimensional Gaussian process modeling. *Ann. Fac. Sci. Toulouse Math. (6)*, 21(3) :481–499.

[Emmerich et al., 2011] Emmerich, M. T., Deutz, A. H., and Klinkenberg, J. W. (2011). Hypervolume-based expected improvement : Monotonicity properties and exact computation. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 2147–2154. IEEE.

[Everitt et al., 2011] Everitt, B. S., Landau, S., Leese, M., and Stahl, D. (2011). *Cluster analysis*. Wiley Series in Probability and Statistics. John Wiley & Sons, Ltd., Chichester, fifth edition.

[Fonseca et al., 2006] Fonseca, C. M., Paquete, L., and López-Ibáñez, M. (2006). An improved dimension-sweep algorithm for the hypervolume indicator. In *Proceedings of the 2006 Congress on Evolutionary Computation (CEC 2006)*, pages 1157–1163. IEEE Press, Piscataway, NJ.

[Fricker et al., 2013] Fricker, T. E., Oakley, J. E., and Urban, N. M. (2013). Multivariate Gaussian process emulators with nonseparable covariance structures. *Technometrics*, 55(1) :47–56.

[Friedman et al., 2001] Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

[Gabrel et al., 2014] Gabrel, V., Murat, C., and Thiele, A. (2014). Recent advances in robust optimization : An overview. *European journal of operational research*, 235(3) :471–483.

[Gao et al., 2002] Gao, J., Gunn, S., and Kandola, J. (2002). Adapting kernels by variational approach in SVM. In *AI 2002 : Advances in artificial intelligence*, volume 2557 of *Lecture Notes in Comput. Sci.*, pages 395–406. Springer, Berlin.

[Ginsbourger, 2009] Ginsbourger, D. (2009). *Multiples métamodèles pour l'approximation et l'optimisation de fonctions numériques multivariables*. PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne.

[Ginsbourger et al., 2010] Ginsbourger, D., Le Riche, R., and Carraro, L. (2010). Kriging is well-suited to parallelize optimization. In *Computational Intelligence in Expensive Optimization Problems*, pages 131–162. Springer.

[Ginsbourger et al., 2016] Ginsbourger, D., Roustant, O., Schuhmacher, D., Durrande, N., and Lenz, N. (2016). On ANOVA decompositions of kernels and Gaussian random field paths. In *Monte Carlo and quasi-Monte Carlo methods*, volume 163 of *Springer Proc. Math. Stat.*, pages 315–330. Springer, [Cham].

[Göhler et al., 2016] Göhler, S. M., Eifler, T., and Howard, T. J. (2016). Robustness metrics : Consolidating the multiple approaches to quantify robustness. *Journal of Mechanical Design*, 138(11) :111407.

[Hartigan and Wong, 1979] Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136 : A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1) :100–108.

[Henkenjohann and Kunert, 2007] Henkenjohann, N. and Kunert, J. (2007). An efficient sequential optimization approach based on the multivariate expected improvement criterion. *Quality Engineering*, 19(4) :267–280.

[Iooss and Lemaître, 2015] Iooss, B. and Lemaître, P. (2015). *A Review on Global Sensitivity Analysis Methods*, pages 101–122. Springer US, Boston, MA.

[J. Darlington and Rustem, ated] J. Darlington, C. Pantelides, B. T. and Rustem, B. (undated). An Efficient Approximate Algorithm for Robust Optimal Decisions under Uncertainty. Technical report.

[Janusevskis and Le Riche, 2013] Janusevskis, J. and Le Riche, R. (2013). Simultaneous kriging-based estimation and optimization of mean response. *Journal of Global Optimization*, 55(2) :313–336.

[Jeong and Obayashi, 2005] Jeong, S. and Obayashi, S. (2005). 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. IEEE.

[Jin et al., 2005] Jin, R., Chen, W., and Sudjianto, A. (2005). An efficient algorithm for constructing optimal design of computer experiments. *Journal of Statistical Planning and Inference*, 134(1) :268–287.

[Jones, 2001] Jones, D. R. (2001). A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4) :345–383.

[Jones et al., 1998] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4) :455–492.

[Jozefowiez, 2013] Jozefowiez, N. (2013). *Optimisation combinatoire multi-objectif : des méthodes aux problèmes, de la Terre à (presque) la Lune*. PhD thesis, Institut National Polytechnique de Toulouse (INP Toulouse).

[Knowles, 2006] Knowles, J. (2006). Parego : A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1) :50–66.

[Le Gratiet, 2013] Le Gratiet, L. (2013). *Multi-fidelity Gaussian process regression for computer experiments*. PhD thesis, Université Paris-Diderot-Paris VII.

[Lelièvre et al., 2016] Lelièvre, N., Beaurepaire, P., Mattrand, C., Gayton, N., and Otsmane, A. (2016). On the consideration of uncertainty in design : optimization - reliability - robustness. *Structural and Multidisciplinary Optimization*, 54(6) :1423–1437.

[Liu et al., 2007] Liu, W., Zhang, Q., Tsang, E., Liu, C., and Virginas, B. (2007). On the performance of metamodel assisted moea/d. In *International Symposium on Intelligence Computation and Applications*, pages 547–557. Springer.

[Maatouk, 2015] Maatouk, H. (2015). *Correspondance entre régression par processus Gaussien et splines d'interpolation sous contraintes linéaires de type inégalité. Théorie et applications.* PhD thesis. Thèse de doctorat dirigée par Roustant, Olivier et Grammont, Laurence Mathématiques appliquées Saint-Etienne, EMSE 2015.

[Marmin et al., 2015] Marmin, S., Chevalier, C., and Ginsbourger, D. (2015). Differentiating the multipoint expected improvement for optimal batch design. In Pardalos, P., Pavone, M., Farinella, G. M., and Cutello, V., editors, *Machine Learning, Optimization, and Big Data*, pages 37–48, Cham. Springer International Publishing.

[Marrel et al., 2008] Marrel, A., Iooss, B., Van Dorpe, F., and Volkova, E. (2008). An efficient methodology for modeling complex computer codes with Gaussian processes. *Comput. Statist. Data Anal.*, 52(10) :4731–4744.

[Marzat et al., 2013] Marzat, J., Walter, E., and Piet-Lahanier, H. (2013). Worst-case global optimization of black-box functions through kriging and relaxation. *Journal of Global Optimization*, 55(4) :707–727.

[Mebane Jr et al., 2011] Mebane Jr, W. R., Sekhon, J. S., et al. (2011). Genetic optimization using derivatives : the rgenoud package for r. *Journal of Statistical Software*, 42(11) :1–26.

[Mkaouer et al., 2014] Mkaouer, M. W., Kessentini, M., Bechikh, S., Deb, K., and Ó Cinnéide, M. (2014). High dimensional search-based software engineering : finding tradeoffs among 15 objectives for automating software refactoring using nsga-iii. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 1263–1270. ACM.

[Monod et al., 2006] Monod, H., Naud, C., and Makowski, D. (2006). Uncertainty and sensitivity analysis for crop models. *Working with dynamic crop models : Evaluation, analysis, parameterization, and applications*, 4 :55–100.

[Morris and Mitchell, 1995] Morris, M. D. and Mitchell, T. J. (1995). Exploratory designs for computational experiments. *Journal of statistical planning and inference*, 43(3) :381–402.

[Muehlenstaedt et al., 2012] Muehlenstaedt, T., Roustant, O., Carraro, L., and Kuhnt, S. (2012). Data-driven Kriging models based on FANOVA-decomposition. *Stat. Comput.*, 22(3) :723–738.

[Paciorek and Schervish, 2006] Paciorek, C. J. and Schervish, M. J. (2006). Spatial modelling using a new class of nonstationary covariance functions. *Environmetrics*, 17(5) :483–506.

[Padonou and Roustant, 2016] Padonou, E. and Roustant, O. (2016). Polar Gaussian processes and experimental designs in circular domains. *SIAM/ASA J. Uncertain. Quantif.*, 4(1) :1014–1033.

[Parr, 2013] Parr, J. (2013). *Improvement criteria for constraint handling and multiobjective optimization*. PhD thesis, University of Southampton.

[Picheny, 2015] Picheny, V. (2015). Multiobjective optimization using gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing*, 25(6) :1265–1280.

[Ponweiser et al., 2008] Ponweiser, W., Wagner, T., Biermann, D., and Vincze, M. (2008). Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection. In *International Conference on Parallel Problem Solving from Nature*, pages 784–794. Springer.

[Pronzato and Éric Thierry, 2003] Pronzato, L. and Éric Thierry (2003). Robust design with non-parametric models : prediction of second-order characteristics of process variability by kriging1. *IFAC Proceedings Volumes*, 36(16) :537 – 542. 13th IFAC Symposium on System Identification (SYSID 2003), Rotterdam, The Netherlands, 27-29 August, 2003.

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA.

[Roustant et al., 2012] Roustant, O., Ginsbourger, D., and Deville, Y. (2012). Dicekriging, diceoptim : Two r packages for the analysis of computer experiments by kriging-based metamodeling and optimization. *Journal of Statistical Software, Articles*, 51(1) :1–55.

[Saltelli et al., 2000] Saltelli, A., Chan, K., Scott, E. M., et al. (2000). *Sensitivity analysis*, volume 1. Wiley New York.

[Santner et al., 2003] Santner, T. J., Williams, B. J., and Notz, W. I. (2003). *The design and analysis of computer experiments*. Springer Series in Statistics. Springer-Verlag, New York.

[Schott, 1995] Schott, J. R. (1995). Fault tolerant design using single and multicriteria genetic algorithm optimization. Technical report, AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH.

[Schwarz, 1978] Schwarz, G. (1978). Estimating the dimension of a model. *Ann. Statist.*, 6(2) :461–464.

[Snelson, 2008] Snelson, E. L. (2008). *Flexible and efficient Gaussian process models for machine learning*. ProQuest LLC, Ann Arbor, MI. Thesis (Ph.D.)–University of London, University College London (United Kingdom).

[Sobol, 1993] Sobol, I. M. (1993). Sensitivity estimates for nonlinear mathematical models. *Mathematical modelling and computational experiments*, 1(4) :407–414.

[Stein, 1999] Stein, M. L. (1999). *Interpolation of spatial data*. Springer Series in Statistics. Springer-Verlag, New York. Some theory for Kriging.

[Stitson et al., 1997] Stitson, M., Gammerman, A., Vapnik, V., Vovk, V., Watkins, C., and Weston, J. (1997). Support vector regression with anova decomposition kernels. *Advances in kernel methods—Support vector learning*, pages 285–292.

[Sudret, 2012] Sudret, B. (2012). Meta-models for structural reliability and uncertainty quantification. *arXiv preprint arXiv :1203.2062*.

[Svenson, 2011] Svenson, J. (2011). *Computer experiments : Multiobjective optimization and sensitivity analysis*. PhD thesis, The Ohio State University.

[Svenson and Santner, 2016] Svenson, J. and Santner, T. (2016). Multiobjective optimization of expensive-to-evaluate deterministic computer simulator models. *Computational Statistics & Data Analysis*, 94 :250–264.

[Tarantola et al., 2007] Tarantola, S., Gatelli, D., Kucherenko, S., Mauntz, W., et al. (2007). Estimating the approximation error when fixing unessential factors in global sensitivity analysis. *Reliability Engineering & System Safety*, 92(7) :957–960.

[Troian et al., 2016] Troian, R., Shimoyama, K., Gillot, F., and Besset, S. (2016). Methodology for the design of the geometry of a cavity and its absorption coefficients as random design variables under vibroacoustic criteria. *Journal of Computational Acoustics*, 24(02) :1650006.

[Ur Rehman and Langelaar, 2015] Ur Rehman, S. and Langelaar, M. (2015). Efficient global robust optimization of unconstrained problems affected by parametric uncertainties. *Structural and Multidisciplinary Optimization*, 52(2) :319–336.

[Ur Rehman et al., 2014] Ur Rehman, S., Langelaar, M., and van Keulen, F. (2014). Efficient kriging-based robust optimization of unconstrained problems. *Journal of Computational Science*, 5(6) :872–881.

[Van Loan, 1996] Van Loan, C. F. (1996). Matrix computations (johns hopkins studies in mathematical sciences).

[Van Veldhuizen, 1999] Van Veldhuizen, D. A. (1999). *Multiobjective Evolutionary Algorithms : Classifications, Analyses, and New Innovations*. PhD thesis, Wright Patterson AFB, OH, USA. AAI9928483.

[Vazquez and Bect, 2010] Vazquez, E. and Bect, J. (2010). Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference*, 140(11) :3088 – 3095.

[Villa-Vialaneix et al., 2012] Villa-Vialaneix, N., Follador, M., Ratto, M., and Leip, A. (2012). A comparison of eight metamodeling techniques for the simulation of n 2 o fluxes and n leaching from corn crops. *Environmental Modelling & Software*, 34 :51–66.

[Wagner et al., 2010] Wagner, T., Emmerich, M., Deutz, A., and Ponweiser, W. (2010). On expected-improvement criteria for model-based multi-objective optimization. In *International Conference on Parallel Problem Solving from Nature*, pages 718–727. Springer.

[Welch et al., 1992] Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J., and Morris, M. D. (1992). Screening, predicting, and computer experiments. *Technometrics*, 34(1) :15–25.

[Yi, 2009] Yi, G. (2009). *Variable selection with penalized Gaussian process regression models*. PhD thesis, University of Newcastle Upon Tyne.

[Zhang et al., 2010] Zhang, Q., Liu, W., Tsang, E., and Virginas, B. (2010). Expensive multiobjective optimization by moea/d with gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14(3) :456–474.

[Zitzler and Thiele, 1999] Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms : a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4) :257–271.